DVP-PLC Application Manual

Programming

Revision III

DVP-PLC
Application Manual

Programming

Revision III

ES / SS / EX
SA / SX /SC
EH / EH2 / SV

www.delta.com.tw/industrialautomation

DVP-PLC Application Manual
【 *Programming*】

Table of Contents

Chapter 4 Step Ladder Instructions

Chapter 5 Categories & Use of Application Instructions

Chapter 6 Application Instructions API 00-49

Chapter 7 Application Instructions API 50-99

Chapter 8 Application Instructions API 100-149

Chapter 9 Application Instructions API 150-199

Chapter 10 Application Instructions API 215-246

Foreword: Background and Functions of PLC

PLC (Programmable Logic Controller) is an electronic device, previously called "sequence controller". In 1978, NEMA (National Electrical Manufacture Association) in the United States officially named it as "programmable logic controller". PLC reads the status of the external input devices, e.g. keypad, sensor, switch and pulses, and execute by the microprocessor logic, sequential, timing, counting and arithmetic operations according the status of the input signals as well as the pre-written program stored in the PLC. The generated output signals are sent to output devices as the switch of a relay, electromagnetic valve, motor drive, control of a machine or operation of a procedure for the purpose of machine automation or processing procedure. The peripheral devices (e.g. personal computer/handheld programming panel) can easily edit or modify the program and monitor the device and conduct on-site program maintenance and adjustment. The widely used language in designing a PLC program is the ladder diagram.

With the development of the electronic technology and wider applications of PLC in the industry, for example in position control and the network function of PLC, the input/output signals of PLC include DI (digital input), AI (analog input), PI (pulse input), NI (numeric input), DO (digital output), AO (analog output), and PO (pulse output). Therefore, PLC will still stand important in the industrial automation field in the future.

1.1   The Working Principles of Ladder Diagram

The ladder diagram was a diagram language for automation developed in the WWII period, which is the oldest and most widely adopted language in automation. In the initial stage, there were only A (normally open) contact, B (normally closed) contact, output coil, timer and counter…the sort of basic devices on the ladder diagram (see the power panel that is still used today). After the invention of PLC, the devices displayable on the ladder diagram are added with differential contact, latched coil and the application commands which were not in a traditional power panel, for example the addition, subtraction, multiplication and division operations.

The working principles of the traditional ladder diagram and PLC ladder diagram are basically the same. The only difference is that the symbols on the traditional ladder diagram are more similar to its original form, and PLC ladder diagram adopts the symbols that are easy to recognize and shown on computer or data sheets. In terms of the logic of the ladder diagram, there are combination logic and sequential logic.

1.   Combination Logic

Examples of traditional ladder diagram and PLC ladder diagram for combination logic:

<div align="center">Traditional Ladder Diagram   PLC Ladder Diagram</div>
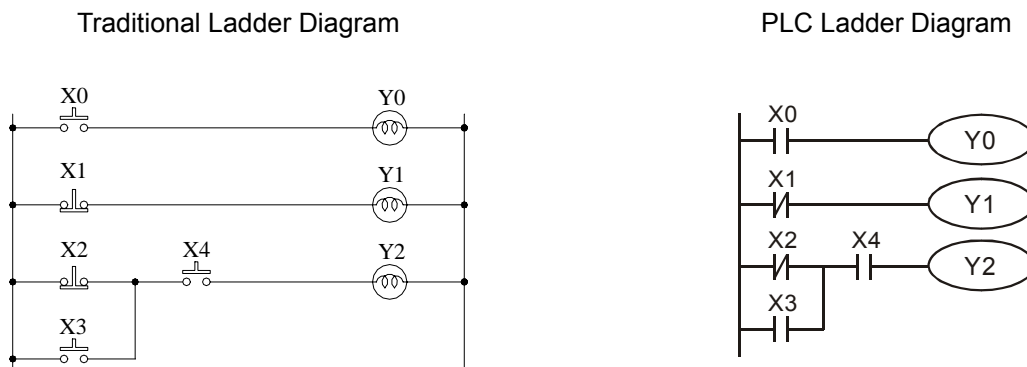
Row 1: Using a normally open (NO) switch X0 ("A" switch or "A" contact). When X0 is not pressed, the contact will be open loop (Off), so Y0 will be Off. When X0 is pressed, the contact will be On, so Y0 will be On.

Row 2: Using a normally closed (NC) switch X1 ("B" switch or "B" contact). When X1 is not pressed, the contact will be On, so Y1 will be On. When X1 is pressed, the contact will be open loop (Off), so Y1 will be Off.

Row 3: The combination logic of more than one input devices. Output Y2 will be On when X2 is not pressed or X3 and X4 are pressed.

2. Sequential Logic

Sequential logic is a circuit with "draw back" structure, i.e. the output result of the circuit will be drawn back as an input criterion. Therefore, under the same input criteria, different previous status or action sequence will follow by different output results.

Examples of traditional ladder diagram and PLC ladder diagram for sequential logic:
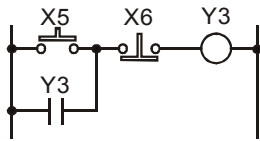
Traditional Ladder Diagram   PLC Ladder Diagram
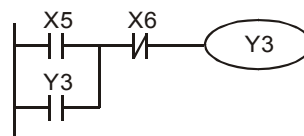
When the circuit is first connected to the power, though X6 is On, X5 is Off, so Y3 will be Off. After X5 is pressed, Y3 will be On. Once Y3 is On, even X5 is released (Off), Y3 can still keep its action because of the draw back (i.e. the self-retained circuit). The actions are illustrated in the table below.

Device status / Action sequence	X5	X6	Y3
1	No action	No action	Off
2	Action	No action	On
3	No action	No action	On
4	No action	Action	Off
5	No action	No action	Off

From the table above, we can see that in different sequence, the same input status can result in different output results. For example, switch X5 and X6 of action sequence 1 and 3 do not act, but Y3 is Off in sequence 1 and On in sequence 3. Y3 output status will then be drawn back as input (the so-called "draw back"), making the circuit being able to perform sequential control, which is the main feature of the ladder diagram circuit. Here we only explain contact A, contact B and the output coil. Other devices are applicable to the same method. See Chapter 3 "Basic instructions" for more details.

1.2 Differences Between Traditional Ladder Diagram and PLC Ladder Diagram

Though the principles of traditional ladder diagram and PLC ladder diagram are the same, in fact, PLC adopts microcomputer to simulate the motions of the traditional ladder diagram, i.e. scan-check status of all the input devices and output coil and calculate to generate the same output results as those from the traditional ladder diagram based on the logics of the ladder diagram. Due to that there is only one microcomputer, we can only check the program of the ladder diagram one by one and calculate the output results according to the program and the I/O status before the cyclic process of sending the results to the output interface → re-reading of the input status → calculation → output. The time spent in the cyclic process is called the "scan time" and the time can be longer with the expansion of the program. The scan time can cause delay from the input detection to output response of the PLC. The longer the delay,

the bigger the error is to the control. The control may even be out of control. In this case, you have to choose a PLC with faster scan speed. Therefore, the scan speed is an important specification requirement in a PLC. Owing to the advancement in microcomputer and ASIC (IC for special purpose), there has been great improvement in the scan speed of PLC nowadays. See the figure below for the scan of the PLC ladder diagram program.

The output result is calculated based on the ladder diagram. (The result has not yet sent to the external output point, but the internal device will perform an immediate output.)

Read input status from outside

Executing in cycles

Send the result to the output point
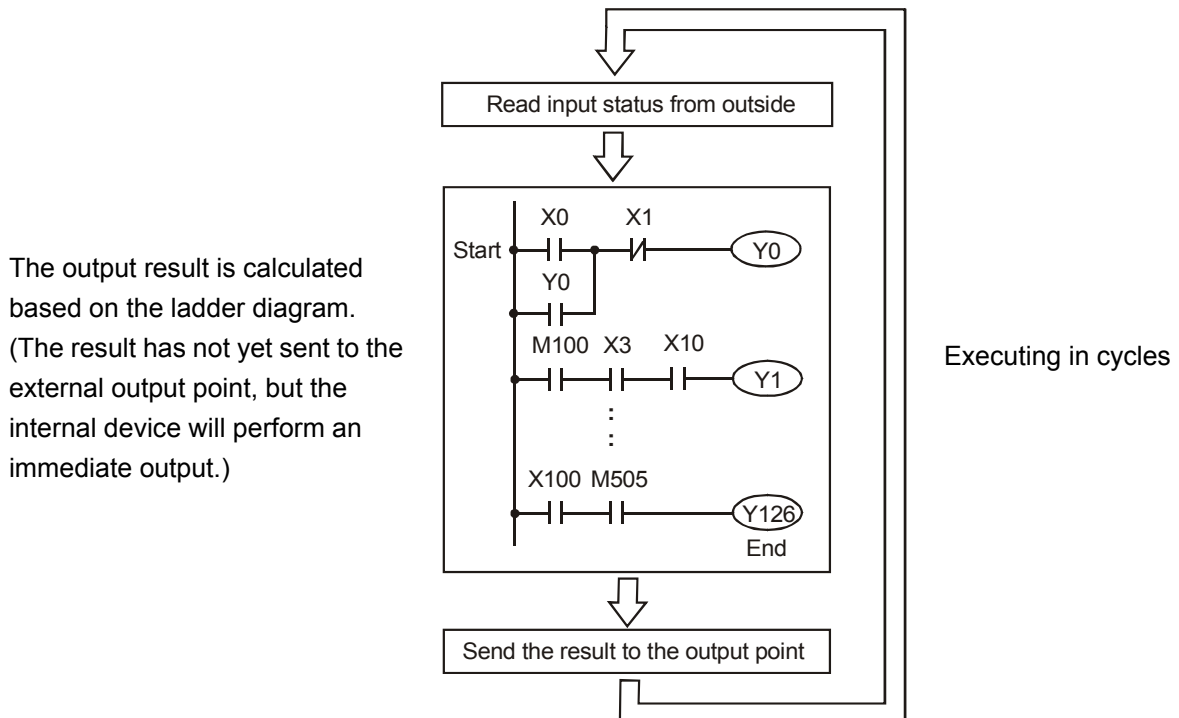
Besides the difference in the scan time, PLC ladder and traditional ladder diagram also differ in "reverse current". For example, in the traditional ladder diagram illustrated below, when X0, X1, X4 and X6 are On and others are Off, Y0 output on the circuit will be On as the dotted line goes. However, the PLC ladder diagram program is scanned from up to down and left to right. Under the same input circumstances, the PLC ladder diagram editing tool WPLSoft will be able to detect the errors occurring in the ladder diagram.

Reverse current of traditional ladder diagram
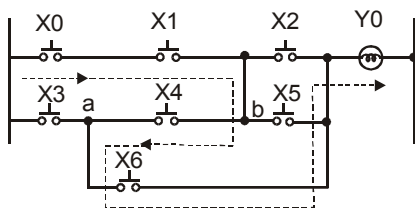
Reverse current of PLC ladder diagram
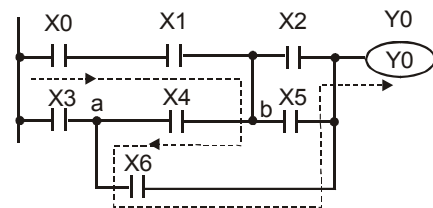
Error detected in the third row

1.3   How to Edit Ladder Diagram

Ladder diagram is a diagram language frequently applied in automation. The ladder diagram is composed of the symbols of electric control circuit. The completion of the ladder diagram by the ladder diagram editor is the completion of the PLC program design. The control flow illustrated by diagram makes the flow more straightforward and acceptable for the technicians of who are familiar with the electric control circuit. Many basic symbols and actions in

the ladder diagram come from the frequently-seen electromechanical devices, e.g. buttons, switches, relay, timer and counter, etc. in the traditional power panel for automation control.

Internal devices in the PLC: The types and quantity of the devices in the PLC vary in different brand names. Though the internal devices in the PLC adopts the names, e.g. transistor, coil, contact and so on, in the traditional electric control circuit, these physical devices do not actually exist inside the PLC. There are only the corresponding basic units (1 bit) inside the memory of the PLC. When the bit is "1", the coil will be On, and when the bit is "0", the coil will be Off. The normally open contact (NO or contact A) directly reads the value of the corresponding bit. The normally close contact (NC or contact B) reads the opposite state of the value of the corresponding bit. Many relays will occupy many bits. 8 bits equal a "byte". 2 bytes construct a "word" and 2 words combined is "double word". Byte, word or double words are used when many relays are processed (e.g. addition/subtraction, displacement) at the same time. The other two devices, timer and counter, in the PLC have coil, timer value and counter value and they have to process some values in byte, word or double word.

All kinds of internal devices in the value storage area in the PLC occupy their fixed amount of storage units. When you use these devices, you are actually read the contents stored in the form of bit, byte or word.

Introductions on the basic internal devices in the PLC (See Ch 2. Functions of Devices in DVP-PLC for more details.)

Device	Functions
Input relay	The input relay is an internal memory (storage) unit in the PLC corresponding to a external input point and is used for connecting to the external input switches and receiving external input signals. The input relay will be driven by the external input signals which make it "0" or "1". Program designing cannot modify the status of the relay, i.e. it cannot re-write the basic unit of a relay, nor can it force On/Off of the relay by HPP/WPLSoft. SA/SX/SC/EH/EH2/SV series MPU can simulate input relay X and force On/Off of the relay. But the status of the external input points will be updated and disabled, i.e. the external input signals will not be read into their corresponding memories inside PLC, but only the input points on the MPU. The input points on the extension modules will still operate normally. There are no limitations on the times of using contact A and contact B of the input relay. The input relays without corresponding input signals can only be left unused and cannot be used for other purposes. ☞　Device indication: X0, X1,…X7, X10, X11,… are indicated as X and numbered in octal form. The No. of input points are marked on MPU and extension modules.
Output relay	The output relay is an internal memory (storage) unit in the PLC corresponding to a external output point and is used for connecting to the external load. The output relay will be driven by the contact of an input relay, contacts of other internal devices and the contacts on itself. A normally open contact of the output relay is connected to the external load. Same as the input contacts, there are no limitations on the times of using other contacts of the output relay. The output relay without corresponding output signals can only be left unused and can be used as input relay if necessary. ☞　Device indication: Y0, Y1,…Y7, Y10, Y11,…are indicated as Y and numbered in octal form. The No. of output points are marked on MPU and extension modules.
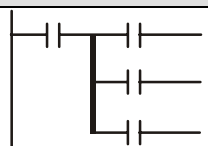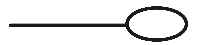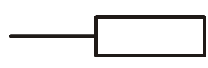
Internal relay	The internal relay does not have connection with the external. It is an auxiliary relay inside the PLC with the functions same as those of the auxiliary (middle) relay in the electric control circuit. Every internal relay corresponds to a basic internal storage unit and can be driven by the contacts of the input relay, contacts of the output relay and the contacts of other internal devices. There are no limitations on the times of using the contacts of the internal relay and there will be no output from the internal relay, but from the output point. ☞ Device indication: M0, M1,…, M4095 are indicated as M and numbered in decimal form.
Step	DVP series PLC offers a step-type control program input method. STL instruction controls the transfer of step S, which makes it easy for the writing of the control program. If you do not use any step program in the control program, step S can be used as a internal relay M as well as an alarm point. ☞ Device indication: S0, S1,…S1023 are indicated as S and numbered in decimal form.
Timer	The timer is used for timing and has coil, contact and register in it. When the coil is On and the estimated time is reached, its contact will be enabled (contact A closed, contact B open). Every timer has its fixed timing period (unit: 1ms/10ms/100ms). Once the coil is Off, the contact iwlwl be disabled (contact A open, contact B closed) and the present value on the timer will become "0". ☞ Device indication: T0, T1,…,T255 are indicated as T and numbered in decimal form. Different No. refers to different timing period.
Counter	The counter is used for counting. Before using the counter, you have to give the counter a set value (i.e. the number of pulses for counting). There are coil, contact and registers in the counter. When the coil goes from Off to On, the counter will regard it as an input of 1 pulse and the present value on the counter will plus "1". We offer 16-bit and 32-bit high-speed counters for our users. ☞ Device indication: C0, C1,…,C255 are indicated as C and numbered in decimal form.
Data register	Data processing and value operations always occur when the PLC conducts all kinds of sequential control, timing and counting. The data register is used for storing the values or all kinds of parameters. Every register is able to store a word (16-bit binary value). Double words will occupy 2 adjacent data registers. ☞ Device indication: D0, D1,…,D9,999 are indicated as D and numbered in decimal form.

File register	The file register is used for storing the data or all kinds of parameters when the data registers required for processing the data and value operations are insufficient. Every file register is able to store a 16-bit word. Double words will occupy 2 adjacent file registers. In SA/SX/SC series MPU, there are 1,600 file registers. In EH/EH2/SV series MPU, there are 10,000 file registers. There is not an actual device No. for a file register. The reading and writing of file registers should be executed by instructions API 148 MEMR, API 149 MEMW, or through the peripheral device HPP02 and WPLSoft. ☞ Device indication: K0 ~ K9,999, numbered in decimal form.
Index register	E and F index registers are 16-bit data registers as other data registers. They can be read and written and can be used in word devices, bit devices or as a constant for index indication. ☞ Device indication: E0 ~ E7, F0 ~ F7 are indicated as E and F and numbered in decimal form.

The structure of a ladder diagram:

Structure	Explanation	Instruction	Devices Used
	Normally open, contact A	LD	X, Y, M, S, T, C
	Normally closed, contact B	LDI	X, Y, M, S, T, C
	Normally open in series connection	AND	X, Y, M, S, T, C
	Normally closed in series connection	ANI	X, Y, M, S, T, C
	Normally open in parallel connection	OR	X, Y, M, S, T, C
	Normally closed in parallel connection	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 connection	ANDP	X, Y, M, S, T, C
	Falling-edge trigger in series connection	ANDF	X, Y, M, S, T, C
	Rising-edge trigger in parallel connection	ORP	X, Y, M, S, T, C
	Falling-edge trigger in parallel connection	ORF	X, Y, M, S, T, C
	Block in series connection	ANB	-
	Block in parallel connection	ORB	-

Structure	Explanation	Instruction	Devices Used
	Multiple output	MPS MRD MPP	-
	Coil driven output instruction	OUT	Y, M, S
	Step ladder	STL	S
	Basic instruction Application instruction	Application instructions	See Ch.3 for basic instructions (RST/SET and CNT/TMR) and Ch.5 ~ 10 for application instructions
	Inverse logic	INV	-

Block:

A block is a series or parallel operation composed of more than 2 devices. There are series block and parallel block.

Series block

Parallel block
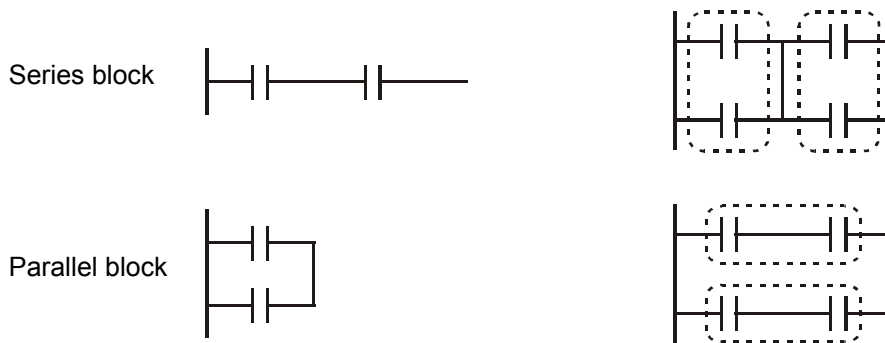
Separation line and combination line:

The vertical line is used for separating the devices. For the devices on the left, the vertical line is a combination line, indicating that there are at least 2 rows of circuits on the left connected with the vertical line. For the devices on the right, the vertical line is a separation line, indicating that there are at least 2 rows of circuits interconnected on the right side of the vertical line).

Combination line for block 1
Separation line for block 2

Combination line for block 2
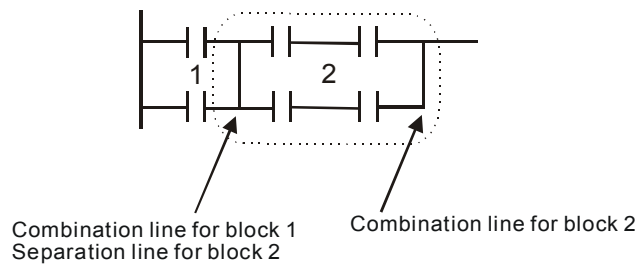
Network:

A complete block network is composed of devices and all kinds of blocks. The blocks or devices connectable by a vertical line or continuous line belong to the same network.

An independent network

An incomplete network
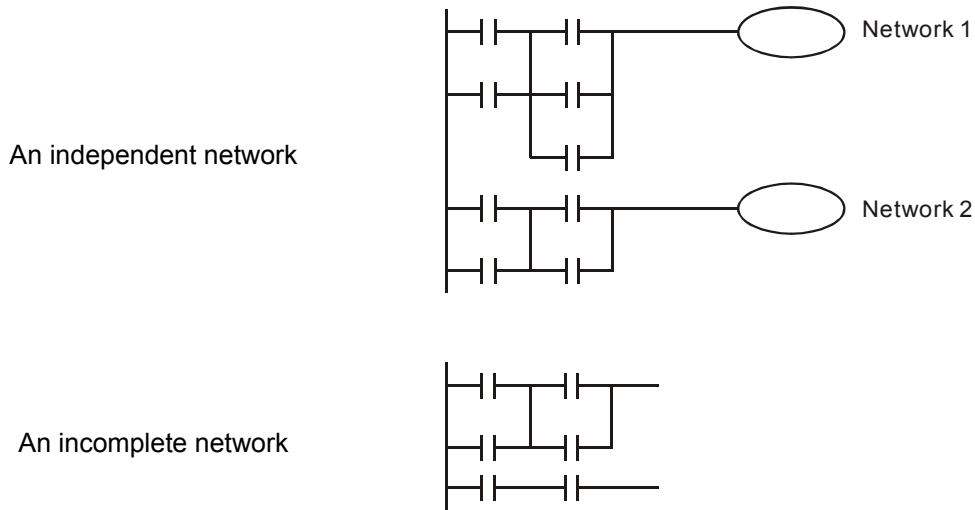
1.4 How to Edit a PLC Ladder Diagram

The editing of the program should start from the left power line and ends at the right power line, a row after another. The drawing of the right power line will be omitted if edited from WPLSoft. A row can have maximum 11 contacts on it. If 11 is not enough, you can continuously connect more devices and the continuous number will be generated automatically. The same input points can be used repeatedly. See the figure below:

Continuous number
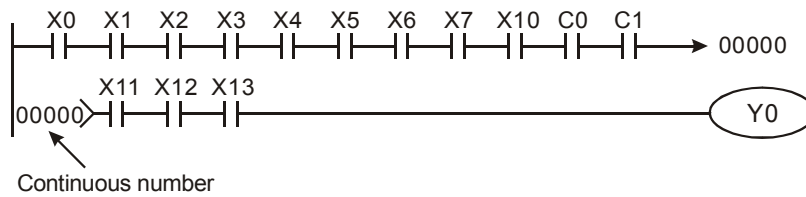
The operation of the ladder diagram program is scanning from top left to bottom right. The coil and the operation frame of the application instruction belong to the output side in the program and are placed in the right if the ladder diagram. Take the figure below for example, we will step by step explain the process of a ladder diagram. The numbers in the black circles indicate the order.
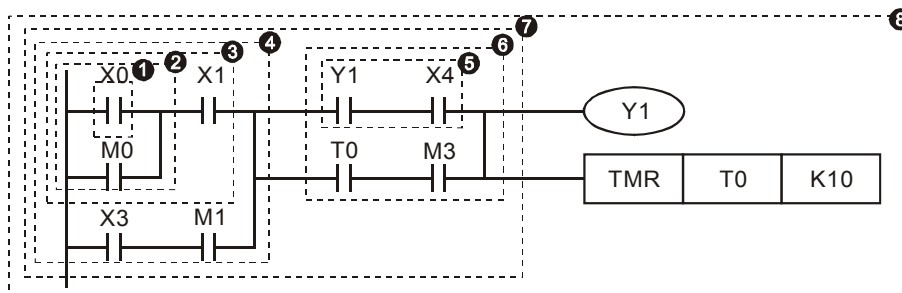
The order of the instructions:

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
```

Explanations on the basic structures in the ladder diagram:

1. LD (LDI) instruction: Given in the start of a block.
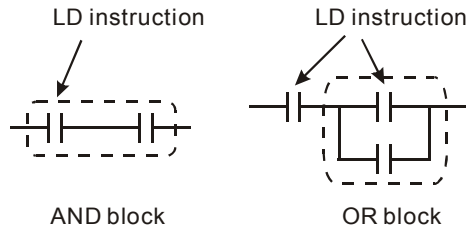
The structure of LDP and LDF instructions are the same as that of LD instruction, and the two only differ in their actions. LDP and LDF instructions only act at the rising edge or falling edge when the contact is On, as shown in the figure below.
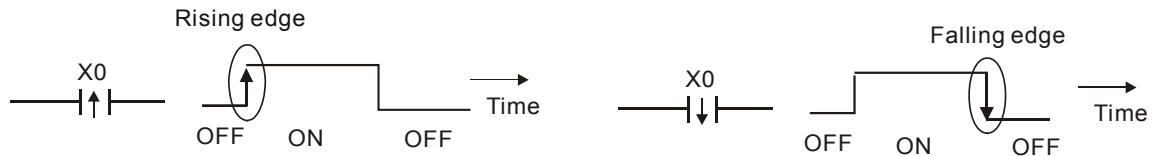
2. AND (ANI) instruction: A single device connects to another single device or a block in series
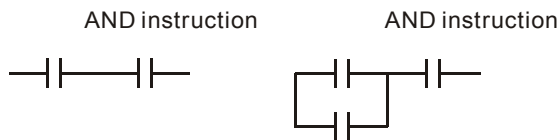
The structure of ANDP and ANDF instructions are the same. ANDP and ANDF instructions only act at the rising edge or falling edge.

3. OR (ORI) instruction: A single device connects to another single device or a block
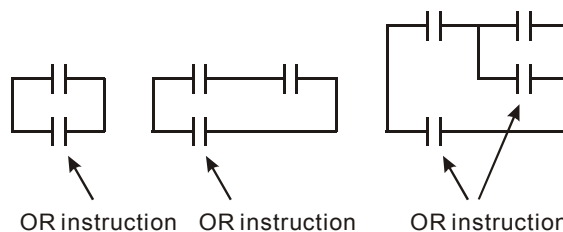
The structure of ORP and ORF instructions are the same. ORP and ORF instructions only act at the rising edge or falling edge.

4. ANB instruction: A block connects to a device or another block in series
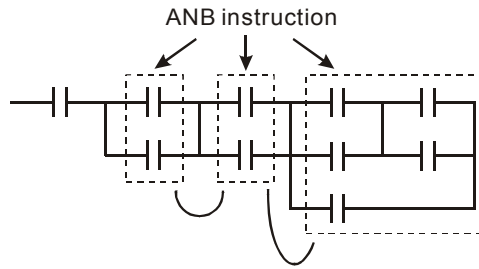
5. ORB instruction: A block connects to a device or another block in parallel
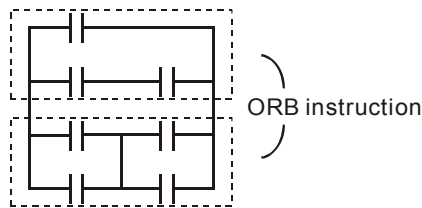
If the ANB and ORB operations are with several blocks, the operation should be performed from up to down or left to right, combining into a block or network.

6. MPS, MRD, MPP instructions: Bifurcation point of multiple outputs, for generating many and diverse outputs.

MPS instruction is the start of the bifurcation point. The bifurcation point is the intersection of the horizontal line and vertical line. We will have to determine whether to give a contact memory instruction by the contact status of the same vertical line. Basically, every contact can be given a memory instruction, but considering the convenience of operating the PLC and the limitation on its capacity, some parts in the ladder diagram will be omitted during the conversion. We can determine the type of contact memory instruction by the structure of the ladder diagram. MPS is recognized as "┬" and the instruction can be given continuously for 8 times.

MRD instruction is used for reading the memory of the bifurcation point. Due to that the same vertical line is of the same logic status, in order to continue analyzing other ladder diagrams, we have to read the status of the original contact again. MRD is recognized as "├".

MPP instruction is used for reading the start status of the top bifurcation point and popping it out from the stack. Since MPP is the last item on the vertical line, the vertical line ends at this point.

MPP is recognized as "└". Using the method given above for the analysis cannot be wrong. However, sometimes the compiling program will ignore the same output status, as shown in the figure.
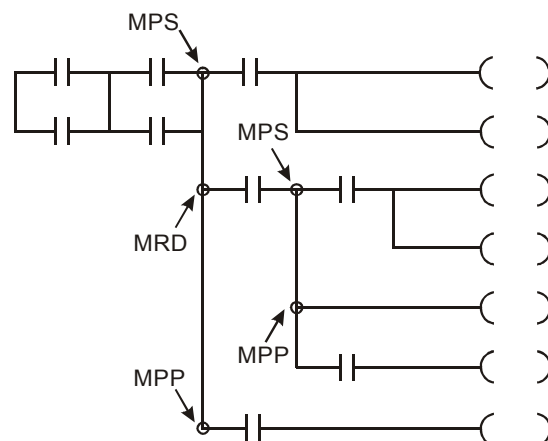
7. STL instruction: Used for designing the syntax of the sequential function chart (SFC).

STL instruction allows the program designer a clearer and readable picture of the sequence of the program as when they draw a sequence chart. From the figure below, we can see clearly the sequence to be planned. When the step S moves to the next step, the original S will be "Off". Such a sequence can then be converted into a PLC ladder diagram and called "step ladder diagram".
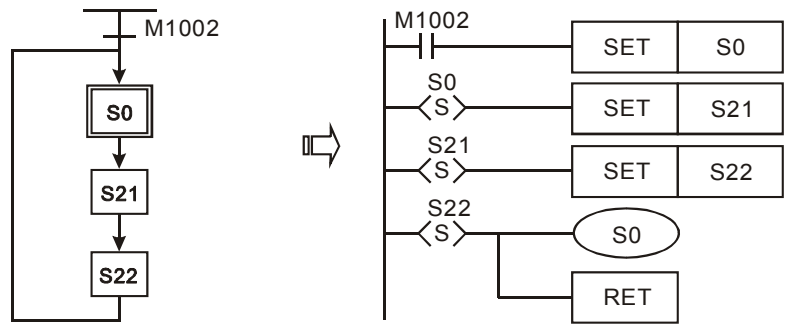
8. RET instruction: Placed after the completed step ladder diagram.

RET also has be placed after STL instruction. See the example below.
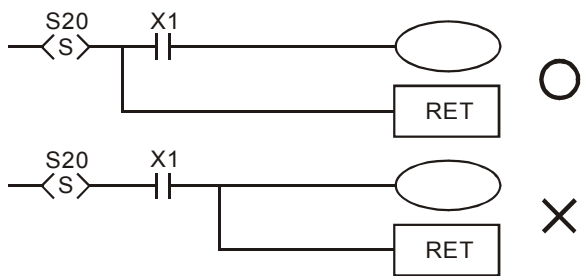
See step ladder instructions [STL], [RET] in Ch. 4 for the structure of the ladder diagram.

1.5 The Conversion of PLC Command and Each Diagram Structure

Ladder Diagram
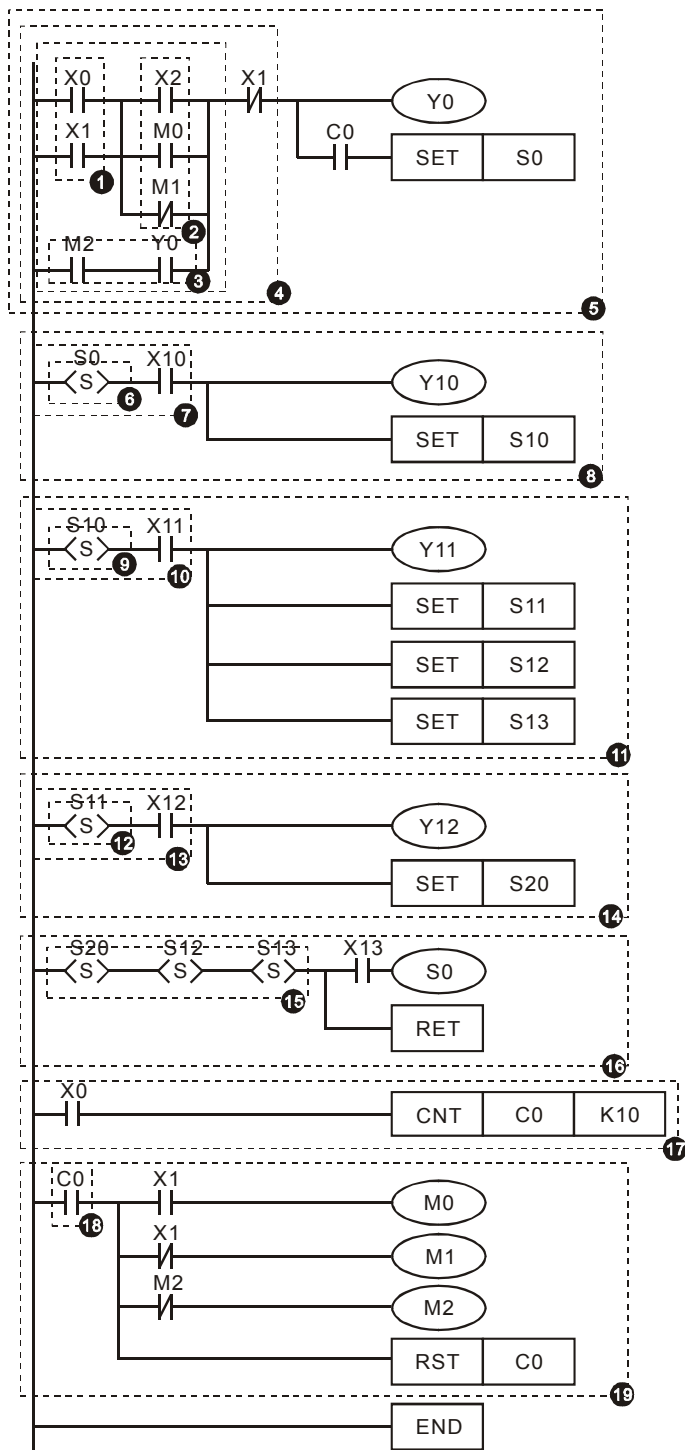
LD	X0	❶ OR block
OR	X1	
LD	X2	❷ OR block
OR	M0	
ORI	M1	
ANB		Series connection blcok
LD	M2	❸ AND block
AND	Y0	
ORB		Parallel connection block
AN I	X1	❹ ANI
OUT	Y0	❺ The output will continue following the status of ❹
AND	C0	Multiple outputs
SET	S0	
STL	S0	Step ladder Start ❻ ❼ ❽
LD	X10	Status S0 and X10 operation
OUT	Y10	Status working item and step point transfer
SET	S10	
STL	S10	❾ Withdraw S10 status
LD	X11	❿ Withdraw X11 status ⓫
OUT	Y11	
SET	S11	Status working item and step point transfer
SET	S12	
SET	S13	
STL	S11	⓬ Withdraw S11 status
LD	X12	⓭ Withdraw X12 status ⓮
OUT	Y12	Status working item and step point transfer
SET	S20	
STL	S20	⓯ Bifurcation convergence
STL	S12	
STL	S13	⓰ End of step ladder
LD	X13	Status working item and step point transfer
OUT	S0	
RET		Return
LD	X0	⓱
CNT	C0 K10	
LD	C0	⓲ Read C0
MPS		
AND	X1	⓳
OUT	M0	
MRD		Multiple outputs
AN I	X1	
OUT	M1	
MPP		
AN I	M2	
OUT	M2	
RST	C0	
END		End of program

■ **Fuzzy Syntax**

The correct ladder diagram analysis and combination should be conducted from up to down and left to right.

However, without adopting this principle, some instructions can make the same ladder diagram.

Example Program 1

See the ladder diagram below. There are 2 ways to indicate the ladder by instruction programs with the same result.
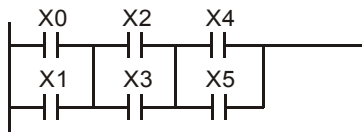
Ideal way		Less ideal way	
LD	X0	LD	X0
OR	X1	OR	X1
LD	X2	LD	X2
OR	X3	OR	X3
ANB		LD	X4
LD	X4	OR	X5
OR	X5	ANB	
ANB		ANB	

The two instruction programs will be converted into the same ladder diagram. The difference between the ideal one and less ideal one is the operation done by the MPU. For the ideal way, the combination is done block by block whereas the less idea way combines all the blocks combine with one another in the last step. Though the length of the program codes of the two ways are equal, the combination done in the last step (by ANB instruction, but ANB cannot be used continuously for more than 8 times) will have to store up the previous calculation results in advance. In our case, there are only two blocks combined and the MPU allows such kind of combination. However, once the number of blocks exceed the range that the MPU allows, problems will occur. Therefore, the best way is to execute the block combination instruction after a block is made, which will also make the logic sequence planned by the programmer more in order.

Example Program 2

See the ladder diagram below. There are 2 ways to indicate the ladder by instruction programs with the same result.
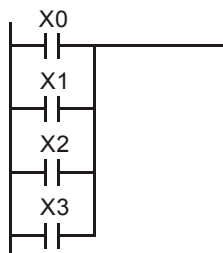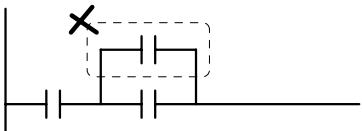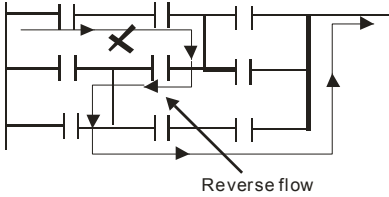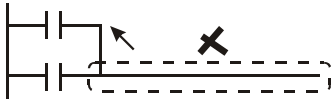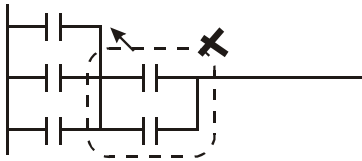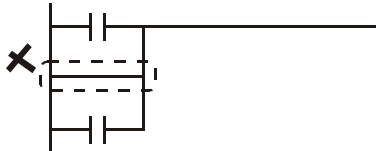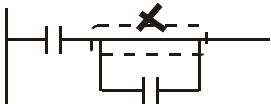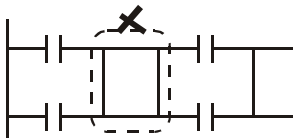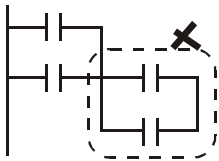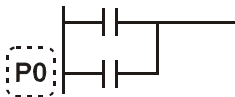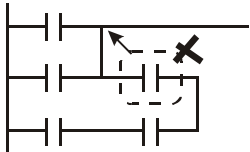
Ideal way		Less ideal way	
LD	X0	LD	X0
OR	X1	LD	X1
OR	X2	LD	X2
OR	X3	LD	X3
		ORB	
		ORB	
		ORB	

In this example, the program codes and the operation memory in the MPU increase in the less ideal way. Therefore, it is better that you edit the program following the defined sequence.

■ **Incorrect Ladder Diagram**

PLC processes the diagram program from up to down and left to right. Though we can use all kinds of ladder symbols to combine into various ladder diagrams, when we draw a ladder diagram, we will have to start the diagram from the left power line and end it at the right power line (In WPLSoft ladder diagram editing area, the right power line is omitted), from left to right horizontally, one row after another from up to down. See bellows for the frequently seen incorrect diagrams:

	OR operation upward is not allowed.
	"Reverse flow" exists in the signal circuit from the beginning of input to output.
	The up-right corner should output first.
	Combining or editing should be done from the up-left to the bottom-right. The dotted-lined area should be moved up.
	Parallel operation with empty device is not allowed.
	Empty device cannot do operations with other devices.
	No device in the middle block.
	Devices and blocks in series should be horizontally aligned.
	Label P0 should be in the first row of a complete network.
	Blocks connected in series should be aligned with the upmost horizontal line.

1.6   Simplified Ladder Diagram

■ When a series block is connected to a parallel block in series, place the block in the front to omit ANB instruction.
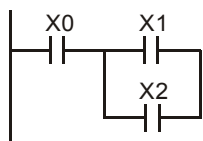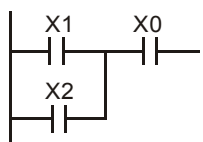
Ladder diagram complied into instruction

LD	X0
LD	X1
OR	X2
ANB	

⇩

Ladder diagram complied into instruction

LD	X1
OR	X2
AND	X0

■ When a single device is connected to a block in parallel, place the block on top to omit ORB instruction.
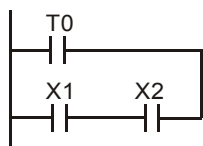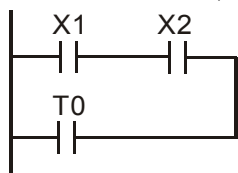
Ladder diagram complied into instruction

LD	T0
LD	X1
AND	X2
ORB	

⇩

Ladder diagram complied into instruction

LD	X1
AND	X2
OR	T0

■ In diagram (a), the block on top is shorter than the block in the bottom, we can switch the position of the two blocks to achieve the same logic. Due to that diagram (a) is illegal, there is a "reverse flow" in it.

(a)
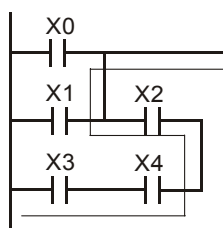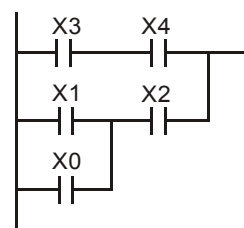
Ladder diagram complied into instruction

LD	X0
OR	X1
AND	X2
LD	X3
AND	X4
ORB	

⇩

Ladder diagram complied into instruction

(b)

LD	X3
AND	X4
LD	X1
OR	X0
AND	X2
ORB	

■ MPS and MPP instruction can be omitted when the multiple outputs in the same horizontal line do not need to operate with other input devices.
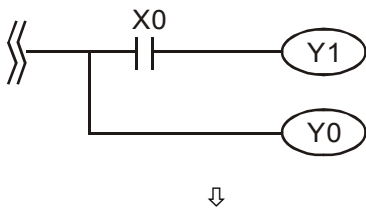
Ladder diagram complied into instruction

MPS	
AND	X0
OUT	Y1
MPP	
OUT	Y0
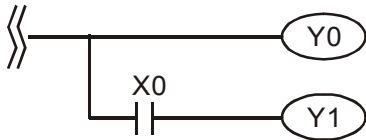
Ladder diagram complied into instruction

OUT	Y0
AND	X0
OUT	Y1

■ Correct the circuit of reverse flow

In the following two examples, the diagram in the left hand side is the ladder diagram we desire. However, the illegal "reverse flow" in it is incorrect according to our definition on the ladder diagram. We modify the diagram into the diagram in the right hand side.

Example 1
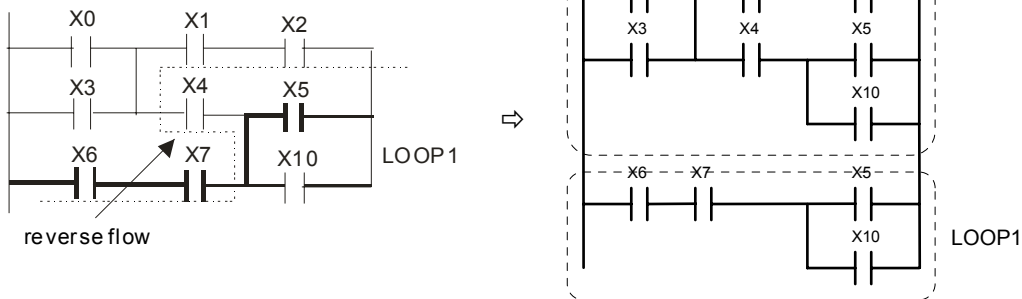
Example 2
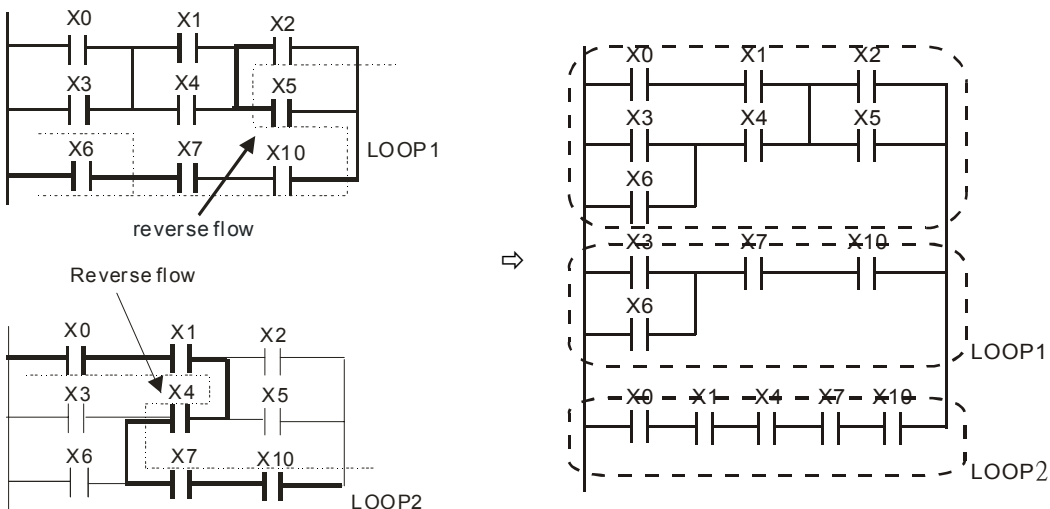
1.7   Basic Program Designing Examples

■ **Start, Stop and Latched**

In some application occasions, we need to use the transient close/open buttons for the start and stop of an equipment. To maintain its continuous action, you have to design latched circuits.

Example 1: Stop first latched circuit

When the normally open contact X1 = On and the normally closed contact X2 = Off, Y1 will be On. If you make X2 = On at this time, Y1 will be Off. It is the reason why this is called "stop first".
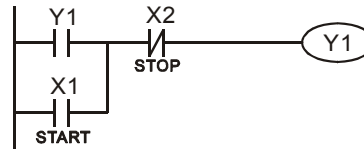
Example 2: Start first latched circuit

When the normally open contact X1 = On and the normally closed contact X2 = Off, Y1 will be On and latched. If you make X2 = On at this time, Y1 will continue to be On because of the latched contact. It is the reason why this is called "start first".
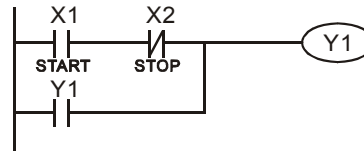
Example 3: Latched circuit for SET and RST instructions

See the diagram in the right hand side for the latched circuit consist of RST and SET instructions.

In the stop first diagram, RST is placed after SET. PLC executes the program from up to down, so the On/Off of Y1 will be determined upon its status in the end of the program. Therefore, when X1 and X2 are enabled at the same time, Y1 will be Off. It is the reason why this is called "stop first".

In the start first diagram, SET is placed after RST. When X1 and X2 are enabled at the same time, Y1 will be On. It is the reason why this is called "start first".
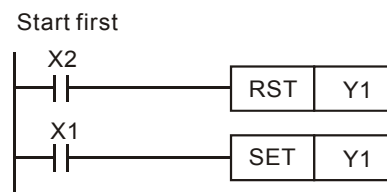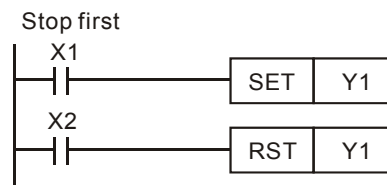
Example 4: Power shutdown latched

The auxiliary relay M512 is latched (see instruction sheets for DVP series PLC MPU). The circuit can not only be latched when the power is on, but also keep the continuity of the original control when the power is shut down and switched on again.
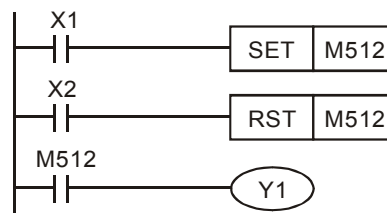
■ **Frequently Used Control Circuit**

Example 5: Conditional control
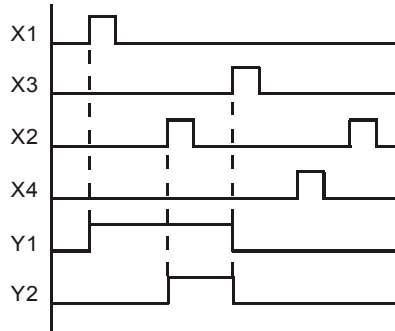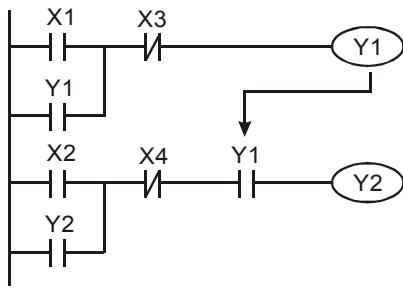
X1 and X3 enables and disables Y1; X2 and X4 enables and disables Y2, and all are latched. Due to that the normally open contact of Y1 is connected to the circuit of Y2 in series, Y1 becomes an AND condition for Y2. Therefore, only when Y1 is enabled can Y2 be enabled.

Example 6: Interlock control
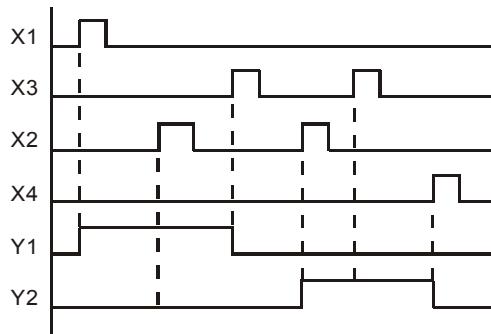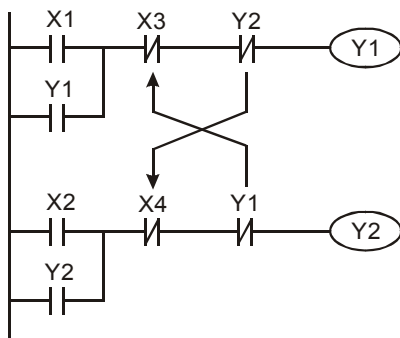
Which of the X1 and X2 is first enabled decides either the corresponding output Y1 or Y2 will be enabled first. Either Y1 or Y2 will be enabled at a time, i.e. Y1 and Y2 will not be enabled at the same time (the interlock). Even X1 and X2 are enabled at the same time, Y1 and Y2 will not be enabled at the same time due to that the ladder diagram program is scanned from up to down. In this ladder diagram, Y1 will be enabled first.

Example 7: Sequential control
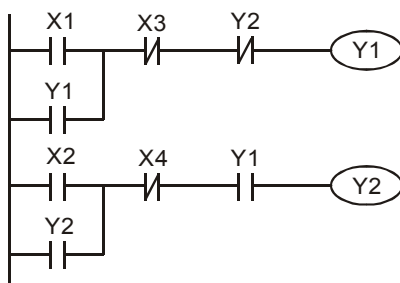
If we serially connect the normally closed contact of Y2 in example 5 to the circuit of Y1 as an AND condition for Y1 (as the diagram in the left hand side), the circuit can not only make Y1 as the condition for Y2, but also allow the stop of Y1 after Y2 is enabled. Therefore, we can make Y1 and Y2 execute exactly the sequential control.

Example 8: Oscillating circuit

An oscillating circuit with cycle $\triangle T + \triangle T$
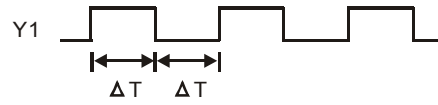
The ladder diagram above is a very simple one. When the program starts to scan the normally closed contact Y1, Y1 will be closed because coil Y1 is Off. When the program then scan to coil Y1 and make it On, the output will be 1. When the program scans to the normally closed contact Y1 again in the next scan cycle, because coil Y1 is On, Y1 will be open and make coil Y1 Off and output 0. The repeated scans will result in coil Y1 outputs oscillating pulses by the cycle $\triangle T$(On)$+\triangle T$(Off).

An oscillating circuit with cycle $nT + \triangle T$
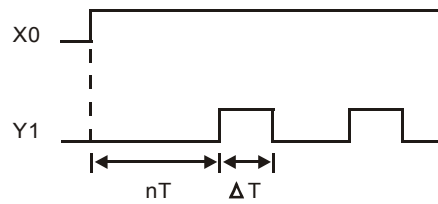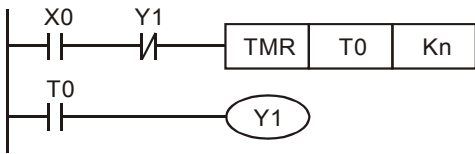
The ladder diagram program controls the On time of coil Y1 by timer T0 and disable timer T0 in the next scan cycle, resulting in the oscillating pulses in the output of Y1. n refers to the decimal set value in the timer and T is the cycle of the clock.

Example 9: Flashing circuit
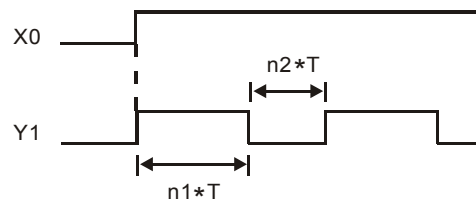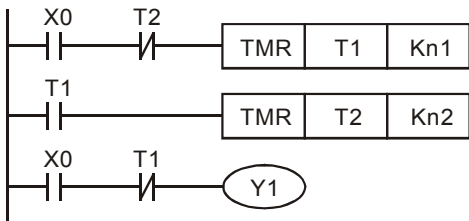
The ladder diagram is an oscillating circuit which makes the indicator flash or enables the buzzer alarms. It uses two timer to control the On/Off time of coil Y1. n1 and n2 refer to the set values in T1 and T2 and T is the cycle of the clock.

Example 10: Trigger circuit
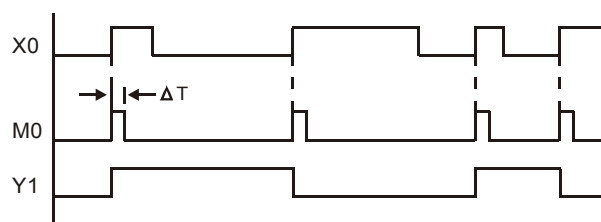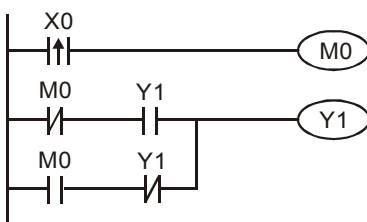
The rising-edge differential instruction of X0 makes coil M0 generate a single pulse of △T (one scan cycle). Coil Y1 will be On during this scan period. In the next scan period, coil M0 will be Off and the normally closed contact M0 and Y1 will all be closed, making coil Y1 continue to be On until another rising-edge arrives in input X0, making coil M0 On for another scan period and Y1 Off. Such kind of circuit relies on an input to make two actions execute interchangeably. Also from the timing diagram on the last page, we can see that input X0 are square pulse signals of the cycle T and coil Y1 output are square pulse signals of the cycle 2T.

Example 11: Delay circuit
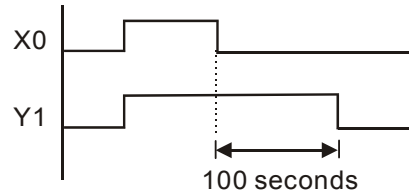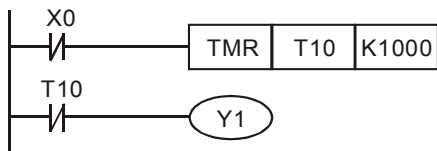
Time base: T = 0.1 sec

When input X0 is On, due to that its corresponding normally closed contact is Off, time T10 will be Off and the output coil Y1 will be On. T10 will be On and start to count until input X0 is Off. Output coil Y1 will be delayed for 100 seconds (K1,000 × 0.1 sec = 100 secs) and be Off. See the timing diagram above.

Example 12: Output delay circuit

The output delay circuit is the circuit composed of two timers. When input X0 is On and Off, output Y4 will be delayed.
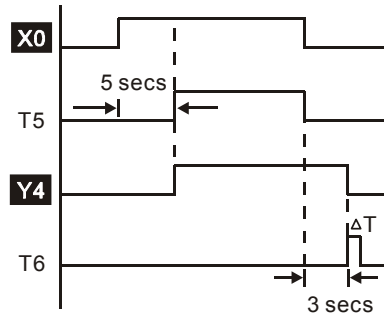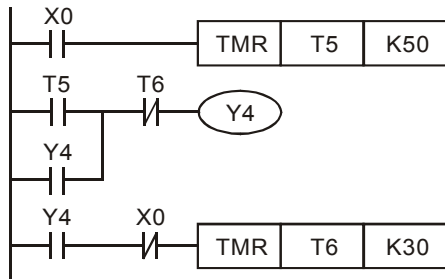
Example13: Timing extension circuit
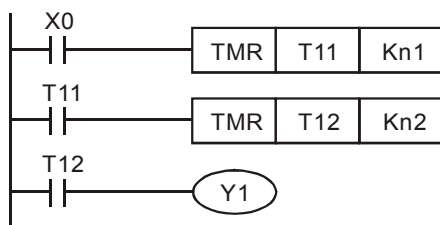
Timer = T11, T12

Clock cycle: T

The total delay time from input X0 is closed to output Y1 is On = (n1+n2)* T. T refers to the clock cycle.
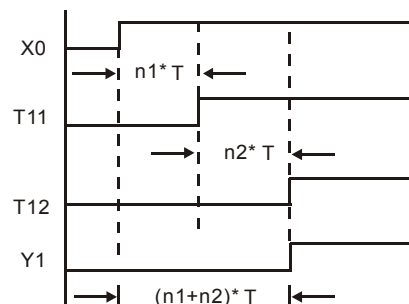
Example 14: How to enlarge the counting range
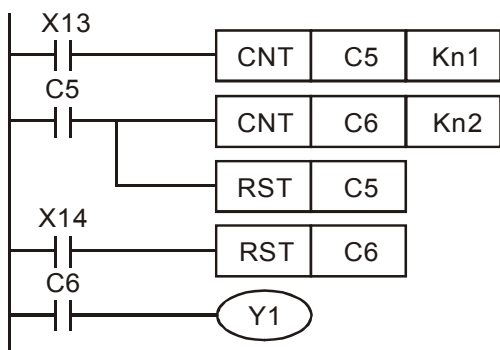
The counting range of a 16-bit counter is 0 ~ 32,767. As the circuit in the left hand side, using two counters can increase the counting range to n1*n2. When the counting of counter C5 reaches n1, C6 will start to count for one time and reset for counting the pulses from X13. When the counting of counter C6 reaches n2, the pulses from input X13 will be n1*n2.

Example 15: Traffic light control (by using step ladder instruction)
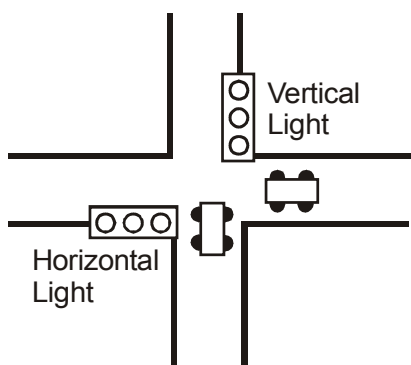
Traffic light control

	Red light	Yellow light	Green light	Green light flashes
Vertical light	Y0	Y1	Y2	Y2
Horizontal light	Y10	Y11	Y12	Y12
On time	35 secs	5 secs	25 secs	5 secs

Timing Diagram:
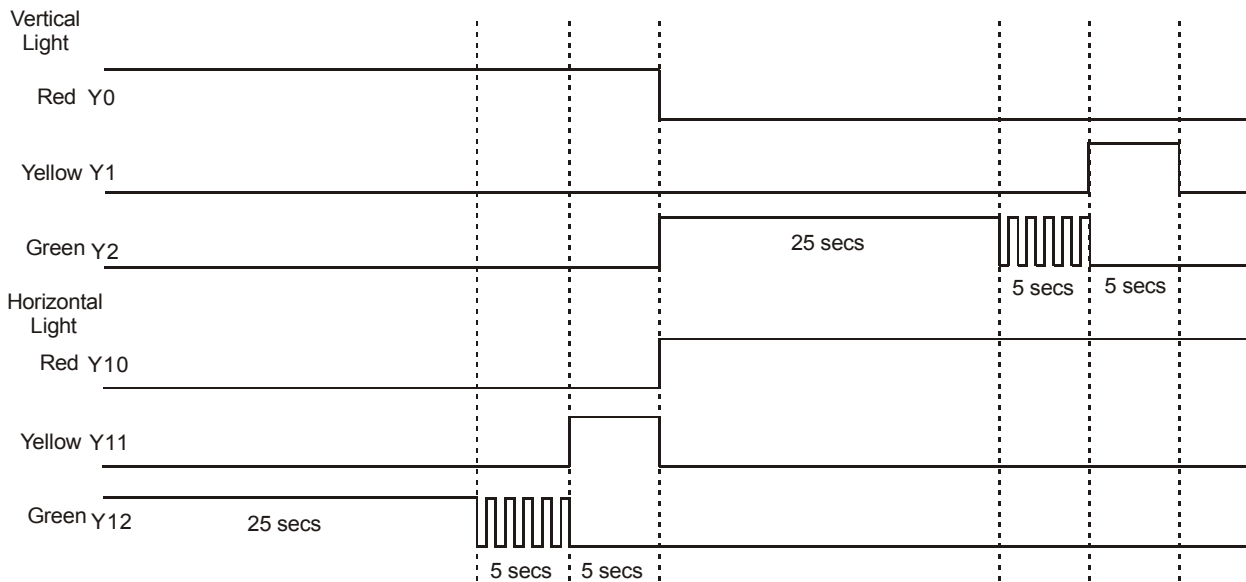
SFC Figure:
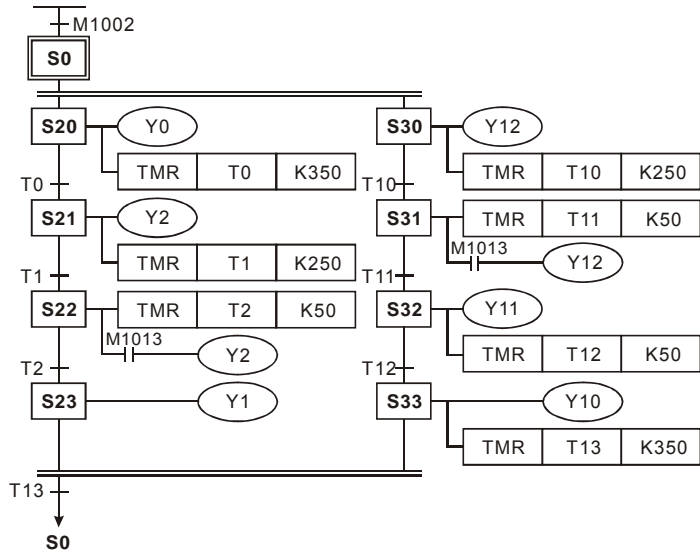
Ladder Diagram:
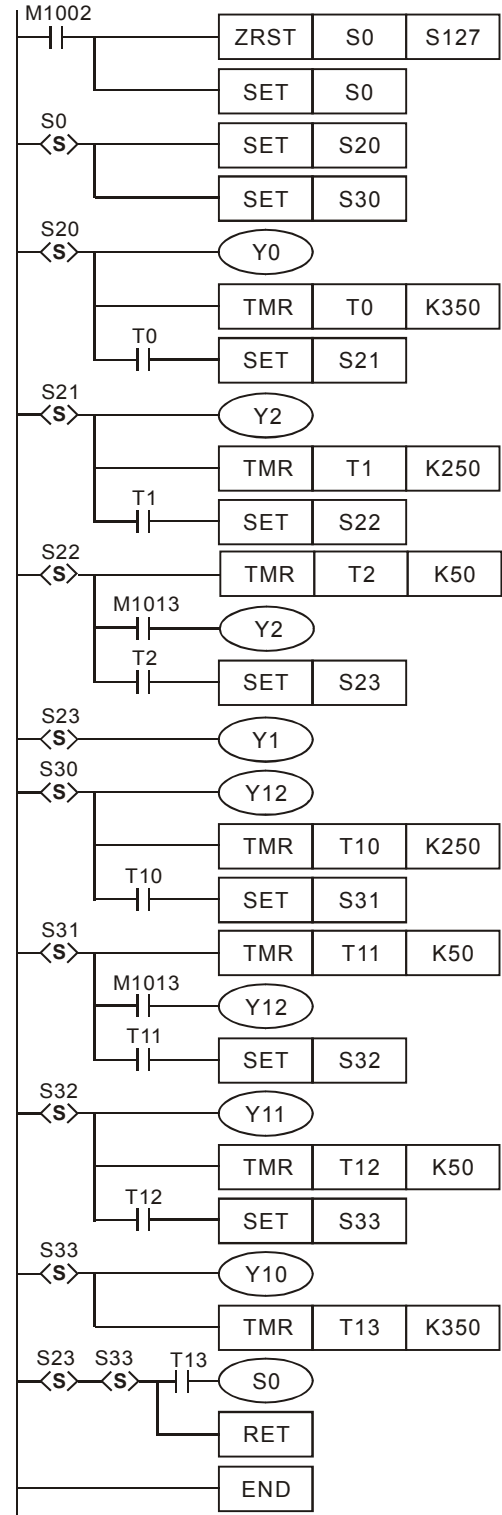
■ Drawing by SFC Editor (WPLSoft)

Drawn by SFC	Internal Ladder Diagram
	■ LAD-0 M1002 —ZRST S0 S127 / SET S0 ■ Transferring Condition 1 T0 — TRANS* ■ S22 TMR T2 K50 M1013 — Y2 ■ Transferring Condition 4 T13 — TRANS* ■ Transferring Condition 7 T12 — TRANS*
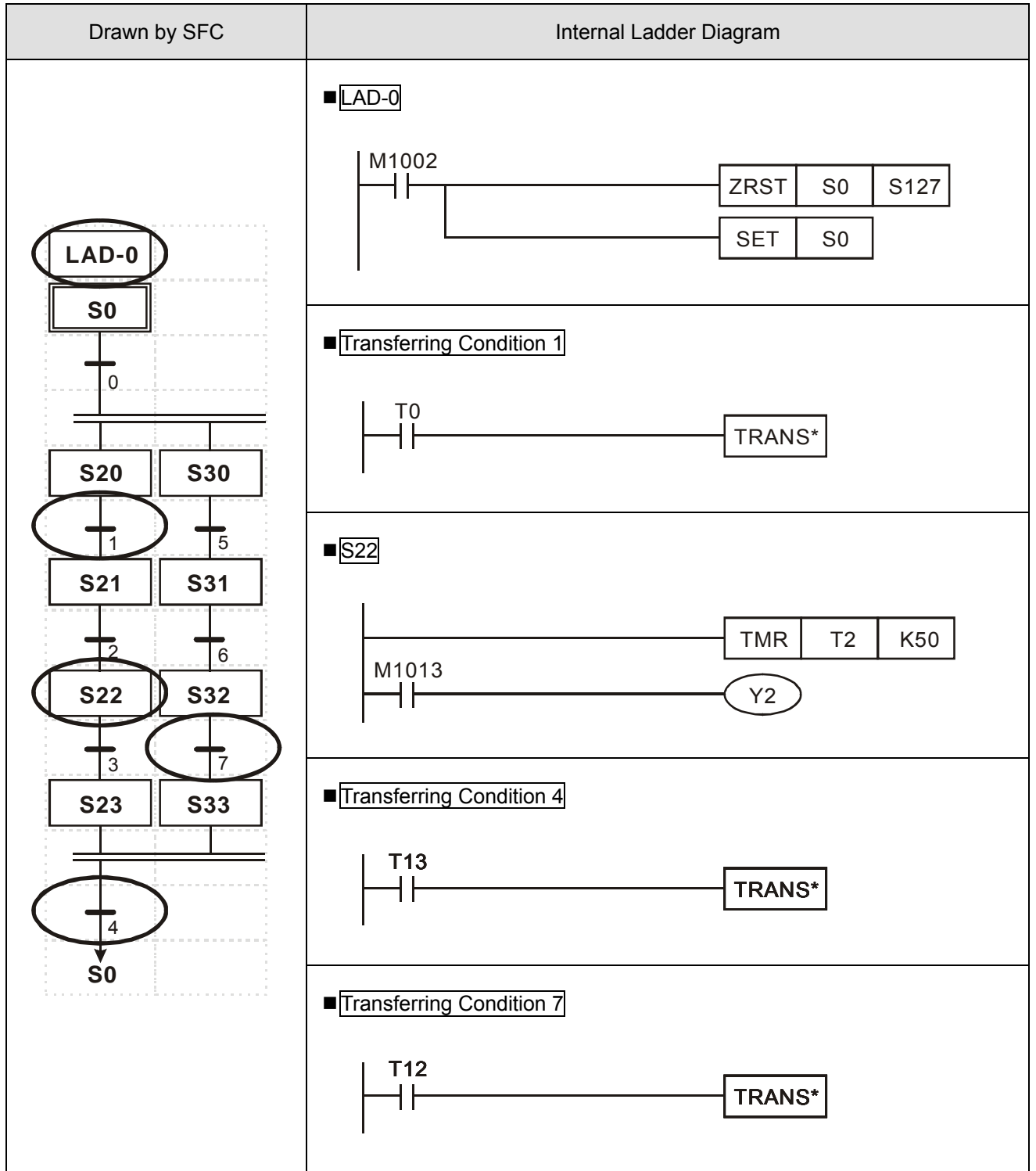
MEMO

2.1 All Devices in DVP-PLC

ES/EX/SS series MPU:

Type	Device	Item			Range		Function
Relay (bit)	X	External input relay			X0 ~ X177, 128 points, octal	Total 256 points	Corresponds to external input points
	Y	External output relay			Y0 ~ Y177, 128 points, octal		Corresponds to external output points
	M	Auxiliary relay	General purpose		M0 ~ M511, M768 ~ M999, 744 points	Total 1,280 points	The contact can be On/Off in the program.
			Latched*		M512 ~ M767, 256 points		
			Special purpose		M1000 ~ M1279, 280 points (some are latched)		
	T	Timer	100ms timer		T0 ~ T63, 64 points	Total 128 points	Timer indicated by TMR instruction. If timing reaches its target, the T contact of the same No. will be On.
			10ms timer (M1028 = On)		T64 ~ T126, 63 points (M1028 = Off: 100ms)		
			1ms timer		T127, 1 points		
	C	Counter	16-bit counting up (general purpose)		C0 ~ C111, 112 points	Total 128 points	Counter indicated by CNT (DCNT) instruction. If counting reaches its target, the C contact of the same No. will be On.
			16-bit counting up (latched*)		C112 ~ C127, 16 points		
			32-bit counting up/down high-speed counter (latched*)	1-phase 1 input	C235 ~ C238, C241, C242, C244, 7 points	Total 13 points	
				1-phase 2 inputs	C246, C247, C249, 3 points		
				2-phase 2 inputs	C251, C252, C254, 3 points		
	S	Step	Initial step (latched*)		S0 ~ S9, 10 points	Total 128 points	Used for SFC.
			Zero return (latched*)		S10 ~ S19, 10 points (used with IST instruction)		
			Latched*		S20 ~ S127, 108 points		
Register (word data)	T	Present value of timer			T0 ~ T127, 128 points		When the timing reaches the target, the contact of the timer will be On.
	C	Present value of counter			C0 ~ C127, 16-bit counter, 128 points C235 ~ C254, 32-bit counter, 13 points		When the counting reaches the target, the contact of the counter will be On.
	D	Data register	General purpose		D0 ~ D407, 408 points	Total 600 points	Memory area for data storage; E, F can be used for index indication.
			Latched*		D408 ~ D599, 192 points		
			Special purpose		D1000 ~ D1311, 312 points	Total 312 points	
			Index indication		E, F, 2 points		
Pointer	N	For master control nested loop			N0 ~ N7, 8 points		Control point for main control loop
	P	For CJ, CALL instructions			P0 ~ P63, 64 points		Position index for CJ and CALL
	I	Interruption	External interruption		I001, I101, I201, I301, 4 points		Position index for interruption subroutine.
			Timed interruption		I6□□, 1 point (□□＝10 ~ 99, time base = 1ms) (for V5.7 and above)		
			Communication interruption		I150, 1 point		
Constant	K	Decimal form			K-32,768 ~ K32,767 (16-bit operation) K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)		
	H	Hexadecimal form			H0000 ~ HFFFF (16-bit operation) H00000000 ~ HFFFFFFFF (32-bit operation)		

* The latched area is fixed and cannot be changed.

SA/SX/SC series MPU:

Type	Device	Item		Range		Function
Relay (bit)	X	External input relay		X0 ~ X177, 128 points, octal	Total 256 points	Corresponds to external input points
	Y	External output relay		Y0 ~ Y177, 128 points, octal		Corresponds to external output points
	M	Auxiliary Relay	General purpose	M0 ~ M511, 512 points (*1)	Total 4,096 points	The contact can be On/Off in the program.
			Latched*	M512 ~ M999, 488 points (*3) M2000 ~ M4095, 2,096 points (*3)		
			Special purpose	M1000 ~ M1999, 1,000 points (some are latched)		
	T	Timer	100ms	T0 ~ T199, 200 points (*1) T192 ~ T199 for subroutine T250 ~ T255, 6 accumulative points (*4)	Total 256 points	Timer indicated by TMR instruction. If timing reaches its target, the T contact of the same No. will be On.
			10ms	T200 ~ T239, 40 points (*1) T240 ~ T245, 6 accumulative points (*4)		
			1ms	T246 ~ T249, 4 accumulative points (*4)		
	C	Counter	16-bit counting up	C0 ~ C95, 96 points (*1) C96 ~ C199, 104 points (*3)	Total 235 points	Counter indicated by CNT (DCNT) instruction. If counting reaches its target, the C contact of the same No. will be On.
			32-bit counting up/down	C200 ~ C215, 16 points (*1) C216 ~ C234, 19 points (*3)		
			For SA/SX, 32-bit high-speed counter	C235 ~ C244, 1-phase 1 input, 9 points (*3) C246 ~ C249, 1-phase 2 inputs, 3 points (*3) C251 ~ C254, 2-phase 2 inputs, 4 points (*3)	Total 16 points	
			For SC, 32-bit high-speed counter	C235 ~ C245, 1-phase 1 input, 11 points (*3) C246 ~ C250, 1-phase 2 inputs, 4 points (*3) C251 ~ C255, 2-phase 2 inputs, 4 points (*3)	Total 19 points	
	S	Step point	Initial step	S0 ~ S9, 10 points (*1)	Total 1,024 points	Used for SFC.
			Zero return	S10 ~ S19, 10 points (used with IST instruction) (*1)		
			General purpose	S20 ~ S511, 492 points (*1)		
			Latched*	S512 ~ S895, 384 points (*3)		
			Alarm	S896 ~ S1023, 128 points (*3)		
Register (word data)	T	Present value of timer		T0 ~ T255, 256 points		When the timing reaches the target, the contact of the timer will be On.
	C	Present value of counter		C0 ~ C199, 16-bit counter, 200 points C200 ~ C254, 32-bit counter, 50 points (SC: 53 points)		When the counting reaches the target, the contact of the counter will be On.
	D	Data register	General purpose	D0 ~ D199, 200 points (*1)	Total 5,000 points	Memory area for data storage; E, F can be used for index indication.
			Latched*	D200 ~ D999, 800 points (*3) D2000 ~ D4999, 3,000 points (*3)		
			Special purpose	D1000 ~ D1999, 1,000 points		
			Index indication	E0 ~ E3, F0 ~ F3, 8 points (*1)		
	N/A	File register		K0 ~ K1,599 (1,600 points) (*4)		Expanded register for data storage.

Pointer	N	For Master control loop	N0 ~ N7, 8 points	Control point for main control loop
	P	For CJ, CALL instructions	P0 ~ P255, 256 points	Position index for CJ and CALL
	I	Interruption — External interruption	I001, I101, I201, I301, I401, I501, total 6 points	Position index for interruption subroutine.
		Interruption — Timed interruption	I6□□, I7□□, 2 points (□□ = 1 ~ 99, time base = 1ms)	
		Interruption — Interruption inserted when high-speed counter reaches target	I010, I020, I030, I040, I050, I060, total 6 points	
		Interruption — Communication interruption	I150, 1 point	
Constant	K	Decimal form	K-32,768 ~ K32,767 (16-bit operation) K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)	
	H	Hexadecimal form	H0000 ~ HFFFF (16-bit operation) H00000000 ~ HFFFFFFFF (32-bit operation)	

*1. Non-latched area cannot be modified.

*2. The preset non-latched area can be modified into latched area by setting up parameters.

*3. The preset latched area can be modified into non-latched area by setting up parameters.

*4. The fixed latched area cannot be modified

Latched settings for all devices in SA/SX/SC series MPU:

	General purpose	Latched	Special auxiliary relay	Latched
M (Auxiliary relay)	M0 ~ M511	M512 ~ M999	M1000 ~ M1999	M2000 ~ M4095
	It is fixed to be non-latched	Default: latched Start: D1200 (K512) End: D1201 (K999)	Some are latched and cannot be modified	Default: latched Start: D1202 (K2,000) End: D1203 (K4,095)

	100 ms	10 ms	10 ms	1 ms	100 ms
T (Timer)	T0 ~ T199	T200 ~ T239	T240 ~ T245	T246 ~ T249	T250 ~ T255
	It is fixed to be non-latched	It is fixed to be non-latched	Accumulative type It is fixed to be latched		

	16-bit counting up		32-bit counting up/down		32-bit high-speed counting up/down
C (Counter)	C0 ~ C95	C96 ~ C199	C200 ~ C215	C216 ~ C234	C235 ~ C255
	It is fixed to be non-latched	Default: latched Start: D1208 (K96) End: D1209 (K199)	It is fixed to be non-latched	Default: latched Start: D1210 (K216) End: D1211 (K234)	Default: latched Start: D1212 (K235) End: D1213 (K255)

	Initial	Zero return	General purpose	Latched	Alarm step
S (Step relay)	S0 ~ S9	S10 ~ S19	S20 ~ S511	S512 ~ S895	S896 ~ S1023
	It is fixed to be non-latched			Default: latched Start: D1214 (K512) End: D1215 (K895)	It is fixed to be latched

	General purpose	Latched	Special register	Latched
D (Register)	D0 ~ D199	D200 ~ D999	D1000 ~ D1999	D2000 ~ D4999
	It is fixed to be non-latched	Default: latched Start: D1216 (K200) End: D1217 (K999)	Some are latched and cannot be modified.	Default: latched Start: D1218 (K2,000) End: D1219 (K4,999)

File Register	K0 ~ K1599
	It is fixed to be latched.

EH/EH2/SV series MPU:

Type	Device	Item		Range		Function
Relay (bit)	X	External input relay		X0 ~ X377, 256 points, octal	Total 512 points	Corresponds to external input points
	Y	External output relay		Y0 ~ Y377, 256 points, octal		Corresponds to external output points
	M	Auxiliary relay	General purpose	M0 ~ M499, 500 points (*2)	Total 4,096 points	The contact can be On/Off in the program.
			Latched	M500 ~ M999, 500 points (*3) M2000 ~ M4095, 2,096 points (*3)		
			Special purpose	M1000 ~ M1999, 1,000 points (some are latched)		
	T	Timer	100ms	T0 ~ T199, 200 points (*2) T192 ~ T199 is for subroutine T250~T255, 6 accumulative points (*4)	Total 256 points	Timer indicated by TMR instruction. If timing reaches its target, the T contact of the same No. will be On.
			10ms	T200 ~ T239, 40 points (*2) T240 ~ T245, 6 accumulative points (*4)		
			1ms	T246 ~ T249, 4 accumulative points (*4)		
	C	Counter	16-bit counting up	C0 ~ C99, 100 points (*2) C100 ~ C199, 100 points (*3)	Total 253 points	Counter indicated by CNT (DCNT) instruction. If counting reaches its target, the C contact of the same No. will be On.
			32-bit counting up/down	C200 ~ C219, 20 points (*2) C220 ~ C234, 15 points (*3)		
			32-bit high-speed counter	C235 ~ C244, 1-phase 1 input, 10 points (*3) C246 ~ C249, 1-phase 2 inputs, 4 points(*3) C251 ~ C254, 2-phases 2 inputs, 4 points (*3)		
	S	Step	Initial step point	S0 ~ S9, 10 points (*2)	Total 1,024 points	Used for SFC.
			Zero return	S10 ~ S19, 10 points (used with IST instruction) (*2)		
			General purpose	S20 ~ S499, 480 points (*2)		
			Latched	S500 ~ S899, 400 points (*3)		
			Alarm	S900 ~ S1023, 124 points (*3)		
Register (word data)	T	Present value of timer		T0 ~ T255, 256 points		When the timing reaches the target, the contact of the timer will be On.
	C	Present value of counter		C0 ~ C199, 16-bit counter, 200 points C200 ~ C254, 32-bit counter, 53 points		When the counting reaches the target, the contact of the counter will be On.
	D	Data register	General purpose	D0 ~ D199, 200 points, (*2)	Total 10,000 points	Memory area for data storage; E, F can be used for index indication.
			Latched	D200 ~ D999, 800 points (*3) D2000 ~ D9999, 8,000 points (*3)		
			Special purpose	D1000 ~ D1999, 1,000 points		
			Index indication	E0 ~ E7, F0 ~ F7, 16 points (*1)		
	N/A	File register		K0 ~ K9,999 (10,000 points) (*4)		Expanded register for data storage.
Pointer	N	For master control loop		N0 ~ N7, 8 points		Control point for main control loop
	P	For CJ, CALL instructions		P0~P255, 256 points		Position index for CJ and CALL
	I	Interruption	External interruption	I00□(X0), I10□(X1), I20□(X2), I30□(X3), I40□(X4), I50□(X5), 6 points (□ = 1, rising-edge trigger ⌐, □ = 0, falling-edge trigger ⌐)		Position index for interruption subroutine.
			Timed interruption	I6□□, I7□□, 2 points(□□ = 1~99ms) time base = 1ms I8□□, 1 point (□□ = 1~99, time base = 0.1ms)		
			Interruption inserted when high-speed counter reaches target	I010, I020, I030, I040, I050, I060, 6 points		
			Pulse interruption	I110, I120, I130, I140, 4 points		
			Communication interruption	I150, I160, I170, 3 points		

Type	Device	Item	Range	Function
		Frequency measurement card interruption	I180, 1 point	
Constant	K	Decimal form	K-32,768 ~ K32,767 (16-bit operation) K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)	
	H	Hexadecimal form	H0000 ~ HFFFF (16-bit operation) H00000000 ~ HFFFFFFFF (32-bit operation)	

*1. Non-latched area cannot be modified.

*2. The preset non-latched area can be modified into latched area by setting up parameters.

*3. The preset latched area can be modified into non-latched area by setting up parameters.

*4. The fixed latched area cannot be modified

Latched settings for all devices in EH/EH2/SV series MPU:

M (Auxiliary relay)	General purpose		Latched		Special auxiliary relay	Latched

M (Auxiliary relay)	General purpose	Latched	Special auxiliary relay	Latched
	M0 ~ M499	M500 ~ M999	M1000 ~ M1999	M2000 ~ M4095
	Start: D1200 (K500) End: D1201 (K999)		Some are latched and cannot be modified.	Start: D1202 (K2,000) End: D1203 (K4,095)

T (Timer)	100 ms	10 ms	10 ms	1 ms	100 ms
	T0 ~ T199	T200 ~ T239	T240 ~ T245	T246 ~ T249	T250 ~ T255
	Default: non-latched	Default: non-latched	Accumulative type It is fixed to be latched.		
	Start: D1204 (K-1)*1 End: D1205 (K-1)*1	Start: D1206 (K-1)*1 End: D1207 (K-1)*1			

C (Counter)	16-bit counting up		32-bit counting up/down		32-bit high-speed counting up/down	
	C0 ~ C99	C100 ~ C199	C200 ~ C219	C220 ~ C234	C235 ~ C245	C246 ~ C255
	Default: non-latched	Default: latched	Default: non-latched	Default: latched	Default: latched	
	Start: D1208 (K100) End: D1209 (K199)		Start: D1210 (K220) End: D1211 (K234)		Start: D1212 (K235) End: D1213 (K255)	

S (Step relay)	Initial	Zero return	General purpose	Latched	Step alarm
	S0 ~ S9	S10 ~ S19	S20 ~ S499	S500 ~ S899	S900 ~ S1023
	Non-latched (default)			Latched (default)	It is fixed to be latched.
	Start: D1214 (K500) End: D1215 (K899)				

D (Register)	General purpose	Latched	Special register	Latched
	D0 ~ D199	D200 ~ D999	D1000 ~ D1999	D2000 ~ D9999
	Default: non-latched	Default: latched	Some is latched and cannot be modified.	Default: latched
	Start: D1216 (K200) End: D1217 (K999)			Start: D1218 (K2,000) End: D1219 (K9,999)
File register	K0 ~ K9,999			
	It is fixed to be latched.			

*1: K-1 refers to the default setting is non-latched.

■ Power On/Off or the MPU switches between RUN/STOP:

Memory of ES/EX/SS V5.5 (and above)

Memory type	Power Off→On	STOP→RUN	RUN→STOP	Clear all non-latched areas (M1031)	Clear all latched areas (M1032)	Default setting
Non-latched	Clear	Clear when M1033 = Off		Clear	Unchanged	0
		Remain unchanged when M1033 = On				
Latched	Unchanged			Unchanged	Clear	Unchanged
Special M, Special D, index register	Initial	Unchanged		Unchanged		Initial setting

Memory of SA/SX/SC/EH/EH2/SV series MPU:

Memory type	Power Off→On	STOP→RUN	RUN→STOP		Clear all non-latched area (M1031)	Clear all latched area (M1032)	Default setting
Non-latched	Clear	Unchanged	Clear when M1033 = Off		Clear	Unchanged	0
			Remain unchanged when M1033 = On				
Latched	Unchanged				Unchanged	Clear	0
Special M, Special D, index register	Initial	Unchanged			Unchanged		Initial setting
File Register	Unchanged						0

2.2 Values, Constants [K] / [H]

Constant	K	Decimal form	K-32,768 ~ K32,767 (16-bit operation) K-2,147,483,648 ~ K2,147,483,647 (32-bit operation)
	H	Hexadecimal form	H0 ~ HFFFF (16-bit operation) H0 ~ HFFFFFFFF (32-bit operation)

For different control purposes, there are five types of values inside DVP-PLC for executing the operations. See the explanations bellows for the functions and works of every type of value.

1. Binary value (BIN)

All the operations and storage of values in PLC are conducted in BIN. Belows are the terms for BIN values.

Bit:	The basic unit for a BIN value, either 1 or 0.
Nibble:	Composed of 4 continuous bits (e.g. b3 ~ b0). Presented as the decimal value 0 ~ 9 of a digit or 0 ~ F in hex.
Byte:	Composed of 2 continuous nibble (i.e. 8 bits, b7 ~ b0). Presented as 00 ~ FF in hex.
Word:	Composed of 2 continuous bytes (i.e. 16 bits, b15 ~ b0). Presented as 4-digit 0000 ~ FFFF in hex.
Double word:	Composed of 2 continuous words (i.e. 32 bits, b31 ~ b0). Presented as 8 digit 00000000 ~ FFFFFFFF.

Bit, nibble, byte, word, and double word in a binary system:
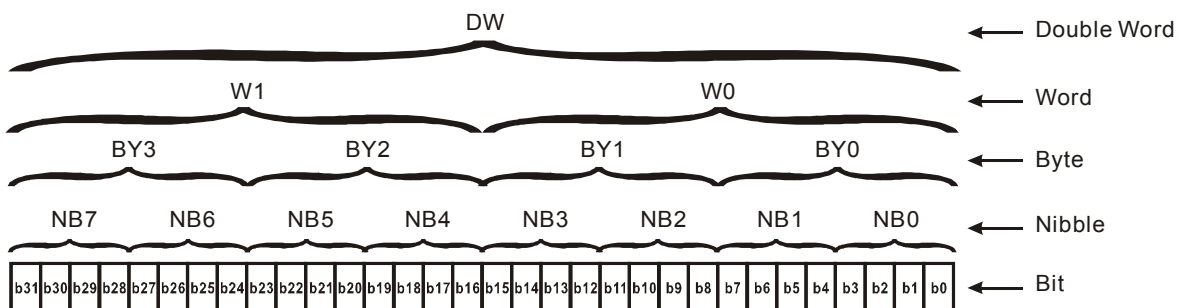
2. Octal value (OCT)

The No. of external input and output terminals in DVP-PLC is numbered in octal system.

For example:

External input: X0 ~ X7, X10 ~ X17…(device No.)

External output: Y0 ~ Y7, Y10 ~ Y17…(device No.)

3. Decimal value (DEC)

Occassions of using decimal values in DVP-PLC:

- Set value in timer T and counter C, e.g. TMR C0 K50 (constant K)

- No. of device S, M, T, C, D, E, F, P, I, e.g. M10, T30. (device No.)

- Operands in application instructions, e.g. MOV K123 D0 (constant K)

4. Binary code decimal (BCD)

A decimal datum is presented by a nibble or 4 bits. Therefore, a continuous 16 bits can be presented as a 4-digit decimal value. BCD is mainly used on reading the input value from the DIP switch or the data output to a 7-section display.

5. Hexadecimal value (HEX)

Occassion of using hexadecimal values:

- Operands in application instructions, e.g. MOV H1A2B D0 (constant H)

Constant K:

"K" is normally placed before a decimal value in the PLC. For example, K100 refers to a decimal value, 100.

Exception:

K and bit devices X, Y, M and S can combine into data in bit, byte, word or double word, e.g. K2Y10, K4M100.

Here K1 refers to a 4-bit data and K2 ~ K4 refer to 8-bit, 12-bit and 16-bit data.

Constant H:

"H" is normally placed before a hexadecimal value in the PLC. For example, H100 refers to a hexadecimal value, 100.

Reference table:

Binary (BIN)		Octal (OCT)	Decimal (DEC)	Binary Code Decimal (BCD)		Hexadecimal (HEX)
For PLC internal operation		No. of device X, Y	Constant K, No. of device M, S, T, C, D, E, F, P, I	For DIP switch and 7-section display		Constant H
0 0 0 0	0 0 0 0	0	0	0 0 0 0	0 0 0 0	0
0 0 0 0	0 0 0 1	1	1	0 0 0 0	0 0 0 1	1
0 0 0 0	0 0 1 0	2	2	0 0 0 0	0 0 1 0	2
0 0 0 0	0 0 1 1	3	3	0 0 0 0	0 0 1 1	3
0 0 0 0	0 1 0 0	4	4	0 0 0 0	0 1 0 0	4
0 0 0 0	0 1 0 1	5	5	0 0 0 0	0 1 0 1	5
0 0 0 0	0 1 1 0	6	6	0 0 0 0	0 1 1 0	6
0 0 0 0	0 1 1 1	7	7	0 0 0 0	0 1 1 1	7
0 0 0 0	1 0 0 0	10	8	0 0 0 0	1 0 0 0	8
0 0 0 0	1 0 0 1	11	9	0 0 0 0	1 0 0 1	9
0 0 0 0	1 0 1 0	12	10	0 0 0 1	0 0 0 0	A
0 0 0 0	1 0 1 1	13	11	0 0 0 1	0 0 0 1	B
0 0 0 0	1 1 0 0	14	12	0 0 0 1	0 0 1 0	C
0 0 0 0	1 1 0 1	15	13	0 0 0 1	0 0 1 1	D
0 0 0 0	1 1 1 0	16	14	0 0 0 1	0 1 0 0	E
0 0 0 0	1 1 1 1	17	15	0 0 0 1	0 1 0 1	F
0 0 0 1	0 0 0 0	20	16	0 0 0 1	0 1 1 0	10

Binary (BIN)		Octal (OCT)	Decimal (DEC)	Binary Code Decimal (BCD)		Hexadecimal (HEX)
For PLC internal operation		No. of device X, Y	Constant K, No. of device M, S, T, C, D, E, F, P, I	For DIP switch and 7-section display		Constant H
0 0 0 1	0 0 0 1	21	17	0 0 0 1	0 1 1 1	11
⋮	⋮	⋮	⋮	⋮	⋮	⋮
0 1 1 0	0 0 1 1	143	99	1 0 0 1	1 0 0 1	63

2.3   Numbering and Functions of External Input/Output Contacts [X] / [Y]

No. of input/output contacts (in octal):

The No. of input and output contacts on the PLC MPU starts from X0 and Y0. The range of the No. varies upon the number of points on the MPU. For I/O extension units, the No. of input and output contacts is calculated according to its connection sequence with the MPU.

■ **ES/EX/SS series MPU:**

Model	DVP-14ES	DVP-14SS	DVP-20EX	DVP-24ES	DVP-32ES	DVP-40ES	DVP-60ES	I/O Extension Unit
Input X	X0 ~ X7 (8 points)	X0 ~ X7 (8 points)	X0 ~ X7 (8 points)	X0 ~ X17 (16 points)	X0 ~ X17 (16 points)	X0 ~ X27 (24 points)	X0 ~ X43 (36 points)	X20/30/50 ~ X177 (Note)
Output Y	Y0 ~ Y5 (6 points)	Y0 ~ Y5 (6 points)	Y0 ~ Y5 (6 points)	Y0 ~ Y7 (8 points)	Y0 ~ Y17 (16 points)	Y0 ~ Y17 (16 points)	Y0 ~ Y27 (24 points)	Y20/30 ~ Y177 (Note)

Note: The input points on I/O extension units start from X20 and output points from Y20, except input points on DVP-40ES start from X30 and output from Y20; input points on DVP-60ES start from X50 and output from Y30. The No. of input/output points on the I/O extension units increases by 8's multiple. If the number of points is less than 8, it will be counted as 8.

■ **SA/SX/SC serues MPU:**

Model	DVP-10SX (Note1)	DVP-12SA	DVP-12SC	I/O Extension Unit (Note 2)
Input X	X0 ~ X3 (4 points)	X0 ~ X7 (8 points)	X0 ~ X5, X10 ~ X11 (8 points)	X20 ~ X177
Output Y	Y0 ~ Y1 (2 points)	Y0 ~ Y3 (4 points)	Y0 ~ Y1, Y10 ~ Y11 (4 points)	Y20 ~ Y177

Note 1: Besides 4DI and 2DO, SX series MPU has also 2AI (12-bit) and 2AO (12-bit) of analog input/output.
Note 2: SX/SA/SC series MPU share the extension units with SS series MPU. The input points on I/O extension units start from X20 and output points start from Y20. The calculation on the No. of I/O points is the same as that in SS series.

■ **EH series MPU:**

Model	DVP-16EH	DVP-20EH (Note 1)	DVP-32EH (Note1, 2)	DVP-40EH	DVP-48EH	DVP-64EH	DVP-80EH	I/O Extension Unit (Note 3)
Input X	X0 ~ X7 (8 points)	X0 ~ X13 (12 points)	X0 ~ X17 (16 points)	X0 ~ X27 (24 points)	X0 ~ X27 (24 points)	X0 ~ X37 (32 points)	X0 ~ X47 (40 points)	X※~X377
Output Y	Y0 ~ Y7 (8 points)	Y0 ~ Y7 (8 points)	Y0 ~ Y17 (16 points)	Y0 ~ Y17 (16 points)	Y0 ~ Y27 (24 points)	Y0 ~ Y37 (32 points)	Y0 ~ Y47 (40 points)	Y※~Y377

Note 1: The output type of 20EH00T and 32EH00T is transistor, among which Y0 and Y2 are high-speed transistor output (200KHz) and other outputs are normal transistor output (10KHz). The output type of other MPUs with 16/48/64/80 points is transistor and all outputs are normal transistor output (10KHz).
Note 2: The terminal layouts of 32EH00T, 32EH00R and 32EH00M are different. See the instruction sheets of EH series MPU. In 32EH00M, CH0 (Y0, Y1) and CH1 (Y2, Y3) are high-speed differential output.
Note 3: The start No. of the input and output points on the I/O extension unit resumes from the last No. in the MPU. The start No. of input points on the I/O extension unit of DVP-16EH and DVP-20EH start from X20 and output points start from Y20. The No. on the I/O extension unit are in sequence, with max. input point No. X377 and max. output point No. Y377.

■ **EH2 series MPU:**

Model	DVP-16EH2	DVP-20EH2 (Note 1)	DVP-32EH2 (Note1)	DVP-40EH2 (Note 2)	DVP-48EH2	DVP-64EH2	DVP-80EH2	I/O Extension Unit (Note 3)
Input X	X0 ~ X7 (8 points)	X0 ~ X13 (12 points)	X0 ~ X17 (16 points)	X0 ~ X27 (24 points)	X0 ~ X27 (24 points)	X0 ~ X37 (32 points)	X0 ~ X47 (40 points)	X※~X377
Output Y	Y0 ~ Y7 (8 points)	Y0 ~ Y7 (8 points)	Y0 ~ Y17 (16 points)	Y0 ~ Y17 (16 points)	Y0 ~ Y27 (24 points)	Y0 ~ Y37 (32 points)	Y0 ~ Y47 (40 points)	Y※~Y377

Note 1: The output type of 20EH00T2 and 32EH00T2 is transistor, among which Y0 and Y2 are high-speed transistor output (200KHz) and other outputs are normal transistor output (10KHz). The output type of other MPUs with 16/48/64/80 points is transistor and all outputs are normal transistor output (10KHz).

Note 2: The output type of 40EH00T2 is transistor, among which CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4) and CH3 (Y6) are high-speed transistor output (200KHz). The output type of other output points is normal transistor output (10KHz).

Note 3: The numbering of the I/O points on EH2 series MPU is the same as that of EH series MPU.

■ **SV series MPU:**

Model	DVP-28SV (Note 1)	I/O Extension Unit (Note 2)
Input X	X0 ~ X17 (16 points)	X20 ~ X377
Output Y	Y0 ~ Y13 (12 points)	Y20 ~ Y377

Note 1: The output type of 28SV11T is transistor output, among which CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4) and CH3 (Y6) are high-speed transistor output (200KHz); others are normal transistor output (10KHz).

Note 2: The input points on I/O extension units start from X20 and output points start from Y20. The calculation on the No. of I/O points is the same as that in SS series.

■ **Input Relay X0 ~ X377**

The numbering of input relays (or input terminals) is in octal form. EH series MPU can have up to 256 points and the range is: X0 ~ X7, X10 ~ X17, …, X370 ~ X377.

■ **Output Relay Y0 ~ Y377**

The numbering of output relays (or output terminals) is also in octal form. EH series MPU can have up to 256 points and the range is: Y0 ~ Y7, Y10 ~ Y17, …, Y370 ~ Y377.

■ **Functions of input contact X**

The input contact X is connected to the input device and reads the input signals into the PLC. There is no limitation on the times of using contact A or B of input contact X in the program. On/Off of the input contact X only changes with On/Off of the input device. You cannot use the peripheral devices (HPP or WPLSoft) to force On/Off of the input contact X.

The special relay M1304 in SA/SX/SC/EH series MPU allows the peripheral devices HPP or WPLSoft to force On/Off of the input contact X, but the PLC will not be able to receive external input signals at this time.

■ **Functions of output contact Y**

Output contact Y sends out On/Off signals to drive the load connected to output contact Y. There are two types of output contacts, relay and transistor. Thereis no limitation on the times of using contact A or B of output contact Y in the program, but the No. of output coil Y can only be used once in the program; otherwise according to the scan principle of the PLC program, the output status will be determined by the circuit of the last output Y in the program.
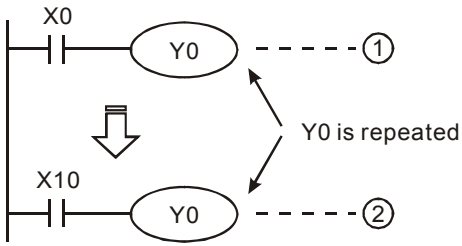
The output of Y0 will be determined by circuit ②, i.e. On/Off of X10 will determine the output status of Y0.

The Handling Process of PLC Program (Batch I/O)
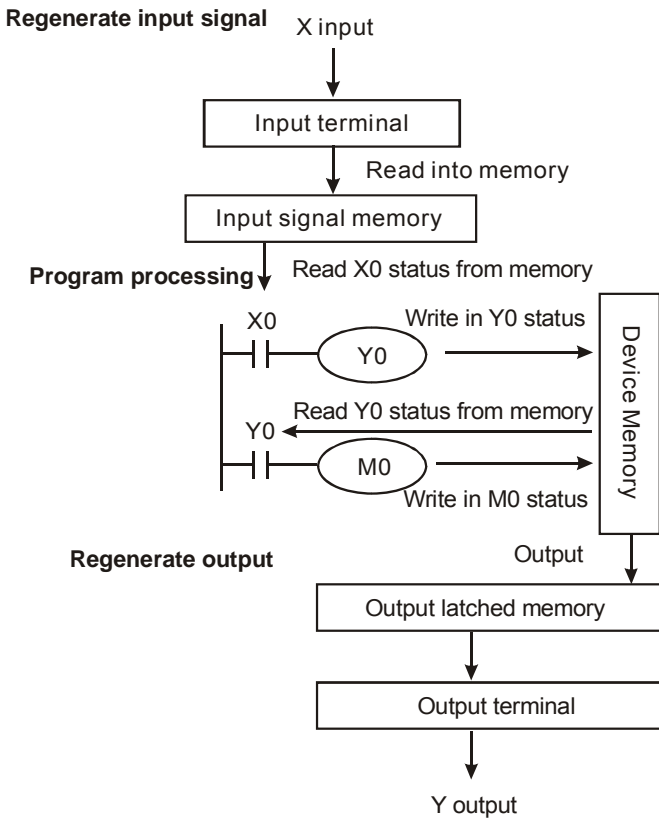
- **Regenerate input signal**
  1. Before the execution of the program, PLC reads the On/Off status of the external input signals into the input signal memory at a time.
  2. The On/Off status of the input signal during the execution of the program will not change the signal status in the input signal memory. The new On/Off status will be read in in the next scan.
  3. There will be approximately a 10ms delay from the On→Off or Off→On changes to the status being recognized by the contact in the program. The delay time may be affected by the scan time in the program.

- **Program processing**

  After the PLC reads the On/Off status of every input signal in the input signal memory, it will start to execute every instruction in the program in order starting from address 0. The execution result (On/Off of every output coil) will be stored in order into the device memory.

- **Regenerate output**
  1. When the program executes to END instruction, it will send the On/Off status of Y in the device memory to the output latched memory. The output latched memory is the coil of the output relay.
  2. There will be a 10ms delay from On→Off or Off→On of the relay coil to the On/Off status of the contact.
  3. There will be a 10 ~ 20us delay from On→Off or Off→On of the transistor module to the On/Off status of the contact.

2.4   Numbering and Functions of Auxiliary Relays [M]

No. of auxiliary relays (in decimal)

■ **ES/EX/SS series MPU:**

Auxiliary relay M	General purpose	M0 ~ M511, M768 ~ M999, 744 points. Fixed to be non-latched.	Total 1,280 points
	Latched	M512 ~ M767, 256 points. Fixed to be latched.	
	Special purpose	M1000 ~ M1279, 280 points. Some are latched.	

■ **SA/SX/SC series MPU:**

Auxiliary relay M	General purpose	M0 ~ M511, 512 points. Fixed to be non-latched.	Total 4,096 points
	Latched	M512 ~ M999, M2000 ~ M4095, 2,584 points. Can be modified to be non-latched by setting up parameters.	
	Special purpose	M1000 ~ M1999, 1000 points. Some are latched.	

■ **EH/EH2/SV series MPU:**

Auxiliary relay M	General purpose	M0 ~ M499, 500 points. Can be modified to be latched by setting up parameters.	Total 4,096 points
	Latched	M500 ~ M999, M2000 ~ M4095, 2,596 points. Can be modified to be non-latched by setting up parameters.	
	Special purpose	M1000 ~ M1999, 1,000 points. Some are latched.	

Functions of auxiliary relays:

Both auxiliary relay M and output relay Y have output coils and contact A, B, and there is no limitation on the times of using the contact. You can use auxiliary relay M to assemble a control loop, but it cannot directly drive the external load. There are three types of auxiliary relays:

1. **General purpose auxiliary relay:** If the relay encouters power cut during the operation of the PLC, its status will be reset to Off and stay Off when the power is on again.

2. **Latched auxiliary relay:** If the relay encounters power cut during the operation of the PLC, its status will be retained and stay at the status before the power cut when the power is on again.

3. **Special purpose auxiliary relay:** Every relay of this kind has its specific function. Do not use undefined special purpose auxiliary relay. See 2.10 for special puspose auxiliary relay of all series MPU and 2.11 for its functions.

2.5   Numbering and Functions of Step Relays [S]

No. of step relays (in decimal)

■ **ES/EX/SS series MPU:**

Step relay S	Initial latched	S0 ~ S9, 10 points. Fixed to be latched.	Total 128 points
	Zero return latched	S10 ~ S19, 10 points, used with IST instruction. Fixed to be latched.	
	Latched	S20 ~ S127, 108 points. Fixed to be latched.	

■ **SA/SX/SC series MPU:**

Step relay S	Initial	S0 ~ S9, 10 points. Fixed to be non-latched.	Total 1,024 points
	Zero return	S10 ~ S19, 10 points, used with IST instruction. Fixed to be non-latched.	
	General purpose	S20 ~ S511, 492 points. Fixed to be non-latched.	
	Latched	S512 ~ S895, 384 points. Can be modified to be non-latched by setting up parameters.	
	Alarm	S896 ~ S1023, 128 points. Fixed to be latched.	

■ **EH/EH2/SV series MPU:**

Step relay S	Initial	S0 ~ S9, 10 points. Can be modified to be latched by setting up parameters.	Total 1,024 points
	Zero return	S10 ~ S19, 10 points, used with IST instruction. Can be modified to be latched by setting up parameters.	
	General purpose	S20 ~ S499, 480 points. Can be modified to be latched by setting up parameters.	
	Latched	S500 ~ S899, 400 points. Can be modified to be non-latched by setting up parameters.	
	Alarm	S900 ~ S1023, 124 points. Can be modified to be latched by setting up parameters.	

Functions of step relays:

The step relay S can easily set up the procedure in the industrial automation, which is the most basic device in the the sequential function chart (SFC) and has to be used with STL, RET instructions.

The device No. of S is S0 ~ S1023 (total 1,024 points) and both step relay S and output relay Y have output coils and contact A, B, and there is no limitation on the times of using the contact. S cannot directly drive the external load. When the step relay is not used in SFC, it can be used as a normal auxiliary relay. There are four types of step relays:

1. **Initial step relay:** S0 ~ S9, total 10 points, used for initial steps.
2. **Zero return step relay:** S10 ~ S19, total 10 points. S10 ~ S19 are planned for zero return when used with API 60 IST instruction in the program. If they are not used with IST, they will become normal step relays.
3. **General purpose step relay:** S20 ~ S511, total 492 points (for SA/SX/SC series MPU); S20 ~ S499, total 480 points (for EH/EH2/SV series MPU). Used for general purposes in SFC and their status will all be cleared when the operation of the PLC encounters power cut.
4. **Latched step relay:** S512 ~ S895, total 384 points (for SA/SX/SC series MPU); S20 ~ S127, total 108 points (for ES/EX/SS series MPU); S500 ~ S899, total 400 points (for EH/EH2/SV series MPU). Used for latched function in SFC and their status will all be retained when the operation of the PLC encounters power cut. They will remain at the status before the power cut when the PLC is powered again.
5. **Alarm step relay:** S896 ~ S1023, total 128 points (for SA/SX/SC series MPU); S900 ~ S1023, total 124 points (for EH/EH2/SV series MPU). Used with alarm driving instruction API 46 ANS as an alarm contact for recording the alarm messages or eliminating external malfunctions.

2.6 Numbering and Functions of Timers [T]

No. of timers (in decimal)

■ **ES/EX/SS series MPU:**

Timer T	100ms general purpose	T0 ~ T63, 64 points	Total 128 points
	10ms general purpose	T64 ~ T126, 63 points (M1028 = On:10ms; M1028 = Off:100ms)	
	1ms general purpose	T127, 1 point	

■ **SA/SX/SC series MPU:**

Time T	100ms general purpose	T0 ~ T199, 200 points. T192 ~ T199 are the timers for subroutine. Fixed to be non-latched	Total 256 points
	100ms accumulative	T250 ~ T255, 6 points. Fixed to be latched.	
	10ms general purpose	T200 ~ T239, 40 points. Fixed to be non-latched	
	10ms accumulative	T240 ~ T245, 6 points. Fixed to be latched.	
	1ms accumulative	T246 ~ T249, 4 points. Fixed to be latched.	

■ **EH/EH2/SV series MPU:**

Timer T	100ms general purpose	T0 ~ T199, 200 points. Can be latched by setting up parameters. T192 ~ T199 are the timers for subroutine.	Total 256 points
	100ms accumulative	T250 ~ T255, 6 points. Fixed to be latched.	
	10ms general purpose	T200 ~ T239, 40 points. Can be latched by setting up parameters.	
	10ms accumulative	T240 ~ T245, 6 points. Fixed to be latched.	
	1ms accumulative	T246 ~ T249, 4 points. Fixed to be latched.	

Functions of timers:

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 output coil will be On. The set value should be a K value in decimal and the data register D can also be a set value.

The actual set time in the timer = timing unit × set value

There are three types of timers:

1. **General purpose timer:**

For ES/SA series MPU: The timer executes once when the program reaches END instruction. When TMR instruction is executed, the output coil will be On when the timing reaches its target.

For EH/EH2/SV series MPU: The timer executes once when the program reaches TMR instruction. When TMR instruction is executed, the output coil will be On when the timing reaches its target.
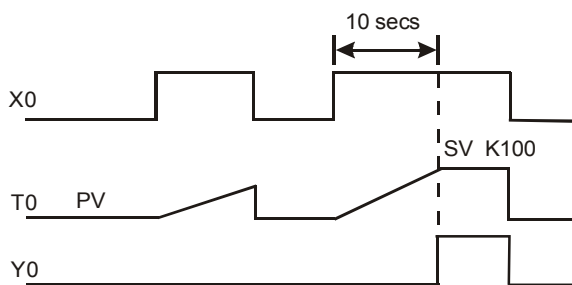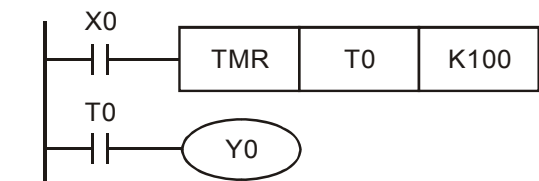
■ When X0 = On, The PV in timer T0 will count up by 100ms. When the PV = SV K100, the output coil T0 will be On.

■ When X0 = Off or the power is off, the PV in timer T0 will be cleared as 0, and the output coil T0 will be Off.

2. **Accumulative type timer:**

For ES/SA series MPU: The timer executes once when the program reaches END instruction. When TMR instruction is executed, the output coil will be On when the timing reaches its target.

For EH/EH2/SV series MPU: The timer executes once when the program reaches TMR instruction. When TMR instruction is executed, the output coil will be On when the timing reaches its target.
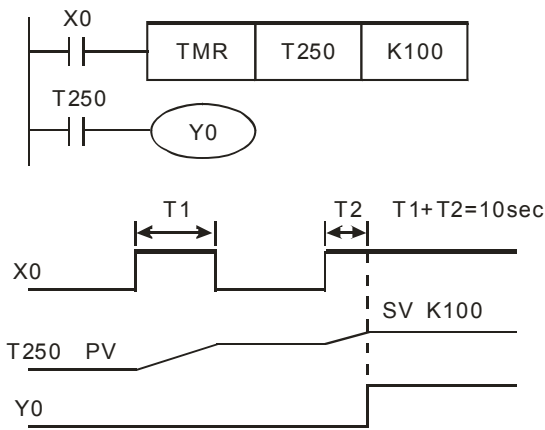
- When X0 = On, The PV in timer T250 will count up by 100ms. When the PV = SV K100, the output coil T0 will be On.

- When X0 = Off or the power is off, timer T250 will temporarily stop the timing and the PV remain unchanged. When X0 is On again, the timing will resume and the PV will count up and when the PV = SV K100, the output coil T0 will be On.

3. **Subrountine timer:**

Timer T192 ~ T199 are used in subrountines or interruption subrountines.

For SA series MPU: The timer executes once when the program reaches END instruction. When END instruction is executed, the output coil will be On when the timing reaches its target.

For EH/EH2/SV series MPU: The timer executes once when the program reaches TMR or END instruction. When TMR or END instruction is executed, the output coil will be On when the PV equals SV.

If the general purpose timer is used in a subroutine or interruption subroutine but the subroutine is not being executed, the timer will not be able to time correctly.

How to designate SV: The actual set time in the timer = timing unit × set value

a) Designating constant K: SV is a constant K
b) Indirectly designating D: SV is data register D

2.7 Numbering and Functions of Counters [C]

No. of counters (in decimal)

■ **ES/EX/SS series MPU:**

Counter C	16-bit counting up, for general purpose	C0 ~ C111, 112 points. Fixed to be non-latched.	Total 141 points
	16-bit counting up, for latched	C112 ~ C127, 16 points. Fixed to be latched.	
32-bit counting up/down high-speed counter C	1-phase 1 input	C235 ~ C238, C241, C242, C244, 7 points. Fixed to be latched.	
	1-phase 2 inputs	C246, C247, C249, 3 points. Fixed to be latched.	
	2-phase 2 inputs	C251, C252, C254, 3 points. Fixed to be latched.	

■ **SA/SX/SC series MPU:**

Counter C	16-bit counting up, for general purpose	C0 ~ C95, 96 points. Fixed to be non-latched.	Total 235 points
	16-bit counting up, for latched	C96 ~ C199, 104 points. Can be modified to be non-latched by setting up parameters.	
	32-bit counting up/down, for general purpose	C200 ~ C215, 16 points. Fixed to be non-latched.	
	32-bit counting up/down, for latched	C216 ~ C234, 19 points. Can be modified to be non-latched by setting up parameters.	

(SA/SX) 32-bit counting up/down high-speed counter C	1-phase 1 input, for latched	C235 ~ C242, C244, 9 points	Can be modified to be non-latched by setting up parameters.	Total 16 points
	1-phase 2 inputs, for latched	C246, C247, C249, 3 points		
	2-phase 2 inputs, for latched	C251 ~ C254, 4 points		
(SC) 32-bit counting up/down high-speed counter C	1-phase 1 input, for latched	C235 ~ C245, 11 points		Total 19 points
	1-phase 2 inputs, for latched	C246 ~ C250, 4 points		
	2-phase 2 inputs, for latched	C251 ~ C255, 4 points		

■ **EH/EH2/SV series MPU:**

Counter C	16-bit counting up, for general purpose	C0 ~ C99, 100 points. Can be modified to be latched by setting up parameters.	Total 253 points
	16-bit counting up, for latched	C100 ~ C199, 100 points. Can be modified to be non-latched area by setting up parameters.	
	32-bit counting up/down, for general purpose	C200 ~ C219, 20 points. Can be modified to be latched by setting up parameters.	
	32-bit counting up/down, for latched	C220 ~ C234, 15 points. Can be modified to be non-latched by setting up parameters.	
32-bit counting up/down high-speed counter C	Software 1-phase 1 input	C235 ~ C240, 6 points	Can be modified to be non-latched by setting up parameters.	
	Hardware 1-phase 1 input	C241 ~ C244, 4 points	
	Hardware 1-phase 2 inputs	C246 ~ C249, 4 points	
	Hardware 2-phase 2 inputs	C251 ~ C254, 4 points	

■ **Features of counter:**

	16 bits counters	32 bits counters	
	General purpose	General purpose	High speed
Type			
Counting direction	Counting up	Counting up, counting down	
Set value	0 ~ 32,767	-2,147,483,648 ~ +2,147,483,647	
SV designation	Constant K or data register D	Constant K or data register D (designating 2 values)	
Present value	Counting will stop when the SV is reached.	Counter will continue when the SV is reached.	
Output contact	On and being retained when the counting reaches SV.	On and keeps being On when counting up reaches SV. Reset to Off when counting down reaches SV.	
Reset	PV will be return to 0 when RST instruction is executed and the contact will be reset to Off.		
Contact action	Acts when the scanning is completed.	Acts when the scanning is completed.	Acts immediately when the counting reaches its target, has nothing to do with the scan period.

Functions of counters:

When the pulse input signals of the counter go from Off to On and the present value in the counter equals the set value, the output coil will be On. The set value should be a K value in decimal and the data register D can also be a set value.

16-bit counters C0 ~ C199:

1. The setup range of 16-bit counter: K0 ~ K32,767. K0 is the same as K1. The output contact will be On immediately when the first counting starts.

2. PV in the general purpose counter will be cleared when the power of the PLC is switched off. If the counter is a latched type, the counter will retain the PV and contact status before the power is off and resume the counting

after the power is on again.

3. If you use MOV instruction, WPLSoft or HPP to send a value bigger than the SV to the present value register of C0, next time when X1 goes from Off to On, the contact of counter C0 will be On and its PV will equal SV.

4. The SV in the counter can be constant K (set up directly) or the values in register D (set up indirectly, excluding special data registers D1000~ D1999).

5. If you set up a constant K as the SV, it should be a positive value. Data register D as SV can be positive or negative. When the PV reaches up to 32,767, the next PV will turn to -32,768.

Example:

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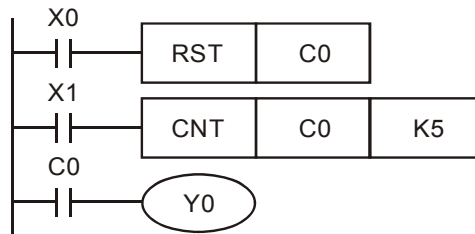

a) When X0 = On, RST instruction will be executed, PV in C0 will be "0" and the output contact will be reset to Off.

b) When X1 goes from Off to On, the PV in the counter will count up (plus 1).

c) When the counting of C0 reaches SV K5, the contact of C0 will be On and PV of C0 = SV = K5. The X1 trigger signal comes afterwards will not be accepted by C0 and the PV of C0 will stay at K5.
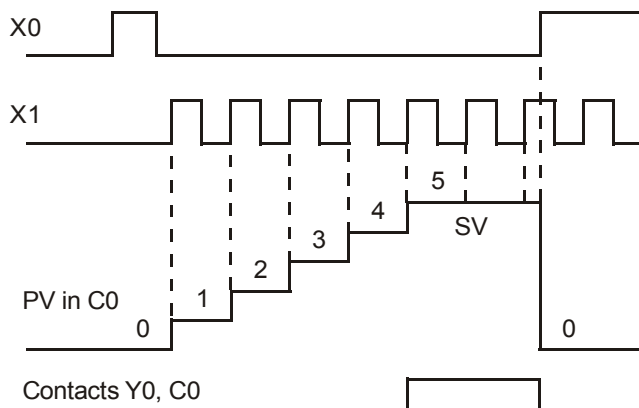
32-bit general purpose addition/subtraction counters C200 ~ C234:

1. The setup range of 32-bit counter: K-2,147,483,648 ~ K2,147,483,647 (not available for ES/EX/SS series MPU).

2. Addition or subtraction of the counters is designated by On/Off status of special auxiliary relays M1200 ~ M1234. For example, when M1200 = Off, C200 will be an addition counter; when M1200 = On, C200 will be a subtraction counter.

3. The SV can be constant K or data register D (excluding special data registers D1000 ~ D1999). Data register D as SV can be a positive or negative value and an SV will occupy two consecutive data registers.

4. PV in the general purpose counter will be cleared when the power of the PLC is switched off. If the counter is a latched type, the counter will retain the PV and contact status before the power is off and resume the counting after the power is on again.

5. When the PV reaches up to 2,147,483,647, the next PV will turn to -2,147,483,648. When the PV reaches down to -2,147,483,648, the next PV will turn to 2,147,483,647.

Example:

LD	X10
OUT	M1200
LD	X11
RST	C200
LD	X12
CNT	C200 K-5
LD	C200
OUT	Y0
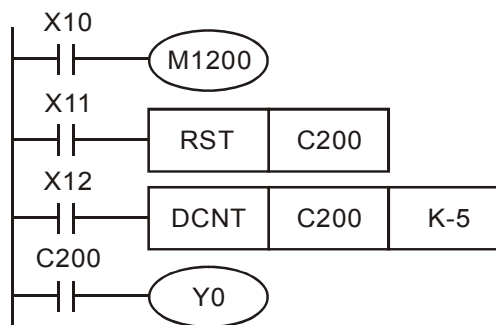
a) X10 drives M1200 to determine whether C200 is an addition or subtraction counter.

b) When X11 goes from Off to On, RST instsruction will be executed and the PV in C200 will be cleared to "0" and the contact will be Off.

c) When X12 goes from Off to On, the PV in the counter will count up (plus 1) or count down (minus 1).

d) When the PV in C200 changes from K-6 to K-5, the contact of C200 will go from Off to On. When the PV in C200 changes from K-5 to K-6, the contact of C200 will go from On to Off.

e) If you use MOV instruction, WPLSoft or HPP to send a value bigger than the SV to the present value register of C0, next time when X1 goes from Off to On, the contact of counter C0 will be On and its PV will equal SV.
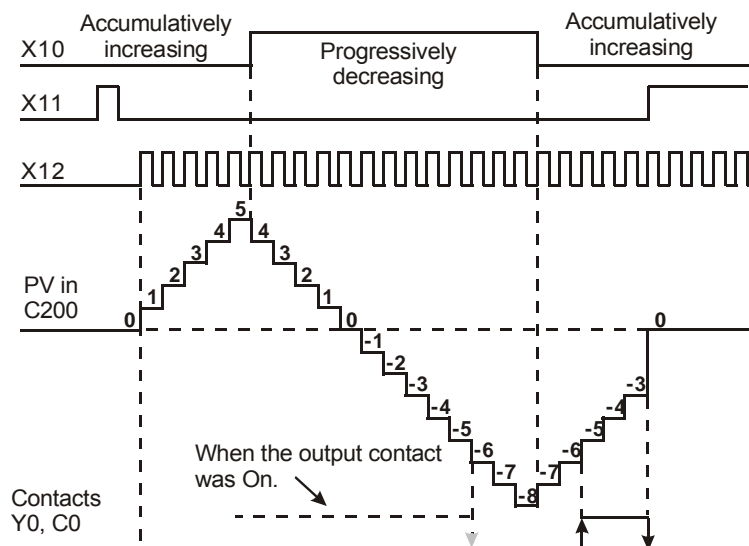
32-bit high-speed addition/subtraction counters C235 ~ C255:

1. The setup range of 32-bit counter: K-2,147,483,648 ~ K2,147,483,647

2. Addition or subtraction of C235 ~ C244 is designated by On/Off status of special auxiliary relays M1235 ~ M1244. For example, when M1235 = Off, C235 will be an addition counter; when M1235 = On, C235 will be a subtraction counter.

3. Addition or subtraction of C246 ~ C255 is designated by On/Off status of special auxiliary relays M1246 ~ M1255. For example, when M1246 = Off, C246 will be an addition counter; when M1246 = On, C246 will be a subtraction counter.

4. The SV can be constant K or data register D (excluding special data registers D1000 ~ D1999). Data register D

as SV can be a positive or negative value and an SV will occupy two consecutive data registers.

5. If using DMOV instruction, WPLSoft or HPP to send a value which is large than the setting to any high-speed counter, next time when the input point X of the counter goes from Off to On, this contact will remain unchanged and it will perform addition and subtraction with the present value.

6. When the PV reaches up to 2,147,483,647, the next PV will turn to -2,147,483,648. When the PV reaches down to -2,147,483,648, the next PV will turn to 2,147,483,647.

■ **High-speed counters for ES/EX/SS series MPU, total bandwidth: 20KHz**

Type / Input	1-phase input							1-phase 2 inputs			2-phase 2 inputs		
	C235	C236	C237	C238	C241	C242	C244	C246	C247	C249	C251	C252	C254
X0	U/D				U/D		U/D	U	U	U	A	A	A
X1		U/D					R	D	D	D	B	B	B
X2			U/D			U/D			R	R		R	R
X3				U/D		R	S			S			S

U: Progressively increasing input   A: A phase input   S: Input started
D: Progressively decreasing input   B: B phase input   R: Input cleared

1. Input points X0 and X1 can be planned as counters of higher speed with 1-phase 1 input reaching 20KHz. But the two counting frequencies added together has to be smaller or equal 20KHz. If the input is a 2-phase 2 input signal, the counting frequency will be approximately 4KHz. The 1-phase input of high-speed counters X2 and X3 and reach 10KHz.

2. The use of DHSCS instruction together with DHSCR instruction in ES series MPU cannot exceed 4 times.

■ **High-speed counters for SA/SX series MPU, total bandwidth: 40KHz**

Type / Input	1-phase input									1-phase 2 inputs			2-phase 2 inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C244	C246	C247	C249	C251	C252	C253	C254
X0	U/D						U/D		U/D	U	U	U	A	A	B	A
X1		U/D					R		R	D	D	D	B	B	A	B
X2			U/D					U/D			R	R		R		R
X3				U/D				R	S			S				S
X4					U/D											
X5						U/D										

U: Progressively increasing input   A: A phase input   S: Input started
D: Progressively decreasing input   B: B phase input   R: Input cleared

1. The frequency of input points X0 and X1 of 1-phase input can reach up to 20KHz, X2 ~ X5 can reach 10KHz. The frequency of C251, C252 and C254 of 2-phase input (X0, X1) can reach up to 4KHz. The maximum frequency of C253 is 25KHz (only supports 4 times frequency counting).

2. Input point X5 has two functions:

    a) When M1260 = Off, C240 will be normal U/D high-speed counter.

    b) When M1260 = On and DCNT instruction enables C240, X5 will be the shared reset signal for C235 ~ C239. Counter C240 will continue to receive the input signals from X5.

■ **High-speed counters for SC series MPU, total bandwidth: 130KHz**

Type / Input	1-phase input											1-phase 2 inputs				2-phase 2 inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245	C246	C247	C249	C250	C251	C252	C254	C255
X0	U/D						U/D			U/D		U	U	U		A	A	A	
X1		U/D					R			R		D	D	D		B	B	B	
X2			U/D					U/D					R	R			R	R	
X3				U/D				R		S				S				S	
X4					U/D														
X5						U/D													
X10									U/D						U				A
X11											U/D				D				B

U: Progressively increasing input A: A phase input S: Input started
D: Progressively decreasing input B: B phase input R: Input cleared

1. The functions of high-speed counters of input points X0 ~ X5 are the same of those in SA/SX series MPU.

2. The maximum frequency of the input points X10 (C243), X11 (C245) and (X10, X11)(C250) of 1-phase input is 100KHz. The total bandwidth of X10 ~ X11 high-speed counting is 130KHz. The maximum input frequency of C255 (2-phase input X10, X11) is 50KHz.

3. The use of DHSCS instruction together with DHSCR instruction in SA/SX/SC series MPU cannot exceed 6 times. The use of DHSZ instruction cannot exceed 6 times as well. When DHSCS instruction designates I interruption, the designated high-speed counter cannot be used in DHSCS, DHSCR and DHSZ instruction.

4. Functions of high-speed counters X10 ~ X11 in SC series MPU:

   a) When X10 and X11 are set to be 1-phase 1 input or 1-phase 2 outputs, the maximum frequency can reach 100KHz. When set to be 2-phase 2 inputs, the maximum frequency can reach 50KHz.

   b) X10 and X11 can be set to be rising-edge or falling-edge counting. X10 is set by D1166 and X11 by D1167. K0: rising-edge counting; K1: falling-edge counting; K2: rising-/falling-edge counting (only available in X10).

   c) Counting up or down of C243 is determined by On/Off status of M1243 anc that of C245 is determined by On/Off of M1245. Rising-edge and falling-edge countings cannot be performed at the same time. Rising edge or falling edge of C250 is determined by the content (K0 or K1) in D1166. C255 can only be used in 4 times frequency counting and rising-edge and falling-edge triggers are not available for C255.

   d) When you use C243 or C245, you will not be able to use C250 or C255, and vice versa.

   e) High-speed counter and high-speed comparator:
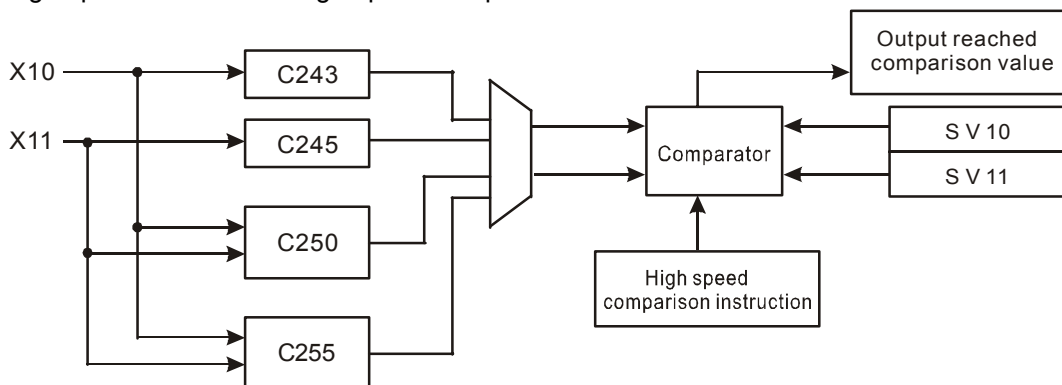
   f) Explanations on high-speed counter and high-speed comparator:

   (i) When DHSCS and DHSCR instructions use new added high-speed counters, they can only use two groups of SVs in high speed comparison instruction. Assume you have used the comparison instruction DHSCS D0 C243 Y10, you can only set another group of instruction DHSCR D2 C243 Y10 or DHSCS D4 C245 Y10.

(ii) When DHSZ instruction uses new added high-speed counters, it can only use one group of SV in the comparator.

(iii) The number of SVs in high speed comparison instructions offered by SA/SX series MPU will not decrease owing to the increasing of high-speed counters.

(iv) If the output device of the high-speed comparison instruction DHSCS requires high-speed output, it is recommended you use Y10 or Y11 for the output. If you use other general devices, the output will delay for 1 scan period for its setup or clearing. For example, if I0x0 interruption is set, C243 will correspond to I020, C245 to I040, and C250 and C255 to I060.

(v) The high speed comparison instruction DHSCR is able to clear the counter, but only the counters used in the same instruction, e.g. DHSCR K10 C243 C243. This function only applies to 4 special high-speed counters C243, C245, C250 and C255.

5. Counting modes:

a) The 2-phase 2 inputs counting mode of the high speed counters in ES/EX/SS (V5.5 and above) and SA/SX/SC series MPU is set by special D1022 with normal frequency, double frequency and 4 times frequency modes. The contents in D1022 will be loaded in in the first scan when PLC is switched from STOP to RUN.

Device No.	Function
D1022	Setting up the multiplied frequency of the counter
D1022 = K1	Normal frequency mode selected
D1022 = K2 or 0	Double frequency mode selected (default)
D1022 = K4	4 times frequency mode selected

b) Multiplied frequency mode (↑↓ indicates the occurrence of counting)
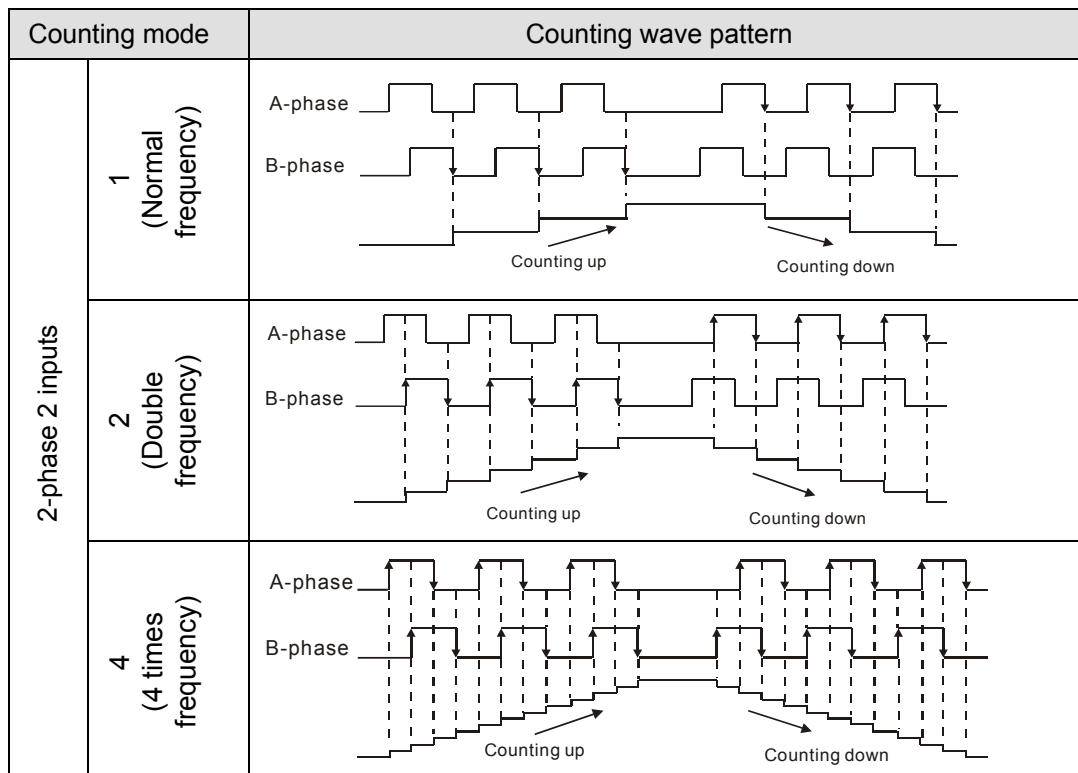
Counting mode		Counting wave pattern
2-phase 2 inputs	1 (Normal frequency)	
	2 (Double frequency)	
	4 (4 times frequency)	

EH/EH2/SV series MPU supports high speed counters. C235 ~ C240 are program-interruption 1-phase high

speed counter with a total bandwidth of 20KHz, can be used alone with a counting frequency of up to 10KHz. C241 ~ C254 are hardware high speed counter (HHSC). There are four HHSC in EH/EH2/SV series MPU, HHSC0 ~ 3. The pulse input frequency of HHSC0 and HHSC1 can reach 200KHz and that of HHSC2 and HHSC3 can reach 20KHz (1 phase or A-B phase). The pulse input frequency of HHSC0 ~ 3 of 40EH2 series MPU can reach 200KHz, among which:

- C241, C246 and C251 share HHSC0
- C242, C247 and C252 share HHSC1
- C243, C248 and C253 share HHSC2
- C244, C249 and C254 share HHSC3

1. Every HHSC can only be designated to one counter by DCNT instruction.
2. There are three counting modes in every HHSC (see the table below):
    a) 1-phase 1 input refers to "pulse/direction" mode.
    b) 1-phase 2 inputs refers to "clockwise/counterclockwise (CW/CCW)" mode.
    c) 2-phase 2 inputs refers to "A-B phase" mode.

Counter type / Input	Program-interruption high speed counter 1-phase 1 input C235	C236	C237	C238	C239	C240	Hardware high speed counter 1-phase 1 input C241	C242	C243	C244	1-phase 2 inputs C246	C247	C248	C249	2-phase 2 inputs C251	C252	C253	C254
X0	U/D						U/D				U				A			
X1		U/D									D				B			
X2			U/D				R				R				R			
X3				U/D			S				S				S			
X4					U/D			U/D				U				A		
X5						U/D						D				B		
X6								R				R				R		
X7								S				S				S		
X10									U/D				U				A	
X11													D				B	
X12									R				R				R	
X13									S				S				S	
X14										U/D				U				A
X15														D				B
X16										R				R				R
X17										S				S				S

U: Progressively increasing input   A: A phase input   S: Input started
B: Progressively decreasing input   B: B phase input   R: Input cleared

3. System structure of the hardware high speed counters:

a) HHSC0 ~ 3 have reset signals and start signals from external inputs. Settings in M1272, M1274, M1276 and M1278 are reset signals of HHSC0, HHSC1, HHSC2 and HHSC3. Settings in M1273, M1275, M1277 and M1279 are start signals of HHSC0, HHSC1, HHSC2 and HHSC3.

b) If the external control signal inputs of R and S are not in use, you can set M1264/M1266/M1268/M1270 and M1265/M1267/M1269/M1271 as True and disable the input signals. The corresponding external inputs can be used again as general input points (see the figure below).

c) When special M is used as a high speed counter, the inputs controlled by START and RESET will be affected by the scan time.
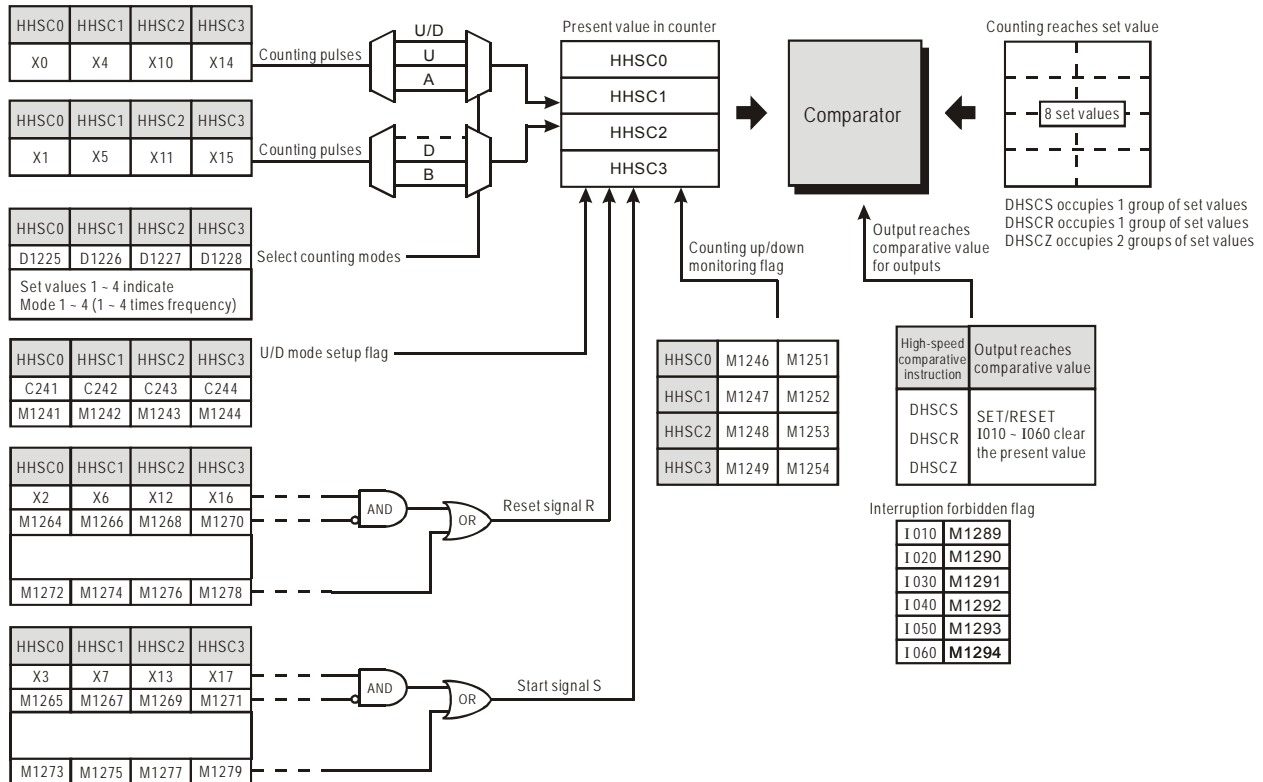
4. Counting modes:

The counting modes of the hardware high-speed counters in EH/EH2/SV series MPU can be set in D1225 ~ D1228.

Counting modes		Wave pattern	
Type	Set value in special D	Counting up(+1)	Counting down(-1)
1-phase 1 input	1 (Normal frequency)	U/D ⎍ U/D FLAG ___	U/D ⎍ U/D FLAG ___
	2 (Double frequency)	U/D ⎍ U/D FLAG ___	U/D ⎍ U/D FLAG ___
1-phase 2 inputs	1 (Normal frequency)	U ⎍ D ___	U ___ D ⎍
	2 (Double frequency)	U ⎍ D ___	U ___ D ⎍

2-phase 2 inputs	1 (Normal frequency)	A	
		B	
	2 (Double frequency)	A	
		B	
	3 (Triple frequency)	A	
		B	
	4 (4 times frequency)	A	
		B	

5. Special registers for relevant flags and settings of high speed counters:

Flag	Function
M1150	DHSZ instruction in multiple set values comparison mode
M1151	The execution of DHSZ multiple set values comparison mode is completed.
M1152	Set DHSZ instruction as frequency control mode
M1153	DHSZ frequency control mode has been executed.
M1235 ~ M1244	Designating the counting direction of high speed counters C235 ~ C245 When M12☐☐ = Off, C2☐☐ will perform a counting up. When M12☐☐ = On, C2☐☐ will perform a counting down.
M1245~ M1255	Designating the counting direction of high speed counters C246 ~ C255 When M12☐☐ = Off, C2☐☐ will perform a counting up. When M12☐☐ = On, C2☐☐ will perform a counting down.
M1160	X5 as the reset input signal of all high speed counters
M1261	High-speed comparison flag for DHSCR instruction
M1264	Disable the external control signal input point of HHSC0 reset signal point (R)
M1265	Disable the external control signal input point of HHSC0 start signal point (S)
M1266	Disable the external control signal input point of HHSC1 reset signal point (R)
M1267	Disable the external control signal input point of HHSC1 start signal point (S)
M1268	Disable the external control signal input point of HHSC2 reset signal point (R)
M1269	Disable the external control signal input point of HHSC2 start signal point (S)
M1270	Disable the external control signal input point of HHSC3 reset signal point (R)
M1271	Disable the external control signal input point of HHSC3 start signal point (S)
M1272	Internal control signal input point of HHSC0 reset signal point (R)
M1273	Internal control signal input point of HHSC0 start signal point (S)
M1274	Internal control signal input point of HHSC1 reset signal point (R)
M1275	Internal control signal input point of HHSC1 start signal point (S)
M1276	Internal control signal input point of HHSC2 reset signal point (R)
M1277	Internal control signal input point of HHSC2 start signal point (S)
M1278	Internal control signal input point of HHSC3 reset signal point (R)
M1279	Internal control signal input point of HHSC3 start signal point (S)
M1289	High speed counter I010 interruption forbidden

Flag	Function
M1290	High speed counter I020 interruption forbidden
M1291	High speed counter I030 interruption forbidden
M1292	High speed counter I040 interruption forbidden
M1293	High speed counter I050 interruption forbidden
M1294	High speed counter I060 interruption forbidden
M1312	C235 Start input point control
M1313	C236 Start input point control
M1314	C237 Start input point control
M1315	C238 Start input point control
M1316	C239 Start input point control
M1317	C240 Start input point control
M1320	C235 Reset input point control
M1321	C236 Reset input point control
M1322	C237 Reset input point control
M1323	C238 Reset input point control
M1324	C239 Reset input point control
M1325	C240 Reset input point control
M1328	Enable Start/Reset of C235
M1329	Enable Start/Reset of C236
M1330	Enable Start/Reset of C237
M1331	Enable Start/Reset of C238
M1332	Enable Start/Reset of C239
M1333	Enable Start/Reset of C240
D1022	Multiplied frequency of A-B phase counters for ES/EX/SS and SA/SX/SC series MPU
D1150	Table counting register for DHSZ multiple set values comparison mode
D1151	Register for DHSZ instruction frequency control mode (counting by table)
D1152 (low word) D1153 (high word)	In frequency control mode, DHSZ reads the upper and lower limits in the table counting register D1153 and D1152.
D1166	Switching between rising/falling edge counting modes of X10 (for SC_V1.4 series MPU only)
D1167	Switching between rising/falling edge counting modes of X11 (for SC_V1.4 series MPU only)
D1225	The counting mode of the 1st group counters (C241, C246, C251)
D1226	The counting mode of the 2nd group counters (C242, C247, C252)
D1227	The counting mode of the 3rd group counters (C243, C248, C253)
D1228	The counting mode of the 4th group counters (C244, C249, C254)

Flag	Function
D1225 ~ D1228	Counting modes of HHSC0 ~ HHSC3 in EH/EH2/SV series MPU (default = 2) 1: Normal frequency counting mode 2: Double frequency counting mode 3: Triple frequency counting mode 4: 4 times frequency counting mode

1-phase 1 input high-speed counter

Example:

```
LD      X10
RST     C241
LD      X11
OUT     M1241
LD      X12
DCNT    C241   K5
LD      C241
OUT     Y0
```
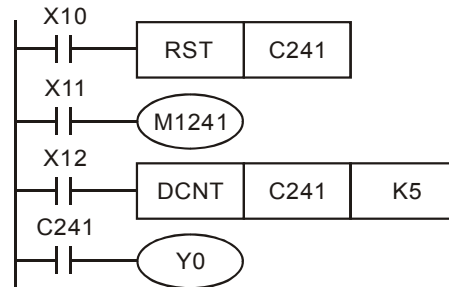

1. X11 drives M1241 to determine whether C241 is an addition or subtraction counter.

2. When X10 is On, RST instsruction will be executed and the PV in C241 will be cleared to "0" and the contact will be Off.

3. In C241, when X12 is On and C241 receives the signals from X0, the PV in the counter will count up (plus 1) or count down (minus 1).

4. When the counting of C241 reaches SV K5, the contact of C241 will be On. If there are still input signals from X0, the counting will continue.

5. C241 in ES/EX/SS and SA/SX/SC series MPU has external input signals to reset X1.

6. C241 in EH/EH2/SV series MPU has external input signals to reset X2 and start X3.

7. The external input contact of reset signal of C241 (HHSC0) in EH/EH2/SV series MPU is disabled by M1264. The external input contact of start signal is disabled by M1265.

8. The internal input contact of reset signal of C241 (HHSC0) in EH/EH2/SV series MPU is disabled by M1272. The internal input contact of start signal is disabled by M1273.

9. The counting modes (normal frequency or double frequency) of C246 (HHSC0) in EH/EH2/SV series MPU can be set up by D1225. The default setting is double frequency mode.
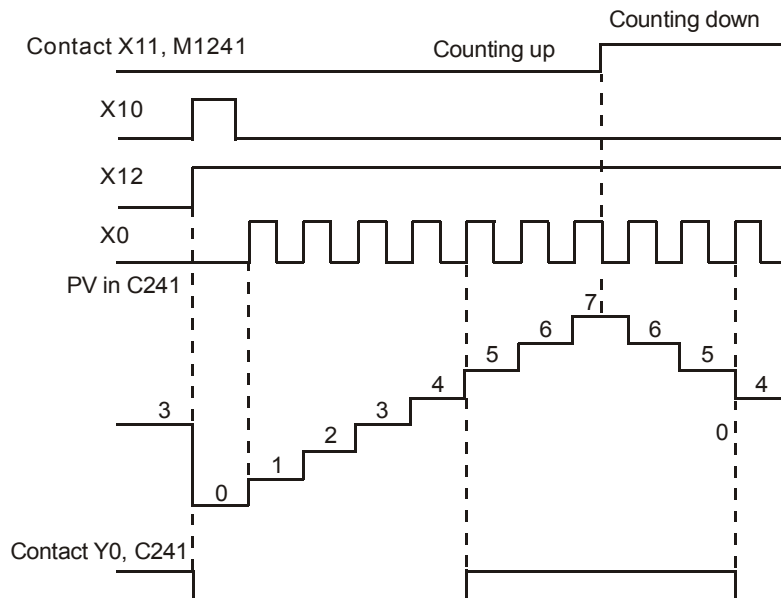
1-phase 2 inputs high-speed counter

Example:

LD	X10
RST	C246
LD	X11
DCNT	C246 K5
LD	C246
OUT	Y0
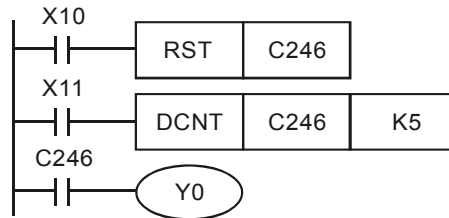
1. When X10 is On, RST instsruction will be executed. The PV in C246 will be cleared to "0" and the output contact will be reset to be Off.

2. In C246, when X11 is On and C246 receives the signals from X0, the PV in the counter will count up (plus 1) or count down (minus 1).

3. When the counting of C246 reaches SV K5, the contact of C246 will be On. If there are still input signals from X0, the counting will continue.
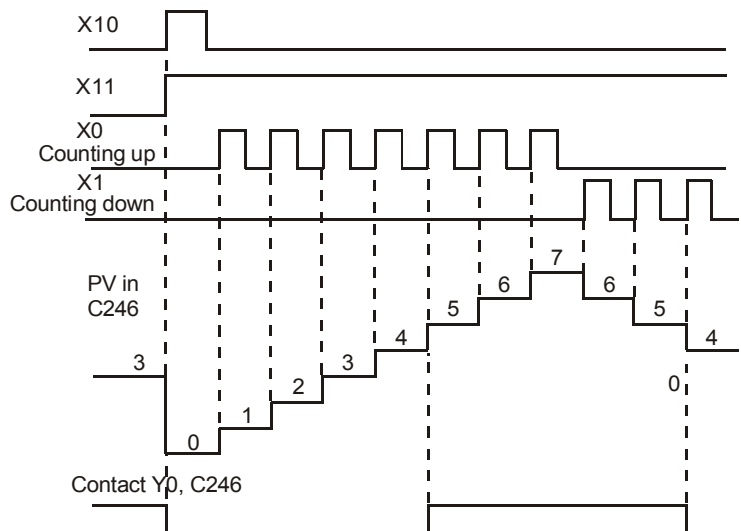
4. C246 in EH/EH2/SV series MPU has external input signals to reset X2 and start X3.

5. The counting modes (normal frequency or double frequency) of C246 (HHSC0) in EH/EH2/SV series MPU can be set up by D1225. The default setting is double frequency mode.

6. The external input contact of reset signal of C246 (HHSC0) in EH/EH2/SV series MPU is disabled by M1264. The external input contact of start signal is disabled by M1265.

7. The internal input contact of reset signal of C246 (HHSC0) in EH/EH2/SV series MPU is disabled by M1272. The internal input contact of start signal is disabled by M1273.

2-phase AB input high-speed counter

Example:

```
LD      X10
RST     C251
LD      X11
DCNT    C251 K5
LD      C251
OUT     Y0
```
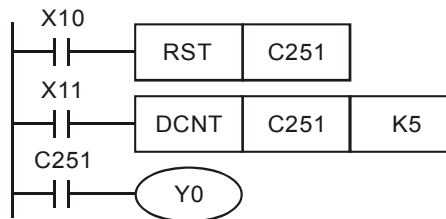

1. When X10 is On, RST instsruction will be executed. The PV in C251 will be cleared to "0" and the output contact will be reset to be Off.

2. In C251, when X11 is On and C251 receives the A-phase signals from X0 and B-phase signals from X1, the PV in the counter will count up (plus 1) or count down (minus 1). You can select different counting modes if you use EH/EH2/SV series MPU.

3. When the counting of C251 reaches SV K5, the contact of C251 will be On. If there are still input signals coming in, the counting will continue.

4. The counting modes (normal frequency, double frequency or 4 times frequency) of C251 (HHSC0) in ES/SA series MPU can be set up by D1022. The default setting is double frequency mode.

5. C251 in EH/EH2/SV series MPU has external input signals to reset X2 and start X3.

6. The counting modes (normal frequency, double frequency, triple frequency or 4 times frequency) of C251 (HHSC0) in EH/EH2/SV series MPU can be set up by D1225. The default setting is double frequency mode.

7. The external input contact of reset signal of C246 (HHSC0) in EH/EH2/SV series MPU is disabled by M1264. The external input contact of start signal is disabled by M1265.

8. The internal input contact of reset signal of C246 (HHSC0) in EH/EH2/SV series MPU is disabled by M1272. The internal input contact of start signal is disabled by M1273.

ES/EX/SS and SA/SX/SC series MPU (double frequency)
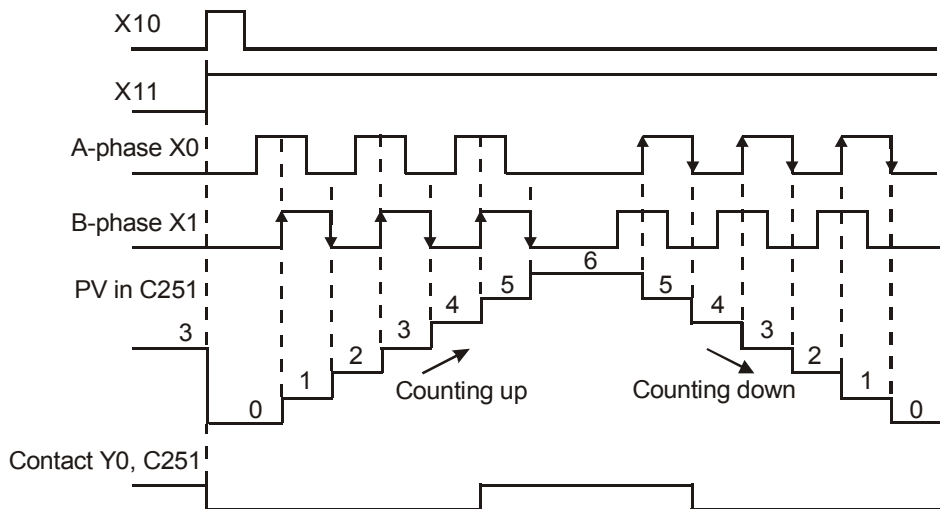
EH/EH2/SV series MPU (double frequency)
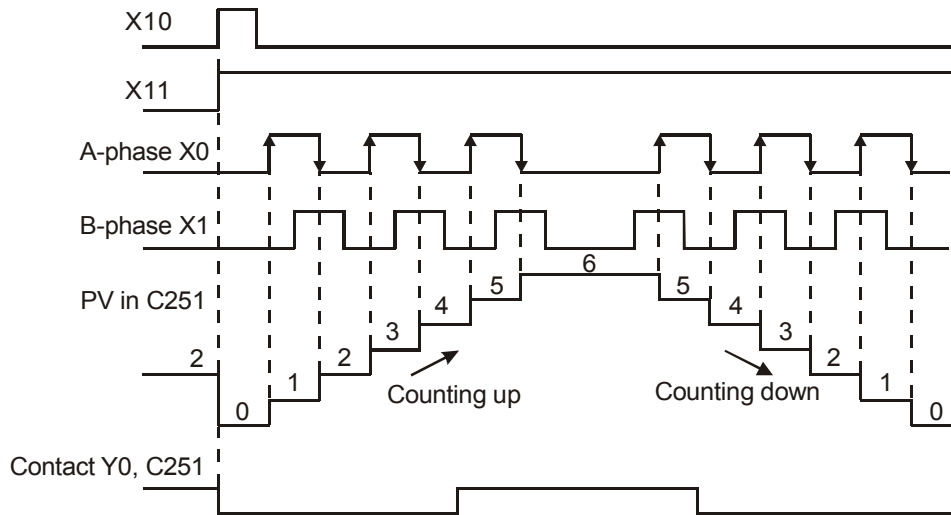
2.8 Numbering and Functions of Registers [D], [E], [F]

2.8.1 Data register [D]

A data register is for storing a 16-bit datum of values between -32,768 to +32,767. The highest bit is "+" or "-" sign. Two 16-bit registers can be combined into a 32-bit register (D + 1; D of smaller No. is for lower 16 bits). The highest b it is "+" or "-" sign and it can store a 32-bit datum of values between -2,147,483,648 to +2,147,483,647.

■ **ES/EX/SS series MPU:**

Data register D	General purpose	D0 ~ D407, 408 points	Total 744 points
	Latched	D408 ~ D599, 192 points. Fixed to be latched.	
	Special purpose	D1000 ~ D1143, 144 points. Some are latched.	
	Index register E, F	E, F, 2 points	

■ **SA/SX/SC series MPU:**

Data register D	General purpose	D0 ~ D199, 200 points. Fixed to be non-latched.	Total 5,000 points
	Latched	D200 ~ D999, D2000 ~ D4999, 3,800 points. Can be modified to be non-latched by setting up parameter.	
	Special purpose	D1000 ~ D1999, 1,000 points. Some are latched.	
	Index register E, F	E0 ~ E3, F0 ~ F3, 8 points	
File register		K0 ~ K1,599, MPU 1,600 points. Fixed to be latched.	1,600 points

■ **EH/EH2/SV series MPU:**

Data register D	General purpose	D0 ~ D199, 200 points. Can be modified to be latched by setting up parameters.	Total 10,000 points
	Latched	D200 ~ D999, D2000 ~ D9999, 8,800 points. Can be modified to be non-latched by setting up parameters.	
	Special purpose	D1000 ~ D1999, 1,000 pints. Some are latched.	
	Index register E, F	E0 ~ E7, F0 ~ F7, 16 points.	
File register		K0 ~ K9,999, MPU 10,000 points. Fixed to be latched.	10,000 points

There are five types of registers:

1. **General purpose register:** When PLC goes from RUN to STOP or the power of the PLC is switched off, the data in the register will be cleared to "0". When M1033 = On and PLC goes from RUN to STOP, the data will not be cleared, but will still be cleared to "0" when the power is off.

2. **Latched register:** When the power of PLC is switched off, the data in the register will not be cleared but will retain at the value before the power is off. You can use RST or ZRST instruction to clear the data in the latched register.

3. **Special purpose register:** Every register of this kind has its special definition and purpose, mainly for storing the system status, error messages and monitored status. See 2.10 and 2.11 for more details.

4. **Index register E, F:** The index register is a 16-bit register. There are 2 points of E, F in ES/EX/SS series MPU; 8 points (E0 ~ E3, F0 ~ F3) in SA/SX/SC series MPU; 16 points (E0 ~ E7, F0 ~ F7) in EH/EH2/SV series MPU. If the index register is to be used as a 32-bit register, please designate E. When E is already designated in a 32-bit instruction, using also F will not be allowed.

5. **File register:** There are 1,600 file registers (K0 ~ K1,599) in SA/SX/SC series MPU and 10,000 file registers (K0 ~ K9,999) in EH/EH2/SV series MPU. The file register does not have an exact device No.; therefore the read/write function of file registers has to be executed by instruction API 148 MEMR, API 149 MEMW or through peripheral devices HPP and WPLSoft.

2.8.2 Index Register [E], [F]
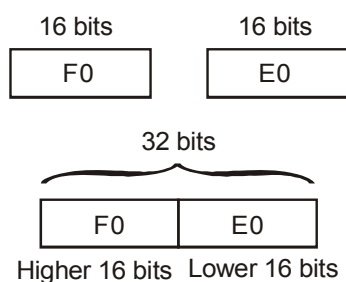
Index registers E, F are 16-bit data registers and can be written and read.

If you need to use a 32-bit register, you have to designate E. In this case, F will be covered by E and cannot be used anymore; otherwise, the content in E (32-bit) will be incorrect. We suggest you use DMOVP K0 E instruction, the content in E (including F) will be cleared to "0" when the power of PLC is switched on.

The combination of E, F when you use a 32-bit index register:
(F0, E0), (F1, E1), (F2, E2), …(F7, E7)
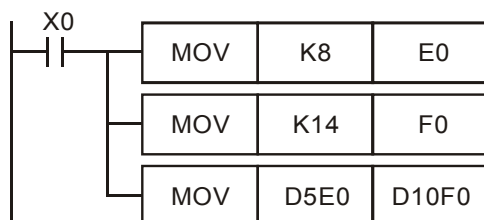
When X0 = On, E0 = 8, F0 = 14, D5E0 = D (5 + 8) = D13, D10F0 = D (10 + 14) = D24. At this moment, the content in D13 will be moved to D24.

The index register is the same as normal operands, can be used for moving or comparison on word devicies (KnX, KnY, KnM, KnS, T, C, D) and bit devices (X, Y, M, S). ES/SA series MPU does not support constant (K, H) index register, but EH/EH2/SV series MPU supports constant (K, H) index register.

ES/EX/SS series MPU has 2 points of index registers E0, F0

SA/SX/SC series MPU has 8 points of index registers E0 ~ E3, F0 ~ F3

EH/EH2/SV series MPU has 16 points of index registers E0 ~ E7, F0 ~ F7

■ Some instructions do not support index registers. For how to use index register E, F to modify the operands, see Chapter 5.4 for more details.

■ When you use the instruction mode in WPLSoft to generate constant (K, H) index register function, please use symbol "@". For example, "MOV K10@E0 D0F0"

▪ When you use index register E, F to modify the operands, the modification range CANNOT exceed the range of special purpose registers D1000 ~ D1999 and special auxiliary registers M1000 ~ M1999 in case errors may occur.

2.8.3   Functions and Features of File Registers

When the power of PLC is switched on, SA/SX/SC and EH/EH2/SV series MPU will check the following devices:

1. M1101 (whether the file register is enabled)

2. D1101 (No. of file registers in SA/SX/SC series MPU: K0 ~ K1,599; No. of file registers in EH/EH2/SV series MPU: K0 ~ K9,999)

3. D1102 (Number of file registers to be read in SA/SX/SC series MPU: K0 ~ K1,600; number of file registers to be read in EH/EH2/SV series MPU: K0 ~ K8,000)

4. D1103 (devices for storing the data read from file registers; the No. of designated data register D starts from K2,000 ~ K9,999; determining whether to automatically send the content in the file register to the designated data register.)

Note:

1. When D1101 of SA/SX/SC series MPU is bigger than 1,600, D1101 of EH/EH2/SV series MPU is bigger than 8,000 and D1103 is smaller than 2,000 or bigger than 9,999, the data read from file registers will not be sent to data register D.

2. When the program starts to send the data read from the file register to data register D and the address of the file register or the data register D exceed their ranges, PLC will stop the reading.

3. There are 1,600 file registers in SA/SX/SC series MPU and 10,000 in EH/EH2/SV series MPU. The file register does not have an exact device No.; therefore the read/write function of file registers has to be executed by instruction API 148 MEMR, API 149 MEMW or through peripheral devices HPP and WPLSoft.

4. If you tend to read a file register with an address that is not within the range, the read value will be "0".

2.9   Pointer [N], Pointer [P], Interruption Pointer [I]

▪ **ES/EX/SS series MPU:**

Pointer	N	For master control loop	N0 ~ N7, 8 points	Control point of master control loop
	P	For CJ, CALL instructions	P0 ~ P63, 64 points	Position pointer of CJ, CALL
	I	Interruption	External interruption	I001, I101, I201, I301, 4 points	Position pointer of interruption subroutine
			Timed interruption	I6□□, 1 point (□□＝10 ~ 99, time base = 1ms) (for V5.7)	
			Communication interrupt	I150, 1 point	

▪ **SA/SX/SC series MPU:**

Pointer	N	Master control loop	N0 ~ N7, 8 points	Control point of master control loop
	P	For CJ, CALL instructions	P0 ~ P255, 256 points	Position pointer of CJ, CALL
	I	Interruption	External interruption	I001, I101, I201, I301, I401, I501, 6 points	Position pointer of interruption subroutine
			Timer interruption	I6□□, I7□□, 2 points (□□ = 1 ~ 99, time base = 1ms)	
			High-speed counter interruption	I010, I020, I030, I040, I050, I060, 6 points	
			Communication interruption	I150, 1 point	

Note: Among the 6 pairs of interruption No. (I001, I010), (I101, I020), (I201, I030), (I301, I040), (I401, I050), (I501, I060), only 1 No. in the pair is allowed to be used in the program. If you use both No. in the pair and write them into the program, there may be syntax errors occurring.

■ **EH/EH2/SV series MPU:**

	N	Master control loop	N0 ~ N7, 8 points	Control point of master control loop
	P	For CJ, CALL instructions	P0 ~ P255, 256 points	Position pointer of CJ, CALL
Pointer	I	Interruption	External interruption	I00□(X0), I10□(X1), I20□(X2), I30□(X3), I40□(X4), I50□(X5), 6 points (□ = 1, rising-edge trigger ⌐, □ = 0, falling-edge trigger ⌐)	Position pointer of interruption subroutine
			Timed interruption	I6□□, I7□□, 2 points (□□ = 1 ~ 99, time base = 1ms) I8□□, 1 point (□□ = 1 ~ 99, time base = 0.1ms)	
			High-speed counter interruption	I010, I020, I030, I040, I050, I060, 6 points	
			Pulse interruption	I110, I120, I130, I140, 4 points	
			Communication interruption	I150, I160, I170, 3 points	
			Frequency measurement card triggered interruption	I180, 1 point	

Note: Input point X as a high-speed counter cannot be used as an external interruption signal. For example, if C251 occupies X0, X1, X2 and X3, the external input interruption No. I00□(X0), I10□(X1), I20□(X2), and I30□(X3) cannot be used.

Pointer N: Used with MC and MCR instructions. MC is the master control start instruction. When MC instruction is executer, the instructions between MC and MCR will still be executed normally. See Chapter 3 explanations on MC and MCR instructions for more details.

Pointer P: Used with API 00 CJ, API 01 CALL and API 02 SRET. See Chapter 6 explanations on CJ, CALL and SRET instructions for more details.

CJ Conditional Jump:
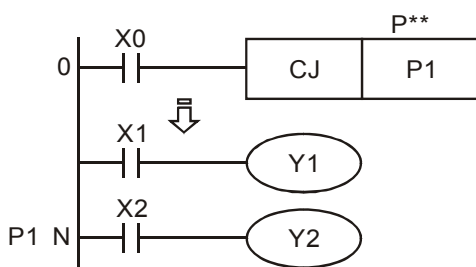
■ When X0 = On, the program will jump from address 0 to N (designated label P1) and keep on the execution. The addresses in the middle will be ignored.

■ When X0 = Off, the program will execute from address 0 and keep on executing. At this time, CJ instruction will not be executed.

CALL Call Subroutine, SRET Subroutine Return:
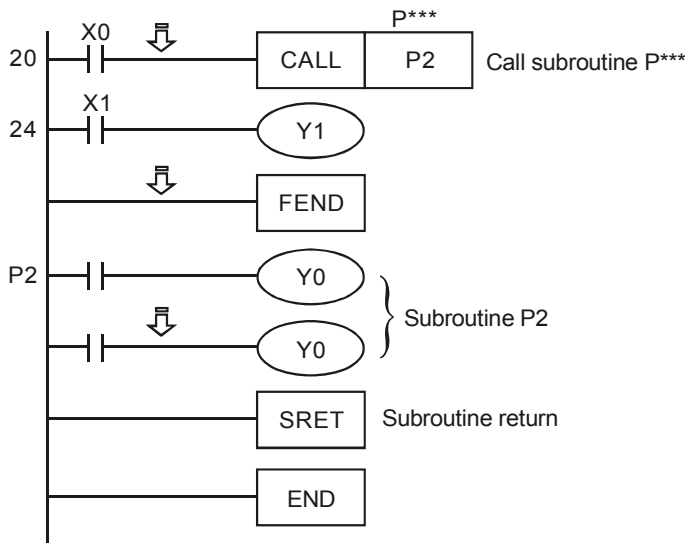
- When X0 = On, CALL instruction will be executed and the program will jump to P2 and executed the designated subroutine. When SRET instruction is executed, the program will return to address 24 and keep on the execution.

Interruption Pointer I: Used with API 04 EI, API 05 DI, API 03 IRET. See Chapter 5.5 for more details. There are 6 types of interruption pointer. To insert an interruption, you have to combine the action with EI (enable interruption), DI (disable interruption), IRET (interruption return) instructions.

1. External interruption: Due to the special hardware design inside the MPU, the input signals coming in at input terminals X0 ~ X5 when rising-edge or falling-edge triggers will not be affected by the scan cycle. The currently executed program will be interrupted immediately and the execution will jump to the designated interruption subroutine pointer I00□(X0), I10□(X1), I20□(X2), I30□(X3), I40□(X4), I50□(X5). Till the execution reaches IRET instruction, the program will return to the original position and keep on its execution.
    In SA/SX series MPU, X0 (pulse input point) works with X4 (external interruption point), corresponding to C235, C251 and C253 work with I401, which will be able to interrupt and intercept the present value in the high-speed counter. D1181 is the device to store the 32-bit value. X1 (pulse input point) works with X5 (external interruption point), corresponding to C236 works with I501, which will be able to interrupt and intercept the present value in the high-speed counter. D1198 and D1199 are the devices to store the 32-bit value.
    In SC series MPU, X10 (pulse input point) works with X4 (external interruption point), corresponding to C243 and C255 work with I401, which will be able to interrupt and intercept the present value in the high-speed counter. D1180 and D1181 are the devices to store the 32-bit value. X11 works with X5, corresponding to C245 works with I501, which will be able to interrupt and intercept the present value in the high-speed counter. D1198 and D1199 are the devices to store the 32-bit value.

2. Timed interruption: PLC automatically interrupts the currently executed program every a fixed period of time and jumps to the execution of a designated interruption subroutine.

3. Interruption when the counting reaches the target: The high-speed counter comparison instruction API 53 DHSCS can designates that when the comparison reaches the target, the currently executed program will be interrupted and jump to the designated interruption subroutine executing the interruption pointers I010, I020, I030, I040, I050 and I060.

4. Pulse interruption: The pulse output instruction API 57 PLSY can be set up that the interruption signal is sent out synchronously when the first pulse is sent out by enabling flags M1342 and M1343. The corresponding

interruptions are I130 and I140. You can also set up that the interruption signal is sent out after the last pulse is sent out by enabling flags M1340 and M1341. The corresponding interruptions are I110 and I120.

5. Communication interruption:

I150: When the communication instruction RS is being used, you can send out interruption request when the program receives a specific word by interruption I150. The specific word is set up in the low byte of D1168. This function can be adopted when the PLC receives data of different length during the connection with the communication device. Set up the end word in D1168 and write the interruption subroutine I150 and when the PLC receives this end word, the program will execute I150.

I160: RS instruction sends out interruption request when receiving a specific length of data. When the data received equals the low byte of D1169, I160 will be triggered. When D1169 = 0, I160 will not be triggered.

I170: In Slave mode, interruption I170 will be generated when the data receiving is completed. Normally when the communication terminal of the PLC is in Slave mode, PLC will not immediately process the communication data entered but process it after the END is executed. Therefore, when the scan time is very long and you need the communication data to be processed immediately, you can use interruption I170 for this matter.

6. Frequency measurement card triggered interruption:

I180: When the PLC sets up the frequency measurement card in mode 1 (pulse cycle measurement) and mode 3 (pulse number counting) by M1019 and D1034, I180 will be supported as well.

2.10 Special Auxiliary Relays and Special Data Registers

The types and functions of special auxiliary relays (special M) and special data registers (special D) are listed in the table below. Please be noted 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 the 2.11. Columns marked with "R" regers to "read only", "R/W" refers to "read and write", "-" refers to the status remains unchanged and "#" refers to the system will set it up according to the status of the PLC.

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1000*	Monitoring normally open contact (A)	○	○	○	Off	On	Off	R	NO	Off
M1001*	Monitoring normally closed contact (B)	○	○	○	On	Off	On	R	NO	On
M1002*	Enabling positive pulses (On when RUN)	○	○	○	Off	On	Off	R	NO	Off
M1003*	Enabling negative pulses (Off when RUN)	○	○	○	On	Off	On	R	NO	On
M1004*	On when syntax errors occur	○	○	○	Off	Off	-	R	NO	Off
M1005*	Password of data backup memory card and password of MPU do not match	✕	✕	○	Off	-	-	R	NO	Off
M1006*	Data backup memory card has not been initialized	✕	✕	○	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1007*	Data do not exist in the program area of data backup memory card	✕	✕	○	Off	-	-	R	NO	Off
M1008*	Scanning watchdog timer (WDT) On	○	○	○	Off	Off	-	R	NO	Off
M1009	Insufficient 24V DC supply, LV signal has been occurred.	○	○	○	Off	-	-	R	NO	Off
M1010	ES/SA: PLSY Y0 mode selection, continuous output when On EH/EH2/SV: Pulse output when reaching END instruction	○	○	○	Off	-	-	R/W	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*	1min clock pulse, 30s On / 30s Off	○	○	○	Off	-	-	R	NO	Off
M1015*	Enabling high-speed counter	✕	○	○	Off	-	-	R/W	NO	Off
M1016*	Displaying real time clock in A.D.	✕	○	○	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	Enabling frequency measurement card	✕	✕	○	Off	Off	-	R	NO	Off
M1020	Zero flag	○	○	○	Off	-	-	R	NO	Off
M1021	Borrow flag	○	○	○	Off	-	-	R	NO	Off
M1022	Carry flag	○	○	○	Off	-	-	R	NO	Off
M1023	PLSY Y1 mode selection, continuous output when On	○	○	✕	Off	-	-	R/W	NO	Off
M1024	Requesting COM1 monitoring	○	○	○	Off	-	-	R	NO	Off
M1025	There is incorrect request for communication.	○	○	○	Off	-	-	R	NO	Off
M1026	Enabling RAMP module	✕	○	○	Off	-	-	R/W	NO	Off
M1027	Number of PR outputs	✕	○	○	Off	-	-	R/W	NO	Off
M1028	10ms time switching flag. Off: time base of T64 ~ T126 = 100ms On: time base of T64 ~ T126 = 10ms	○	✕	✕	Off	-	-	R/W	NO	Off
M1029*	ES/SA: Y0 pulse output of PLSY, PLSR instructions is completed, or other relevant instructions complete their executions. EH/EH2/SV: the 1st group pulse output CH0 (Y0, Y1) is completed, or other relevant instructions complete their executions.	○	○	○	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1030*	ES/SA: Y1 pulse output of PLSY, PLSR instructions is completed, or other relevant instructions complete their executions. EH/EH2/SV: the 2nd group pulse output CH1 (Y2, Y3) is completed, or other relevant instructions complete their executions.	○	○	○	Off	-	-	R	NO	Off
M1031*	Clear all non-latched areas	○	○	○	Off	-	-	R/W	NO	Off
M1032*	Clear all latched areas	○	○	○	Off	-	-	R/W	NO	Off
M1033*	Memory latched when STOP	○	○	○	Off	-	-	R/W	NO	Off
M1034*	Disabling all Y outputs	○	○	○	Off	-	-	R/W	NO	Off
M1035*	Enabling input point X as the RUN/STOP switch, corresponding to D1035 (SA designates X7 only; SX designates X3 only; SC designates X5 only)	✕	○	○	-	-	-	R/W	YES	Off
M1036	EH2/SV: the 3rd group pulse output CH2 (Y4, Y5) is completed. (Not available in EH)	✕	✕	○	Off	-	-	R	NO	Off
M1036	SPD instruction is able to use X0 ~ X5 to detect the flag (only available in SC_V1.4 and above)	✕	○	✕	Off	Off	-	R/W	NO	Off
M1037	EH2/SV: the 4th group pulse output CH3 (Y6, Y7) is completed. (Not available in EH)	✕	✕	○	Off	-	-	R	NO	Off
M1039*	Fixing time scan mode	○	○	○	Off	-	-	R/W	NO	Off
M1040	Disabling step	○	○	○	Off	-	-	R/W	NO	Off
M1041	Starting step	○	○	○	Off	-	Off	R/W	NO	Off
M1042	Enabling pulses	○	○	○	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	Diabling all output reset	○	○	○	Off	-	-	R/W	NO	Off
M1046	Setting STL status as On	○	○	○	Off	-	-	R	NO	Off
M1047	Enabling STL monitoring	✕	○	○	Off	-	-	R/W	NO	Off
M1048	Alarm status	✕	○	○	Off	-	-	R	NO	Off
M1049	Setting up alarm monitoring	✕	○	○	Off	-	-	R/W	NO	Off
M1050	Inhibiting I001	○	○	✕	Off	-	-	R/W	NO	Off
M1051	Inhibiting I101	○	○	✕	Off	-	-	R/W	NO	Off
M1052	Inhibiting I201	○	○	✕	Off	-	-	R/W	NO	Off
M1053	Inhibiting I301	○	○	✕	Off	-	-	R/W	NO	Off
M1054	Inhibiting I401	✕	○	✕	Off	-	-	R/W	NO	Off
M1055	Inhibiting I501	✕	○	✕	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1056	Inhibiting I6☐☐	○	○	✕	Off	-	-	R/W	NO	Off
M1057	Inhibiting I7☐☐	✕	○	✕	Off	-	-	R/W	NO	Off
M1059	Inhibiting I010 ~ I060	✕	○	✕	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*	Calculation error	○	○	○	Off	Off	-	R	NO	Off
M1068*	Calculation error locked (D1068)	○	○	○	Off	-	-	R	NO	Off
M1070	ES/SA: Y1 time base switching for PWM instruction (On: 100us; Off: 1ms) EH/EH2/SV: Y0 time base switching for PWM instruction (On: 100us; Off: 1ms) EH_V1.2 and above/EH2/SV: when On, D1371 will decide the time base)	○	○	○	Off	-	-	R/W	NO	Off
M1071	Y2 time base switching for PWM instruction (On: 100us; Off: 1ms) EH_V1.2 and above/EH2/SV: when On, D1372 will decide the time base)	✕	✕	○	Off	-	-	R/W	NO	Off
M1072	Executing PLC RUN instruction	○	○	○	Off	On	Off	R/W	NO	Off
M1075	Error occurring when writing FLASH card	✕	✕	○	Off	-	-	R	NO	Off
M1076*	Real time clock malfunction	✕	○	○	Off	-	-	R	NO	Off
M1077	Battery in low voltage, malfunction or no battery	✕	○	○	Off	-	-	R	NO	Off
M1078	Immediately stopping Y0 pulse output for PLSY instruction	○	○	✕	Off	-	-	R/W	NO	Off
M1079	Immediately stopping Y1 pulse output for PLSY instruction	○	○	✕	Off	-	-	R/W	NO	Off
M1080	Requesting COM2 monitoring	○	○	○	Off	-	-	R	NO	Off
M1081	Changing direction for FLT instruction	✕	○	○	Off	-	-	R/W	NO	Off
M1082	Real time clock has been changed	✕	○	○	Off	-	-	R	NO	Off
M1083	Allowing interruption subroutine in FROM/TO instructions	✕	○	○	Off	-	-	R/W	NO	Off
M1084*	Detecting bandwidth (only available in ES/EX/SS_V6.4, SA/SX_V1.6, SC_V1.4 and above)	○	○	✕	Off	Off	Off	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1085	Selecting DVP-PCC01 duplicating function	○	○	○	Off	-	-	R/W	NO	Off
M1086	Setting up the switch for enabling password function of DVP-PCC01	○	○	○	Off	-	-	R/W	NO	Off
M1087*	Enabling LV signal	✕	✕	○	Off	-	-	R/W	NO	Off
M1088	Matrix comparison. Comparing between equivalent values (M1088 = 1) or different values (M1088 = 0).	✕	○	○	Off	Off	-	R/W	NO	Off
M1089	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.	✕	○	○	Off	Off	-	R	NO	Off
M1090	Matrix search start flag. Comparing from bit 0 (M1090 = 1).	✕	○	○	Off	Off	-	R	NO	Off
M1091	Matrix bit search flag. When the comparison is completed, the comparison will stop immediately (M1091=1).	✕	○	○	Off	Off	-	R	NO	Off
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.	✕	○	○	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. Clearing the current value of the Pr to 0.	✕	○	○	Off	Off	-	R/W	NO	Off
M1095	Matrix rotation/displacement/output carry flag	✕	○	○	Off	Off	-	R	NO	Off
M1096	Matrix displacement/input complement flag	✕	○	○	Off	Off	-	R/W	NO	Off
M1097	Matrix rotation/displacement direction flag	✕	○	○	Off	Off	-	R/W	NO	Off
M1098	Matrix counting the number of bits which are "1" or "0"	✕	○	○	Off	Off	-	R/W	NO	Off
M1099	On when the matrix counting result is "0"	✕	○	○	Off	Off	-	R/W	NO	Off
M1100	SPD instruction sampling once	✕	✕	○	Off	-	-	R/W	NO	Off
M1101*	Whether to enable file registers	✕	○	○	-	-	-	R/W	Yes	Off
M1102	Y10 pulse output ends (For SC)	✕	○	✕	Off	-	-	R/W	NO	Off
M1103	Y11 pulse output ends (For SC)	✕	○	✕	Off	-	-	R/W	NO	Off
M1104*	Status of SW1 on digital switch card/AX0 input point on 4DI card (photpcoupler isolation)	✕	✕	○	Off	Off	-	R	NO	Off
M1105*	Status of SW2 on digital switch card/AX1 input point on 4DI card (photpcoupler isolation)	✕	✕	○	Off	Off	-	R	NO	Off
M1106*	Status of SW3 on digital switch card/AX2 input point on 4DI card (photpcoupler isolation)	✕	✕	○	Off	Off	-	R	NO	Off
M1107*	Status of SW4 on digital switch card/AX3 input point on 4DI card (photpcoupler isolation)	✕	✕	○	Off	Off	-	R	NO	Off
M1108*	Status of SW5 on digital switch card	✕	✕	○	Off	Off	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1109*	Status of SW6 on digital switch card	✕	✕	○	Off	Off	-	R	NO	Off
M1110*	Status of SW7 on digital switch card	✕	✕	○	Off	Off	-	R	NO	Off
M1111*	Status of SW8 on digital switch card	✕	✕	○	Off	Off	-	R	NO	Off
M1112*	AY0 output point on 2DO card (transistor)	✕	✕	○	Off	-	Off	R/W	NO	Off
M1113*	AY1 output point on 2DO card (transistor)	✕	✕	○	Off	-	Off	R/W	NO	Off
M1115*	Switch for enabling pulse accelerating/decelerating output (not available in SC_V1.4 and above)	○	○	✕	Off	Off	Off	R/W	NO	Off
M1116*	Pulse output is accelerating (not available in SC_V1.4 and above)	○	○	✕	Off	Off	Off	R/W	NO	Off
M1117*	Accelerating/decelerating pulse output reaches its target (not available in SC_V1.4 and above)	○	○	✕	Off	Off	Off	R/W	NO	Off
M1118*	Pulse output is decelerating (not available in SC_V1.4 and above)	○	○	✕	Off	Off	Off	R/W	NO	Off
M1119*	Accelerating/decelerating pulse output is completed (not available in SC_V1.4 and above)	○	○	✕	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	Waiting for the sending of RS-485 communication data	○	○	○	Off	On	-	R	NO	Off
M1122	Sending request	○	○	○	Off	Off	-	R/W	NO	Off
M1123	Receiving is completed	○	○	○	Off	Off	-	R/W	NO	Off
M1124	Waiting for receiving	○	○	○	Off	Off	-	R	NO	Off
M1125	Communication reset	○	○	○	Off	Off	-	R/W	NO	Off
M1126	Selecting STX/ETX user defined or system defined	○	○	○	Off	Off	-	R/W	NO	Off
M1127	Sending/receiving data of communication instruction is completed (RS instruction not included)	○	○	○	Off	Off	-	R/W	NO	Off
M1128	Sending…/receiving... indication	○	○	○	Off	Off	-	R	NO	Off
M1129	Receiving time-out	○	○	○	Off	Off	-	R/W	NO	Off
M1130	Selectin STX/ETX user defined or system defined	○	○	○	Off	Off	-	R/W	NO	Off
M1131	On during MODRD/RDST/MODRW data are converted to hex data	○	○	○	Off	Off	-	R	NO	Off
M1132	On when there are no communication related instructions in the program	○	○	○	Off	-	-	R	NO	On
M1133*	Enabling special high-speed pulse output Y0 (50KHz) SC_V1.4 and above: 2-axis synchronous control, enabling Y10 output	✕	○	✕	Off	Off	Off	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1134*	Special high-speed pulse output Y0 (50KHz) On: continuous output (Not available in SC_V1.4 and above)	✕	○	✕	Off	Off	-	R/W	NO	Off
M1135*	Special high-speed pulse output Y0 (50KHz) reaches the target number of pulses. SC_V1.4 and above: 2-axis synchronous control, enabling Y11 output	✕	○	✕	Off	Off	Off	R/W	NO	Off
M1136*	Retaining the communication setting of COM3	✕	✕	○	Off	-	-	R/W	NO	Off
M1138*	Retaining the communication setting of COM1 (RS-232), modifying D1036 will be invalid when M1138 is set.	○	○	○	Off	-	-	R/W	NO	Off
M1139*	Selecting ASCII or RTU mode of COM1 (RS-232) when in Slave mode Off: ASCII; On: RTU	○	○	○	Off	-	-	R/W	NO	Off
M1140	MODRD/MODWR/MODRW data receiving error	○	○	○	Off	Off	-	R	NO	Off
M1141	MODRD/MODWR/MODRW parameter error	○	○	○	Off	Off	-	R	NO	Off
M1142	Data receiving of VFD-A commands error	○	○	○	Off	Off	-	R	NO	Off
M1143*	Selecting ASCII or RTU mode of COM2 (RS-485) when in Slave mode Off: ASCII; On: RTU Selecting ASCII or RTU mode of COM2 (RS-485) when in Master mode (used together with MODRD/MODWR/MODRW instructions) Off: ASCII; On: RTU	○	○	○	Off	-	-	R/W	NO	Off
M1144*	Switch for enabling adjustable pulse accelerating/decelerating output Y0	✕	○	✕	Off	Off	Off	R/W	NO	Off
M1145*	Adjustable pulse output Y0 is accelerating	✕	○	✕	Off	Off	-	R/W	NO	Off
M1146*	Adjustable pulse output Y0 reaches the target frequency	✕	○	✕	Off	Off	-	R/W	NO	Off
M1147*	Adjustable pulse output Y0 is decelerating	✕	○	✕	Off	Off	-	R	NO	Off
M1148*	Adjustable pulse output Y0 is completed	✕	○	✕	Off	Off	Off	R/W	NO	Off
M1149*	Adjustable pulse output Y0 temporarily stops counting the number of pulses.	✕	○	✕	Off	Off	-	R/W	NO	Off
M1150	DHSZ instruction in multiple set values comparison mode	✕	✕	○	Off	-	-	R/W	NO	Off
M1151	The execution of DHSZ multiple set values comparison mode is completed.	✕	✕	○	Off	-	-	R	NO	Off
M1152	Setting up DHSZ instruction as frequency control mode	✕	✕	○	Off	-	-	R/W	NO	Off
M1153	DHSZ frequency control mode has been executed.	✕	✕	○	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1154*	Enabling the deceleration function of adjustable pulse output Y0	✕	○	✕	Off	-	-	R/W	NO	Off
M1161	8-bit mode On: in 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
M1165	When On, the program and password on flash will be copied to the PLC when PLC is powered. (Not available in EH)	✕	✕	○	-	-	-	R/W	YES	Off
M1166	When On, the recipe on flash will be copied to the PLC when PLC is powered. (Not available in EH)	✕	✕	○	-	-	-	R/W	YES	Off
M1167	16-bit mode for HKY input	✕	○	○	Off	-	-	R/W	NO	Off
M1168	Designating work mode of SMOV	✕	○	○	Off	-	-	R/W	NO	Off
M1169	Selecting PWD modes	✕	✕	○	Off	-	-	R/W	NO	Off
M1170*	Enabling single step execution	✕	✕	○	Off	-	-	R/W	NO	Off
M1171*	Single step execution	✕	✕	○	Off	-	-	R/W	NO	Off
M1172*	Switch for 2-phase pulse output On: switch on	✕	○	✕	Off	Off	Off	R/W	NO	Off
M1173*	On: continuous output	✕	○	✕	Off	-	-	R/W	NO	Off
M1174*	The number of output pulses reaches the target	✕	○	✕	Off	Off	-	R/W	NO	Off
M1175	Losing PLC parameter data (not available in EH)	✕	✕	○	-	-	-	R	YES	Off
M1176	Losing the data in PLC program (not available in EH)	✕	✕	○	-	-	-	R	YES	Off
M1178*	Enabling VR0 rotary switch	✕	○	○	Off	-	-	R/W	NO	Off
M1179*	Enabling VR1 rotary switch	✕	○	○	Off	-	-	R/W	NO	Off
M1184*	Enabling modem function (not available in SV)	✕	✕	○	Off	-	-	R/W	NO	Off
M1185*	Enabling initialization of modem (not available in SV)	✕	✕	○	Off	-	-	R/W	NO	Off
M1186*	Initialization of modem fails (not available in SV)	✕	✕	○	Off	-	-	R/W	NO	Off
M1187*	Initialization of modem is completed (not available in SV)	✕	✕	○	Off	-	-	R/W	NO	Off
M1188*	Displaying whether modem is connecting currently (not available in SV)	✕	✕	○	Off	-	-	R/W	NO	Off
M1196	Setting up the content type in the display (for SX) On: hex; Off: decimal	✕	○	✕	Off	-	-	R/W	NO	Off
M1197	Setting up the display of the 100ths digit (for SX)	✕	○	✕	Off	-	-	R/W	NO	Off
M1198	Setting up the display of the 10ths digit (for SX)	✕	○	✕	Off	-	-	R/W	NO	Off
M1200	Counting mode of C200 (On: counting down)	✕	○	○	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1201	Counting mode of C201 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1202	Counting mode of C202 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1203	Counting mode of C203 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1204	Counting mode of C204 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1205	Counting mode of C205 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1206	Counting mode of C206 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1207	Counting mode of C207 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1208	Counting mode of C208 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1209	Counting mode of C209 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1210	Counting mode of C210 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1211	Counting mode of C211 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1212	Counting mode of C212 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1213	Counting mode of C213 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1214	Counting mode of C214 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1215	Counting mode of C215 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1216	Counting mode of C216 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1217	Counting mode of C217 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1218	Counting mode of C218 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1219	Counting mode of C219 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1220	Counting mode of C220 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1221	Counting mode of C221 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1222	Counting mode of C222 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1223	Counting mode of C223 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1224	Counting mode of C224 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1225	Counting mode of C225 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1226	Counting mode of C226 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1227	Counting mode of C227 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1228	Counting mode of C228 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1229	Counting mode of C229 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1230	Counting mode of C230 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1231	Counting mode of C231 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1232	Counting mode of C232 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off
M1233	Counting mode of C233 (On: counting down)	✕	◯	◯	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1234	Counting mode of C234 (On: counting down)	✕	○	○	Off	-	-	R/W	NO	Off
M1235	Counting mode of C235 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1236	Counting mode of C236 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1237	Counting mode of C237 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1238	Counting mode of C238 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1239	Counting mode of C239 (On: counting down)	✕	○	○	Off	-	-	R/W	NO	Off
M1240	Counting mode of C240 (On: counting down)	✕	○	○	Off	-	-	R/W	NO	Off
M1241	Counting mode of C241 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1242	Counting mode of C242 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1243	Counting mode of C243 (On: counting down)	✕	○	○	Off	-	-	R/W	NO	Off
M1244	Counting mode of C244 (On: counting down)	○	○	○	Off	-	-	R/W	NO	Off
M1245	Counting mode of C245 (On: counting down)	✕	○	✕	Off	-	-	R/W	NO	Off
M1246	C246 counter monitoring (On: counting down)	○	○	○	Off	-	-	R	NO	Off
M1247	C247 counter monitoring (On: counting down)	○	○	○	Off	-	-	R	NO	Off
M1248	C248 counter monitoring (On: counting down)	✕	✕	○	Off	-	-	R	NO	Off
M1249	C249 counter monitoring (On: counting down)	○	○	○	Off	-	-	R	NO	Off
M1250	C250 counter monitoring (On: counting down)	✕	○	✕	Off	-	-	R	NO	Off
M1251	C251 counter monitoring (On: counting down)	○	○	○	Off	-	-	R	NO	Off
M1252	C252 counter monitoring (On: counting down)	○	○	○	Off	-	-	R	NO	Off
M1253	C253 counter monitoring (On: counting down)	✕	✕	○	Off	-	-	R	NO	Off
M1254	C254 counter monitoring (On: counting down)	○	○	○	Off	-	-	R	NO	Off
M1255	C255 counter monitoring (On: counting down)	✕	○	✕	Off	-	-	R	NO	Off
M1258	Y0 pulse output signal reversing for PWM instruction	✕	✕	○	Off	-	-	R/W	NO	Off
M1259	Y2 pulse output signal reversing for PWM instruction	✕	✕	○	Off	-	-	R/W	NO	Off
M1260	X5 as the reset input signal for all high-speed counters	✕	○	✕	Off	-	-	R/W	NO	Off
M1261	High-speed comparator comparison flag for DHSCR instruction	✕	✕	○	Off	-	-	R/W	NO	Off
M1264	Enabling reset function of HHSC0	✕	✕	○	Off	-	-	R/W	NO	Off
M1265	Enabling start function of HHSC0	✕	✕	○	Off	-	-	R/W	NO	Off
M1266	Enabling reset function of HHSC1	✕	✕	○	Off	-	-	R/W	NO	Off
M1267	Enabling start function of HHSC1	✕	✕	○	Off	-	-	R/W	NO	Off
M1268	Enabling reset function of HHSC2	✕	✕	○	Off	-	-	R/W	NO	Off
M1269	Enabling start function of HHSC2	✕	✕	○	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1270	Enabling reset function of HHSC3	✕	✕	◯	Off	-	-	R/W	NO	Off
M1271	Enabling start function of HHSC3	✕	✕	◯	Off	-	-	R/W	NO	Off
M1272	Reset control of HHSC0	✕	✕	◯	Off	-	-	R/W	NO	Off
M1273	Start control of HHSC0	✕	✕	◯	Off	-	-	R/W	NO	Off
M1274	Reset control of HHSC1	✕	✕	◯	Off	-	-	R/W	NO	Off
M1275	Start control of HHSC1	✕	✕	◯	Off	-	-	R/W	NO	Off
M1276	Reset control of HHSC2	✕	✕	◯	Off	-	-	R/W	NO	Off
M1277	Start control of HHSC2	✕	✕	◯	Off	-	-	R/W	NO	Off
M1278	Reset control of HHSC3	✕	✕	◯	Off	-	-	R/W	NO	Off
M1279	Start control of HHSC3	✕	✕	◯	Off	-	-	R/W	NO	Off
M1280	Inhibiting I00□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1281	Inhibiting I10□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1282	Inhibiting I20□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1283	Inhibiting I30□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1284	Inhibiting I40□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1285	Inhibiting I50□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1286	Inhibiting I6□□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1287	Inhibiting I7□□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1288	Inhibiting I8□□	✕	✕	◯	Off	-	-	R/W	NO	Off
M1289	Inhibiting I010	✕	✕	◯	Off	-	-	R/W	NO	Off
M1290	Inhibiting I020	✕	✕	◯	Off	-	-	R/W	NO	Off
M1291	Inhibiting I030	✕	✕	◯	Off	-	-	R/W	NO	Off
M1292	Inhibiting I040	✕	✕	◯	Off	-	-	R/W	NO	Off
M1293	Inhibiting I050	✕	✕	◯	Off	-	-	R/W	NO	Off
M1294	Inhibiting I060	✕	✕	◯	Off	-	-	R/W	NO	Off
M1295	Inhibiting I110	✕	✕	◯	Off	-	-	R/W	NO	Off
M1296	Inhibiting I120	✕	✕	◯	Off	-	-	R/W	NO	Off
M1297	Inhibiting I130	✕	✕	◯	Off	-	-	R/W	NO	Off
M1298	Inhibiting I140	✕	✕	◯	Off	-	-	R/W	NO	Off
M1299	Inhibiting I150	✕	◯	◯	Off	-	-	R/W	NO	Off
M1300	Inhibiting I160	✕	✕	◯	Off	-	-	R/W	NO	Off
M1301	Inhibiting I170	✕	✕	◯	Off	-	-	R/W	NO	Off
M1302	Inhibiting I180	✕	✕	◯	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1303	High/low bits exchange for XCH instruction	✕	◯	◯	Off	-	-	R/W	NO	Off
M1304*	Enabling force On/Off of input point X	✕	◯	◯	Off	-	-	R/W	NO	Off
M1305	Reverse operation of the 1st group pulse CH0 (Y0, Y1) for PLSV/DPLSV/DRVI/DDRVI/DRVA/DDRVA instruction	✕	✕	◯	Off	-	-	R	NO	Off
M1306	Reverse operation of the 2nd group pulse CH1 (Y2, Y3) for PLSV/DPLSV/DRVI/DDRVI/DRVA/DDRVA instruction	✕	✕	◯	Off	-	-	R	NO	Off
M1310*	Immediately disabling Y10 pulse output (for SC_V1.4 and above)	✕	◯	✕	Off	Off	-	R/W	NO	Off
M1311*	Disabling Y11 pulse output (for SC_V1.4 and above)	✕	◯	✕	Off	Off	-	R/W	NO	Off
M1312	Controlling start input point of C235	✕	✕	◯	Off	-	-	R/W	NO	Off
M1313	Controlling start input point of C236	✕	✕	◯	Off	-	-	R/W	NO	Off
M1314	Controlling start input point of C237	✕	✕	◯	Off	-	-	R/W	NO	Off
M1315	Controlling start input point of C238	✕	✕	◯	Off	-	-	R/W	NO	Off
M1316	Controlling start input point of C239	✕	✕	◯	Off	-	-	R/W	NO	Off
M1317	Controlling start input point of C240	✕	✕	◯	Off	-	-	R/W	NO	Off
M1320	Controlling reset input point of C235	✕	✕	◯	Off	-	-	R/W	NO	Off
M1321	Controlling reset input point of C236	✕	✕	◯	Off	-	-	R/W	NO	Off
M1322	Controlling reset input point of C237	✕	✕	◯	Off	-	-	R/W	NO	Off
M1323	Controlling reset input point of C238	✕	✕	◯	Off	-	-	R/W	NO	Off
M1324	Controlling reset input point of C239	✕	✕	◯	Off	-	-	R/W	NO	Off
M1325	Controlling reset input point of C240	✕	✕	◯	Off	-	-	R/W	NO	Off
M1328	Enabling start/reset of C235	✕	✕	◯	Off	-	-	R/W	NO	Off
M1329	Enabling start/reset of C236	✕	✕	◯	Off	-	-	R/W	NO	Off
M1330	Enabling start/reset of C237	✕	✕	◯	Off	-	-	R/W	NO	Off
M1331	Enabling start/reset of C238	✕	✕	◯	Off	-	-	R/W	NO	Off
M1332	Enabling start/reset of C239	✕	✕	◯	Off	-	-	R/W	NO	Off
M1333	Enabling start/reset of C240	✕	✕	◯	Off	-	-	R/W	NO	Off
M1334*	EH/EH2/SV: stopping the 1st group pulse output CH0 (Y0, Y1) SC_V1.4 and above: selecting Y10 pulse output stop modes	✕	◯	◯	Off	-	-	R/W	NO	Off
M1335*	EH/EH2/SV: stopping the 2nd group pulse output CH1 (Y2, Y3) SC_V1.4 and above: selecting Y11 pulse output stop modes	✕	◯	◯	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1336	Sending out the 1st group pulse output CH0 (Y0, Y1)	✕	✕	◯	Off	Off	Off	R	NO	Off
M1337	Sending out the 2nd group pulse output CH1 (Y2, Y3)	✕	✕	◯	Off	Off	Off	R	NO	Off
M1338	Enabling offset pulses of the 1st group pulse output CH0 (Y0, Y1)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1339	Enabling offset pulses of the 2nd group pulse output CH1 (Y2, Y3)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1340	Generating interruption I110 after the 1st group pulse output CH0 (Y0, Y1) is sent out	✕	✕	◯	Off	-	-	R/W	NO	Off
M1341	Generating interruption I120 after the 2nd group pulse output CH1 (Y2, Y3) is sent out	✕	✕	◯	Off	-	-	R/W	NO	Off
M1342	Generating interruption I130 when the 1st group pulse output CH0 (Y0, Y1) is sent out	✕	✕	◯	Off	-	-	R/W	NO	Off
M1343	Generating interruption I140 when the 2nd group pulse output CH1 (Y2, Y3) is sent out	✕	✕	◯	Off	-	-	R/W	NO	Off
M1344	Enabling the offset of the 1st group pulse output CH0 (Y0, Y1)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1345	Enabling the offset of the 2nd group pulse output CH1 (Y2, Y3)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1346	Enabling ZRN CLEAR output signal	✕	✕	◯	Off	-	-	R/W	NO	Off
M1347	Reset after the 1st group pulse output CH0 (Y0, Y1) is completed for PLSY instruction	✕	✕	◯	Off	-	-	R/W	NO	Off
M1348	Reset after the 2nd group pulse output CH1 (Y2, Y3) is completed for PLSY instruction	✕	✕	◯	Off	-	-	R/W	NO	Off
M1350*	Enabling EASY PLC LINK	✕	◯	◯	Off	-	-	R/W	NO	Off
M1351*	Enabling auto mode on EASY PLC LINK	✕	◯	◯	Off	-	-	R/W	NO	Off
M1352*	Enabling manual mode on EASY PLC LINK	✕	◯	◯	Off	-	-	R/W	NO	Off
M1353*	Enable 32 slave unit linkage and up to 100 data length of data exchange on EASY PLC LINK	✕	✕	◯	Off	-	-	R/W	NO	Off
M1354*	Enable simultaneous data read/write in a polling of EASY PLC LINK	✕	◯	◯	Off	-	-	R/W	NO	Off
M1360*	Slave ID#1 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1361*	Slave ID#2 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1362*	Slave ID#3 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1363*	Slave ID#4 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1364*	Slave ID#5 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1365*	Slave ID#6 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1366*	Slave ID#7 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1367*	Slave ID#8 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1368*	Slave ID#9 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1369*	Slave ID#10 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1370*	Slave ID#11 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1371*	Slave ID#12 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1372*	Slave ID#13 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1373*	Slave ID#14 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1374*	Slave ID#15 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1375*	Slave ID#16 status on EASY PLC LINK network	✕	◯	◯	Off	-	-	R	NO	Off
M1376*	Indicating Slave ID#1 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1377*	Indicating Slave ID#2 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1378*	Indicating Slave ID#3 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1379*	Indicating Slave ID#4 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1380*	Indicating Slave ID#5 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1381*	Indicating Slave ID#6 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1382*	Indicating Slave ID#7 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1383*	Indicating Slave ID#8 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1384*	Indicating Slave ID#9 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1385*	Indicating Slave ID#10 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1386*	Indicating Slave ID#11 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1387*	Indicating Slave ID#12 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1388*	Indicating Slave ID#13 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1389*	Indicating Slave ID#14 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off
M1390*	Indicating Slave ID#15 data transaction status on EASY PLC LINK	✕	◯	◯	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1391*	Indicating Slave ID#16 data transaction status on EASY 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
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*	Indicating reading from Salve ID#1 is completed	✕	○	○	Off	-	-	R	NO	Off
M1409*	Indicating reading from Salve ID#2 is completed	✕	○	○	Off	-	-	R	NO	Off
M1410*	Indicating reading from Salve ID#3 is completed	✕	○	○	Off	-	-	R	NO	Off
M1411*	Indicating reading from Salve ID#4 is completed	✕	○	○	Off	-	-	R	NO	Off
M1412*	Indicating reading from Salve ID#5 is completed	✕	○	○	Off	-	-	R	NO	Off
M1413*	Indicating reading from Salve ID#6 is completed	✕	○	○	Off	-	-	R	NO	Off
M1414*	Indicating reading from Salve ID#7 is completed	✕	○	○	Off	-	-	R	NO	Off
M1415*	Indicating reading from Salve ID#8 is completed	✕	○	○	Off	-	-	R	NO	Off
M1416*	Indicating reading from Salve ID#9 is completed	✕	○	○	Off	-	-	R	NO	Off
M1417*	Indicating reading from Salve ID#10 is completed	✕	○	○	Off	-	-	R	NO	Off
M1418*	Indicating reading from Salve ID#11 is completed	✕	○	○	Off	-	-	R	NO	Off
M1419*	Indicating reading from Salve ID#12 is completed	✕	○	○	Off	-	-	R	NO	Off
M1420*	Indicating reading from Salve ID#13 is completed	✕	○	○	Off	-	-	R	NO	Off
M1421*	Indicating reading from Salve ID#14 is completed	✕	○	○	Off	-	-	R	NO	Off
M1422*	Indicating reading from Salve ID#15 is completed	✕	○	○	Off	-	-	R	NO	Off
M1423*	Indicating reading from Salve ID#16 is completed	✕	○	○	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1424*	Indicating writing to Salve ID#1 is completed	✕	○	○	Off	-	-	R	NO	Off
M1425*	Indicating writing to Salve ID#2 is completed	✕	○	○	Off	-	-	R	NO	Off
M1426*	Indicating writing to Salve ID#3 is completed	✕	○	○	Off	-	-	R	NO	Off
M1427*	Indicating writing to Salve ID#4 is completed	✕	○	○	Off	-	-	R	NO	Off
M1428*	Indicating writing to Salve ID#5 is completed	✕	○	○	Off	-	-	R	NO	Off
M1429*	Indicating writing to Salve ID#6 is completed	✕	○	○	Off	-	-	R	NO	Off
M1430*	Indicating writing to Salve ID#7 is completed	✕	○	○	Off	-	-	R	NO	Off
M1431*	Indicating writing to Salve ID#8 is completed	✕	○	○	Off	-	-	R	NO	Off
M1432*	Indicating writing to Salve ID#9 is completed	✕	○	○	Off	-	-	R	NO	Off
M1433*	Indicating writing to Salve ID#10 is completed	✕	○	○	Off	-	-	R	NO	Off
M1434*	Indicating writing to Salve ID#11 is completed	✕	○	○	Off	-	-	R	NO	Off
M1435*	Indicating writing to Salve ID#12 is completed	✕	○	○	Off	-	-	R	NO	Off
M1436*	Indicating writing to Salve ID#13 is completed	✕	○	○	Off	-	-	R	NO	Off
M1437*	Indicating writing to Salve ID#14 is completed	✕	○	○	Off	-	-	R	NO	Off
M1438*	Indicating writing to Salve ID#15 is completed	✕	○	○	Off	-	-	R	NO	Off
M1439*	Indicating writing to Salve ID#16 is completed	✕	○	○	Off	-	-	R	NO	Off
M1440*	Slave ID#17 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1441*	Slave ID#18 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1442*	Slave ID#19 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1443*	Slave ID#20 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1444*	Slave ID#21 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1445*	Slave ID#22 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1446*	Slave ID#23 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1447*	Slave ID#24 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1448*	Slave ID#25 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1449*	Slave ID#26 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1450*	Slave ID#27 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1451*	Slave ID#28 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1452	Slave ID#29 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1453*	Slave ID#30 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1454*	Slave ID#31 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1455*	Slave ID#32 status on EASY PLC LINK network	✕	✕	○	Off	-	-	R	NO	Off
M1456*	Indicating Slave ID#17 data transaction status on EASY PLC LINK	✕	✕	○	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1457*	Indicating Slave ID#18 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1458*	Indicating Slave ID#19 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1459*	Indicating Slave ID#20 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1460*	Indicating Slave ID#21 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1461*	Indicating Slave ID#22 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1462*	Indicating Slave ID#23 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1463*	Indicating Slave ID#24 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1464*	Indicating Slave ID#25 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1465*	Indicating Slave ID#26 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1466*	Indicating Slave ID#27 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1467*	Indicating Slave ID#28 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1468*	Indicating Slave ID#29 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1469*	Indicating Slave ID#30 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Default
M1470*	Indicating Slave ID#31 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1471*	Indicating Slave ID#32 data transaction status on EASY PLC LINK	✕	✕	◯	Off	-	-	R	NO	Off
M1472*	Slave ID#17 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1473*	Slave ID#18 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1474*	Slave ID#19 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1475*	Slave ID#20 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1476*	Slave ID#21 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1477*	Slave ID#22 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1478*	Slave ID#23 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1479*	Slave ID#24 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1480*	Slave ID#25 linking error	✕	✕	◯	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1481*	Slave ID#26 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1482*	Slave ID#27 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1483*	Slave ID#28 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1484*	Slave ID#29 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1485*	Slave ID#30 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1486*	Slave ID#31 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1487*	Slave ID#32 linking error	✕	✕	◯	Off	-	-	R	NO	Off
M1488*	Indicating reading from Salve ID#17 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1489*	Indicating reading from Salve ID#18 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1490*	Indicating reading from Salve ID#19 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1491*	Indicating reading from Salve ID#20 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1492*	Indicating reading from Salve ID#21 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1493*	Indicating reading from Salve ID#22 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1494*	Indicating reading from Salve ID#23 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1495*	Indicating reading from Salve ID#24 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1496*	Indicating reading from Salve ID#25 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1497*	Indicating reading from Salve ID#26 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1498*	Indicating reading from Salve ID#27 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1499*	Indicating reading from Salve ID#28 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1500*	Indicating reading from Salve ID#29 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1501*	Indicating reading from Salve ID#30 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1502*	Indicating reading from Salve ID#31 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1503*	Indicating reading from Salve ID#32 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1504*	Indicating writing to Salve ID#17 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1505*	Indicating writing to Salve ID#18 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1506*	Indicating writing to Salve ID#19 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1507*	Indicating writing to Salve ID#20 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1508*	Indicating writing to Salve ID#21 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1509*	Indicating writing to Salve ID#22 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1510*	Indicating writing to Salve ID#23 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1511*	Indicating writing to Salve ID#24 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1512*	Indicating writing to Salve ID#25 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1513*	Indicating writing to Salve ID#26 is completed	✕	✕	◯	Off	-	-	R	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1514*	Indicating writing to Salve ID#27 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1515*	Indicating writing to Salve ID#28 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1516*	Indicating writing to Salve ID#29 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1517*	Indicating writing to Salve ID#30 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1518*	Indicating writing to Salve ID#31 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1519*	Indicating writing to Salve ID#32 is completed	✕	✕	◯	Off	-	-	R	NO	Off
M1520	EH2/SV: stopping the 3rd group pulse output CH2 (Y4, Y5) (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1521	EH2/SV: stopping the 4th group pulse output CH3 (Y6, Y7) (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1522	EH2/SV: sending out the 3rd group pulse output CH2 (Y4, Y5) (Not available in EH)	✕	✕	◯	Off	-	Off	R	NO	Off
M1523	EH2/SV: sending out the 4th group pulse output CH3 (Y6, Y7) (Not available in EH)	✕	✕	◯	Off	-	Off	R	NO	Off
M1524	EH2/SV: reset after the 3rd group pulse output CH2 (Y4, Y5) is completed for PLSY instruction (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1525	EH2/SV: reset after the 4th group pulse output CH3 (Y6, Y7) is completed for PLSY instruction (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1526	EH2/SV: reversing Y4 pulse output signal for PWM instruction (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1527	EH2/SV: reversing Y6 pulse output signal for PWM instruction (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1530	EH2/SV: switching time base unit of Y4 output for PWM instruction On: 100us; Off: 1ms (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1531	EH2/SV: switching time base unit of Y6 output for PWM instruction On: 100us; Off: 1ms (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1532	EH2/SV: reverse operation of the 3rd group pulse CH2 (Y4, Y5) for PLSV/DPLSV/DRVI/DDRVI/DRVA /DDRVA instruction (Not available in EH)	✕	✕	◯	Off	-	-	R/W	NO	Off
M1533	EH2/SV: reverse operation of the 4th group pulse CH3 (Y6, Y7) for PLSV/DPLSV/DRVI/DDRVI/DRVA /DDRVA instruction	✕	✕	◯	Off	-	-	R/W	NO	Off
M1534	EH2/SV: CH0 being able to designate deceleration time. Has to be used with D1348.	✕	✕	◯	Off	-	-	R/W	NO	Off
M1535	EH2/SV: CH1 being able to designate deceleration time. Has to be used with D1349.	✕	✕	◯	Off	-	-	R/W	NO	Off

Special M	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
M1536	EH2/SV: CH2 being able to designate deceleration time. Has to be used with D1350.	✕	✕	○	Off	-	-	R/W	NO	Off
M1537	EH2/SV: CH3 being able to designate deceleration time. Has to be used with D1351.	✕	✕	○	Off	-	-	R/W	NO	Off

Special D	Function		ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1000*	Scanning watchdog timer (WDT) (Unit: ms)		○	○	○	200	-	-	R/W	NO	200
D1001	Displaying the program version of DVP-PLC (initial factory setting)	EH	○	○	○	#	#	#	R	NO	#
		ES/SA				-	-	-			
D1002*	Program capacity # -> EH:15,872; SA: 7,920; ES: 3,792		○	○	○	#	-	-	R	NO	#
D1003	Sum of program memory # -> EH: -15,873; SA: -7,920; ES: -3,792		○	○	○	-	-	-	R	YES	#
D1004*	Syntax check error code		○	○	○	0	0	-	R	NO	0
D1008*	STSC address when WDT is On		○	○	○	0	-	-	R	NO	0
D1009	ES/SA: recording number of occurrances of LV signals EH2/SV: register for SRAM lost data error code		○	○	○	-	-	-	R	YES	0
D1010*	Current scan time (Unit: 0.1ms)		○	○	○	0	-	-	R	NO	0
D1011*	Minimum scan time (Unit: 0.1ms)		○	○	○	0	-	-	R	NO	0
D1012*	Maximum scan time (Unit: 0.1ms)		○	○	○	0	-	-	R	NO	0
D1015*	0~32,767 (unit: 0.1ms) accumulative high-speed timer		✕	○	○	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)		○	○	○	10	-	-	R/W	NO	10
D1021*	ES/EH/EH2/SV: X10 ~ X17 input filter (Unit: ms) SC: X10 ~ X17 input filter (time base: scan cycle), range: 0 ~ 1,000 (Unit: times)		○	○	○	10	-	-	R/W	NO	10
D1022	Multiplied frequency of A-B phase counters for ES/SA		○	○	✕	0	-	-	R/W	NO	0
D1023*	Register for detected pulse width, Unit: 0.1ms (Available in ES/EX/SS_V6.4, SA/SX_V1.6, SC_V1.4 and above)		○	○	✕	0	-	-	R/W	NO	0
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
D1030*	Number of Y0 output pulses (low word)		○	○	✕	0	-	-	R	NO	0
D1031*	Number of Y0 output pulses (high word)		○	○	✕	0	-	-	R	NO	0
D1032	Number of Y1 output pulses (low word)		○	○	✕	0	-	-	R	NO	0
D1033	Number of Y0 output pulses (high word)		○	○	✕	0	-	-	R	NO	0
D1034	Work mode of frequency measurement card		✕	✕	○	-	-	-	R	YES	1
D1035*	No. of input point X as RUN/STOP		✕	✕	○	-	-	-	R/W	YES	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1036*	COM1 communication protocol	○	○	○	H'86	-	-	R/W	NO	H'86
D1037	Repetition time of HKY key	✕	✕	○	-	-	-	R/W	NO	0
D1038*	Delay time of data response when PLC MPU as slave in RS-485 communication, range: 0 ~ 10,000 (unit: 0.1ms) SA/EH: delay time for sending the next communication data in EASY EASY PLC LINK (unit for SA/SX/SC: 1 scan cycle; EH/EH2/SV: 0.1ms)	○	○	○	-	-	-	R/W	YES	0
D1039*	Fixed scan time (ms)	○	○	○	0	-	-	R/W	NO	0
D1040	On status of step No. 1	✕	○	○	0	-	-	R	NO	0
D1041	On status of step No. 2	✕	○	○	0	-	-	R	NO	0
D1042	On status of step No. 3	✕	○	○	0	-	-	R	NO	0
D1043	On status of step No. 4	✕	○	○	0	-	-	R	NO	0
D1044	On status of step No. 5	✕	○	○	0	-	-	R	NO	0
D1045	On status of step No. 6	✕	○	○	0	-	-	R	NO	0
D1046	On status of step No. 7	✕	○	○	0	-	-	R	NO	0
D1047	On status of step No. 8	✕	○	○	0	-	-	R	NO	0
D1049	No. of alarm On	✕	○	○	0	-	-	R	NO	0
D1050 ↓ D1055	Process of data for Modbus communication instruction. PLC automatically converts the ASCII data in D1070 ~ D1085 in to hex data.	○	○	○	0	-	-	R	NO	0
D1056*	Present value at analog input channel CH0 in SX/EX or at CH0 on AD card in EH/EH2	○	○	○	0	-	-	R	NO	0
D1057*	Present value at analog input channel CH1 in SX/EX or at CH1 on AD card in EH/EH2	○	○	○	0	-	-	R	NO	0
D1058*	Present value at analog input channel CH2 in EX	○	✕	✕	0	-	-	R	NO	0
D1059*	Present value at analog input channel CH3 in EX	○	✕	✕	0	-	-	R	NO	0
D1061	System error message: number of errors recorded in latched area	○	✕	✕	-	-	-	R	YES	0
D1062	Average times of AD0, AD1 in SX (2 ~ 4 times)	✕	○	✕	2	-	-	R/W	NO	2
D1067*	Error code for operational error	○	○	○	0	0	-	R	NO	0
D1068*	Locking the address of operational error	○	○	○	0	-	-	R	NO	0
D1070 ↓ D1085	Process of data for Modbus communication instruction. When the RS-485 communication instruction built-in the PLC sent out is received, the response messages will be stored in D1070 ~ D1085. You can view the response messages by checking these registers.	○	○	○	0	-	-	R	NO	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1086	High word of the set password in DVP-PCC01 (displayed in hex corresponded by its ASCII characters)	○	○	○	0	-	-	R/W	NO	0
D1087	Low word of the set password in DVP-PCC01 (displayed in hex corresponded by its ASCII characters)	○	○	○	0	-	-	R/W	NO	0
D1089 ↓ D1099	Process of data for Modbus communication instruction. When the RS-485 communication instruction built-in the PLC is executed, the words of the instruction will be stored in D1089 ~ D1099. You can check whether the instruction is correct by the contents in these registers.	○	○	○	0	-	-	R	NO	0
D1100	Corresponding status after LV signal is enabled	✕	✕	○	0	-	-	R/W	NO	0
D1101*	Start address of file registers	✕	○	○	-	-	-	R/W	YES	0
D1102*	Number of data copied in file register	✕	○	○	-	-	-	R/W	YES	1,600
D1103*	Start No. of file register D for storing data (has to be bigger than 2,000)	✕	○	○	-	-	-	R/W	YES	2,000
D1104*	Start No. of register D for Y0 acceleration/ deceleration pulse output (Not available in SC_V1.4 and above)	○	○	✕	0	-	-	R/W	NO	0
D1109*	COM3 communication protocol	✕	✕	○	0	-	-	R/W	NO	0
D1110*	Average value at analog input channel CH0 in SX/EX or at CH0 on AD card in EH/EH2	○	○	○	0	-	-	R	NO	0
D1111*	Average value at analog input channel CH1 in SX/EX or at CH1 on AD card in EH/EH2	○	○	○	0	-	-	R	NO	0
D1112*	Average value at analog input channel CH2 in EX	○	✕	✕	0	-	-	R	NO	0
D1113*	Average value at analog input channel CH3 in EX	○	✕	✕	0	-	-	R	NO	0
D1116*	CH0 of analog output in SX/EX CH0 of DA card in EH/EH2	○	○	○	0	0	0	R/W	NO	0
D1117*	CH1 of analog output in SX/EX CH1 of DA card in EH/EH2	○	○	○	0	0	0	R/W	NO	0
D1118*	Sampling time of analog/digital converstion in SX/EX/EH/EH2 (ms)	○	○	○	5	-	-	R/W	NO	5
D1120*	COM2 (RS-485) communication protocol	○	○	○	H'86	-	-	R/W	NO	H'86
D1121	PLC communication address (latched)	○	○	○	-	-	-	R/W	YES	1
D1122	Remaining number of words of sent data	○	○	○	0	0	-	R	NO	0
D1123	Remaining number of words of received data	○	○	○	0	0	-	R	NO	0
D1124	Definition of start word (STX)	○	○	○	H'3A	-	-	R/W	NO	H'3A
D1125	Definition of the first end word	○	○	○	H'0D	-	-	R/W	NO	H'0D
D1126	Defunition of the second end word	○	○	○	H'0A	-	-	R/W	NO	H'0A

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1127	Interruption request for receiving specific word in RS instruction (I150)	○	✕	✕	0	-	-	R/W	NO	0
D1129	Abnormal communication time-out (time: ms)	○	○	○	0	-	-	R/W	NO	0
D1130	Error code returning from Modbus	○	○	○	0	-	-	R	NO	0
D1133*	SA/SX: start No. of register D for Y0 special high-speed pulse output (50KHz) SC_V1.4 and above: start No. of register D for Y10 2-axis synchronous control output	✕	○	✕	0	-	-	R/W	NO	0
D1134*	Number of sections in Y10 output for 2-axis synchronous control (available in SC_V1.4 and above)	✕	○	✕	0	-	-	R/W	NO	0
D1135*	SC_V1.4 and above: start No. of register D for Y11 2-axis synchronous control output	✕	○	✕	0	-	-	R/W	NO	0
D1136*	Number of sections in Y11 output for 2-axis synchronous control (available in SC_V1.4 and above)	✕	○	✕	0	-	-	R/W	NO	0
D1137*	Address where incorrect use of operand occurs	○	○	○	0	0	-	R	NO	0
D1140*	Number of right-side special extension modules (max. 8)	○	○	○	0	-	-	R	NO	0
D1142*	Number of points X in digital extension unit	○	○	○	0	-	-	R	NO	0
D1143*	Number of points Y in digital extension unit	○	○	○	0	-	-	R	NO	0
D1144*	Start No. of register D for Y0 adjustable acceleration/deceleration pulse output	✕	○	✕	0	-	-	R/W	NO	0
D1145*	Number of left-side special extension modules (max. 8) (only available in SV)	✕	✕	○	0	-	-	R	NO	0
D1147	Type of memory card b0 = 0: no card existing (H0000) b0 = 1: with flash card b8 = 0: flash card Off (H0001) b8 = 1: flash card On (H0101)	✕	✕	○	0	-	-	R	NO	0
D1149	Type of function extension card 0: no card 1: RS-232 card, DU-01 2: RS-422 card 3: COM3 card 4: analog rotary switch card 5: digital switch card 6: transistor output card 8: 2AD card 9: 2DA card 10: frequency measurement card	✕	✕	○	0	-	-	R	NO	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1150	Table count register in multi-group setting comparison mode of DHSZ commnad	✕	✕	○	0	0	0	R	NO	0
D1151	Table counting register for DHSZ multiple set values comparison mode	✕	✕	○	0	0	0	R	NO	0
D1152	High word of changed D value for DHSZ instruciton	✕	✕	○	0	0	0	R	NO	0
D1153	Low word of changed D value for DHSZ instruction	✕	✕	○	0	0	0	R	NO	0
D1154*	Suggested deceleration time interval (10 ~ 32,767ms) for adjustable acceleration/deceleration pulse output Y0	✕	○	✕	200	-	-	R/W	NO	200
D1155*	Suggested deceleration frequency (-1 ~ -32,700Hz) for adjustable acceleration/deceleration pulse output Y0	✕	○	✕	-1,000	-	-	R/W	NO	-1,000
D1156 ↓ D1165	Designated special D for RTMU, RTMD instructions	✕	✕	○	0	-	-	R/W	NO	0
D1166*	Switching rising-/falling-edge counting mode of X10 (available in SC_V1.4 and above)	✕	○	✕	0	-	-	R/W	NO	0
D1167*	Switching rising-/falling-edge counting mode of X11 (available in SC_V1.4 and above)	✕	○	✕	0	-	-	R/W	NO	0
D1168	Interruption request for receiving specific word in RS instruction (I150)	✕	○	○	0	-	-	R/W	NO	0
D1169	Interruption request for receiving specific word in RS instruction (I160)	✕	✕	○	0	-	-	R/W	NO	0
D1170*	PC value when executing single step	✕	✕	○	0	0	0	R	NO	0
D1172*	Frequency of 2-phase pulse output (12Hz ~ 20KHz)	✕	○	✕	0	-	-	R/W	NO	0
D1173*	Modes of 2-phase pulse output (K1 and K2)	✕	○	✕	0	-	-	R/W	NO	0
D1174*	Low 16 bits of target numbers of 2-phase output pulses	✕	○	✕	0	-	-	R/W	NO	0
D1175*	High 16 bits of target numbers of 2-phase output pulses	✕	○	✕	0	-	-	R/W	NO	0
D1176*	Low 16 bits of current numbers of 2-phase output pulses	✕	○	✕	0	-	-	R/W	NO	0
D1177*	High 16 bits of current numbers of 2-phase output pulses	✕	○	✕	0	-	-	R/W	NO	0
D1178*	VR0 value	✕	○	○	0	-	-	R	NO	0
D1179*	VR1 value	✕	○	○	0	-	-	R	NO	0
D1180*	Low 16 bytes of high-speed counter value extracted by interruption I401	✕	○	✕	0	-	-	R	NO	0
D1181*	High 16 bytes of high-speed counter value extracted by interruption I401	✕	○	✕	0	-	-	R	NO	0
D1182	Index register E1	✕	○	○	0	-	-	R/W	NO	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
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
D1196	Content in the display (available in SX)	✕	○	✕	0	-	-	R/W	NO	0
D1198*	Low 16 bytes of high-speed counter value extracted by interruption I501	✕	○	✕	0	-	-	R	NO	0
D1199*	Low 16 bytes of high-speed counter value extracted by interruption I501	✕	○	✕	0	-	-	R	NO	0
D1200*	Start latched address for auxiliary relays M0 ~ M999 # -> EH: 500; SA/SX/SC: 512	✕	○	○	-	-	-	R/W	YES	#
D1201*	End latched address for auxiliary relays M0 ~ M999	✕	○	○	-	-	-	R/W	YES	999
D1202*	Start latched address for auxiliary relays M2000 ~ M4095	✕	○	○	-	-	-	R/W	YES	2,000
D1203*	End latched address for auxiliary relays M2000 ~ M4095	✕	○	○	-	-	-	R/W	YES	4,095
D1204*	Start latched address for 100ms timers T0 ~ T199	✕	✕	○	-	-	-	R/W	YES	H'FFFF
D1205*	End latched address for 100ms timers T0 ~ T199	✕	✕	○	-	-	-	R/W	YES	H'FFFF
D1206*	Start latched address for 10ms timers T200 ~ T239	✕	✕	○	-	-	-	R/W	YES	H'FFFF
D1207*	End latched address for 10ms timers T200 ~ T239	✕	✕	○	-	-	-	R/W	YES	H'FFFF
D1208*	Start latched address for16-bit counters C0 ~ C199 # -> EH: 100; SA/SX/SC: 96	✕	○	○	-	-	-	R/W	YES	#
D1209*	End latched address for 16-bit counters C0 ~ C199	✕	○	○	-	-	-	R/W	YES	199
D1210*	Start latched address for 32-bit counters C200 ~ C234 # -> EH: 220; SA/SX/SC: 216	✕	○	○	-	-	-	R/W	YES	#
D1211*	End latched address for 32-bit counters C200 ~ C234	✕	○	○	-	-	-	R/W	YES	234

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1212*	Start latched address for 32-bit high-speed counters C235 ~ C255	✕	○	○	-	-	-	R/W	YES	235
D1213*	End latched address for 32-bit high-speed counters C235 ~ C255	✕	○	○	-	-	-	R/W	YES	255
D1214*	Start latched address for steps S0 ~ S899 # -> EH: 500; SA/SX/SC: 512	✕	○	○	-	-	-	R/W	YES	#
D1215*	End latched address for steps S0 ~ S899 # -> EH: 899; SA/SX/SC: 895	✕	○	○	-	-	-	R/W	YES	#
D1216*	Start latched address for registers D0 ~ D999	✕	○	○	-	-	-	R/W	YES	200
D1217*	End latched address for registers D0 ~ D999	✕	○	○	-	-	-	R/W	YES	999
D1218*	Start latched address for registers D2000 ~ D9999	✕	○	○	-	-	-	R/W	YES	2,000
D1219*	End latched address for registers D2000 ~ D9999	✕	○	○	-	-	-	R/W	YES	#
D1220	Phase of the 1st group pulse output CH0 (Y0, Y1)	✕	✕	○	0	-	-	R/W	NO	0
D1221	Phase of the 2nd group pulse output CH1 (Y2, Y3)	✕	✕	○	0	-	-	R/W	NO	0
D1222	Time difference between direction signal and pulse output for the 1st group pulse CH0 (Y0, Y1) in DRVI, DDRVI, DRVA, DDRVA, PLSV, DPLSV	✕	✕	○	0	-	-	R/W	NO	0
D1223	Time difference between direction signal and pulse output for the 2nd group pulse CH1 (Y2, Y3) in DRVI, DDRVI, DRVA, DDRVA, PLSV, DPLSV	✕	✕	○	0	-	-	R/W	NO	0
D1225	Counting mode of the counter HHSC0	✕	✕	○	2	-	-	R/W	NO	2
D1226	Counting mode of the counter HHSC1	✕	✕	○	2	-	-	R/W	NO	2
D1227	Counting mode of the counter HHSC2	✕	✕	○	2	-	-	R/W	NO	2
D1228	Counting mode of the counter HHSC3	✕	✕	○	2	-	-	R/W	NO	2
D1129	Phase of the 3rd group pulse output CH2 (Y4, Y5) (available in EH2/SV)	✕	✕	○	0	-	-	R/W	NO	0
D1130	Phase of the 4th group pulse output CH3 (Y6, Y7) (available in EH2/SV)	✕	✕	○	0	-	-	R/W	NO	0
D1256 ↓ D1295	When the RS-485 communication instruction MODRW built-in the PLC is executed, the words of sent out by the instruction will be stored in D1256 ~ D1259. You can check whether the instruction is correct by the contents in these registers.	○	○	○	0	-	-	R	NO	0
D1296 ↓ D1311	The RS-485 communication instruction MODRW built in the PLC automatically converts the ASCII data received in the designated register into hex and store the hex data into D1296 ~ D1311.	○	○	○	0	-	-	R	NO	0
D1313*	Second in RTC: 00 ~ 59 #: read RTC and write	✕	○	○	#	-	-	R/W	NO	0
D1314*	Minute in RTC: 00 ~ 59 #: read RTC and write	✕	○	○	#	-	-	R/W	NO	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1315*	Hour in RTC: 00 ~ 23 #: read RTC and write	✕	○	○	#	-	-	R/W	NO	0
D1316*	Day in RTC: 01 ~ 31 #: read RTC and write	✕	○	○	#	-	-	R/W	NO	1
D1317*	Month in RTC: 01 ~ 12 #: read RTC and write	✕	○	○	#	-	-	R/W	NO	1
D1318*	Week in RTC: 1 ~ 7 #: read RTC and write	✕	○	○	#	-	-	R/W	NO	6
D1319*	Year in RTC: 00 ~ 99 (A.D.) #: read RTC and write	✕	○	○	#	-	-	R/W	NO	0
D1320*	ID of the 1st right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1321*	ID of the 2nd right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1322*	ID of the 3rd right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1323*	ID of the 4th right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1324*	ID of the 5th right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1325*	ID of the 6th right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1326*	ID of the 7th right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1327*	ID of the 8th right-side extension module (available in EH/EH2)	✕	✕	○	0	-	-	R	NO	0
D1328	Low word of offset pulse the 1st group pulses CH0 (Y0, Y1)	✕	✕	○	0	-	-	R/W	NO	0
D1329	High word of offset pulse the 1st group pulses CH0 (Y0, Y1)	✕	✕	○	0	-	-	R/W	NO	0
D1330	Low word of offset pulse the 2nd group pulses CH1 (Y2, Y3)	✕	✕	○	0	-	-	R/W	NO	0
D1331	High word of offset pulse the 2nd group pulses CH1 (Y2, Y3)	✕	✕	○	0	-	-	R/W	NO	0
D1332	Low word of the remaining number of pulses of the 1st group pulses CH0 (Y0, Y1)	✕	✕	○	0	-	-	R	NO	0
D1333	High word of the remaining number of pulses of the 1st group pulses CH0 (Y0, Y1)	✕	✕	○	0	-	-	R	NO	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1334	Low word of the remaining number of pulses of the 2nd group pulses CH1 (Y2, Y3)	✕	✕	○	0	-	-	R	NO	0
D1335	High word of the remaining number of pulses of the 2nd group pulses CH1 (Y2, Y3)	✕	✕	○	0	-	-	R	NO	0
D1336	Low word of the present value of the 1st group pulses CH0 (Y0, Y1) (EH2/SV are latached above EH_V1.2)	✕	✕	○	-	-	-	R	YES	0
D1337	High word of the present value of the 1st group pulses CH0 (Y0, Y1) (EH2/SV are latached above EH_V1.2)	✕	✕	○	-	-	-	R	YES	0
D1338	Low word of the present value of the 2nd group pulses CH1 (Y2, Y3) (EH2/SV are latached above EH_V1.2)	✕	✕	○	-	-	-	R	YES	0
D1339	High word of the present value of the 2nd group pulses CH1 (Y2, Y3) (EH_V1.2 and above are latached)	✕	✕	○	-	-	-	R	YES	0
D1340	EH/EH2/SV: start/end frequency of the 1st group pulse output CH0 (Y0, Y1)	✕	○	○	-	-	-	R/W	YES	200
D1340	SC: start/end frequency of Y10 output				200				NO	
D1341	Low word of max. output frequency — Fixed as 200KHz	✕	✕	○	-	-	-	R	YES	H'04D0
D1342	High word of max. output frequency — Fixed as 200KHz	✕	✕	○	-	-	-	R	YES	3
D1343*	EH: acceleration/deceleration time for the 1st group pulse output CH0 (Y0, Y1)	✕	○	○	-	-	-	R/W	YES	100
D1343*	SC: acceleration/deceleration time of Y10 output				200				NO	200
D1344	Low word of the number of compensation pulses of the 1st group pulses CH0 (Y0, Y1)	✕	✕	○	-	-	-	R/W	YES	0
D1345	High word of the number of compensation pulses of the 1st group pulses CH0 (Y0, Y1)	✕	✕	○	-	-	-	R/W	YES	0
D1346	Low word of the number of compensation pulses of the 2nd group pulses CH1 (Y2, Y3)	✕	✕	○	-	-	-	R/W	YES	0
D1347	High word of the number of compensation pulses of the 2nd group pulses CH1 (Y2, Y3)	✕	✕	○	-	-	-	R/W	YES	0
D1348	SC: low word of present value of Y10 pulse output	✕	○	✕	0	-	-	R	NO	0
D1348	EH2/SV: CH0 pulse output. When M1534 = On, it refers to the deceleration time	✕	✕	○	-	-	-	R/W	YES	100
D1349	SC: high word of present value of Y10 pulse output	✕	○	✕	0	-	-	R	NO	0
D1349	EH2/SV: CH1 pulse output. When M1535 = On, it refers to the deceleration time	✕	✕	○	-	-	-	R/W	YES	100

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1350	SC: low word of present value of Y11 pulse output	✕	◯	✕	0	-	-	R	NO	0
	EH2/SV: CH2 pulse output. When M1536 = On, it refers to the deceleration time	✕	✕	◯	-	-	-	R/W	YES	100
D1351	SC: high word of present value of Y11 pulse output	✕	◯	✕	0	-	-	R	NO	0
	EH2/SV: CH3 pulse output. When M1537 = On, it refers to the deceleration time	✕	✕	◯	-	-	-	R/W	YES	100
D1352	EH/EH2/SV: start/end frequency of the 2nd group pulse output CH1 (Y2, Y3)	✕	◯	◯	-	-	-	R/W	YES	200
	SC: start/end frequency of Y11 output				200				NO	
D1353*	EH/EH2/SV: acceleration/deceleration time of the 2nd group pulse output CH1 (Y2, Y3)	✕	◯	◯	-	-	-	R/W	YES	100
	SC: acceleration/deceleration time of Y11 output				200				NO	200
D1355*	Starting reference for Master to read from Salve ID#1	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1356*	Starting reference for Master to read from Salve ID#2	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1357*	Starting reference for Master to read from Salve ID#3	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1358*	Starting reference for Master to read from Salve ID#4	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1359*	Starting reference for Master to read from Salve ID#5	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1360*	Starting reference for Master to read from Salve ID#6	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1361*	Starting reference for Master to read from Salve ID#7	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1362*	Starting reference for Master to read from Salve ID#8	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1363*	Starting reference for Master to read from Salve ID#9	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1364*	Starting reference for Master to read from Salve ID#10	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1365*	Starting reference for Master to read from Salve ID#11	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1366*	Starting reference for Master to read from Salve ID#12	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1367*	Starting reference for Master to read from Salve ID#13	✕	◯	◯	-	-	-	R/W	YES	H'1064
D1368*	Starting reference for Master to read from Salve ID#14	✕	◯	◯	-	-	-	R/W	YES	H'1064

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1369*	Starting reference for Master to read from Salve ID#15	✕	○	○	-	-	-	R/W	YES	H'1064
D1370*	Starting reference for Master to read from Salve ID#16	✕	○	○	-	-	-	R/W	YES	H'1064
D1371	Time unit of PWM Y0 pulse output when M1070=On	✕	✕	○	1	-	-	R/W	NO	1
D1372	Time unit of PWM Y2 pulse output when M1071=On	✕	✕	○	1	-	-	R/W	NO	1
D1373	Time unit of PWM Y4 pulse output when M1530=On (available in EH2/SV)	✕	✕	○	1	-	-	R/W	NO	1
D1374	Time unit of PWM Y6 pulse output when M1531=On (available in EH2/SV)	✕	✕	○	1	-	-	R/W	NO	1
D1375	Low word of the present value of the 3rd group pulses CH2 (Y4, Y5) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	0
D1376	High word of the present value of the 3rd group pulses CH2 (Y4, Y5) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	0
D1377	Low word of the present value of the 4th group pulses CH3 (Y6, Y7) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	0
D1378	High word of the present value of the 4th group pulses CH3 (Y6, Y7) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	0
D1379	Start frequency of the 1st section and end frequency of the last section for the 3rd group pulse output CH2 (Y4, Y5) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	200
D1380	Start frequency of the 1st section and end frequency of the last section for the 4th group pulse output CH3 (Y6, Y7) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	200
D1381	Acceleration/deceleration time for the 3rd pulse output CH2 (Y4, Y5) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	100
D1382	Acceleration/deceleration time for the 4th pulse output CH3 (Y6, Y7) (available in EH2/SV)	✕	✕	○	-	-	-	R/W	YES	100
D1386*	ID of the 1st left-side extension module (available in SV)	✕	✕	○	0	-	-	R	NO	0
D1387*	ID of the 2nd left-side extension module (available in SV)	✕	✕	○	0	-	-	R	NO	0
D1388*	ID of the 3rd left-side extension module (available in SV)	✕	✕	○	0	-	-	R	NO	0
D1389*	ID of the 4th left-side extension module (available in SV)	✕	✕	○	0	-	-	R	NO	0
D1390*	ID of the 5th left-side extension module (available in SV)	✕	✕	○	0	-	-	R	NO	0
D1391*	ID of the 6th left-side extension module (available in SV)	✕	✕	○	0	-	-	R	NO	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1392*	ID of the 7th left-side extension module (available in SV)	✕	✕	◯	0	-	-	R	NO	0
D1393*	ID of the 8th left-side extension module (available in SV)	✕	✕	◯	0	-	-	R	NO	0
D1399*	Starting Salve ID designated by EASY PLC LINK	✕	◯	◯	-	-	-	R/W	YES	1
D1415*	Starting reference for Master to write in Salve ID#1	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1416*	Starting reference for Master to write in Salve ID#2	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1417*	Starting reference for Master to write in Salve ID#3	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1418*	Starting reference for Master to write in Salve ID#4	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1419*	Starting reference for Master to write in Salve ID#5	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1420*	Starting reference for Master to write in Salve ID#6	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1421*	Starting reference for Master to write in Salve ID#7	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1422*	Starting reference for Master to write in Salve ID#8	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1423*	Starting reference for Master to write in Salve ID#9	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1424*	Starting reference for Master to write in Salve ID#10	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1425*	Starting reference for Master to write in Salve ID#11	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1426*	Starting reference for Master to write in Salve ID#12	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1427*	Starting reference for Master to write in Salve ID#13	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1428*	Starting reference for Master to write in Salve ID#14	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1429*	Starting reference for Master to write in Salve ID#15	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1430*	Starting reference for Master to write in Salve ID#16	✕	◯	◯	-	-	-	R/W	YES	H'10C8
D1431*	Times of EASY PLC LINK polling cycle	✕	◯	◯	0	-	-	R/W	NO	0
D1432*	Current times of EASY PLC LINK polling cycle	✕	◯	◯	0	-	-	R/W	NO	0
D1433*	Number of salve units linked to EASY PLC LINK	✕	◯	◯	0	-	-	R/W	NO	0
D1434*	Data length to be read on Salve ID#1	✕	◯	◯	-	-	-	R/W	YES	16
D1435*	Data length to be read on Salve ID#2	✕	◯	◯	-	-	-	R/W	YES	16
D1436*	Data length to be read on Salve ID#3	✕	◯	◯	-	-	-	R/W	YES	16
D1437*	Data length to be read on Salve ID#4	✕	◯	◯	-	-	-	R/W	YES	16
D1438*	Data length to be read on Salve ID#5	✕	◯	◯	-	-	-	R/W	YES	16
D1439*	Data length to be read on Salve ID#6	✕	◯	◯	-	-	-	R/W	YES	16
D1440*	Data length to be read on Salve ID#7	✕	◯	◯	-	-	-	R/W	YES	16
D1441*	Data length to be read on Salve ID#8	✕	◯	◯	-	-	-	R/W	YES	16
D1442*	Data length to be read on Salve ID#9	✕	◯	◯	-	-	-	R/W	YES	16
D1443*	Data length to be read on Salve ID#10	✕	◯	◯	-	-	-	R/W	YES	16

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1444*	Data length to be read on Salve ID#11	✕	○	○	-	-	-	R/W	YES	16
D1445*	Data length to be read on Salve ID#12	✕	○	○	-	-	-	R/W	YES	16
D1446*	Data length to be read on Salve ID#13	✕	○	○	-	-	-	R/W	YES	16
D1447*	Data length to be read on Salve ID#14	✕	○	○	-	-	-	R/W	YES	16
D1448*	Data length to be read on Salve ID#15	✕	○	○	-	-	-	R/W	YES	16
D1449*	Data length to be read on Salve 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
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
D1466	Number of pulses required per revolution of motor at CH0 (low word)	✕	✕	○	-	-	-	R	YES	2,000
D1467	Number of pulses required per revolution of motor at CH0 (high word)	✕	✕	○	-	-	-	R	YES	0
D1468	Number of pulses required per revolution of motor at CH1 (low word)	✕	✕	○	-	-	-	R	YES	2,000
D1469	Number of pulses required per revolution of motor at CH1 (high word)	✕	✕	○	-	-	-	R	YES	0
D1470	Distance created for 1 revolution of motor at CH0 (low word)	✕	✕	○	-	-	-	R	YES	1,000
D1471	Distance created for 1 revolution of motor at CH0 (high word)	✕	✕	○	-	-	-	R	YES	0
D1472	Distance created for 1 revolution of motor at CH1 (low word)	✕	✕	○	-	-	-	R	YES	1,000

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1473	Distance created for 1 revolution of motor at CH1 (high word)	✕	✕	◯	-	-	-	R	YES	0
D1474	Machine unit of CH0 movement (low word)	✕	✕	◯	-	-	-	R	YES	0
D1475	Machine unit of CH0 movement (high word)	✕	✕	◯	-	-	-	R	YES	0
D1476	Machine unit of CH1 movement (low word)	✕	✕	◯	-	-	-	R	YES	0
D1477	Machine unit of CH1 movement (high word)	✕	✕	◯	-	-	-	R	YES	0
D1480* ↓ D1495*	Data buffer to store the data read from Slave ID#1 when M1353 = Off. When M1353 = On, the starting D register No. to store data read from ID#1~16 by Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R	YES	0
D1496* ↓ D1511*	Data buffer to store the data written on Slave ID#1 when M1353 = Off. When M1353 = On, the starting D register No. to store data written on ID#1~16 by Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R/W	YES	0
D1512* ↓ D1527*	Data buffer to store the data read from Slave ID#2 when M1353 = Off. When M1353 = On, the starting D register No. to store data read from ID#17~32 by Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R	YES	0
D1528* ↓ D1543*	Data buffer to store the data written on Slave ID#2 when M1353 = Off. When M1353 = On, the starting D register No. to store data written on ID#17~32 by Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R/W	YES	0
D1544* ↓ D1559*	Data buffer to store the data read from Slave ID#3 when M1353 = Off. When M1353 = On, the length of data read from ID#17~32 by Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R	YES	0
D1560* ↓ D1575*	Data buffer to store the data written on Slave ID#2 when M1353 = Off. When M1353 = On, the length of data written on ID#17~32 by Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R/W	YES	0
D1576* ↓ D1591*	Data buffer to store the data read from Slave ID#4 when M1353 = Off. When M1353=On, the starting D register No. to store data read from ID#17~32 by EH Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R	YES	0
D1592* ↓ D1607*	Data buffer to store the data to be written on Slave ID#4 when M1353 = Off. When M1353=On, the starting D register No. to store data to be written on ID#17~32 by EH Master (available in EH/EH2/SV).	✕	◯	◯	-	-	-	R/W	YES	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1608* ↓ D1623*	Data buffer to store the data read from Slave ID#5	✕	◯	◯	-	-	-	R	YES	0
D1624* ↓ D1639*	Data buffer to store the data to be written on Slave ID#5	✕	◯	◯	-	-	-	R/W	YES	0
D1640* ↓ D1655*	Data buffer to store the data read from Slave ID#6	✕	◯	◯	-	-	-	R	YES	0
D1656* ↓ D1671*	Data buffer to store the data to be written on Slave ID#6	✕	◯	◯	-	-	-	R/W	YES	0
D1672* ↓ D1687*	Data buffer to store the data read from Slave ID#7	✕	◯	◯	-	-	-	R	YES	0
D1688* ↓ D1703*	Data buffer to store the data to be written on Slave ID#7	✕	◯	◯	-	-	-	R/W	YES	0
D1704* ↓ D1719*	Data buffer to store the data read from Slave ID#8	✕	◯	◯	-	-	-	R	YES	0
D1720* ↓ D1735*	Data buffer to store the data to be written on Slave ID#8	✕	◯	◯	-	-	-	R/W	YES	0
D1736* ↓ D1751*	Data buffer to store the data read from Slave ID#9	✕	◯	◯	-	-	-	R	YES	0
D1752* ↓ D1767*	Data buffer to store the data to be written on Slave ID#9	✕	◯	◯	-	-	-	R/W	YES	0
D1768* ↓ D1783*	Data buffer to store the data read from Slave ID#10	✕	◯	◯	-	-	-	R	YES	0
D1784* ↓ D1799*	Data buffer to store the data to be written on Slave ID#10	✕	◯	◯	-	-	-	R/W	YES	0
D1800* ↓ D1815*	Data buffer to store the data read from Slave ID#11	✕	◯	◯	-	-	-	R	YES	0

Special D	Function	ES EX SS	SA SX SC	EH EH2 SV	Off ⇩ On	STOP ⇩ RUN	RUN ⇩ STOP	Attribute	Latched	Default
D1816* ↓ D1831*	Data buffer to store the data to be written on Slave ID#11	✕	◯	◯	-	-	-	R/W	YES	0
D1832* ↓ D1847*	Data buffer to store the data read from Slave ID#12	✕	◯	◯	-	-	-	R	YES	0
D1848* ↓ D1863*	Data buffer to store the data to be written on Slave ID#12	✕	◯	◯	-	-	-	R/W	YES	0
D1864* ↓ D1879*	Data buffer to store the data read from Slave ID#13	✕	◯	◯	-	-	-	R	YES	0
D1880* ↓ D1895*	Data buffer to store the data to be written on Slave ID#13	✕	◯	◯	-	-	-	R/W	YES	0
D1896* ↓ D1911*	Data buffer to store the data read from Slave ID#14	✕	◯	◯	-	-	-	R	YES	0
D1912* ↓ D1927*	Data buffer to store the data to be written on Slave ID#14	✕	◯	◯	-	-	-	R/W	YES	0
D1928* ↓ D1943*	Data buffer to store the data read from Slave ID#15	✕	◯	◯	-	-	-	R	YES	0
D1944* ↓ D1959*	Data buffer to store the data to be written on Slave ID#15	✕	◯	◯	-	-	-	R/W	YES	0
D1960* ↓ D1975*	Data buffer to store the data read from Slave ID#16	✕	◯	◯	-	-	-	R	YES	0
D1976* ↓ D1991*	Data buffer to store the data to be written on Slave ID#16	✕	◯	◯	-	-	-	R/W	YES	0

2.11 Functions of Special Auxiliary Relays and Special Registers

Function Group   PLC Operation Flag

Number   M1000 ~ M1003

M1000:

M1000 (A contact) is constantly "On" during operation and detection. When PLC is in RUN status, M1000 remains "On".

```
    M1000
 ───┤ ├───────────────(  Y0  )   PLC is running

 Normally On contact                    ↘
 in PLC RUN                   Keeps being On
```
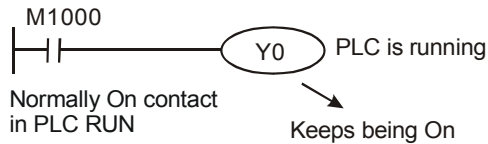
M1001:

M1001 (B contact) is constantly "On" during operation and detection. When PLC is in RUN status, M1001 remains "On".

M1002:

M1002 is "On" during the first scan when PLC starts to RUN and remains "Off" afterward. The pulse width = 1 scan time. Use this contact for all kinds of initial settings.

M1003:

M1003 is "Off" during the first scan when PLC starts to RUN and remains "On" afterward. M1003 enables negative-direction ("Off" immediately when RUN) pulses.

```
PLC RUN ┐      ┌─────────────┐   ┌──────
        └──────┘             └───┘

M1000   ┐      ┌─────────────┐   ┌──────
        └──────┘             └───┘

M1001   ───────┐             ┌───┐
               └─────────────┘   └──────

M1002   ───────┐┐            ───┐┐
               └┘               └┘

M1003   ┐┌─────────────────┐┌──────────
        └┘                 └┘
        →│├←── scan time
```
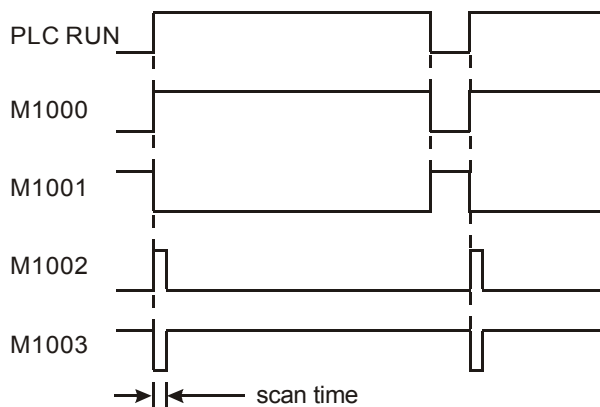
Function Group   Monitor Timer

Number   D1000

Contents:

1.   Monitor timer is used for moitoring PLC scan time. When the scan time exceeds the set time in the moitor timer, the red ERROR LED indicator remains beaconing and all outputs will be "Off".

2.   The initial set value of the time in the monitor timer is 200ms. If the program is long or the operation is too complicated, MOV instruction can be used for changing the set value. See the example below for SV = 300ms.

```
       M1002
  0 ├──┤ ├──────────────────┤ MOV │ K300 │ D1000 │
       Initial pulse
```
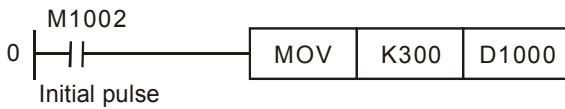
3.   The maximum set value in the monitor timer is 32,767ms. Please be noted that if the SV is too big, the timing of detecting operational errors will be delayed. Therefore, it is suggested that you remain the scan time of shorter than 200ms.

4.   Complicated instruction operations or too many extension modules being connected to the MPU will result in the scan time being too long. Check D1010 ~ D1012 to see if the scan time exceeds the SV in D1000. In this case, besides modifying the SV in D1000, you can also add WDT instruction (API 07) into the PLC program. When the CPU execution progresses to WDT instruction, the internal monitor timer will be cleared as "0" and the scan time will not exceed the set value in the monitor timer.

Function Group   Program Capacity

Number   D1002

Contents:

The program capacity differs in different series of MPUs.

1.   ES/EX/SS series MPU: 3,792 Steps

2.   SA/SX/SC series MPU: 7,920 Steps

3.   EH/EH2/SV series MPU: 15,872 Steps

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 the program is written into PLC.
     c)   When on-line editing is being conducted.

3.   The syntax check may start due to illegal use of instruction operands (devices) or incorrect program syntax loop. The error can be detected by the error code in D1004 and error table. The address where the error exists will be stored in D1137. (The address value in D1137 will be invalid if the error is a general loop error.)

4.   See Chapter 2.12 for error codes for syntax check.

Function Group   Data Backup Memory

Number   M1005~M1007

Contents:

When the data backup memory card is installed in EH/EH2 MPU, MPU will operate according to the On/Off of switch on the card. If the switch is "On", the following comparisons will be conducted and the card will be copied to MPU. If the switch is "Off", MPU will not perform any action.

1. M1005 = On: An error occurs in the comparison between the ciphers of MPU and the data backup memory card and MPU does not perform any action.

2. M1006 = On: The data backup memory card has not been initialized.

3. M1007 = On: Data in the program area of the data backup memory card do not exist, it means data doesn't exist in the program area of data backup memory card.

Function Group   Scan Time-out Timer

Number   M1008, D1008

Contents:

1. M1008 = On: Scan time-out occurs during the execution of the program, and PLC ERROR LED indicator remains beaconing.

2. Users can use WPLSoft or HPP to monitor the content (STEP address when WDT timer is "On").

Function Group   Checking Lost PLC SRAM Data

Number   D1009, M1175, M1176

Contents:

1. bit0 ~ bit7 record the types of data lost. bit = 1 refers to losing data; bit = 0 refers to correct data.

2. What are lost

bit8 ~ 15	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Reserved	PLC program	D register	T register	C register	File register	M relay	S step	password

3. After the PLC is powered, the data in SRAM will be verified. If the SRAM data are lost, the PLC will record the error in D1009 and set on M1175 or M1176 according to the content of the data.

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.

1. D1010: Present scan time value

2. D1011: Minimum scan time value

3. D1012: Maximum scan time value

Function Group   Internal Clock Pulse

Number   M1011 ~ M1014

Contents:

1. All PLC MPUs provide four different clock pulses. When PLC is powered, the four clock pulses will start automatically.
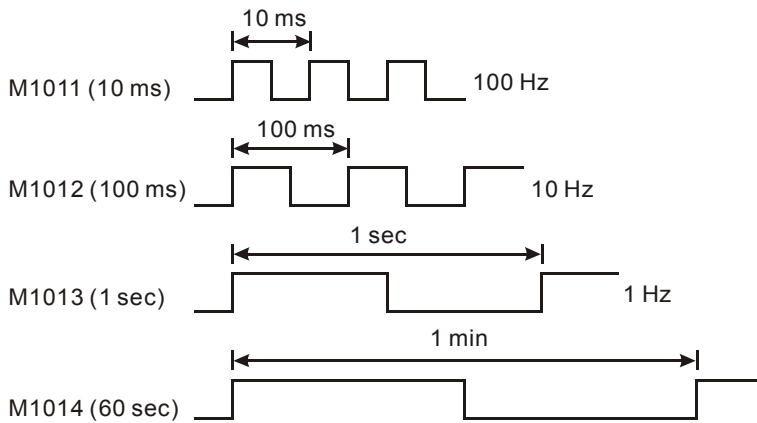
2. The clock pulses also start when PLC is in STOP status. The activation timing of clock pulses and that of RUN will not happen synchronously.

Function Group   High-Speed Timer

Number   M1015, D1015

Contents:

1. The steps for using special M and special D directly:
   a) High-speed counter is valid only when PLC is in RUN status for EH/EH2/SV, but is valid when PLC is in RUN or STOP stauts for SA/SX/SC.
   b) M1015 = On: High-speed counter D1015 is enabled only whtn PLC scans to END instruction. (Min. timing unit of D1015: 100us)
   c) Timing range of D1015: 0~32,767. When the timing reaches 32,767, the next timing restarts from 0.
   d) M1015 = Off: D1015 stops timing immediately.
2. EH/EH2/SV series MPU offers high-speed timer instruction HST. See API 196 HST for more details.
3. Example:
   a) When X10 = On, M1015 will be On. The high-speed timer will start to time and record the present value in D1015.
   b) When X10 = Off, M1015 will be Off. The high-speed timer will be disabled.
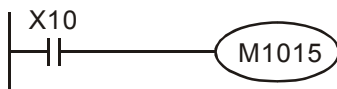
Function Group   Real Time Clock

Number   M1016, M1017, M1076, D1313 ~ D1319

Contents:

1. Special M and special D relevant to RTC

No.	Name	Function
M1016	Year (in A.D.) in RTC	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	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
M1076	RTC malfunction	Set value exceeds the range; dead battery
M1082	Flag change on RTC	On: Modification on RTC
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 the set value in RTC is incorrect, the time will be recovered as "Saturday, 00:00 Jan. 1, 2000" when PLC is powered and restarted.

3. D1313 ~ D1319 will immediately update the RTC only when in TRD instruction or WPLSoft monitoring mode.

4. How to make corrections on RTC:

   a) Use TWR instruction fir SA/SX/SX/EH/EH2/SV series MPU. See API 167 TWR for more details.

   b) Use WPLSoft or digital display panel DU-01.

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 of π (PI)

2. loating 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 SS/ES/EX/SA/SX/SC series MPU. (Setup range: 0 ~ 20; Unit: ms)

2. D1021 can be used for setting up the response time of receiving pulses at X10 ~X17 for ES series MPU. (Setup range: 0 ~ 20; Unit: ms)

3. D1021 can be used for setting up the response time of receiving pulses at X10 ~X11 for SC series MPU. (Setup range: 0 ~ 1,000; Unit: time)

4. D1020 can be used for setting up the response time of receiving pulses at X0 ~X7 for EH/EH2/SV series MPU. (Setup range: 0 ~ 60; Unit: ms)

5. D1021 can be used for setting up the response time of receiving pulses at X10 ~X17 for EH/EH2/SV series MPU. (Setup range: 0 ~ 60; Unit: ms)

6. When the power of PLC goes from "Off" to "On", the content of D1020 and D1021 turn to 10 automatically.
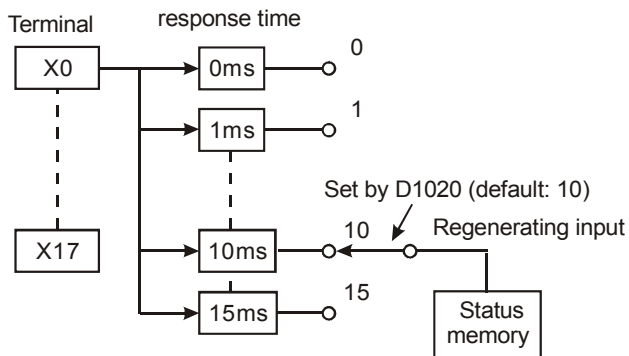
7. If the following programs are executed during the program, the response time of X0 ~ X7 will be set to 0ms. The fastest response time of input terminals is 50μs due to that all terminals are connected with RC filter loop.
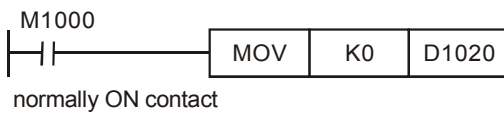
8. There is no need to make adjustment on response time when using high-speed counters and interruptions during the program.

9. Using API 51 REFF instruction has the same effect as modifying D1020 and D1021.

Function Group	Execution Completed Flag
Number	M1029, M1030, M1036, M1037, M1102, M1103

Contents:

Using execution completed flag:

1. API 52 MTR, API 71 HKY, API 72 DSW, API 74 SEGL, API 77 PR: M1029 = On whenever the instruction completes one scan period.

2. API 57 PLSY, API 59 PLSR:

    a) M1029 will be "On" after Y0 pulse output of SA/SX/SC/ES/EX/SS is completed. M1030 will be "On" after Y1 pulse output is compeleted. When PLSY and PLSR instruction is "Off", M1029 and M1030 turn "Off". You have to reset M1029 and M1030 after the action is completed.

    b) M1029 will be "On" after Y0 and Y1 pulse output of EH/EH2/SV is completed. M1030 will be "On" after Y2 and Y3 pulse output is compeleted. M1036 will be "On" after Y4 and Y5 pulse output of EH2/SV is completed. M1037 will be "On" after Y6 and Y7 pulse output is completed.When PLSY and PLSR instruction is "Off", M1029, M1030, M1036 and M1037 turn "Off". When the instruction is re-executed for the next time, M1029, M1030, M1036 and M1037 will turn "Off" and "On" again when the execution is completed.

3. API 63 INCD: M1029 will be "On" for a scan period when the assigned group number of data are compared.

4. API 67 RAMP, API 69 SORT:

a) When the execution of the instruction is completed, M1029= On. You have to reset M1029.

b) M1029 turns "Off" when the instruction is "Off".

5. API 155 DABSR, API 156 ZRN, API 158 DRVI, API 159 DRVA for EH/EH2/SV series MPU:

a) M1029 = On when the 1st output group Y0 and Y1 is completed. M1030 = On when the 2nd output group Y2 and Y3 is completed.

b) M1036 = On when the 3rd output group Y4 and Y5 of EH2/SV is completed. M1037 = On when the 4th output group Y6 and Y7 is completed.

c) When the instruction is re-executed for the next time, M1029 or M1030 will turn "Off" and "On" again when the execution is completed.

6. In API 57 PLSY, API 156 DZRN, API 158 DDRVI and API 159 DDRVA for SC series MPU: M1102 will be set On when Y10 pulse output is completed. M1103 will be set On when Y11 pulse output is completed. After PLSY instruction is disabled,M1102 and M1103 will be set Off. In DDRVA, DDRVI and DZRN, M1102 and M1103 will be set Off when next time these instructions are enabled.

Function Group	Communication Error Code
Number	M1025, D1025

Contents:

When HPP, PC or HMI is connected to the PLC and the PLC receives illegal communication request during the transmission of data, M1025 will be On and the error code will be written in D1025. See the error codes below.

01: Illegal instruction code

02: Illegal device address

03: Requested data exceed the range

07: Checksum error

Function Group	Clear Instruction
Number	M1031, M1032

Contents:

M1031 (clearing non-latched area) , M1032 (clearing latched area)

Device No.	Cleared Device
M1031	■ 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

Device No.	Cleared Device
M1032	■ Contact status of M and S for latched ■ Contact and timing coil of accumulative timer T ■ Contac 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 Latched During STOP

Number   M1033

Contents:

When M1033 = On and PLC goes from "RUN" to "STOP", the On/Off status of output will be ratained.

Assume the output contact load of the PLC is a heater, when PLC switches from RUN to STOP, the status of the heater will be retained. After the PLC program is modified, the PLC will RUN again.

Function Group   All Output Y Inhibited

Number   M1034

Contents:

When M1034 = On, all Y outputs will turn "Off".

(M1034)   all outputs inhibited

Function Group   RUN/STOP Switch

Number   M1035, D1035

Contents:

1. When M1035 = On, EH/EH2/SV series MPU will determine the content (K0 ~ K15) in D1035 to enable input points X0 ~ X17 as the RUN/STOP switch.

2. When M1035 = On, SA/SX/SC series MPU will enable the input point X7 (in SA), X3 (in SX) and X5 (in SC) as the RUN/STOP switch.

Function Group   Detecting Speed of X0 ~ X5

Number   M1036

Contents:

1. For SC_V1.4 and above, SPD can detect the speed of X0 ~ X5 at the same time. The total bandwidth is 40KHz.

2. Program example:

```
      X7
   ┤├─────┬──── [ SET    M1036 ]
          │
          └──── [ SPD   X1   K1000   D0 ]
```
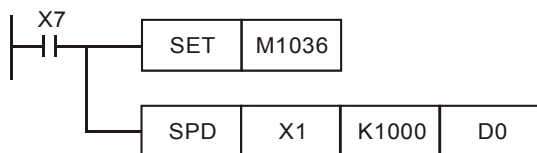
3. Parameter D0 when X7 = On

Start No. of D0 + index value	Functions
+0	Low 16 bits of the 32-bit speed detected at input point X0.
+1	High 16 bits of the 32-bit speed detected at input point X0.
+2	Low 16 bits of the 32-bit speed detected at input point X1.
+3	High 16 bits of the 32-bit speed detected at input point X1.
+4	Low 16 bits of the 32-bit speed detected at input point X2.
+5	High 16 bits of the 32-bit speed detected at input point X2.
+6	Low 16 bits of the 32-bit speed detected at input point X3.
+7	High 16 bits of the 32-bit speed detected at input point X3.
+8	Low 16 bits of the 32-bit speed detected at input point X4.
+9	High 16 bits of the 32-bit speed detected at input point X4.
+10	Low 16 bits of the 32-bit speed detected at input point X5.
+11	High 16 bits of the 32-bit speed detected at input point X5.
+12	Remaining time for speed detection (unit: ms)

Function Group   X0 Detecting Pulse Width

Number   M1084, D1023

Contents:

When M1084 = On, X0 of ES/EX/SS_V6.4/SA/SX_V1.6/SC_V1.4 can detect pulse width. Whenever X0 turns from "On" to "Off", the value is updated once and stored in D1023 (unit: 0.1ms). The minimum detectable width is 0.1ms and maximum 10,000ms.

Function Group   Communication Port Function

Number   M1120, M1136, M1138, M1139, M1143, D1036, D1109, D1120

Contents:

1. Supports ES/EX/SS_V6.0/SA/SX_V1.2/SC_V1.0/EH_V1.1/SV_V1.0 and versions above.

2. COM ports (COM1: RS-232; COM2: RS-485) in SA/SX/SC series MPU and COM ports (COM1: RS-232; COM2: RS-232/RS-485/RS-422) in EH/EH2 series MPU support Modbus ASCII/RTU communication format with speed of up to 115,200bps. COM1 and COM2 can be used at the same time. COM3: RS-232/RS-485 in EH/EH2 series MPU supports Modbus ASCII communication format with speed of up to 38,400bps.

    COM1:   For slave stations only. Supports ASCII/RTU communication format, adjustable baud rate with speed of up to 115,200bps, and modification on data length (data bits, parity bits, stop bits).

COM2: For master or slave stations. Supports ASCII/RTU communication format, adjustable baud rate with speed of up to 115,200bps, and modification on data length (data bits, parity bits, stop bits).

COM3: For slave stations only. Supports ASCII communication format (data bits, parity bits, stop bits) 7, E, 1, adjustable baud rate with speed of up to 38,400bps. COM2 or COM3 cannot be used for slave stations at the same time.

■ Communication Format Settings:

COM1: 1. Communication format is set in D1036. b8 ~ b15 do not support the communication protocol of COM1 (RS-232) Slave

2. Communication setting in M1138 remains

3. M1139 is set in ASCII/RTU mode

COM2: 1. Communication format is set in D1120. Communication protocol of COM2 (RS-232/RS-485/RS-422) Master or Slave

2. Communication setting in M1120 remains

3. M1143 is set in ASCII/RTU mode

COM3: 1. Communication format is set in D1109. b0 ~ b3 and b8 ~ b15 do not support the communication protocol of COM3 (RS-232/RS-485) Slave

2. Communication setting in M1136 remains

Communication protocols and how to set:

	Content	0	1
b0	Data length	b0 = 0:7	b0 = 1:8
b1 b2	parity bit	b2, b1 = 00 : None b2, b1 = 01 : Odd b2, b1 = 11 : Even	
b3	stop bits	b3 = 0:1 bit	b3 = 1:2 bit
b7 ~ b4	b7 ~ b4 = 0001 (H1) : 110 bps b7 ~ b4 = 0010 (H2) : 150 bps b7 ~ b4 = 0011 (H3) : 300 bps b7 ~ b4 = 0100 (H4) : 600 bps b7 ~ b4 = 0101 (H5) : 1,200 bps b7 ~ b4 = 0110 (H6) : 2,400 bps b7 ~ b4 = 0111 (H7) : 4,800 bps b7 ~ b4 = 1000 (H8) : 9,600 bps b7 ~ b4 = 1001 (H9) : 19,200 bps b7 ~ b4 = 1010 (HA) : 38,400 bps b7 ~ b4 = 1011 (HB) : 57,600 bps b7 ~ b4 = 1100 (HC) : 115,200 bps		
b8	Select start bit	b8 = 0:None	b8 = 1:D1124
b9	Select the 1st end bit	b9 = 0:None	b9 = 1:D1125
b10	Select the 2nd end bit	b10 = 0:None	b10 = 1:D1126
b15 ~ b11	Not defined		

Example 1: Modifying communication format of COM2

1. Add the program code below on top of the program to modify the communication format of COM2. When PLC switches from STOP to TUN, the program will detect whether M1120 is On in the first scan time. If M1120 is On, the program will modify the relevant settings of COM2 according to the value set in D1120.

2. Modify the communication format of COM2 into ASCII mode, 57,600bps, 7 data bits, even parity, 1 stop bit (57,600, 7, E, 1)

```
  M1002
───┤├─────────┬──────┬─────┬──────┐
              │ MOV  │ H86 │ D1120│
              ├──────┴─────┴──────┤
              │      ┌─────┬──────┐
              └──────┤ SET │ M1120│
                     └─────┴──────┘
```
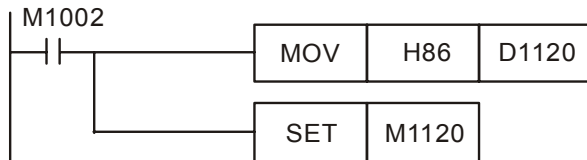
Notes:

1. If COM2 is to be used as a Slave terminal, make sure there is no communication instruction existing in the program.

2. After the communication format is modified, the format will stay intact when PLC switches from RUN to STOP.

3. If you shut down the power of the PLC and repower it again, the modified communication format will return to default setting.

Example 2: Modifying the communication format of COM1

1. Add the program code below on top of the program to modify the communication format of COM1. When PLC switches from STOP to TUN, the program will detect whether M1138 is On in the first scan time. If M1138 is On, the program will modify the relevant settings of COM1 according to the value set in D1036.

2. Modify the communication format of COM1 into ASCII mode, 115,200bps, 7 data bits, even parity, 1 stop bit (115,200, 7, E, 1)
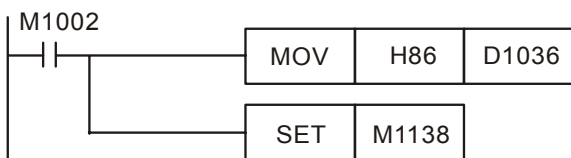
```
  M1002
───┤├─────────┬──────┬─────┬──────┐
              │ MOV  │ H86 │ D1036│
              ├──────┴─────┴──────┤
              │      ┌─────┬──────┐
              └──────┤ SET │ M1138│
                     └─────┴──────┘
```

Notes:

1. After the communication format is modified, the format will stay intact when PLC switches from RUN to STOP.

2. If you shut down the power of the PLC and repower it again, the modified communication format will return to default setting.

Example 3: Modifying the communication format of COM3

1. The communication format of COM3 is fixed as 7 data bits, even parity, 1 stop bit. Add the program code below on top of the program to modify the baud rate of COM3 into 38,400bps. When PLC switches from STOP to TUN,

the program will detect whether M1136 is On in the first scan time. If M1136 is On, the program will modify the relevant settings of COM3 according to the value set in D1109.

2. Modify the baud rate of COM3 into 38,400bps
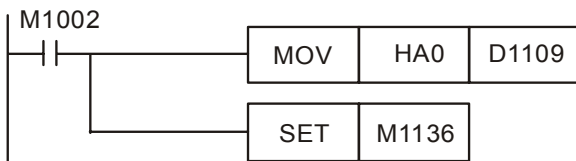
```
M1002
 ├┤├─────────┬──── MOV | HA0 | D1109
             │
             └──── SET | M1136
```

Notes:

1. After the communication format is modified, the format will stay intact when PLC switches from RUN to STOP.

2. If you shut down the power of the PLC and repower it again, the modified communication format will return to default setting.

Example 4: Setting up RTU mode of COM1 and COM2

1. COM1 and COM2 support ASCII/RTU mode. COM1 is set by M1139 and COM2 is set by M1143. When the flags are On, they are in RTU mode; when the flags are Off, they are in ASCII mode.

2. How to set up RTU mode

COM1: (9,600, 8, E, 1, RTU)

```
M1002
 ├┤├─────────┬──── MOV | H86 | D1036
             │
             ├──── SET | M1138
             │
             └──── SET | M1139
```

COM2: (9,600, 8, E, 1, RTU)

```
M1002
 ├┤├─────────┬──── MOV | H86 | D1120
             │
             ├──── SET | M1120
             │
             └──── SET | M1143
```
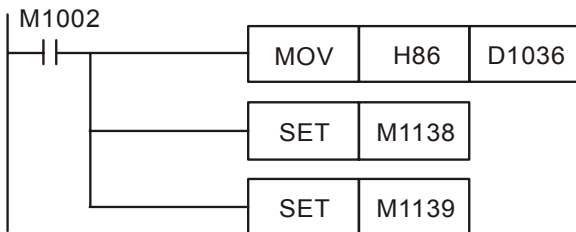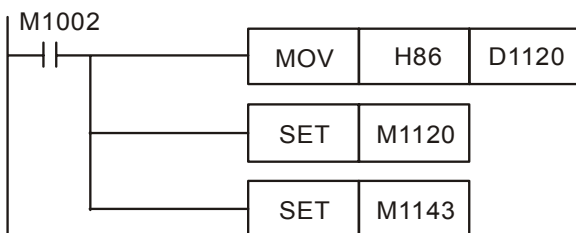
3. COM2 of EH/EH2/SV series MPU supports the generation of interruption I170 when the data receiving is completed in Slave mode.

4. Normally when the communication terminal of the PLC is in Slave mode, PLC will not immediately process the communication data entered but process it after the END is executed. Therefore, when the scan time is very long and you need the communication data to be processed immediately, you can use interruption I170 for this matter.

5. Example of interruption I170 (after the data receiving is completed in Slave mode)

```
             ┌──────┐
             │  EI  │
    M1002    ├──────┬──────┬──────┐
   ──┤├──────┤ MOV  │ H86  │D1120 │
             ├──────┴──────┴──────┘
             │ SET  │M1120 │
             ├──────┴──────┤
             │ SET  │M1143 │
             ├──────┴──────┘
             │ FEND │
             ├──────┤
    I170     │ IRET │
   ──────────├──────┤
             │ END  │
             └──────┘
```
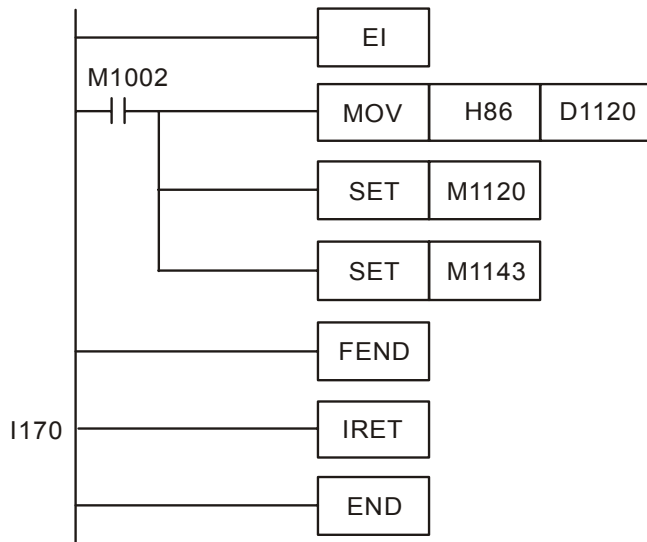
With I170 in the program, when COM2 is in Slave mode and there are communication data coming in, PLC will process the data and respond immediately.

Notes:

1. DO NOT updat program on-line when using I170.

2. The scan time of PLC will be slightly longer.

Function Group   Communication Response Delay

Number   D1038

Contents:

1. When PLC is used as slave station, in RS-485 communication interface, users can set up communication response delay time ranging from 0 to 10,000 (0 ~ 1 second). If the time is without the range, D1038 = O (time unit: 0.1ms). The set value of time must be less than that in D1000(scan time-out timer WDT).

2. In EASY PLC LINK, you can set up delayed transmission of the next communication data (unit: 1 scan period for SA/SX/SC; 0.1ms for EH/EH2/SV).

Function Group   Fixed Scan Time

Number   M1039, D1039

Contents:

1. When M1039 = On, the scan time of program is determined by the content in D1039. When the execution of the program is completed, the next scan will take place when the fixed scan time is reached. If the content in D1039 is less than the actual scan time of the program, the scan time will follow the actual scan time of the program.

```
    M1000
   ──┤├──────────────( M1039 )  Fixed scan time
   Normally On      ┌──────┬──────┬──────┐
   contact     ─────┤ MOV P│ K20  │D1039 │
                    └──────┴──────┴──────┘
              Scan time is fixed to 20ms
```
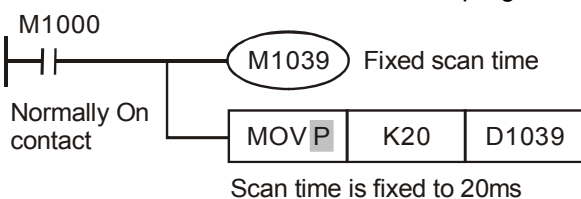
2. Instructions related to scan time, RAMP (API 67), HKY (API 71), SEGL (API 74), ARWS (API 75) and PR (API 77) should be used together with "fixed scan time" or "constant interruption".

3. Particularly for HKY instruction, when the 16-digit button input is operated by 4x4 matrix, the scan time has to be fixed to longer than 20ms.

4. The scan time in D1010 ~ D1012 also includes fixed scan time.

Function Group   Analog Function

Number   D1056 ~ D1059, D1062, D1110 ~ D1113, D1116 ~ D1118

Contents:

1. Resolution of analog input channel: 10 bits for EX, corresponding to 0 ~ ±10 V (-512 ~ +511) or 0 ~ ±20 mA (-512 ~ +511); 12 bits for SX, corresponding to 0 ~ ±10 V (-2,000 ~ +2,000) or 0 ~ ± 20 mA (-1,000 ~ +1,000).

2. Resolution of analog output channel: 8 bits for EX, corresponding to 0 ~ 10 V (0 ~ 255) or 0 ~ 20 mA (0 ~ 255); 12 bits for SX, corresponding to 0 ~ ±10 V (-2,000 ~ +2,000) or 0 ~ ±20 mA (-2,000 ~ +2,000).

3. Sampling time of analog/digital conversion. Default setting = 5; unit: ms. If D1118 ≤ 5, it will be regarded as 5ms.

4. Resolution of EH/EH2 analog input AD card (DVP-F2AD): 12 bits 0 ~ 10 V (0 ~ +4,000) or 11 bits 0 ~ 20 mA (0~+2,000)

5. Resolution of EH/EH2 analog input DA card (DVP-F2DA): 12 bits 0 ~ 10 V (0 ~ +4,000) or 0 ~ 20 mA (0 ~ +4,000)

Device No.	Function
D1056	Present value of EX/SX analog input channel CH0 and EH/EH2 AD card channel CH0
D1057	Present value of EX/SX analog input channel CH1 and EH/EH2 AD card channel CH1
D1058	Present value of EX analog input channel CH 2
D1059	Present value of EX analog input channel CH 3
D1062	Average times (2 ~ 4) of SX AD0 and AD1
D1110	Average value of EX/SX analog input channel CH0 and EH/EH2 AD card channel CH0
D1111	Average value of EX/SX analog input channel CH1 and EH/EH2 AD card channel CH1
D1112	Average value of EX analog input channel CH2
D1113	Average value of EX analog input channel CH3
D1116	EX/SX analog output channel CH0 and EH/EH2 DA card channel CH0
D1117	EX/SX analog output channel CH1 and EH/EH2 DA card channel CH1
D1118	Sampling time (ms) of EX/SX/EH/EH2 analog/digital conversion

Function Group   Operational Error Flag

Number   M1067 ~ M1068, D1067 ~ D1068

Contents:

1. Operational error flag:

Device	Description	Latched	STOP → RUN	RUN → STOP
M1067	Operational error flag	None	Cleared	Latched
M1068	Operational error locked flag	None	Latched	Latched

Device	Description	Latched	STOP → RUN	RUN → STOP
D1067	Operational error code	None	Cleared	Latched
D1068	STEP value when operational error occurs	None	Latched	Latched

2. Error code explanation:

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

Function Group   Low Voltage

Number   M1087, D1100

Contents:

1. When PLC detects LV (Low Voltage) signal, it will check if M1087 is "On" or not. If M1087 is "On", the content in D1100 will be stored in Y0 ~ Y17.

2. bit0 (LSB) of D1100 corresponds to Y0, bit1 corresponds to Y1, bit8 corresponds to Y10 and so on.

Function Group   File Register

Number   M1101, D1101 ~ D1103

Contents:

1. When the power of PLC turns from "Off" to "On", PLC determines whether to automatically send the content in the file register to the assigned data register by checking M1101, D1101 ~ D1103 (for SA/SX/SC/EH/EH2/SV).

M1101:	Whether to automatically downland data from file register
D1101:	Start No. of file register K0 ~ K1,599 (for SA/SX/SC) Start No. of file register K0 ~ K9,999 (for EH/EH2/SV)
D1102:	Number of data read from file register K0 ~ K1,600 (for SA/SX/SC) Number of data read from file register K0 ~ K8,000 (for EH/EH2/SV)
D1103:	Location for storing data read from file register Start No. of assigned data register D K2,000 ~ K4,999 (for SA/SX/SC) Start No. of assigned data register D K2,000 ~ K9,999 (for EH/EH2/SV)

2. See API 148 MEMR and API 149 MEMW for more details.

Function Group   DIP Switch Function Card

Number   M1104 ~ M1111

Contents:

1. When PLC is in RUN status with digital switch function card inserted, the 8 DIP switches amd their status orrespond respectively to M1104 ~ M1111.

2. See API 109 SWRD for more details.

3. When PLC is in RUN status with 4DI card inserted into the input AX0 (photocoupler isolation), the status of AX0 ~ AX3 correspond respectively to M1104 ~ M1107.

Function Group	Transistor Output Function Card
Number	M1112, M1113

Contents:

When PLC is in RUN status with 2DO function card inserted, M1112 and M1113 will correspond respectively to 2 transistors output points, AY0 and AY1.

Function Group	Pulse Output With Speed Acceleration/Deceleration
Number	M1115 ~ M1119, D1104

Contents:

1. Special D and special M for acceleration/ deceleration of speed pulse output for SA/SX/SC (not applicable to SC_V1.4 and versions above):

Device No.	Function
M1115	Activation switch
M1116	"Accelerating" flag
M1117	"Target frequency reached" flag
M1118	"Decelerating" flag
M1119	"Function completed" flag
D1104	Start No. of control register (D)

2. Parameters for D1104 (frequency range: 25Hz ~ 10KHz)

Index	Function	
+ 0	Start frequency (SF)	
+ 1	Gap frequency (GF)	
+ 2	Target frequency (TF)	
+ 3	The lower 16 bits of the 32 bits for the total number of output pulses	(TP)
+ 4	The higher 16 bits of the 32 bits for the total number of output pulses	
+ 5	The lower 16 bits of the 32 bits for the total number of output pulses in accelerating/decelerating section	(AP)
+ 6	The higher 16 bits of the 32 bits for the total number of output pulses in accelerating/decelerating section	

3. No instruction is needed, users need only to fill out the parameter table and enable M1115 (in RUN mode). This functio only supports Y0 output and the timing chart is as below.

Number of accel/decel sections = (TF-SF)/GF

Number of output pulses in every section:
GP = AP/Number of accel/decel sections

AP = number of accel/decel pulses
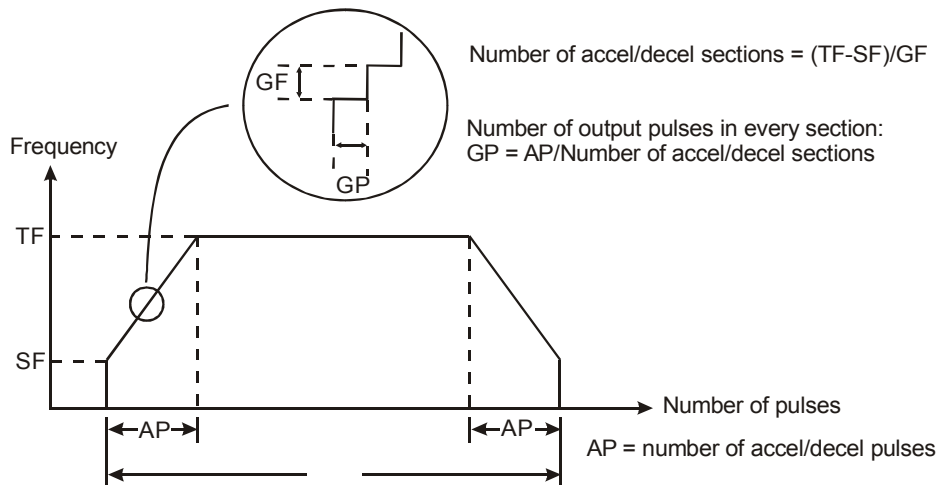
4. Note: this function is applicable only when "all" the conditions below are met.

    a) Start frequency < target frequency.

    b) Gap frequency ≤ (target frequency – start frequency)

    c) Total number of pulses > (accel/decel number of pulses × 2)

    d) For start frequency and target frequency: Min. 25Hz; Max. 10KHz

    e) Number of accel/decel pulses > number of accel/decel sections

When M1115 turns from "On" to "Off", M1119 will be reset and M1116, M1117 and M1118 remain unchanged. When PLC goes from "STOP" to "RUN", M1115 ~ M1119 will be reset as "Off". D1104 will only be cleared as "0" when it turns from "Off" to "On".

Either accel/decel pulse output function or PLSY Y0 output can be executed at a time when PLC is operating.

5. How to calculate the action time of each section

    Assume the start frequency is set as 1KHz, gap frequency as 1KHz, target frequency as 5KHz, total number of pulses as 100 and number of acceleration pulses as 40, the timing diagram of the acceleration sections is as the figure below.
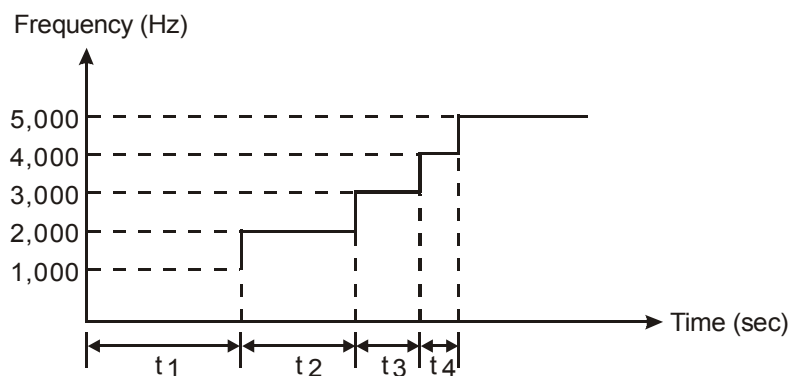
    From the conditions above, we can obtain the number of acceleration/deceleration sections is (5K – 1K)/1K = 4 and the number of output pulses in each section is 40/4 = 10. Therefore, in the diagram, t1 = (1/1K) × 10 = 10ms, t2 = (1/2K) × 10 = 5ms, t3 = (1/3K) × 10 = 3.33ms, t4 = (1/4K) × 10 = 2.5ms.

6. Program example: Forward/reverse acceleration/deceleration step motor control

```
M1002
──┤├──────┬──── MOV │ K500  │ D1104      D500 ~ D506 as parameter addresses
          │
          ├──── MOV │ K1000 │ D500       Start frequency: 1KHz
          │
          ├──── MOV │ K100  │ D501       Interval frequency: 100Hz
          │
          ├──── MOV │ K10000│ D502       Target frequency: 10KHz
          │
          ├──── DMOV│ K80000│ D503       Number of output pulses: 80,000
          │
          ├──── DMOV│ K10000│ D505       Number of pulses in accel/decel section: 10,000
          │
          └──── SET │ M1115
```
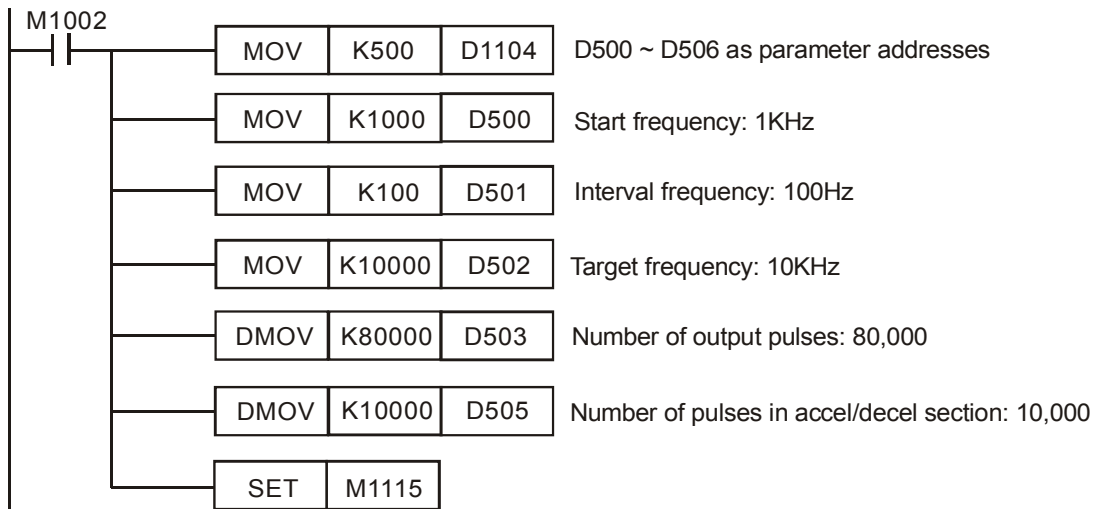
a) When PLC is in RUN status, store all parameter settings into the registers designated in D1104.

b) When M1115 = On, the acceleration/deceleration pulse output will start.

c) M1116 = On in the acceleration process. When the speed reaches its target, M1117 will be On. M1118 = On in the deceleration process. When the speed reaches its target, M1119 will be On.

d) M1115 will not be reset automatically. You have to check the conditions during the process and reset it.

e) Pulse output curves:
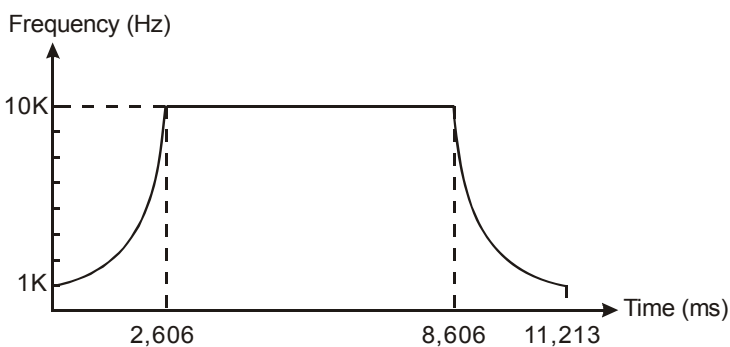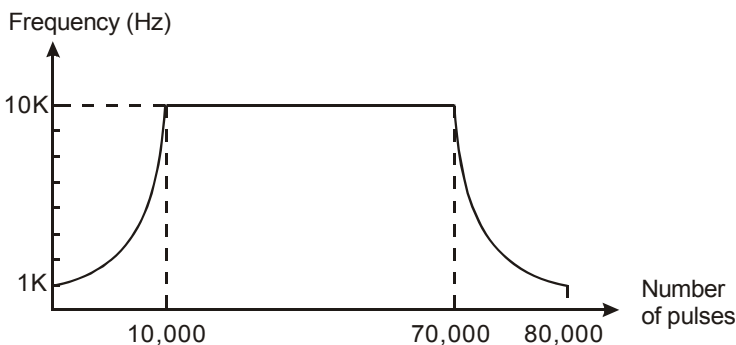
Function Group	Special High-Speed Pulse Output
Number	M1133 ~ M1135, D1133

Contents:

1. Special D and special M for special high-speed pulse Y0 (50KHz)for SA/SX/SC (not applicable to SC_V1.4 and versions above):

No.	Function
M1133	Output switch for special high-speed pulse Y0 (50KHz) (On = enabled)
M1134	On = Continuous output switch for special high-speed pulse Y0 (50KHz)
M1135	"Number of pulses reached" flag for special high-speed pulse Y0 (50KHz)
D1133	Start No. of control register (D) for special high-speed pulse Y0 (50KHz)

2. Parameters for D1133:

Index	Function
+ 0	The lower 16 bits of the 32 bits for output frequency of special high-speed pulse Y0
+ 1	The higher 16 bits of the 32 bits for output frequency of special high-speed pulse Y0
+ 2	The lower 16 bits of the 32 bits for number of output pulses of special high-speed pulse Y0
+ 3	The higher 16 bits of the 32 bits for number of output pulses of special high-speed pulse Y0
+ 4	The lower 16 bits of the 32 bits of the present number of special high-speed pulses Y0
+ 5	The higher 16 bits of the 32 bits of the present number of special high-speed pulses Y0

3. The function:

All output frequency and number of pulses stated in the table above can be modified when M1133 = On and M1135 = On. The modification will not affect the present output pulses. The present number of output pulses is updated in every scan time. When M1133 turns from "Off" to "on", the number will be cleared as "0". When 1133 turns from "On" to "Off", the last numner of output pulses will be shown.

Note:

The special high-speed pulse output function can only be used on specific Y0 output point when PLC is in RUN status. It can coexist with PLSY (Y0) in the program and PLSY (Y1) will not be affected. If PLSY (Y0) instruction is executed prior to this function, the function cannot be used and vice versa. When the function is executed, the general function, general Y0 output will be invalid but Y1 ~ Y7 can be used.

The output frequency of this function is higher (max. 50KHz) than that of PLSY instruction.

Function Group   2-axis Synchronous Control (PH)

Number   M1133, M1135, D1133 ~ D1136

Contents:

1. Special D and special M for 2-axis synchronous drawing oblique and arc for SC_V1.4 and versions above:

Device No.	Function
M1133	Start flag for Y10 output for two-axis synchronous control
M1135	Start flag for Y11 output for two-axis synchronous control
D1133	Start No. of control register (D) for Y10 output for two-axis synchronous control
D1134	Number of sections for Y10 output for two-axis synchronous control
D1135	Start No. of control register (D) for Y11 output for two-axis synchronous control
D1136	Number of sections for Y11 output for two-axis synchronous control

2. Parameters for D1133, D1135:

Index	Function
+ 0	Y10, Y11 2-axis synchronous control; output frequency of 1st section = low 16 bits of 32 bits
+ 1	Y10, Y11 2-axis synchronous control; output frequency of 1st section = high 16 bits of 32 bits
+ 2	Y10, Y11 2-axis synchronous control; output pulse number of 1st section = low 16 bits of 32 bits
+ 3	Y10, Y11 2-axis synchronous control; output pulse number of 1st section = high 16 bits of 32 bits

3. The functions:

a) Definition of the 2 axes:

X axis: Y0 (direction output) and Y10 (pulse output)

Y axis: Y1 (direction output) and Y11 (pulse output)

b) Define the format of output table:

Assume D1133 = K100 and D1134 = K3 and the output table has to be set as:

Section No.	Device D	Output frequency	Device D	Number of Output pulses	Description
1	D101,D100	K10,000	D103,D102	K1,000	Section 1 outputs 1,000 pulses in 10KHz
2	D105,D104	K15,000	D107,D106	K2,000	Section 2 outputs 2,000 pulses in 15KHz
3	D109,D108	K5,000	D111,D110	K3,000	Section 3 outputs 3,000 pulses in 5KHz

Note: The frequency and number of output pulses are all in 32-bit. Thus, the 3 sections will continuously occupy 12 D devices (3 × 2 × 2 = 12).

4. Note:

a) Make sure that the output frequency and the number of pulses have been set before using this function. The output frequency and the number of pulses cannot be modified during the execution of the function.

b) When PLC program scans to END instruction, it will auto-check whether this function needs to be enabled.

c) When M1133 and M1135 are set in the same scan period, the two axes will output pulses synchronously.

d) When the output frequency < 100Hz, the output will be executed in 100Hz. When the output frequency > 100KHz, the output will be executed by 100KHz.

e) Only device D (D0 ~ D999 and D2000 ~ D4999) can be used for this function. DO NOT use other devices or exceed the range of device D.

f) The maximum number of segments for this function is 50. When the number of segments < 1 or > 50, this function will be disabled.

g) After this function is enabled, M1102 = "On" indicates Y10 output is completed and M1103 = "On" indicates Y11 output is completed.

5. Examples:

a) Draw oblique lines in 2 axes

Destination: Draw 2 oblique lines (as figure 1)

Program explanation: Y0 and Y10 belong to X axis, and Y1 and Y11 belong to Y axis (as figure 2)

Output frequency and number of pulses: see table 1

(Figure 1)

(Figure 2)
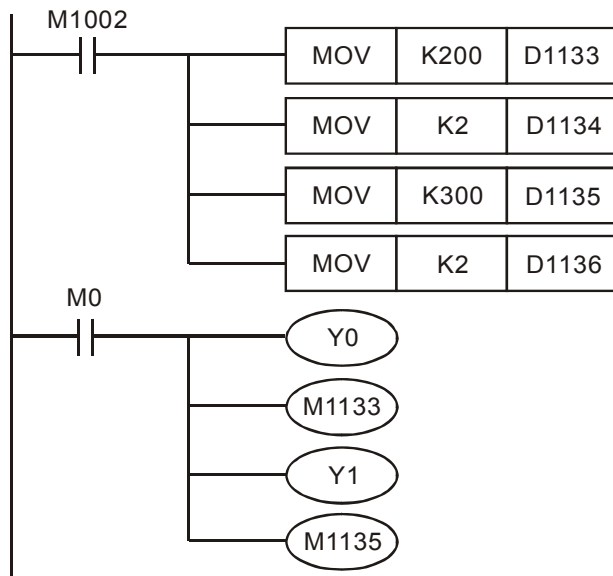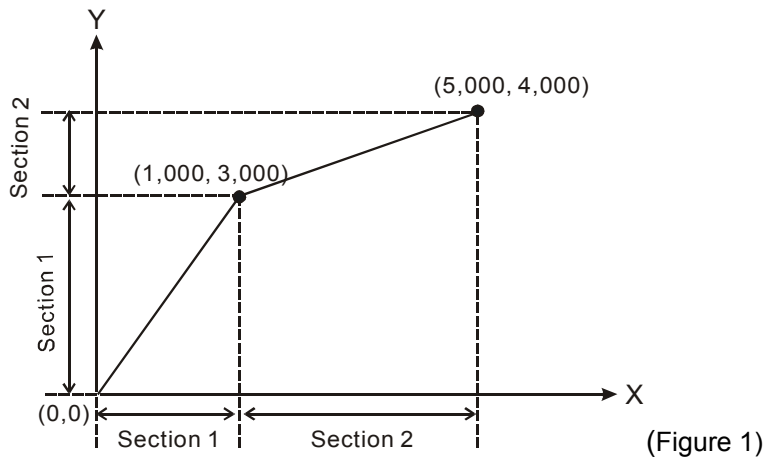
Axis	Section	Device D	Output frequency	Device D	Number of output pulses
X	1	D201, D200	K1,000	D203, D202	K1,000
	2	D205, D204	K4,000	D207, D206	K4,000
Y	1	D301, D300	K3,000	D303, D302	K3,000
	2	D305, D304	K1,000	D307, D306	K1,000

(Table 1)

b) Draw an arc in 2 axes

Destination: Draw a 90° arc (see figure 3)

Program explanation: Same as the the example of draw oblique lines, except that D1134 and D1136 are modified as K10 (output 10 sections)

Output frequency and number of pulses: see table 2

(Figure 3)
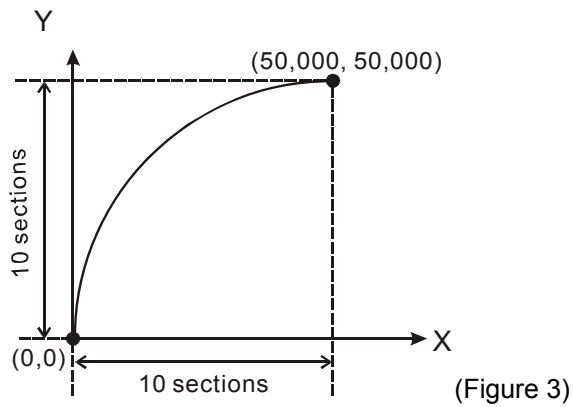
Axis	Section	Device D	Output frequency	Device D	Number of output pulses
X	1	D201, D200	K1,230	D203, D202	K615
	2	D205, D204	K3,664	D207, D206	K1,832
	3	D209, D208	K6,004	D211, D210	K3,002
	4	D213, D212	K8,200	D215, D214	K4,100
	5	D217, D216	K10,190	D219, D218	K5,095
	6	D221, D220	K11,932	D223, D222	K5,966
	7	D225, D224	K13,380	D227, D226	K6,690
	8	D229, D228	K14,498	D231, D230	K7,249
	9	D233, D232	K15,258	D235, D234	K7,629
	10	D237, D236	K15,644	D239, D238	K7,822
Y	1	D301, D300	K15,644	D303, D302	K7,822
	2	D305, D304	K15,258	D307, D306	K7,629
	3	D309, D308	K14,498	D311, D310	K7,249
	4	D313, D312	K13,380	D315, D314	K6,690
	5	D317, D316	K11,932	D319, D318	K5,966
	6	D321, D320	K10,190	D323, D322	K5,095
	7	D325, D324	K8,200	D327, D326	K4,100
	8	D329, D328	K6,004	D331, D330	K3,002
	9	D333, D332	K3,664	D335, D334	K1,832
	10	D337,D336	K1,230	D339, D338	K615

(Table 2)

c)   Draw arcs in the four quadrants

Destination: Draw four 90° arcs (see figure 4)

Program explanation: When the direction signal is On, the direction will be a positive one; when the direction signal is Off, the direction will be a negative one (see figure 5)

Output frequency and number of pulses: see table 2
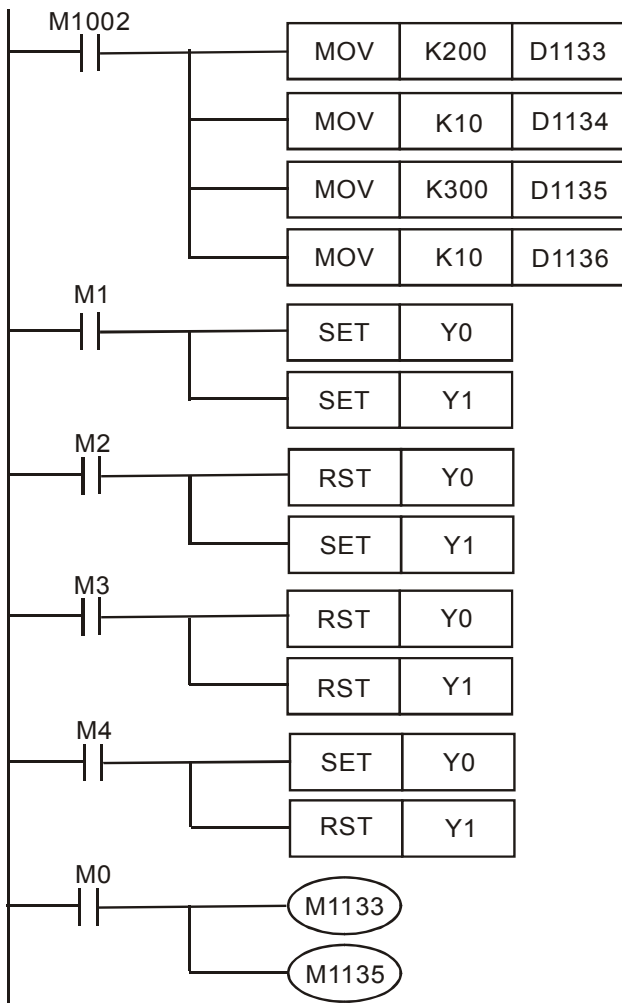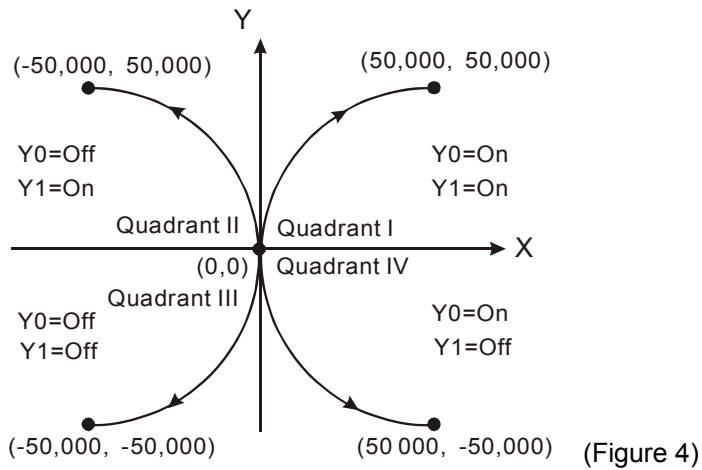
(Figure 4)

(Figure 5)

- M0, M1 = On refers to drawing a 90° arc in Quadrant I; M0, M2 = On refers to drawing a 90° arc in Quadrant II; M0, M3 = On refers to drawing a 90° arc in Quadrant III; M0, M4 = On refers to drawing a 90° arc in Quadrant IV.

- The four 90° arcs are drawn when acceleration in X and deceleration in Y. To draw the arcs when deceleration in X and acceleration in X, modify the program into figure 6, i.e. D1333 = K300 and D1335 = K200.

M1002
MOV	K300	D1133
MOV	K10	D1134
MOV	K200	D1135
MOV	K10	D1136

M1
| SET | Y0 |
| SET | Y1 |

M2
| RST | Y0 |
| SET | Y1 |

M3
| RST | Y0 |
| RST | Y1 |

M4
| SET | Y0 |
| RST | Y1 |

M0
(M1133)
(M1135)
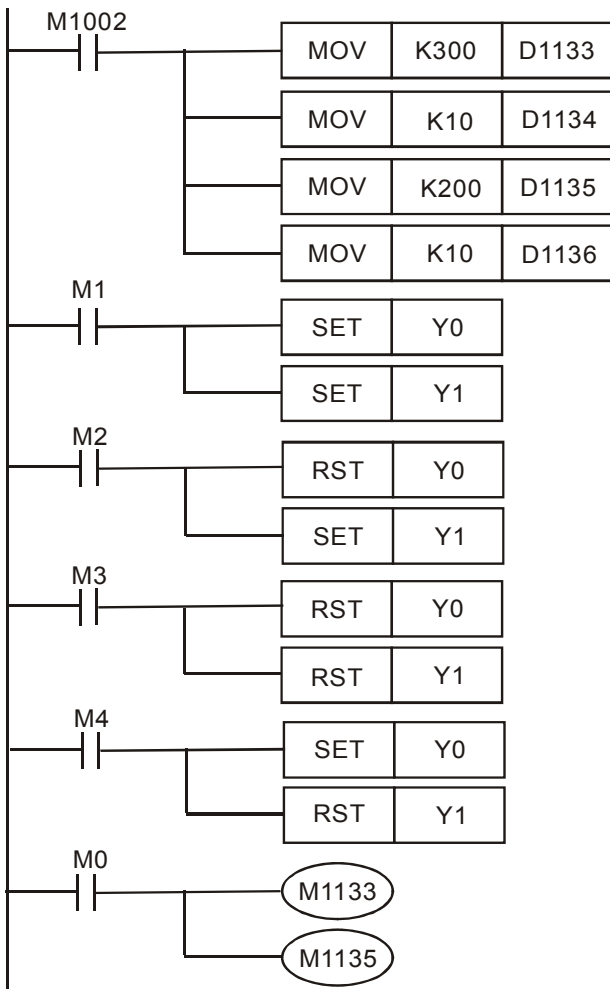
(Figure 6)

■ M0, M1 = On refers to drawing a 90° arc in Quadrant I; M0, M2 = On refers to drawing a 90° arc in Quadrant II; M0, M3 = On refers to drawing a 90° arc in Quadrant III; M0, M4 = On refers to drawing a 90° arc in Quadrant IV. (See figure 7)
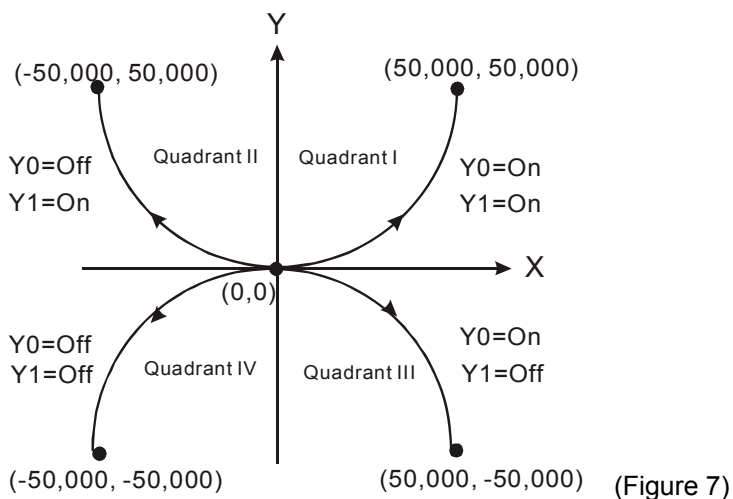
(Figure 7)

d)   Draw a circle

Destination: Extract four 90° arcs from figure 4 and figure 7 and combine them into a circle (see figure 8).

Program explanation: When the direction control pin is On, the direction will be a positive one; otherwier, it will be a negative one (see figure 9). When X0 = On, D0 will accumulate once and the 2 axes will draw a 90° arc.

Output frequency and number of pulses: see table 2.
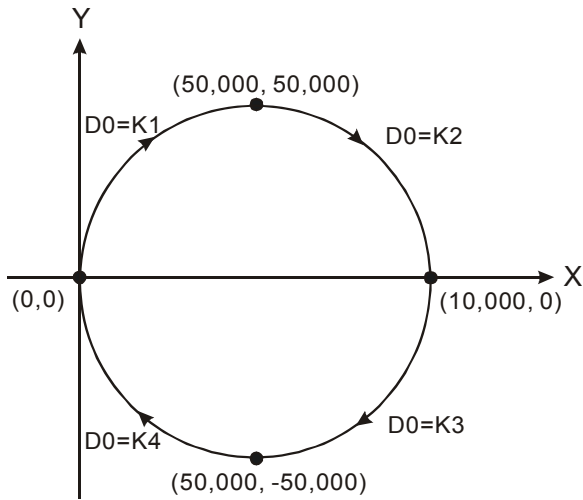
(Figure 8)
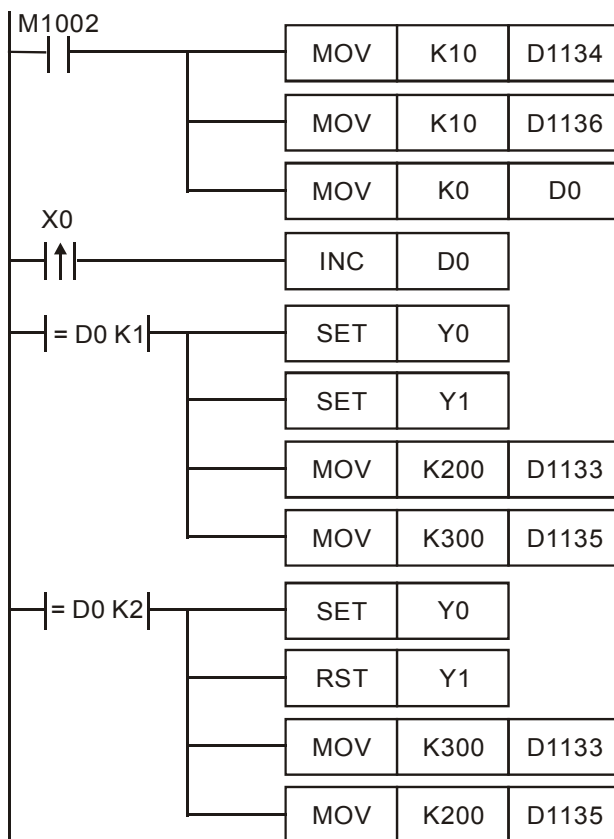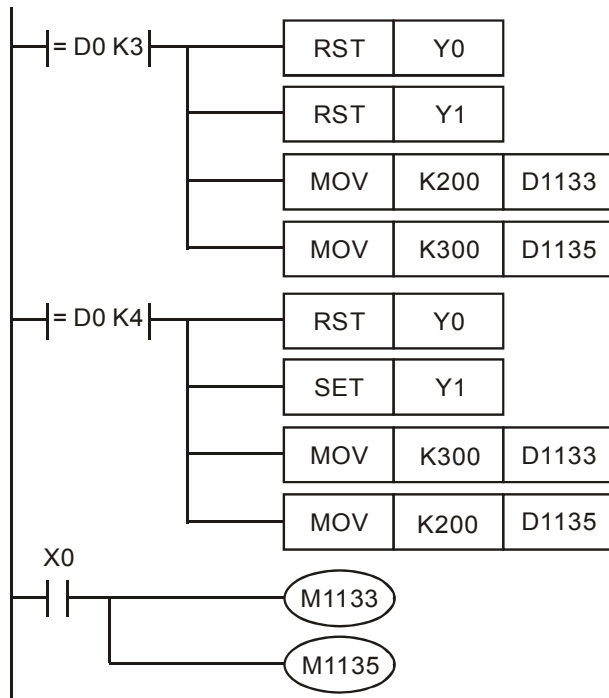
(Figure 9)

e) Calculate the frequency and number of output pulses in each section

Destination: Draw 10 sections of arcs clockwise until they reach (50,000, 50,000) (see figure 10)

Rx = target value in X; Ry = target value in Y; N = number of sections; π = 3.1416
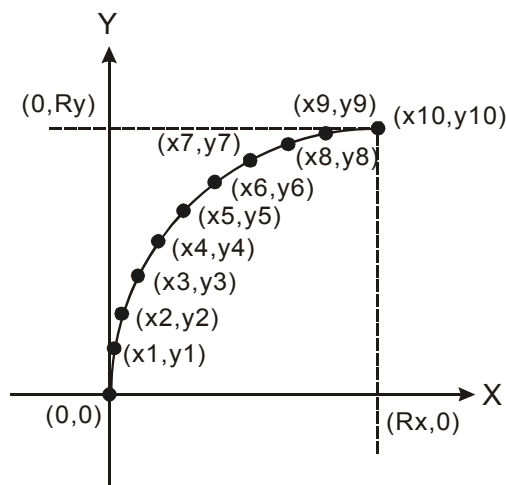
(Figure 10)

■ **Step 1:** Calculate the position for each section

$x_1 = Rx - Rx \times \sin[(N-1) \times \pi \div (2 \times N)]$

$x_2 = Rx - Rx \times \sin[(N-2) \times \pi \div (2 \times N)]$ …. See table 3

$y_1 = Ry \times \sin[1 \times \pi \div (2 \times N)]$

$y_2 = Ry \times \sin[2 \times \pi \div (2 \times N)]$…. See table 4

Position	x_1	x_2	x_3	x_4	x_5
With decimal point	615.55	2,447.12	5,449.61	9,549.08	14,464.59
Without decimal point	615	2,447	5,449	9,549	14,464

Position	x_6	x_7	x_8	x_9	x_{10} (R_X)
With decimal point	20,610.67	27,300.42	34,549.11	42,178.25	50,000
Without decimal point	20,610	27,300	34,549	42,178	50,000

(Table 3)

Position	y_1	y_2	y_3	y_4	y_5
With decimal point	7,821.74	15,450.88	22,699.57	29,389.32	35,355.40
Without decimal point	7,821	15,450	22,699	29,389	35,355
Position	y_6	y_7	y_8	y_9	y_{10} (R_y)
With decimal point	40,450.91	44,550.38	47,552.87	49,384.44	50,000
Without decimal point	40,450	44,550	47,552	49,384	50,000

(Table 4)

- **Step 2:** Calculate the distance (number of pulses) between every section

    In X axis: $x_1 = x_1 - 0$, $x_2 = x_2 - x_1$, ... $x_{10} = x_{10} - x_9$ (see table 5)

    In Y axis: $y_1 = y_1 - 0$, $y_2 = y_2 - y_1$, … $y_{10} = y_{10} - y_9$ (see table 5)

Position	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Number of pulses	615	1,832	3,002	4,100	5,095	5,966	6,690	7,249	7,629	7,822
Position	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
Number of pulses	7,821	7,629	7,249	6,690	5,966	5,095	4,100	3,002	1,832	616

(Table 5)

- **Step 3:** Decide the execution time of every section and obtain the frequency of every section by table 5

    Assume every section executes for 500ms, the equation for frequency (Hz) of each section is: $fx_1 = 1 \div 0.5 \times$

    $x1$、$fx_2 = 1 \div 0.5 \times x_2$…(see table 6)

Position	fx_1	fx_2	fx_3	fx_4	fx_5	fx_6	fx_7	fx_8	fx_9	fx_{10}
Frequency	1,230	3,664	6,004	8,200	10,190	11,932	13,380	14,498	15,258	15,644
Position	fy_1	fy_2	fy_3	fy_4	fy_5	fy_6	fy_7	fy_8	fy_9	fy_{10}
Frequency	15,642	15,258	14,498	13,380	11,932	10,190	8,200	6,004	3,644	1,232

(Table 6)

- **Step 4:** Fill Device D into table 2 and complete all steps.

    Reminder 1: When Rx=Ry, you can calculate X axis, and copy X axis to Y axis (as fy1 = fx10, fy2 = fx9, …fy10 = fx1,and y1 = x10, y2 = x9,…y10 = x1)

    Reminder 2: When drawing a counterclockwise arc, switch the index value of X axis with that of Y axis.

Function Group   Detecting Extension

Number   D1140, D1142, D1143, D1145

Contents:

1. D1140: Number of special right-side extension modules (AD, DA, XA, PT, TC…); Max. 8

2. D1142: Number of X input points on digital extension device

3. D1143: Number of Y output points on digital extension device

4. D1145: Number of special left-side extension modules (AD, DA, XA, PT, TC…); Max. 8 (available in SV only)

Function Group	Adjustable Pulse Speed Acceleration/Deceleration
Number	M1144 ~ M1149, M1154, D1030, D1031, D1144, D1154, D1155

Contents:

1. Special D and special M of Y0 adjustable pulse speed acceleration/deceleration for SA/SX/SC:

Device No.	Function
M1144	Activation switch for Y0 adjustable pulse speed acceleration/deceleration
M1145	Accerlerating flag for Y0 adjustable pulse speed acceleration/deceleration
M1146	"Target frequency reached" flag for Y0 adjustable pulse speed acceleration/deceleration
M1147	"Decerlerating" flag for Y0 adjustable pulse speed acceleration/deceleration
M1148	"Function completed" flag for Y0 adjustable pulse speed acceleration/deceleration
M1149	"Counting temporarily stops" flag for Y0 adjustable pulse speed acceleration/deceleration
M1154	"Enabling deceleration" flag for Y0 adjustable pulse speed acceleration/deceleration
D1030	The lower 16 bits in the 32-bit data register for accumulative Y0 output pulses
D1031	The higher 16 bits in the 32-bit data register for accumulative Y0 output pulses
D1144	Starting No. of the register (D) for Y0 adjustable pulse speed acceleration/deceleration
D1154	Recommended value for indicated gap time of deceleration(10 ~ 32,767 ms)
D1155	Recommended value for indicated gap frequency of deceleration (-1 ~ -32,700 Hz)

2. Parameters for D1144:

Index	Function
+ 0	Total number of sections (n) (max. 10)
+ 1	Currently executed section (read only)
+ 2	Start frequency of the 1st section (SF1)
+ 3	Gap time of the 1st section (GT1)
+ 4	Gap frequency of the 1st section (GF1)
+ 5	Target frequency of the 1st section (TF1)
+ 6	The lower 16 bits of the 32 bits for the target number of output pulses in the 1st section (SE1)
+ 7	The higher 16 bits of the 32 bits for the target number of output pulses in the 1st section (SE1)
+ 8	Start frequency of the 2nd section (SF2); Cannot be the same as TF1
+ 9	Gap time of the 2nd section (GT2)
+ 10	Gap frequency of the 2nd section (GF2)
+ 11	Target frequency of the 2nd section (TF2)
+ 12	The lower 16 bits of the 32 bits for the target number of output pulses in the 2nd section (SE2)
+ 13	The higher 16 bits of the 32 bits for the target number of output pulses in the 2nd section (SE2)
:	:
+ n*6 + 2	Start frequency of the nth section (SFn); Cannot be the same as the start frequency of the n-1th section (TFn-1)
+ n*6 + 3	Gap time of the nth section (GTn)

+ n*6 + 4	Gap frequency of the nth section (GFn)
+ n*6 + 5	Target frequency of the nth section (TFn)
+ n*6 + 6	The lower 16 bits of the 32 bits for the target number of output pulses in the nth section (SEn)
+ n*6 + 7	The higher 16 bits of the 32 bits for the target number of output pulses in the nth section (SEn)

3. The functions:

    This function can only be used on Y0 output point and the timing chart is as follows. After filling out the parameter table, setup M1144 to start the function (should be applied in RUN mode).
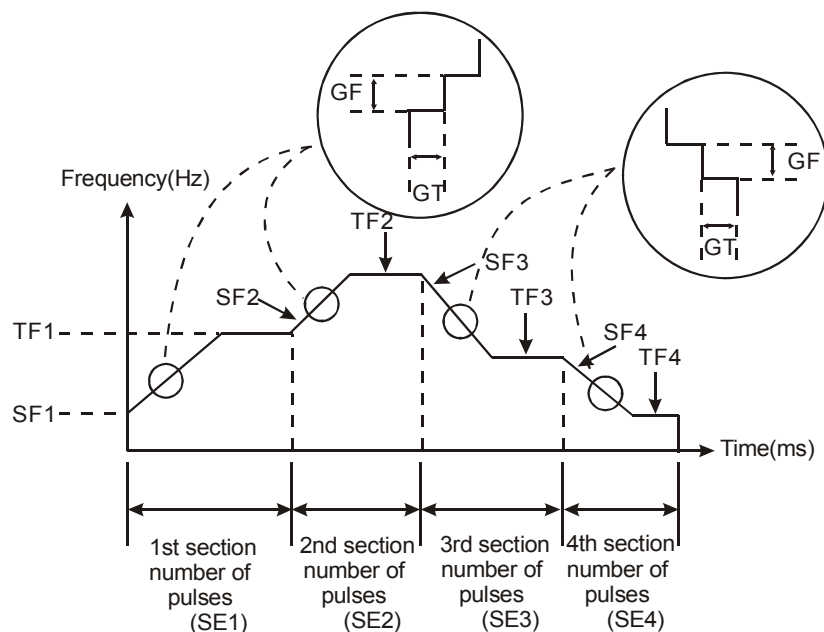
4. How to use and the restrictions:

    a) The start frequency and target frequency have to be ≥ 200Hz; otherwise, the function will not be executed or complete execution.

    b) The start frequency and target frequency have to be < 32,700Hz. Frequency > 32,700Hz will be executed in 32,700Hz.

    c) Range of gap time: 1 ~ 32,767ms (Min. unit: ms)

    d) The range of gap frequency within acceleration section: 1Hz ~ 32,700Hz; within deceleration section: -1 ~ -32,700Hz. If the gap frequency is set as 0Hz, it will fail to reach target frequency in the executed section but in the next section when the target number of pulses is reached.

    e) The target number of pulses in a section has to be > (GF × GT/1,000) × [(TF - SF)/GF]; otherwise the target may not be reached. To correct it, the user may extend gap time or increase the target number of pulses.

    f) When PLC is in RUN status and there is a high-speed instruction assigning Y0 input, the instruction will be executed prior to other instructions.

    g) After M1144 = On, if M1148 has not be reached and M1144 is "Off", deceleration will be enabled. And if M1154 = Off at the moment, the deceleration rule will be "decelerating 200Hz every 200ms" and M1147 will be set. The pulse output will stop when the frequency falls under 200Hz. If M1154 = On at the moment, the output will be executed following the gap time and frequency as set by the user. The time shall not be ≤ 0 (if ≤ 0, it will follow the initial setting 200ms) and the frequency shall not be ≥ 0 (if = 0, it will follow the initial setting -1KHz; if > 0, negative sign comes before the value).

h) When M1148 = On but M1144 = Off, deceleration will not be enabled and M1148 will be reset. Whenever M1144 = Off, M1149 will be reset.

i) The number of sections being executed is determined upon the total number of sections. (Max. number of sections = 10)

j) Acceleration or deceleration is determined upon the start frequency of the next section. That is, if the target frequency of the current section < the start frequency of the next section, acceleration will take place in the next section and the target frequency of the next section must > its start frequency. If the target frequency of the current section > the start frequency of the next section, deceleration will take place in the next section and the target frequency of the next section must < its start frequency. Correct pulse output cannot be guaranteed if the user does not follow the rules.

k) When PLC goes from STOP to RUN, M1144 ~ M1149 will be reset to "Off". When PLC goes from RUN to STOP, only M1144 will be reset, not M1145 ~ M1149.

l) SA/SX/SC uses parameter table D0 ~ D999 and D2000 ~ D4999. If the used parameter table (including all the used section parameters) falls off the range, the instruction will not be executed and M1144 will be "Off".

5. Example 1: Calculate the number of output pulses in every acceleration/deceleration section

Assume you set the start frequency of a section as 200Hz, gap time as 100ms, gap frequency as 100Hz, target frequency as 500Hz, and target number of pulses as 1,000:

- The number of output pulses at start frequency = 200 × 100 / 1,000 = 20
- The number of output pulses in the first acceleration gap = 300 × 100 / 1,000 = 30
- The number of output pulses in the second acceleration gap = 400 × 100 / 1,000 = 40
- The number of output pulses at target frequency = 1,000 – (40 + 30 + 20) = 910
  (Please be noted that we suggest this number be bigger than 10.)
- Output time for target frequency = 1 / 500 × 910 = 1,820 ms
- Total time spent for this section = 1,820 + 3 × 100 = 2,120 ms

6. Example 2: Pulse output program for 1 acceleration section and 1 deceleration section

```
M1002
├─┤├──────┬──┤ MOV │ K200 │ D1144 │
         │
         ├──┤ MOV │ K2   │ D200  │
         │
         ├──┤ MOV │ K250 │ D202  │
         │
         ├──┤ MOV │ K500 │ D203  │
         │
         ├──┤ MOV │ K250 │ D204  │
         │
         ├──┤ MOV │ K1000│ D205  │
         │
         ├──┤ DMOV│ K2000│ D206  │
         │
         ├──┤ MOV │ K750 │ D208  │
         │
         ├──┤ MOV │ K500 │ D209  │
         │
         ├──┤ MOV │ K-250│ D210  │
         │
         ├──┤ MOV │ K250 │ D211  │
         │
         └──┤ DMOV│ K200 │ D212  │

M0
├─┤├──────────( M1144 )

├──────────────┤ END │
```
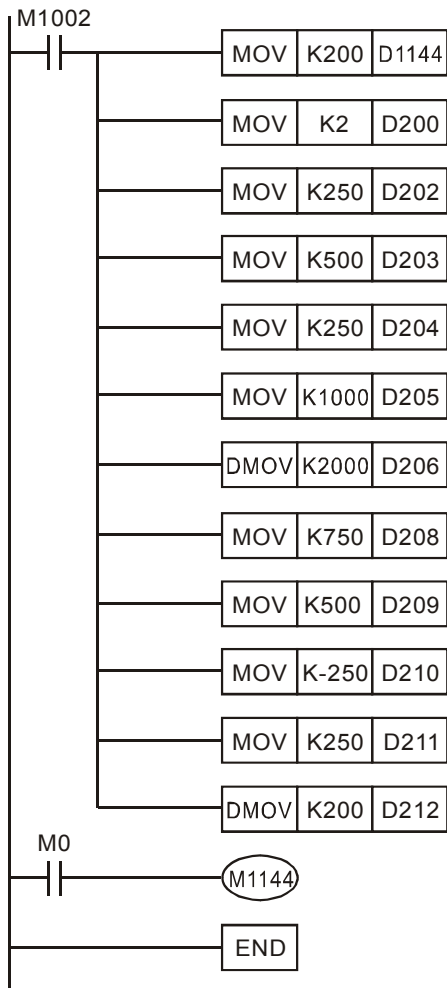
7. Example 3: Acceleration and deceleration in 1 section and the pulse output program with direction switch
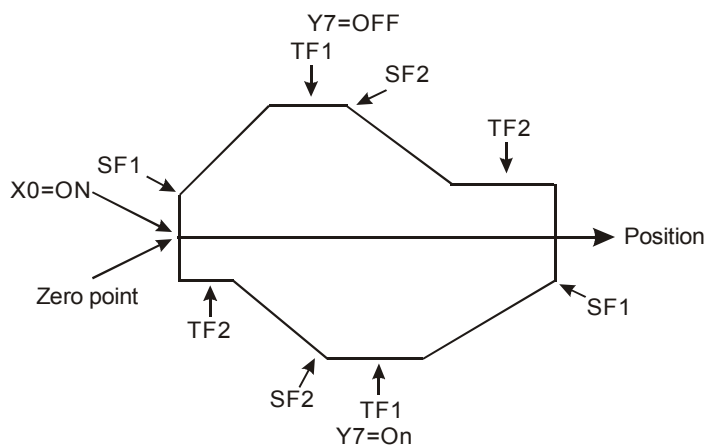
Explanation:

a) See example 2 for the settings for acceleration and deceleration. The acceleration/deceleration frequency is stored in the latched area; therefore, you do not have to write it in in the program.

b) The figure above is the example of the motion. When X0 = On, it will start the motion back and forth; when X0 = Off, the motion will stop. Y7 is a direction switch.
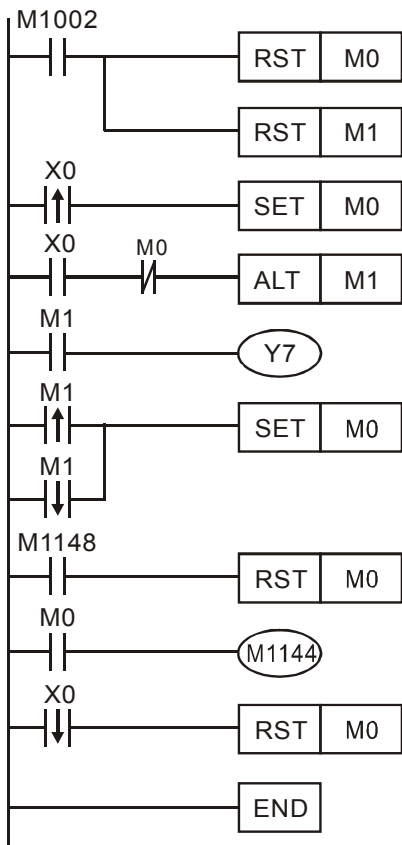
c) The program:

8. Example 4: Program of zero return for 1 acceleration section and 1 deceleration section

  ▪ The timing diagram of relevant flags:
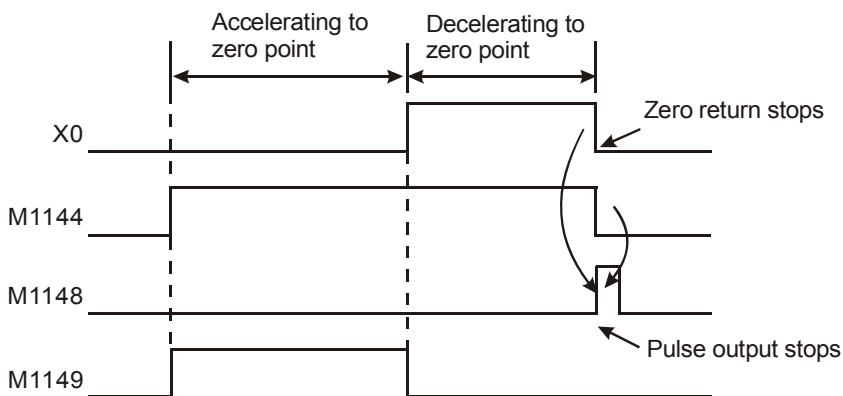
  ▪ Frequency and the positions:
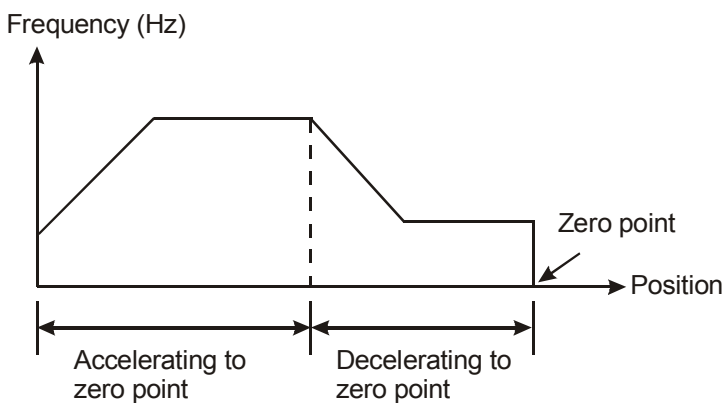
■ Settings of acceleration/deceleration time, frequency and number of pulses:

Started No. D + index value	Setting
+0	2
+2	250 (Hz)
+3	100 (ms)
+4	500 (Hz)
+5	10,000 (Hz)
+6, +7	10 (pulses)
+8	9,750 (Hz)
+9	50 (ms)
+10	-500 (Hz)
+11	250 (Hz)
+12, +13	30,000 (pulses)

■ The program: (Assume X7 is the switch for triggering zero return)

```
X7
 |↑|————————————[ SET  M1144 ]
     |
     └──────────[ SET  M1149 ]

X0
 |↑|————————————[ RST  M1149 ]

X0
 |↓|————————————[ SET  M1148 ]
     |
     └──────────[ RST  M1144 ]

 ————————————————[ END ]
```
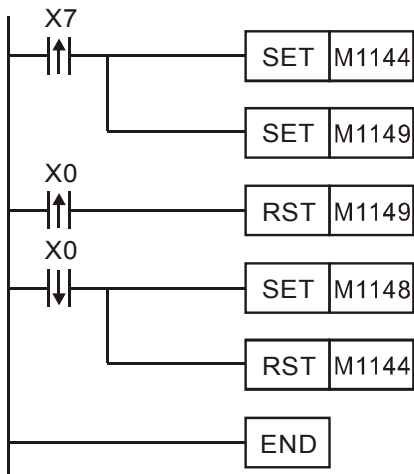
■ Program explanations:

a) When X7 is triggered, M1144 will enable an acceleration. Set M1149 (counting temporarily stops) and 10 pulses will be sent out before the deceleration switch X0 is triggered and the program will enter the deceleration section.

b) When X0 is disabled (i.e. zero return is completed), set M1148 to disable this function.

Note: The example offered here is one of the applications. Please adjust the settings of the parameters according to the features of and restrictions on your machines.

Function Group   Single Step Execution

Number   M1170, M1171, D1170

Contents:

1. Special D and special M for single step execution for EH/EH2/SV:

Device No.	Function
M1170	Start flag
M1171	Action flag
D1170	STEP No. of the currently executed instruction

2. The function:

   a) Execution timing: The flag is valid only when PLC is in RUN status.

   b) Action steps:

      i) When M1170 is enabled, PLC enters the single step execution mode. PLC stays at a specific instruction, stores the location of STEP in D1170 and executes the instsruction once.

      ii) When M1171 is forced "On", PLC executes the next instruction and stops. At the same time, PLC auto-force "O ff" M1171 and stops at the next instruction. D1170 stores the present STEP value.

      iii) When Y output is in single step execution mode, Y outputs immediately without having to wait until END instruction is being executed.

3. Note:

   a) Instruction that will be affected by scan time will be executed incorrectly due to the single step execution. For example, when HKY instruction is executed, it takes 8 scan times to obtain a valid input value from a key. Therefore, the single step execution will result in incorrect actions.

   b) High-speed pulse input/output and high-speed counter comparison instructions are executed by hardware; therefore, they will not be affected by the single step execution.

Function Group   2-phase Pulse Output

Number   M1172 ~ M1174, D1172 ~ D1177

Contents:

1. Special D and special M for two-phase pulse output for SA/SX/SC:

Device No.	Function
M1172	Switch for two-phase pulse output (On = enabled)
M1173	On = Continuous output switch
M1174	"Number of pulses reached" flag
D1172	Output frequency (12Hz ~ 20KHz)
D1173	Output mode (K1 and K2)
D1174	The lower 16 bits of the 32 bits for the target number of pulses
D1175	The higher 16 bits of the 32 bits for the target number of pulses
D1176	The lower 16 bits of the 32 bits for the present number of pulses
D1177	The higher 16 bits of the 32 bits for the present number of pulses

2. The function:

Output frequency = 1/1 pulse cycle period (i.e. 1/T; as the figure below)

There are two output modes. K1 refers to "A-phase ahead of B-phase" and K2 refers to "B-phase ahead of A-phase". The number of pulses accumulates once whenever a phase gap occurs. For example, the number of pulses in the figure below = 8, and when the number is reached, M1174 turns "On". To clear the number, simply turn "Off" M1172.
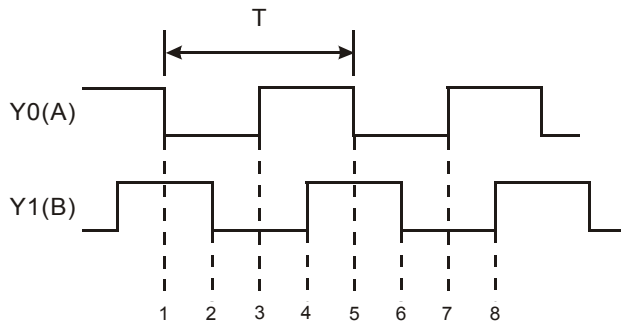
The output frequency, target number of pulses and selection of modes can be modified when M1172 = On and M1174 = Off. Modification on output frequency and target number of pulses will not affect the present number of pulses, but when the mode is modified, the present number of pulses will be cleared as "0". The present number of output pulses is updated in every scan time. When M1133 turns from "Off" to "on", the number will be cleared as "0". When M1172 is cleared as "0" when PLC goes from STOP to RUN. When PLC goes from RUN to STOP, the last number of pulses will be shown.

3.   Note:

This function can only be used when PLC is in RUN status and can coexist with PLSY instruction in the program. If PLSY instruction is executed prior to this function, the function cannot be used and vice versa.

Function Group	VR Volume
Number	M1178 ~ M1179, D1178 ~ D1179

Contents:

1.   Special D and special M for built-in 2-point VR volume for EH/EH2/SV/SA/SC:

Device No.	Function
M1178	Enable VR0 volume
M1179	Enable VR1 volume
D1178	VR0 value
D1179	VR1 value

2.   The function:

This function should be used when PLC is in RUN status. When M1178 = On, VR0 value will be converted into a value of 0 ~ 255 and stored in D1178. When M1179 = On, VR1 value will be converted into a value of 0 ~ 255 and stored in D1179.

3.   See API 85 VRRD for more details.

Function Group	Interruption Instruction for Reading the Number of Pulses
Number	D1180 ~ D1181，D1198 ~ D1199

Contents:

1. SA/SX/SC can use external interruption to store the present value in the middle-high-speed counter into D1180 ~ D1181, D1198 ~ D1199

2. The function:

a) For SA/SX, X0 (pulse input point) has to work with X4 (external interruption point), C235/C251/C253 (high-speed counter) and I401 (interruption No.). D1180 and D1181 are the registers to store the 32-bit values. X1 (pulse input point) has to work with X5 (external interruption point), C236 and I501. D1198 and D1199 are the registers to store the 32-bit values.

b) For SC, X10 (pulse input point) has to work with X4 (external interruption point), C243/C255 (high-speed counter) and I401 (interruption No.). D1180 and D1181 are the registers to store the 32-bit values. X11 (pulse input point) has to work with X5 (external interruption point), C245 and I501. D1198 and D1199 are the registers to store the 32-bit values.

Function Group	MODEM Connection Function
Number	M1184 ~ M1188

Contents:

1. The system connection
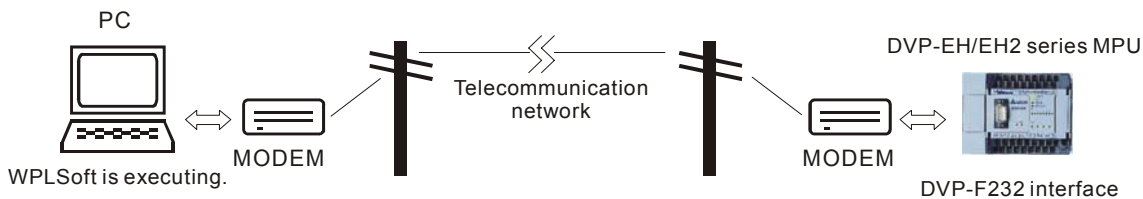
2. Special M for MODEM connection for EH/EH2:

Device No.	Function	Note
M1184	Enable MODEM	On: The following actions are valid
M1185	Initialize MODEM	Off: Initialization is completed
M1186	Fail to initialize MODEM	Off: M1185 = On
M1187	MODEM initialization is completed	Off: M1185 = On
M1188	Shows if MODEM is connected	On: Connecting

Note: The special Ms are both applicable when PLC is in RUN or STOP status.

3. How to conntect (Please follow the steps below):

   a) Set "On" M1184 (Enable PLC MODEM connection).

   b) Set "On" M1185 (Enable initialization of MODEM from PLC).

   c) Check if the initialization of MODEM is successful from M1186, M1187.

   d) Wait for the connection.

4. Note:

   a) When PLC is to be connected with MODEM, a RS-232 extension card is required. If there is no RS-232 extension card, all special M above will be invalid.

b) After enabling MODEM (M1184 = On), PLC has to initialize MODEM first (M1185 = On). If PLC fails to initialize MODEM, the auto-answering function of the MODEM will not be enabled.

c) After MODEM is initialized, it will enter auto-answering mode automatically.

d) If the remote PC is disconnected, MODEM will enter stand-by mode automatically and if the user turns off MODEM now, MODEM will have to be initialized again when it is turned on again.

e) The connection speed is set by PLC as 9,600bps fixed and modification on the speed is not allowed. MODEM has to be able to support the speed of 9,600bps and above.

f) The initialization format from PLC to MODEM are ATZ and ATS0 = 1.

g) If PLC fails to initialize MODEM, use the super terminal in PC to initialize it by the format ATZ and ATS0 = 1.

Function Group   Latched Area

Number   D1200 ~ D1219

Contents:

1. The latched area for EH/EH2/SV/SA/SX/SC is from the start address No. to the end address No.

2. See the tables in Chapter 2.1 for more details.

Function Group   Force On/Off of Input Point X

Number   M1304

Contents:

1. For SA/SX/SC, when M1304 = On, peripheral devices, e.g. WPLSoft or HPP, can force On/Off of X0 ~ X17, but the hardware LED will not respond to it.

2. For EH/EH2/SV, when M1304 = On, peripheral devices, e.g. WPLSoft or HPP, can force On/Off of input point X, and the hardware LED will respond to it.

Function Group   High-speed Output Pulse Stop Mode

Number   M1310 ~ M1311, M1334 ~ M1335, D1166 ~ D1167, D1343 ~ D1353

Contents:

1. Special D and special M for high-speed pulse output stop mode: (SC_V1.4 and versions above are with an additional mode 3)

Device No.	Function
M1334	Select stop mode for Y10 pulse
M1335	Select stop mode for Y11 pulse
M1310	Immediately stop Y10 pulse output
M1311	Immediately stop Y11 pulse output
D1166	X10 rising-edge/falling-edge counting mode switch
D1167	X11 rising-edge/falling-edge counting mode switch
D1343	Acceleration/deceleration time for Y10 pulse output
D1353	Acceleration/deceleration time for Y11 pulse output

2. How do Y10 pulse output stop modes work:

a) Using Y10 pulse output

■ **Mode 1 – Planned deceleration**

Applicable to: DDRVI and DDRVA instructions

Criteria for executing planned deceleration: Shut down the criteria contact for pulse output instruction and turn "Off" M1334.

The time from executing planned deceleration to the end of pulse output: The time set in D1343 (for acceleration/deceleration)

The solid lines in the figure below are the originally planned routes and the dotted lines refer to the routes after planned deceleration is executed.
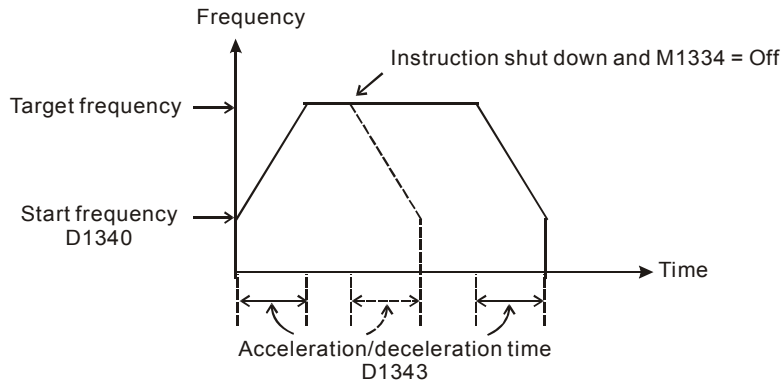
■ **Mode 2 – Output shutdown**

Applicable to: DDRVI, DDRVA, PLSY instructions

Criteria for executing output shutdown: Shut down the criteria contact for pulse output instruction and turn "On" M1334.(Because PLSY does not have acceleration/deceleration setting, M1334 does not need to be set in PLSY)

The time from executing output shutdown to the end of pulse output: Max. 1 scan cycle.

The solid lines in the figure below are the originally planned routes and the dotted lines refer to the routes after output shutdown is executed.
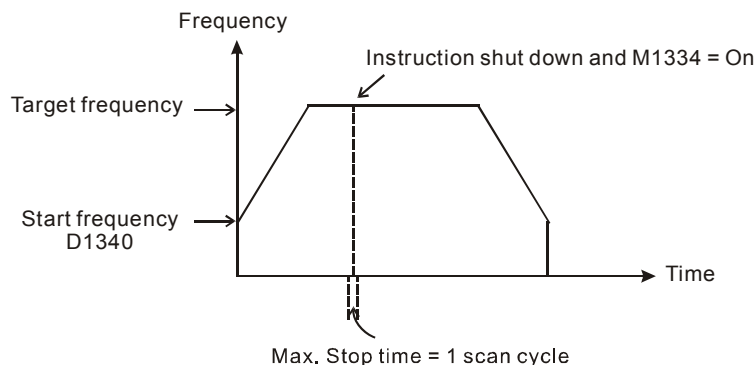
■ **Mode 3 – Immediate output shutdown**

Applicable to: DDRVI, DDRVA, PLSY instructions

Criteria for executing immediate output shutdown: M1310 = On (set before executing the instruction) and the criteria triggers set in X10 (D1166 = K0 refers to rising-edge; D1166 = K1 refers to falling-edge)

The time from executing immediate output shutdown to the end of pulse output: Max. 1 pulse time.

The solid lines in the figure below are the originally planned routes and the dotted lines refer to the routes after X10 is triggered.
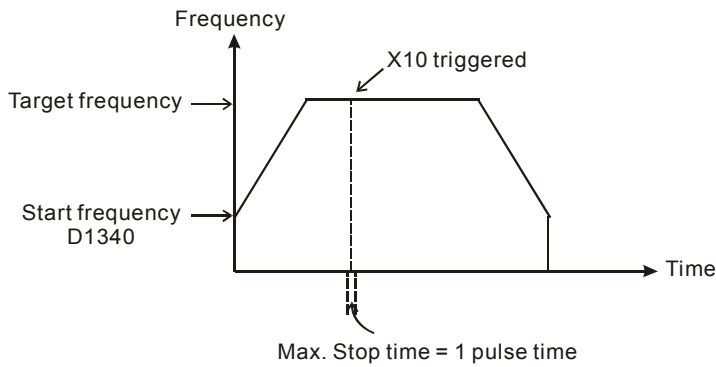
b) Using Y11 pulse output

■ **Mode 1 – Planned deceleration**

Applicable to: DDRVI and DDRVA instructions

Criteria for executing planned deceleration: Shut down the criteria contact for pulse output instruction and turn "Off" M1335.

The time from executing planned deceleration to the end of pulse output: The time set in D1353 (for acceleration/deceleration)

■ **Mode 2 – Output shutdown**

Applicable to: DDRVI, DDRVA, PLSY instructions

Criteria for executing output shutdown: Shut down the criteria contact for pulse output instruction and turn "On" M1335. (Because PLSY does not have acceleration/deceleration setting, M1335 does not need to be set in PLSY)

The time from executing output shutdown to the end of pulse output: Max. 1 scan cycle.

■ **Mode 3 – Immediate output shutdown**

Applicable to: DDRVI, DDRVA, PLSY instructions

Criteria for executing immediate output shutdown: M1311 = On (set before executing the instruction) and the criteria triggers set in X11 (D1167 = K0 refers to rising-edge; D1167 = K1 refers to falling-edge)

The time from executing immediate output shutdown to the end of pulse output: Max. 1 pulse time.

3. Note:

a) The execution criteria M1334 and M1335 for mode 1 and 2 have to be set before executing pulse output shutdown instruction. The execution criteria M1310, M1311 and trigger criteria D1166, D1167 for mode 3 have to be set before the pulse output instruction is executed.

b) In mode 3 (immediate output shutdown), Y10 can only be used with X10 and Y11 with X11.

c) When using X10 or X11 in mode 3, DO NOT use X10 or X11 as the input high-speed counter.

Function Group   Right-Side Special Extension Module ID

Number   D1320 ~ D1327

Contents:

1. The ID of special extension module, if any, connected to EH/EH2/SV are stored in D1320 ~ D1327 in sequence.

2. Special extension module ID for EH:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-H	H'0400	DVP01PU-H	H'0110

DVP04DA-H	H'0401	DVP01HC-H	H'0120
DVP04PT-H	H'0402	DVP02HC-H	H'0220
DVP04TC-H	H'0403	DVP01DT-H	H'0130
DVP06XA-H	H'0604	DVP02DT-H	H'0230

3.   Special extension module ID for EH2:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-H2	H'6400	DVP01PU-H2	H'6110
DVP04DA-H2	H'6401	DVP01HC-H2	H'6120
DVP04PT-H2	H'6402	DVP02HC-H2	H'6220
DVP04TC-H2	H'6403	DVP01DT-H2	H'6130
DVP06XA-H2	H'6604	DVP02DT-H2	H'6230

4.   Special extension module ID for SV:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-S	H'0088	DVP04PT-S	H'008A
DVP06AD-S	H'00C8	DVP04TC-S	H'008B
DVP02DA-S	H'0049	DVP06XA-S	H'00CC
DVP04DA-S	H'0089	DVP01PU-S	H'0110

Function Group   Left-Side High-Speed Special Extension Module ID

Number   D1386 ~ D1393

Contents:

1.   The ID of left-side special extension module, if any, connected to SV are stored in D1386 ~ D1393 in sequence.

2.   Left-side special extension module ID for SV:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-SL	H'4400	DVP01HC-SL	H'4120
DVP04DA-SL	H'4401	DVP02HC-SL	H'4220
DVP04PT-SL	H'4402	DVPDNET-SL	H'4130
DVP04TC-SL	H'4403	DVPEN01-SL	H'4050
DVP06XA-SL	H'6404	DVPMDM-SL	H'4040
DVP01PU-SL	H'4110		

Function Group   EASY EASY PLC LINK

Number   M1350 ~ M1354, M1360 ~ M1519, D1399, D1355 ~ D1370, D1415 ~ D1465, D1480 ~ D1991

Contents:

1. Special D and special M for ID1 ~ ID8 of the 16 stations in EASY EASY PLC LINK (M1353 = Off) for

    SA/SX/SC/EH/EH2/SV:

MASTER PLC															
SLAVE ID 1		SLAVE ID 2		SLAVE ID 3		SLAVE ID 4		SLAVE ID 5		SLAVE ID 6		SLAVE ID 7		SLAVE 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 = Off: Disable 32 stations in the Link and the function of reading/writing more than 16 data (RST M1353); the No. of special D for storing the 16 read/written data.															
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
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
D1434	D1450	D1435	D1451	D1436	D1452	D1437	D1453	D1438	D1454	D1439	D1455	D1440	D1456	D1441	D1457
Start Communication Address															
D1355	D1415	D1356	D1416	D1357	D1417	D1358	D1418	D1359	D1419	D1360	D1420	D1361	D1421	D1362	D1422
LINK in SLAVE PLC?															
M1360		M1361		M1362		M1363		M1364		M1365		M1366		M1367	
Action flag for SLAVE PLC from MASTER PLC															
M1376		M1377		M1378		M1379		M1380		M1381		M1382		M1383	
"Read/write error" flag															
M1392		M1393		M1394		M1395		M1396		M1397		M1398		M1399	
"Reading completed" flag (turns "Off" whenever read/write a station is completed)															
M1408		M1409		M1410		M1411		M1412		M1413		M1414		M1415	
"Writing completed" flag (turns "Off" whenever read/write a station is completed)															
M1424		M1425		M1426		M1427		M1428		M1429		M1430		M1431	

| ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |

SLAVE ID 1		SLAVE ID 2		SLAVE ID 3		SLAVE ID 4		SLAVE ID 5		SLAVE ID 6		SLAVE ID 7		SLAVE 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

- Default start communication address D1355 ~ D1362 to be read = H1064 (D100)

- Default start communication address D1415 ~ D1422 to be written = H10C8 (D200)

2. Special D and special M for ID9 ~ ID16 of the 16 stations in EASY EASY PLC LINK (M1353 = Off) for

   SA/SX/SC/EH/EH2/SV:

MASTER PLC															
SLAVE ID 9		SLAVE ID 10		SLAVE ID 11		SLAVE ID 12		SLAVE ID 13		SLAVE ID 14		SLAVE ID 15		SLAVE 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 = Off: Disable 32 stations in the Link and the function of reading/writing more than 16 data (RST M1353); the No. of special D for storing the 16 read/written data.															
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
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
D1442	D1458	D1443	D1459	D1444	D1460	D1445	D1461	D1446	D1462	D1447	D1463	D1448	D1464	D1449	D1465
Start Communication Address															
D1363	D1423	D1364	D1424	D1365	D1425	D1366	D1426	D1367	D1427	D1368	D1428	D1369	D1429	D1370	D1430
LINK in SLAVE PLC?															
M1368		M1369		M1370		M1371		M1372		M1373		M1374		M1375	
Action flag for SLAVE PLC from MASTER PLC															
M1384		M1385		M1386		M1387		M1388		M1389		M1390		M1391	
"Read/write error" flag															
M1400		M1401		M1402		M1403		M1404		M1405		M1406		M1407	
"Reading completed" flag (turns "Off" whenever read/write a station is completed)															
M1416		M1417		M1418		M1419		M1420		M1421		M1422		M1423	
"Writing completed" flag (turns "Off" whenever read/write a station is completed)															
M1432		M1433		M1434		M1435		M1436		M1437		M1438		M1439	

↓	↓	↓	↓	↓	↓	↓	↓

SLAVE ID 9		SLAVE ID 10		SLAVE ID 11		SLAVE ID 12		SLAVE ID 13		SLAVE ID 14		SLAVE ID 15		SLAVE 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

- Default start communication address D1363 ~ D1370 to be read = H1064 (D100)

- Default start communication address D1423 ~ D1430 to be written = H10C8 (D200)

3. Special D and special M for ID1 ~ ID8 of the 32 stations in EASY EASY PLC LINK (M1353 = On) for EH/EH2/SV:

MASTER PLC															
SLAVE ID 1		SLAVE ID 2		SLAVE ID 3		SLAVE ID 4		SLAVE ID 5		SLAVE ID 6		SLAVE ID 7		SLAVE 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 32 stations in the Link and the function of reading/writing more than 16 data (SET M1353); the No. of D registers for storing the read/written data.															
D1480	D1496	D1481	D1497	D1482	D1498	D1483	D1499	D1484	D1500	D1485	D1501	D1486	D1502	D1487	D1503
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
D1434	D1450	D1435	D1451	D1436	D1452	D1437	D1453	D1438	D1454	D1439	D1455	D1440	D1456	D1441	D1457
Start Communication Address															
D1355	D1415	D1356	D1416	D1357	D1417	D1358	D1418	D1359	D1419	D1360	D1420	D1361	D1421	D1362	D1422
LINK in SLAVE PLC?															
M1360		M1361		M1362		M1363		M1364		M1365		M1366		M1367	
Action flag for SLAVE PLC from MASTER PLC															
M1376		M1377		M1378		M1379		M1380		M1381		M1382		M1383	
"Read/write error" flag															
M1392		M1393		M1394		M1395		M1396		M1397		M1398		M1399	
"Reading completed" flag (turns "Off" whenever read/write a station is completed)															
M1408		M1409		M1410		M1411		M1412		M1413		M1414		M1415	
"Writing completed" flag (turns "Off" whenever read/write a station is completed)															
M1424		M1425		M1426		M1427		M1428		M1429		M1430		M1431	

SLAVE ID 1		SLAVE ID 2		SLAVE ID 3		SLAVE ID 4		SLAVE ID 5		SLAVE ID 6		SLAVE ID 7		SLAVE 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

- Default start communication address D1355 ~ D1362 to be read = H1064 (D100)

- Default start communication address D1415 ~ D1422 to be written = H10C8 (D200)

4. Special D and special M for ID9 ~ ID16 of the 32 stations in EASY EASY PLC LINK (M1353 = On) for EH/EH2/SV:

MASTER PLC															
SLAVE ID 9		SLAVE ID 10		SLAVE ID 11		SLAVE ID 12		SLAVE ID 13		SLAVE ID 14		SLAVE ID 15		SLAVE 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 32 stations in the Link and the function of reading/writing more than 16 data (SET M1353); the No. of D registers for storing the read/written data.															
D1488	D1504	D1489	D1505	D1490	D1506	D1491	D1507	D1492	D1508	D1493	D1509	D1494	D1510	D1495	D1511
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
D1442	D1458	D1443	D1459	D1444	D1460	D1445	D1461	D1446	D1462	D1447	D1463	D1448	D1464	D1449	D1465
Start Communication Address															
D1363	D1423	D1364	D1424	D1365	D1425	D1366	D1426	D1367	D1427	D1368	D1428	D1369	D1429	D1370	D1430
LINK in SLAVE PLC?															
M1368		M1369		M1370		M1371		M1372		M1373		M1374		M1375	
Action flag for SLAVE PLC from MASTER PLC															
M1384		M1385		M1386		M1387		M1388		M1389		M1390		M1391	
"Read/write error" flag															
M1400		M1401		M1402		M1403		M1404		M1405		M1406		M1407	
"Reading completed" flag (turns "Off" whenever read/write a station is completed)															
M1416		M1417		M1418		M1419		M1420		M1421		M1422		M1423	
"Writing completed" flag (turns "Off" whenever read/write a station is completed)															
M1432		M1433		M1434		M1435		M1436		M1437		M1438		M1439	

SLAVE ID 9		SLAVE ID 10		SLAVE ID 11		SLAVE ID 12		SLAVE ID 13		SLAVE ID 14		SLAVE ID 15		SLAVE 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

- Default start communication address D1363 ~ D1370 to be read = H1064 (D100)

- Default start communication address D1423 ~ D1430 to be written = H10C8 (D200)

5.   Special D and special M for ID17 ~ ID24 of the 32 stations in EASY EASY PLC LINK (M1353 = On) for EH/EH2/SV:

MASTER PLC															
SLAVE ID 17		SLAVE ID 18		SLAVE ID 19		SLAVE ID 20		SLAVE ID 21		SLAVE ID 22		SLAVE ID 23		SLAVE 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
M1353 = On: Enable 32 stations in the Link and the function of reading/writing more than 16 data (SET M1353); the No. of D registers for storing the read/written data.															
D1576	D1592	D1577	D1593	D1578	D1594	D1579	D1595	D1580	D1596	D1581	D1597	D1582	D1598	D1583	D1599
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
start Communication Address															
D1512	D1528	D1513	D1529	D1514	D1530	D1515	D1531	D1516	D1532	D1517	D1533	D1518	D1534	D1519	D1535
LINK in SLAVE PLC?															
M1440		M1441		M1442		M1443		M1444		M1445		M1446		M1447	
Action flag for SLAVE PLC from MASTER PLC															
M1456		M1457		M1458		M1459		M1460		M1461		M1462		M1463	
"Read/write error" flag															
M1472		M1473		M1474		M1475		M1476		M1477		M1478		M1479	
"Reading completed" flag (turns "Off" whenever read/write a station is completed)															
M1488		M1489		M1490		M1491		M1492		M1493		M1494		M1495	
"Writing completed" flag (turns "Off" whenever read/write a station is completed)															
M1504		M1505		M1506		M1507		M1508		M1509		M1510		M1511	

SLAVE ID 17		SLAVE ID 18		SLAVE ID 29		SLAVE ID 20		SLAVE ID 21		SLAVE ID 22		SLAVE ID 23		SLAVE 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)

6. Special D and special M for ID25 ~ ID32 of the 32 stations in EASY EASY PLC LINK (M1353 = On) for EH/EH2/SV:

MASTER PLC															
SLAVE ID 25		SLAVE ID 26		SLAVE ID 27		SLAVE ID 28		SLAVE ID 29		SLAVE ID 30		SLAVE ID 31		SLAVE 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
M1353 = On: Enable 32 stations in the Link and the function of reading/writing more than 16 data (SET M1353); the No. of D registers for storing the read/written data.															
D1584	D1600	D1585	D1601	D1586	D1602	D1587	D1603	D1588	D1604	D1589	D1605	D1590	D1606	D1591	D1607
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
Start Communication Address															
D1520	D1536	D1521	D1537	D1522	D1538	D1523	D1539	D1524	D1540	D1525	D1541	D1526	D1542	D1527	D1543
LINK in SLAVE PLC?															
M1448		M1449		M1450		M1451		M1452		M1453		M1454		M1455	
Action flag for SLAVE PLC from MASTER PLC															
M1464		M1465		M1466		M1467		M1468		M1469		M1470		M1471	
"Read/write" error flag															
M1480		M1481		M1482		M1483		M1484		M1485		M1486		M1487	
"Reading completed" flag (turns "Off" whenever read/write a station is completed)															
M1496		M1497		M1498		M1499		M1500		M1501		M1502		M1503	
"Writing completed" flag (turns "Off" whenever read/write a station is completed)															
M1512		M1513		M1514		M1515		M1516		M1517		M1518		M1519	

SLAVE ID 25		SLAVE ID 26		SLAVE ID 27		SLAVE ID 28		SLAVE ID 29		SLAVE ID 30		SLAVE ID 31		SLAVE 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)

7. Note:

    a) EASY EASY PLC LINK is based on MODBUS communication protocol.

    b) EH/EH2/SV supports 32 stations in the LINK and reading/writing of more than 16 data (SET1353) (M1353 = On). SA/SX/SC supports 16 devices in the LINK and reading/writing of 16 data.

    c) EH/EH2/SV: When a MASTER PLC and a Slave PLC is connected, they are able to read/write maximum 100 WORD data (M1353 = On). SA/SX/SC: Does not support M1353. When a Master PLC and a Slave PLC is connected, they are able to read/write maximum 16 WORD data.

    d) When the Master PLC is connected through COM2 (RS-485), baud rates and communication formats of all Slave PLCs must be the same (set in D1120). When SA/SX/SC/EH/EH2/SV serves as Master, it supports ASCII and RTU format.

    e) When the Slave PLC is connected through COM2 (RS-232/RS-485/RS-422), baud rates and communication formats of all connected Slave PLCs must be the same as those in the Master PLC (set in D1120). When SA/SX/SC/EH/EH2/SV serves as Slave, it supports ASCII and RTU format.

    f) When the Slave PLC is connected through COM1 (RS-232), baud rates and communication formats of all connected Slave PLCs must be the same as those in the Master PLC (set in D1036). When SA/SX/SC/EH/EH2/SV serves as Slave, it supports ASCII and RTU format.

    g) When the Slave PLC is connected through COM3 (RS-232/RS-485), baud rates and communication formats of all connected Slave PLCs must be the same as those in the Master PLC (set in D1109). When SA/SX/SC/EH/EH2/SV serves as Slave, it only supports ASCII format (Max. baud rate = 38,400bps).

    h) The start station No. (K1 ~ K214) of Slave ID1 is assigned by D1399 of Master PLC. Station No. of every Slave and Master PLC can not be the same (set in D1121).

    i) For one-to-one LINK: Connected through RS-232, RS-485, RS-422. PLC COM1, COM2, COM3 support many communication formats.

    j) For one-to-many LINK: Connected through RS-485. PLC COM1, COM2, COM3 support many communication formats.

8. How to operate EASY EASY PLC LINK:

    a) Set up the baud rates and communication formats of Master PLC and all connected Slave PLCs and make them the same. COM1_RS-232: D1136; COM2_RS-232/RS-485/RS-422: D1120; COM3_RS-232/RS-485: D1109.

    b) Set up the station No. of Master PLC (in D1121) and assign the start station No. of Slave PLC from D1399 of Master PLC. Next, set up the station No. of Slave PLC. Station No. of Master and Slave cannot be the same.

    c) Set up the number of connected Slave stations and the number of data to be read in/written to Slave stations. For EH/EH2/SV (M1353 = On): Enable the function of the 32 conncected Slaves and reading/writing of more than 16 data (Max. 100 data). Next, set up the No. of D registers for storing the read data (D1480 ~ D1495, D1576 ~ D1591) and written data (D1496 ~ D1511, D1592 ~ D1607) (See the explanations above on special D). SA/SX/SC only supports reading/writing of 16 data.

    d) Set up the length of data to be read from/written into the Slave. (If the user does not set up set them up, PLC will follow the initial setting or the setting set in the previous operation.) (See the explanations above on special D.)

e) Set up the start communication address of the Slave to be read/written. (See the explanations above on special D). The default start communication address of Slave to be read: H1064 = D100. The default start communication address of Slave to be written: H10C8 = D200.

f) Operation procedure:

   i) Enable the function of more than 32 stations connected to EASY EASY PLC LINK and reading/writing of 16 data (Max. 100 data) (M1353).

   ii) Enable reading/writing of EASY EASY PLC LINK in the same polling (M1354).

   iii) Set up EASY EASY PLC LINK as auto mode (M1351), or manual mode (M1352) and the times of polling (D1431).

   iv) Enable EASY EASY PLC LINK (M1350).

9. How does Master PLC work:

   a) To detect station No. of Slave: Set up the LINK to be automatic mode (M1351 = On) or manual mode (M1352 = On) (Note: M1351 and M1352 cannot be "On" at the same time) When M1350 = On, Master PLC starts to detect the total number of Slave stations connected to the LINK and records the number in D1433. The detection time may differ upon the number of Slave stations and the setting of communication timeout in D1129.

   b) When M1360 ~ M1375, M1440 ~ M1455 = On, Slave ID1 ~ ID32 exist.

   c) If the detected number Slave stations is 0, M1350 will be "Off" and the LINK will be stopped.

   d) M1353 and M1354 should be set before EASY EASY PLC LINK is enabled. When EASY EASY PLC LINK is executed, the two special M will not afftec the action of EASY EASY PLC LINK.

   e) When 1353 = On, D1480 ~ D1607 (read/write buffer of SLAVE ID1 ~ 4) will be used for storing the No. of registers D for Slave ID1 ~ ID32, The No. of register D shall not be "> 9,900" or "< 0" ; otherwise, PLC will auto-set it as 9,900. The data length shall not be " > 100" or "<0"; otherwise, PLC will auto-set it as 100.

   f) When M1354 = On, set up Modbus Function H17 (synchronous read/write) for EASY EASY PLC LINK communication. If the number of written data is set to be "0", the communication of EASY EASY PLC LINK will be automatically converted into Modbus Function H03 (read multiple words). Similarly, if the number of read data is set to be "0", the communication of EASY EASY PLC LINK will be converted into Modbus Function H06 (write 1 word) or Modbus Function H10 (write multiple words).

   g) If the number of read/written data > 16, the time for timeout (D1129) shall be > 500ms in case communication timeout may occur.

   h) After M1350 = On, Master PLC detects the ID of Slave E PLC only right after the LINK is enabled and will not detect the ID again afterward.

   i) After the detection on Slave PLCs is completed, Master PLC starts to read and write data from/into every Slave PLC. Please be noted that, Master PLC only reads/writes data from/into the detected Slave PLC. Master PLC will not read/write from/into the new Slave PLC to the LINK, unless it re-detects the ID of Slave PLCs.

   j) Master PLC conducts reading before writing. The ranges of Slave PLCs to be read/written will follow the setting.

   k) Master PLC will move to the reading/writing of the next Slave PLC after finishing reading/writing the current Slave PLC.

10. Auto mode and maual mode:

   a) Auto mode (M1351 = On): Master PLC automatically reads/writes from/to Slave PLC and stops when M1350/M1351 = Off to terminate EASY EASY PLC LINK.

   b) Manual mode (M1352 = On): When M1352 = On, you also have to set up the times of polling in D1431. One time of polling refers to the completion of reading and writing all Slaves. After the EASY PLC LINK is enabled, D1432 will start to count how many LINKs have been done. When D1431 = D1432, EASY EASY PLC LINK stops and M1352 is reset. To re-enable EASY EASY PLC LINK in the manual mode, simply turn "On" M1352 and D1431 will starts to count the times of LINK again.

   c) Note:

   i) Automatic mode (M1351) and manual mode (M1352) cannot be "On" at the same time; otherwise EASY EASY PLC LINK will stop and M1350 will be reset.

   ii) For EH/EH2/SV: M1350 has to be reset before switching between automatic mode and manual mode. For SA/SX/SC: No such restriction

   iii) The communication timeout is adjustable (D1129, range: 200 ≤ D1129 ≤ 3,000). If D1129 falls out the range, PLC will determine the time by 200 or 3,000. The timeout setting of EASY EASY PLC LINK is only valid if it is set before the LINK is enabled. If the number of read/written data >16, the communication timeout shall be > 500ms in case a communication timeout may occur.

   iv) EASY EASY PLC LINK is only workable when the baud rate is > 1,200bps. If the baud rate is < 9,600, the communication timeout setting shall be > 1 second.

   v) The communication is unworkable when the number of read/written data = 0.

   vi) EASY EASY PLC LINK does not support the reading/writing from/to 32-bit counters (C200 ~ C255).

   vii) The maximum set value for D1399 is 230. If the set value is bigger than 230, PLC will automatically correct it as 230. The minimum set value for D1300 is 1. If the set value is smaller than 1, PLC will automatically correct it as 1.

   viii) Setting up of D1399 has to be done before EASY EASY PLC LINK is enabled. After EASY EASY PLC LINK is enabled, setting up D1399 will not result in any changes.

   ix) Advantages (when using a multi-layer network): Assume you are using a network with 3 layers and the first and second layer and the second and third layer are using EASY EASY PLC LINK for communication, the IDs in the second and third layer will definitely overlap due to the old version of EASY PLC LINK detects only Slave ID#1 ~ 16. When the IDs of Slave and Master overlap, EASY PLC LINK will ignore the PLC of overlapping ID, resulting in the situation that the third layer can have only 15 PLCs. Therefore, D1399 allows more PLCs connected in a multi-layer network.

11. Operation Procedure of EASY EASY PLC LINK

```
┌─────────────────────────────────────────────────┐
│ Set up the Slave ID# to be read                 │
│ Set up the number of data in Slave to be read   │
│ Set up the Slave ID# to be written              │
│ Set up the number of data in Slave to be written│
│ (If there is no set value, use the previous set │
│ value or the default value.)                    │
└─────────────────────────────────────────────────┘
```

Enable ◄──── Enable 32 slaves linkage and up to 100 data for exchange ────► Disable

```
┌──────────────────────────────────┐   ┌──────────────────────────────────┐
│ 1. Enable 32 slaves linkage and  │   │ Disable 32 slaves linkage and up │
│    up to 100 data for exchange   │   │ to 100 data for exchange.        │
│    (set M1353).                  │   │ (Reset M1353)                    │
│ 2. Set up the register D for     │   └──────────────────────────────────┘
│    storing the read data.        │
│ 3. Set up the register D for     │
│    storing the written data.     │
└──────────────────────────────────┘
```

Enable ◄──── Communicating by Modbus 0X17 ────► Disable

Set M1354   Reset M1354

Enable auto mode ◄──── Enable auto/maual mode of EASY PLC LINK ────► Enable manual mode

Set M1351

```
┌──────────────────────────┐
│ Set M1352                │
│ Set times of communication│
│ (D1431)                  │
└──────────────────────────┘
```

```
┌──────────────────────────────────┐
│ Set M1350                        │
│ Start to execute EASY PLC LINK   │
└──────────────────────────────────┘
```

a) <u>Example 1:</u> Enabling 32 slave units linkage and up to 100 data for exchange in EASY EASY PLC LINK by M1353

```
  M1002
──┤├──────┬─────[ SET    M1353 ]
          │
          ├─────[ MOV    K17    D1121 ]
          │
          ├─────[ MOV    H86    D1120 ]
          │
          ├─────[ SET    M1120 ]
          │
          ├─────[ MOV    K2000  D1480 ]
          │
          ├─────[ MOV    K3000  D1496 ]
          │
          ├─────[ MOV    K300   D1129 ]
          │
          ├─────[ MOV    K48    D1434 ]
          │
          └─────[ MOV    K48    D1450 ]
  X1
──┤├──────┬─────( M1351 )
          │
          └─────( M1350 )

                 [ END ]
```
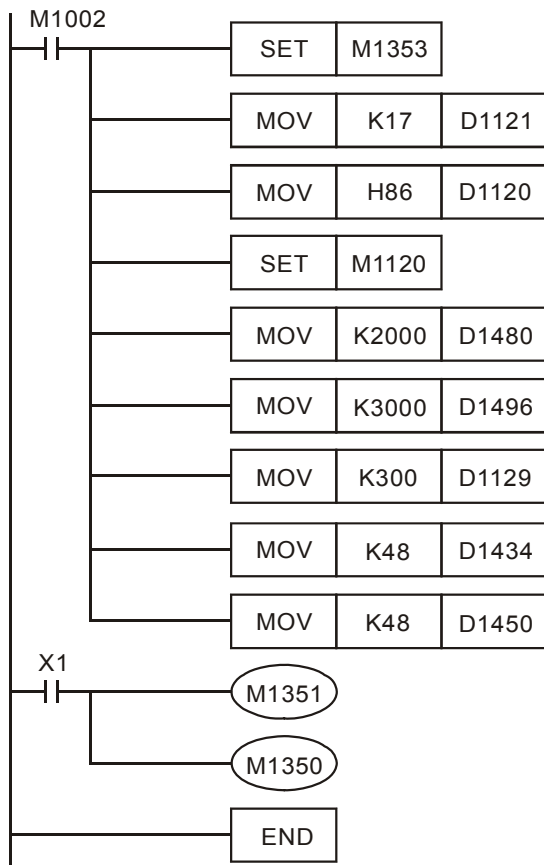
- M1353 has to be set On before EASY EASY PLC LINK is enabled. When EASY EASY PLC LINK is executed, On/Off of M1353 will not affect the execution.

- Registers designated in D1480 ~ D1495, D1576~D1591 (starting register for data read from Slave ID#1 ~ 16) and D1496 ~ D1511, D1592 ~ D1607 (starting register for data written into Slave ID#1 ~ 16) can only be register D, and every special D will correspond to one Slave ID#, e.g. D1480 corresponds to ID1, D1481 to ID2, and so on.

- D1480 ~ D1495, D1576 ~ D1591, D1496 ~ D1511 and D1592 ~ D1607 have to be set before EASY EASY PLC LINK is enabled. In the execution of EASY EASY PLC LINK, you can modify the contents in these special Ds, but the modified results will take effect in the next EASY EASY PLC LINK polling.

- If the ID# designated by D1480 ~ D1495, D1576 ~ D1591, D1496 ~ D1511 and D1592 ~ D1607 is smaller than 0 or bigger than 9,900, PLC will automatically correct the ID# into 9,900.

- If M1353 is not enabled during the execution of EASY PLC LINK, the range for D1434 ~ D1449, D1544 ~ D1559 (number of data read from Slave ID#) and D1450 ~ D1465, D1560 ~ D1575 (numbere of data written into Slave ID#) will be 0 ~ 16. If the setting in the special D exceeds the range, PLC will correct it to 16. When M1353 is enabled, the range will be 0 ~ 100. If the setting in the special D exceeds the range, PLC will correct it to 100.

■ You can modify the settings in D1434 ~ D1449, D1544 ~ D1559, D1450 ~ D1465 and D1560 ~ D1575 during the execution of EASY EASY PLC LINK, but the modified results will take effect in the next EASY EASY PLC LINK polling.

b) Example 2: EASY EASY PLC LINK with M1353 and I170

```
      M1002
      ─┤├──┬──────┌─────┬──────┐
            │      │ SET │ M1353│
            │      └─────┴──────┘
            │      ┌─────┬──────┬──────┐
            ├──────│ MOV │ K17  │ D1121│
            │      └─────┴──────┴──────┘
            │      ┌─────┬──────┬──────┐
            ├──────│ MOV │ H86  │ D1120│
            │      └─────┴──────┴──────┘
            │      ┌─────┬──────┐
            ├──────│ SET │ M1120│
            │      └─────┴──────┘
            │      ┌─────┬──────┬──────┐
            ├──────│ MOV │ K2000│ D1480│
            │      └─────┴──────┴──────┘
            │      ┌─────┬──────┬──────┐
            ├──────│ MOV │ K3000│ D1496│
            │      └─────┴──────┴──────┘
            │      ┌─────┬──────┬──────┐
            ├──────│ MOV │ K300 │ D1129│
            │      └─────┴──────┴──────┘
            │      ┌─────┬──────┬──────┐
            ├──────│ MOV │ K48  │ D1434│
            │      └─────┴──────┴──────┘
            │      ┌─────┬──────┬──────┐
            └──────│ MOV │ K48  │ D1450│
                   └─────┴──────┴──────┘
      X1
      ─┤├──┬──( M1350 )
            │
            └──( M1351 )

      ──────┌──────┐
            │ FEND │
            └──────┘
      I170  ┌──────┐
      ──────│ IRET │
            └──────┘
            ┌──────┐
      ──────│ END  │
            └──────┘
```
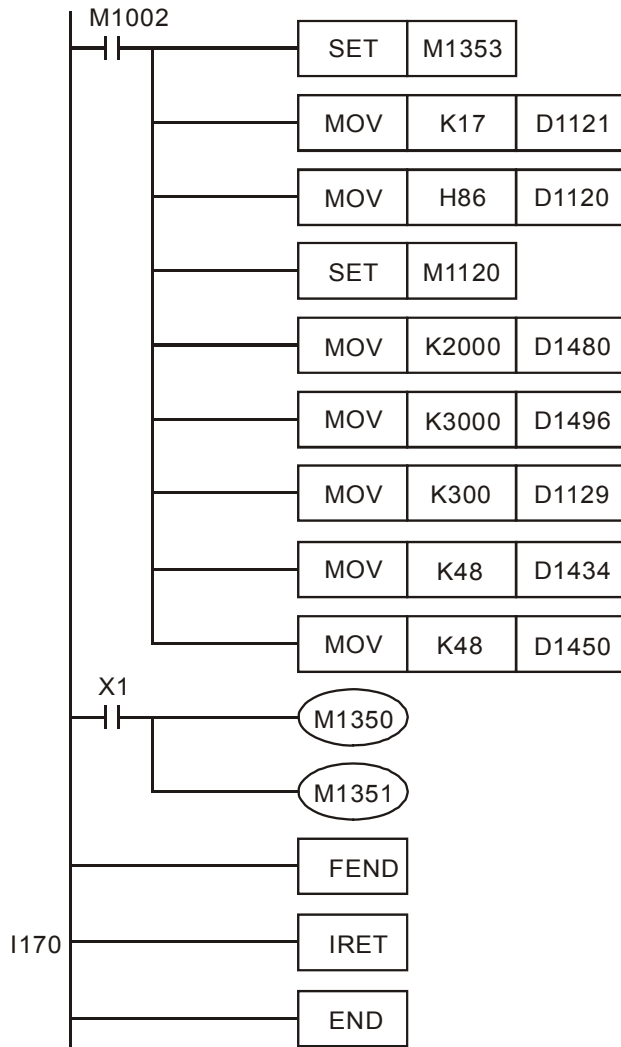
■ With I170, the processing of data in EASY EASY PLC LINK will not be done at END, but at the enabling of I170 immediately after the receiving of data is completed

■ If the reaction speed of the RS-485 IC direction control signal pin of the Slave is slow, it is suggested that you do not enable I170.

■ You can set up D1399 (starting Slave ID designated by EASY PLC LINK), and the ID# of the next 15 Slaves have to be in sequence. For example, when D1399 is set as K20, the Master PLC will detect Slave ID# 20 ~ 35.

c) Example 3: Connection of 1 Master and 2 Slaves by RS-485 and exchange of 16 data between Master and Slaves through EASY PLC LINK (M1353 = Off, linkage of 16 stations, 16 data read/write mode)

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

```
  M1002
───┤├──────┬──────┬─────┬──────┐
            │  MOV │ K17 │ D1121│  Master ID#
            ├──────┼─────┼──────┤
            │  MOV │ H86 │ D1120│  COM2 communication protocol
            ├──────┼─────┼──────┤
            │  SET │ M1120│     │  Retain communication protocol
            ├──────┼─────┼──────┤
            │  MOV │ K16 │ D1434│  Number of data read from Slave ID#1
            ├──────┼─────┼──────┤
            │  MOV │ K16 │ D1450│  Number of data written into Slave ID#1
            ├──────┼─────┼──────┤
            │  MOV │ K16 │ D1435│  Number of data read from Slave ID#2
            ├──────┼─────┼──────┤
            │  MOV │ K16 │ D1451│  Number of data written into Slave ID#2
  X1        └──────┴─────┴──────┘
───┤├──────┬───( M1351 )  Auto mode
            │
            └───( M1350 )  Enable EASY PLC LINK
            │
            ┌──────┐
            │ END  │
            └──────┘
```
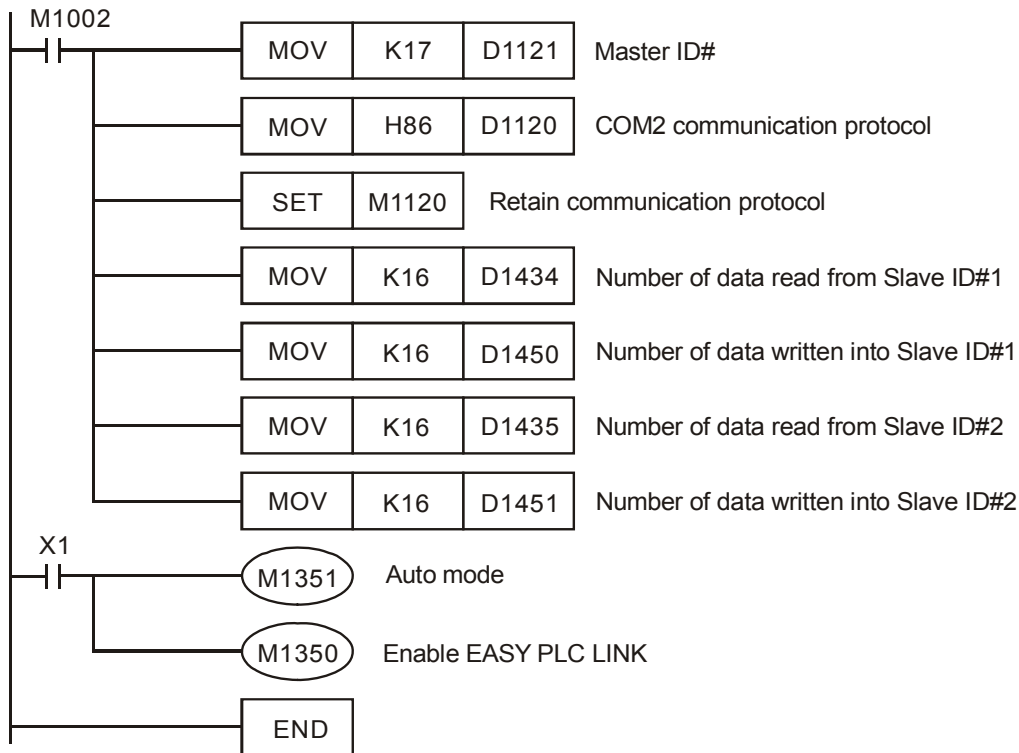
▪ When X1 = On, the data exchange between Master and the two Slaves will be automatically done in EASY PLC LINK, i.e. the data in D100 ~ D115 in the two Slaves will be read into D1480 ~ D1495 and D1512 ~ D1527 of the Master, and the data in D1496 ~ D1511 and D1528 ~ D1543 will be written into D200 ~ D215 of the two Slaves.

Master PLC *1		Slave PLC*2
D1480 ~ D1495	← Read	D100 ~ D115 of Slave ID#1
D1496 ~ D1511	Write →	D200 ~ D215 of Slave ID#1
D1512 ~ D1527	← Read	D100 ~ D115 of Slave ID#2
D1528 ~ D1543	Write →	D200 ~ D215 of Slave ID#2
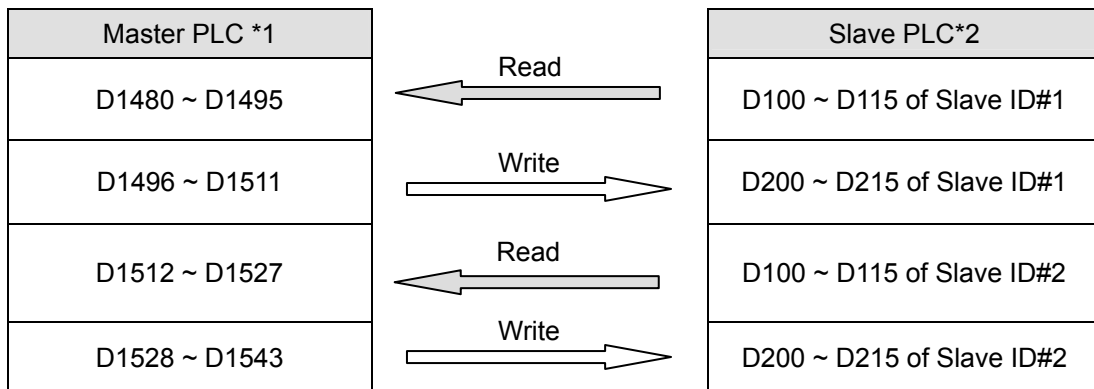
▪ Assume the data in D for data exchange between Master and Slave before EASY PLC LINK is enabled (M1350 = Off) are as the follow:

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

After EASY PLC LINK is enabled (M1350 = On), the data in D for data exchange will become:

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

- The Master PLC has to be SA/SX/SC/EH/EH2/SV series MPU, and the Slave PLC can be any MPU of DVP series.
- There can be maximum 16 Slave PLCs in EASY PLC LINK. See the special Ds in the Master PLC corresponding to D100 ~ D115 and D200 ~ D215 in every Slave PLC in the tables of special M and special D.

d) <u>Example 4:</u> Connection between Delta PLC and Delta VFD-M AC motor drive through EASY PLC LINK for STOP, forward/reverser revolution and writing/reading of frequency.

- Write the ladder diagram program into Master PLC (ID#17)
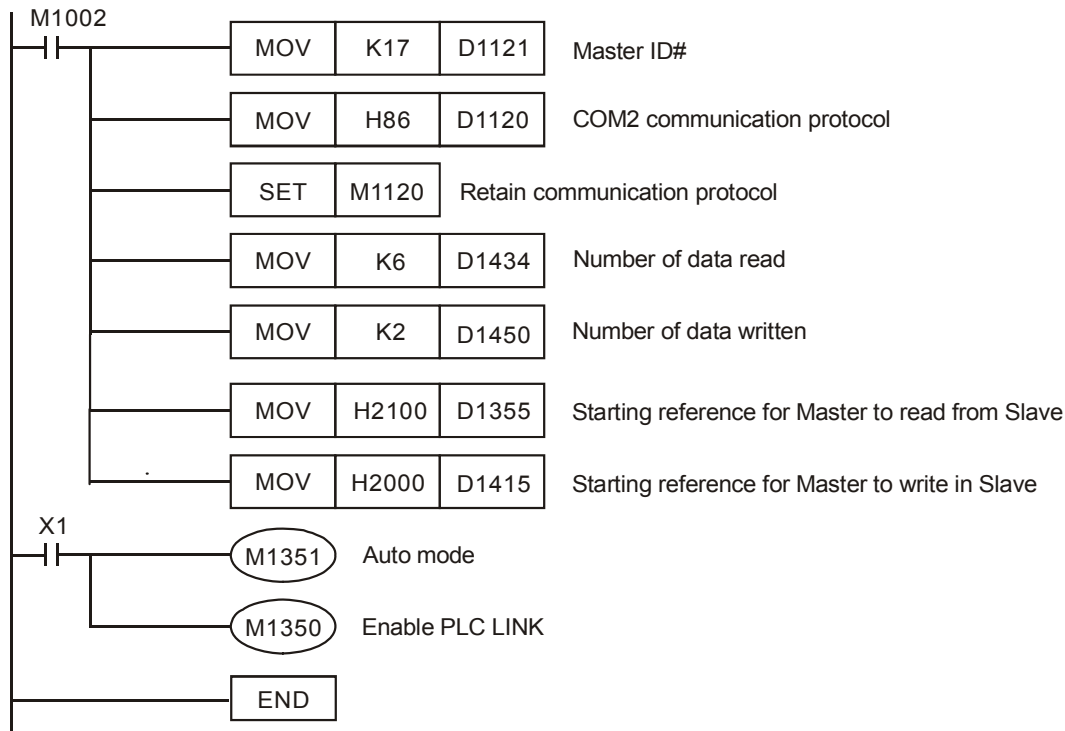
- D1480 ~ D1485 correspond to parameters H2100 ~ H2105 in VFD-M. When X1 = On, EASY PLC LINK will be enabled, and the data in H2100 ~ H2105 will be displayed in D1480 ~ D1485.
- D1496 ~ D1497 correspond to parameters H2000 ~ H2001 in VFD-M. When X1 = On, EASY PLC LINK will be enabled, and the data in H2000 ~ H2001 will be displayed in D1496 ~ D1497.
- Modify D1496 to give command to VFD, e.g. D1496 = H12: enabling forward revolution of VFD-M; D1496 = H11: enabling reverse revolution of VFD.
- Modify D1497 to change the frequency of VFD, e.g. D1497 = K5,000: changing the frequency to 50KHz.
- The Master PLC has to be SA/SX/SC/EH/EH2/SV series MPU, and the Slave AC motor drive can be any VFD series models except VFD-A.

- The Slave can also be Delta temperature controller DTA, DTB, Delta servo ASDA and so on which are compatible to Modbus protocol. Maximum 16 devices are connectable to the LINK.
- See the tables of special M and special D for the starting ID of Slave to be read/written and the number of data to be read/written.

12. Communication addresses of devices in DVP series PLC:

Device	Range		Type	Address	Applicable to		
					ES/EX/SS	SA/SX/SC	EH/EH2/SV
S	000 ~ 255		bit	0000 ~ 00FF	0 ~ 127		
S	246 ~ 511		bit	0100 ~ 01FF		0 ~ 1,023	0 ~ 1,023
S	512 ~ 767		bit	0200 ~ 02FF	-		
S	768 ~ 1,023		bit	0300 ~ 03FF			
X	000 ~ 377 (Octal)		bit	0400 ~ 04FF	0 ~ 177	0 ~ 177	0 ~ 377
Y	000 ~ 377 (Octal)		bit	0500 ~ 05FF	0 ~ 177	0 ~ 177	0 ~ 377
T	000 ~ 255		bit/word	0600 ~ 06FF	0 ~ 127	0 ~ 255	0 ~ 255
M	000 ~ 255		bit	0800 ~ 08FF			
M	256 ~ 511		bit	0900 ~ 09FF			
M	512 ~ 767		bit	0A00 ~ 0AFF	0 ~ 1,279		
M	768 ~ 1,023		bit	0B00 ~ 0BFF			
M	1,024 ~ 1,279		bit	0C00 ~ 0CFF			
M	1,280 ~ 1,535		bit	0D00 ~ 0DFF			
M	1,536 ~ 1,791		bit	B000 ~ B0FF			
M	1,792 ~ 2,047		bit	B100 ~ B1FF			
M	2,048 ~ 2,303		bit	B200 ~ B2FF		0 ~ 4,095	0 ~ 4,095
M	2,304 ~ 2,559		bit	B300 ~ B3FF			
M	2,560 ~ 2,815		bit	B400 ~ B4FF	-		
M	2,816 ~ 3,071		bit	B500 ~ B5FF			
M	3,072 ~ 3,327		bit	B600 ~ B6FF			
M	3,328 ~ 3,583		bit	B700 ~ B7FF			
M	3,584 ~ 3,839		bit	B800 ~ B8FF			
M	3,840 ~ 4,095		bit	B900 ~ B9FF			
C	0 ~ 199	16-bit	bit/word	0E00 ~ 0EC7	0 ~ 127	0 ~ 199	0 ~ 199
	200 ~ 255	32-bit	bit	0EC8 ~ 0EFF	232 ~ 255	200 ~ 255	200 ~ 255
			word	0700 ~ 076F			
D	000 ~ 256		word	1000 ~ 10FF			
D	256 ~ 511		word	1100 ~ 11FF			
D	512 ~ 767		word	1200 ~ 12FF	0 ~ 1,311	0 ~ 4,999	0 ~ 9,999
D	768 ~ 1,023		word	1300 ~ 13FF			
D	1,024 ~ 1,279		word	1400 ~ 14FF			
D	1,280 ~ 1,535		word	1500 ~ 15FF			

Device	Range	Type	Address	Applicable to		
				ES/EX/SS	SA/SX/SC	EH/EH2/SV
D	1,536 ~ 1,791	word	1600 ~ 16FF		0 ~ 4,999	0 ~ 9,999
D	1,792 ~ 2,047	word	1700 ~ 17FF			
D	2,048 ~ 2,303	word	1800 ~ 18FF			
D	2,304 ~ 2,559	word	1900 ~ 19FF			
D	2,560 ~ 2 815	word	1A00 ~ 1AFF			
D	2,816 ~ 3,071	word	1B00 ~ 1BFF			
D	3,072 ~ 3,327	word	1C00 ~ 1CFF			
D	3,328 ~ 3,583	word	1D00 ~ 1DFF			
D	3,584 ~ 3,839	word	1E00 ~ 1EFF			
D	3,840 ~ 4,095	word	1F00 ~ 1FFF			
D	4,096 ~ 4,351	word	9000 ~ 90FF			
D	4,352 ~ 4,607	word	9100 ~ 91FF			
D	4 608 ~ 4863	word	9200 ~ 92FF			
D	4,864 ~ 5,119	word	9300 ~ 93FF			
D	5,120 ~ 5,375	word	9400 ~ 94FF	-	-	
D	5,376 ~ 5,631	word	9500 ~ 95FF			
D	5,632 ~ 5,887	word	9600 ~ 96FF			
D	5,888 ~ 6,143	word	9700 ~ 97FF			
D	6,144 ~ 6,399	word	9800 ~ 98FF			
D	6,400 ~ 6,655	word	9900 ~ 99FF			
D	6,656 ~ 6,911	word	9A00 ~ 9AFF			
D	6,912 ~ 7,167	word	9B00 ~ 9BFF			
D	7,168 ~ 7,423	word	9C00 ~ 9CFF			
D	7,424 ~ 7,679	word	9D00 ~ 9DFF			
D	7,680 ~ 7,935	word	9E00 ~ 9EFF			
D	7,936 ~ 8,191	word	9F00 ~ 9FFF			
D	8,192 ~ 8,447	word	A000 ~ A0FF			
D	8,448 ~ 8,703	word	A100 ~ A1FF			
D	8,704 ~ 8,959	word	A200 ~ A2FF			
D	8,960 ~ 9,215	word	A300 ~ A3FF			
D	9,216 ~ 9,471	word	A400 ~ A4FF			
D	9,472 ~ 9,727	word	A500 ~ A5FF			
D	9,728 ~ 9,983	word	A600 ~ A6FF			
D	9,984 ~ 9,999	word	A700 ~ A70F			

2.12   Error Codes

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	Cause of error	Error code	Cause of error
0001	Use of device S exceeds the range	0F0A	Times of using TTMR, STMR instruction exceed the range
0002	Using P* repeatedly or use of P* exceeds the range	0F0B	Times of using SORT instruction exceed the range
0003	Use of KnSm exceeds the range	0F0C	Times of using TKY instruction exceed the range
0102	Using I* repeatedly or use of I* exceeds the range	0F0D	Times of using HKY instruction exceed the range
0202	Use of MC N* exceeds the range	1000	Improper use of operands of ZRST instruction
0302	Use of MCR N* exceeds the range	10EF	Incorrect use of E, F, or the modification exceeds the range
0401	Use of device X exceeds the range	2000	Times of using TTMR, PR, HOUR instructions exceed the range. Improper use of operands of MRT, ARWS instructions
0403	Use of KnXm exceeds the range		
0501	Use of device Y exceeds the range		
0503	Use of KnYm exceeds the range	C400	Illegal instruction
0601	Use of device T exceeds the range	C401	General loop error
0604	Use of register T exceeds the range	C402	Continuously using LD/LDI instructions for more than 9 times
0801	Use of device M exceeds the range	C403	Continuously using MPS for more than 9 times
0803	Use of KnMm exceeds the range	C404	More than 6 steps in FOR – NEXT
0B01	Incorrect use of KH	C405	Using STL/RET between FOR – NEXT Using SRET/IRET between FOR – NEXT Using MC/MCR between FOR – NEXT Using END/FEND between FOR – NEXT
0D01	Improper use of operands of DECO instruction		
0D02	ES/EX/SS/EH: improper use of operands of ENCO instruction SA/SX/SC: illegal use of the first operand of ANS instruction		
0D03	Improper use of operands of DHSCS instruction	C407	Continuously using STL for more than 9 times
0D04	Improper use of operands of DHSCR instruction	C408	Using MC/MCR in STL, using I/P in STL
0D05	Improper use of operands of pulse output instruction	C409	Using STL/RET in subroutine Using STL/RET in interruption subroutine
0D06	Improper use of operands of PWM instruction	C40A	Using MC/MCR in subroutine Using MC/MCR in interruption subroutine
0D07	Improper use of operands of FROM/TO instruction	C40B	MC/MCR does not start from N0, or is not continuous
0D08	Improper use of operands of PID instruction	C40C	Corresponding N of MC and MCR are different
0D09	Improper use of operands of SPD instruction	C40D	Improper use of I/P
0D0A	Incorrect operands in DHSZ instruction	C40E	IRET does not appear after the last FEND SRET does not appear after the last FEND

Error code	Cause of error	Error code	Cause of error
0D0B	Improper use of operands in IST instruction	C40F	PLC program and data in parameters have not been initialized
0E01	Use of device C exceeds the range	C41B	Invalid RUN/STOP instruction to extension module
0E04	Use of register C exceeds the range	C41C	Points of extension module exceed the range
0E05	Improper use of operand CXXX of DCNT instsruction	C41D	Number of extension modules exceeds the range
0E18	BCD conversion error	C41E	Incorrect hardware setting for extension module
0E19	Division error (divisor = 0)	C41F	Failing to write data into memory
0E1A	Use of device exceeds the range (including E, F index register modification)	C420	Read/write function card error
0E1B	The index of the radical is a negative value	C430	Initializing parallel interface error
0E1C	Communication error of FROM/TO instruction	C440	Hardware error in high-speed counter
0F04	Use of register D exceeds the range	C441	Hardware error in high-speed comparator
0F05	Improper use of operand DXXX of DCNT instruction	C442	Hardware error in MCU pulse output
0F06	Improper use of operands of SFTR instruction	C443	No response from extension unit
0F07	Improper use of operands of SFTL instruction	C4EE	No END instruction in the program
0F08	Improper use of operands of REF instruction	C4FF	Invalid instruction (no such instruction existing)
0F09	Improper use of operands of WSFR, WSFL instructions		

3.1   Basic Instructions and Step Ladder Instructions

📖 **Basic Instructions**

Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
			ES	SA	EH		
LD	Loading in A contact	X, Y, M, S, T, C	3.8	3.8	0.24 (0.56)	1~3	3-3
LDI	Loading in B contact	X, Y, M, S, T, C	3.88	3.88	0.24 (0.56)	1~3	3-3
AND	Series connection- A contact	X, Y, M, S, T, C	2.32	2.32	0.24 (0.56)	1~3	3-3
ANI	Series connection- B contact	X, Y, M, S, T, C	2.4	2.4	0.24 (0.56)	1~3	3-4
OR	Parallel connection- A contact	X, Y, M, S, T, C	2.32	2.32	0.24 (0.56)	1~3	3-4
ORI	Parallel connection- B contact	X, Y, M, S, T, C	2.4	2.4	0.24 (0.56)	1~3	3-5
ANB	Series connection- loop blocks	N/A	1.76	1.76	0.24	1~3	3-5
ORB	Parallel connection- loop blocks	N/A	1.76	1.76	0.24	1~3	3-6
MPS	Store the current result of the internal PLC operations	N/A	1.68	1.68	0.24	1~3	3-6
MRD	Reads the current result of the internal PLC operations	N/A	1.6	1.6	0.24	1	3-6
MPP	Pops (recalls and removes) the currently stored result	N/A	1.6	1.6	0.24	1	3-6

📖 **Output instructions**

Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
			ES	SA	EH		
OUT	Output coil	Y, M, S	5.04	5.04	0.24 (0.56)	1~3	3-7
SET	Latched（On）	Y, M, S	3.8	3.8	0.24 (0.56)	1~3	3-7
RST	Clear the contacts or the registers	Y, M, S, T, C, D, E, F	7.8	7.8	0.24 (0.56)	3	3-8

📖 **Timers, Counters**

API	Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
				ES	SA	EH		
96	TMR	16-bit timer	T-K or T-D	10.6	10.6	9.6	4	3-8
97	CNT	16-bit counter	C-K or C-D (16 bits)	9.7	9.7	12.8	4	3-9
97	DCNT	32-bit counter	C-K or C-D (32 bits)	10.3	10.3	14.3	6	3-9

📖 **Main control instructions**

Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
			ES	SA	EH		
MC	Master control start	N0 ~ N7	5.6	5.6	5.6	3	3-10
MCR	Master control reset	N0 ~ N7	5.7	5.7	5.7	3	3-10

📖 **Instructions for detecting the contacts of rising-/falling-edge**

API	Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
				ES	SA	EH		
90	LDP	Rising-edge detection operation	X, Y, M, S, T, C	5.1	5.1	0.56 (0.88)	3	3-11
91	LDF	Falling-edge detection operation	X, Y, M, S, T, C	5.1	5.1	0.56 (0.88)	3	3-12
92	ANDP	Rising-edge series connection	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	3-12
93	ANDF	Falling-edge series connection	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	3-12
94	ORP	Rising-edge parallel connection	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	3-13
95	ORF	Falling-edge parallel connection	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	3-13

📖 **Rising-/falling-edge output instructions**

API	Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
				ES	SA	EH		
89	PLS	Rising-edge output	Y, M	7.8	7.8	9.92	3	3-13
99	PLF	Falling-edge output	Y, M	7.8	7.8	10.16	3	3-14

📖 **End instruction**

Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
			ES	SA	EH		
END	Program ends	N/A	5	5	0.24	1	3-14

📖 **Other instructions**

API	Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
				ES	SA	EH		
	NOP	No operation	N/A	0.88	0.88	0.16	1	3-15
98	INV	Inverting operation	N/A	1.6	1.6	0.24	1	3-15
	P	Pointer	P0 ~ P255	0.88	0.88	-	1	3-15
	I	Interruption program marker	I□□□	0.88	0.88	-	1	3-16

📖 **Step ladder instructions**

Instruction Code	Function	Operands	Execution speed (us)			STEP	Page
			ES	SA	EH		
STL	Step transition ladder start instruction	S	11.6	10.6	0.56	1	4-1
RET	Step transition ladder return instruction	N/A	7.04	6.04	0.24	1	4-1

Note 1: ES includes ES/EX/SS; SA includes SA/SX/SC; EH includes EH/EH2/SV.

Note 2: For EH series MPU, the execution speed in the brackets ( ) refers to the execution speed of designated operand M1536 ~ M4095.

3.2   Explanations on Basic Instructions

Mnemonic	Function	Program steps	Controllers
LD	Loading in A contact	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The LD instruction is used on the A contact that has its start from the left BUS or the A contact that is the start of a contact circuit. The functions are to save the present contents and store the acquired contact status into the accumulative register.

Program Example:

Ladder diagram:

X0   X1

Y1

Instruction code: Operation:

LD	**X0**	Loading in contact A of X0
AND	X1	Connecting to contact A of X1 in series
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
LDI	Loading in B contact	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The LDI instruction is used on the B contact that has its start from the left BUS or the B contact that is the start of a contact circuit. The functions are to save the present contents and store the acquired contact status into the accumulative register.

Program Example:

Ladder diagram:

X0   X1

Y1

Instruction code: Operation:

LDI	**X0**	Loading in contact B of X0
AND	X1	Connecting to contact A of X1 in series
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
AND	Series connection- A contact	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The AND instruction is used in the series connection of A contact. The functions are to read out the status of present d

series connection contacts and perform the "AND" operation with the logical operation result obtained. The final result will be store in the accumulative register.

Program Example:

Ladder diagram:

X1	X0	
		Y1

Instruction code: Operation:

LDI	X1	Loading in contact B of X1
AND	**X0**	Connecting to contact A of X0 in series
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
ANI	Series connection- B contact	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The ANI instruction is used in the series connection of B contact. The functions are to read out the status of present designated series connection contacts and perform the "AND" operation with the logical operation result obtained. The final result will be store in the accumulative register.

Program Example:

Ladder diagram:

X1	X0	
		Y1

Instruction code: Operation:

LD	X1	Loading in contact A of X1
ANI	**X0**	Connecting to contact B of X0 in series
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
OR	Parallel connection- A contact	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The OR instruction is used in the parallel connection of A contact. The functions are to read out the status of present designated parallel connection contacts and perform the "OR" operation with the logical operation result obtained. The final result will be store in the accumulative register.

Program Example:

Ladder diagram:

X0	
X1	Y1

Instruction code: Operation:

LD	X0	Loading in contact A of X0
OR	**X1**	Connecting to contact A of X1 in parallel
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
ORI	Parallel connection- B contact	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The OR instruction is used in the parallel connection of B contact. The functions are to read out the status of present designated parallel connection contacts and perform the "OR" operation with the logical operation result obtained. The final result will be store in the accumulative register.

Program Example:

Ladder diagram:

Instruction code: Operation:

LD	X0	Loading in contact A of X0
ORI	**X1**	Connecting to contact B of X1 in parallel
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
ANB	Series connection- loop blocks	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	N/A

Explanations:

To perform the "AND" operation of the preserved logic results and content in the accumulative register.

Program Example:

Ladder diagram:

Block A Block B
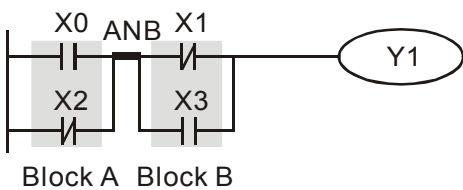
Instruction code: Operation:

LD	X0	Loading in contact A of X0
ORI	X2	Connecting to contact B of X2 in parallel
LDI	X1	Loading in contact B of X1
OR	X3	Connecting to contact A of X3 in parallel
ANB		Connecting circuit block in series
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers							
ORB	Parallel connection- loop blocks	1	ES	EX	SS	SA	SX	SC	EH	SV

Operand	
	N/A

Explanations:

To perform the "OR" operation of the preserved logic results and content in the accumulative register.

Program Example:

Ladder diagram:
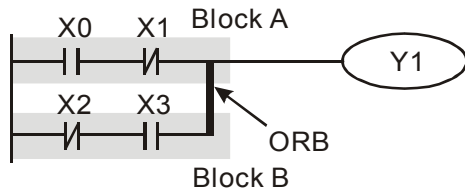
Instruction code:		Operation:
LD	X0	Loading in contact A of X0
ANI	X1	Connecting to contact B of X1 in series
LDI	X2	Loading in contact B of X2
AND	X3	Connecting to contact A of X3 in series
ORB		Connecting circuit block in parallel
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers							
MPS	Store the current result of the internal PLC operations	1	ES	EX	SS	SA	SX	SC	EH	SV

Operand	
	N/A

Explanations:

To save the content in the accumulative register into the operational result (the pointer of operational result will plus 1).

Mnemonic	Function	Program steps	Controllers							
MRD	Reads the current result of the internal PLC operations	1	ES	EX	SS	SA	SX	SC	EH	SV

Operand	
	N/A

Explanations:

To read the operational result and store it into the accumulative register (the pointer of operational result stays intact).

Mnemonic	Function	Program steps	Controllers							
MPP	Pops (recalls and removes) the currently stored result	1	ES	EX	SS	SA	SX	SC	EH	SV

Operand	
	N/A

Explanations:

To retrieve the previous preserved logical operation result and store it into the accumulative register (the pointer of operational result will minus 1).

Program Example:

Ladder diagram:
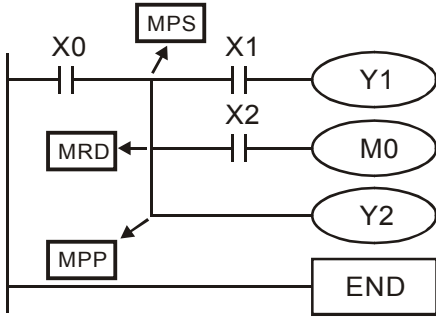
Instruction code:		Operation:
LD	X0	Loading in contact A of X0
MPS		Saving into stack
AND	X1	Connecting to contact A of X1 in series
OUT	Y1	Driving Y1 coil
MRD		Reading from stack
AND	X2	Connecting to contact A of X2 in series
OUT	M0	Driving M0 coil
MPP		Reading from stack and pop pointer
OUT	Y2	Driving Y2 coil
END		Program ends

Mnemonic	Function	Program steps	Controllers							
			ES	EX	SS	SA	SX	SC	EH	SV
OUT	Output coil	1								

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	-	✓	✓	✓	-	-	-

Explanations:

1. To output the logical operation result before OUT instruction into a designated device.

2. Actions of coil contact:

Operational result	OUT instruction		
	Coil	Contact	
		A contact (normally open)	B contact (normally closed)
FALSE	Off	Off	On
TRUE	On	On	Off

Program Example:

Ladder diagram:

Instruction code:		Operation:
LDI	X0	Loading in contact B of X0
AND	X1	Connecting to contact A of X1 in series
OUT	**Y1**	Driving Y1 coil

Mnemonic	Function	Program Steps	Controllers							
			ES	EX	SS	SA	SX	SC	EH	SV
SET	Latched (On)	1								

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	-	✓	✓	✓	-	-	-

Explanations:

When the SET instruction is driven, its designated device will be "On" and keep being On both when SET instruction is still being driven or not driven. Use RST instruction to set "Off" the device.

Program Example:

Ladder diagram:

X0   Y0
├─┤ ├──┤/├──────┤ SET │ Y1 │
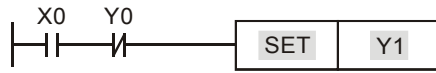
Instruction code:

LD	X0	Loading in contact A of X0
ANI	Y0	Connecting to contact B of Y0 in series
SET	**Y1**	Y1 latched (On)

Operation:

Mnemonic	Function	Program steps	Controllers
RST	Clear the contacts or the registers	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999	E0 ~ E7 F0 ~ F7
	-	✓	✓	✓	✓	✓	✓	✓

Explanations:

1. When the RST instruction is driven, the actions of the designated devices are:

Device	Status
Y, M, S,	Coil and contact will be set to "Off"
T, C	Present values of the timer or counter will be set to "0", and the coil and contact will be set to "Off"
D, E, F	The content will be set to "0".

2. If RST instruction is not being executed, the status of the designated device will stay intact.

Program Example:

Ladder diagram:

X0
├─┤ ├──────┤ RST │ Y5 │
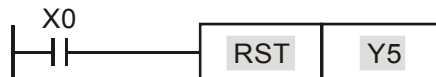
Instruction code:

| LD | X0 | Loading in contact A of X0 |
| **RST** | **Y5** | Resetting contact Y5 |

Operation:

Mnemonic	Function	Program steps	Controllers
TMR	16-bit timer	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	T-K	T0 ~ T255, K0 ~ K32,767
	T-D	T0 ~ T255, D0 ~ D9999

Explanations:

When TMR instruction is executed, the designated coil of the timer will be On and the timer will start to time. When the set value in the timer is reached (present ≥ set value), the contact will be:

NO (Normally Open) contact	Open collector
NC (Normally Closed) contact	Close collector

Program Example:

Ladder diagram:
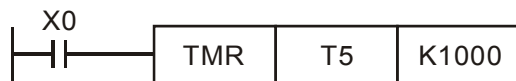
X0	TMR	T5	K1000

Instruction code: Operation:

LD	X0	Loading in contact A of X0 T5 timer
TMR	**T5 K1000**	Set value in timer T5 as K1,000

Remarks:

See the specification of each model for the range of operand T.

Mnemonic	Function	Program steps	Controllers
CNT	16-bit counter	1	ES | EX | SS | SA | SX | SC | EH | SV

	C-K	C0 ~ C199, K0~K32,767
Operand	C-D	C0 ~ C199, D0 ~ D9999

Explanations:

1. When the CNT instruction goes from Off to On, the designated counter coil will be driven, and the present value in the counter will plus 1. When the counting reaches the set value (present value = set value), the contact will be:

NO (Normally Open) contact	Open collector
NC (Normally Closed) contact	Close collector

2. If there are other counting pulse input after the counting reaches its target, the contact and present value will stay intact. Use RST instruction to restart or reset the counting.

Program Example:

Ladder diagram:

X0	CNT	C20	K100

Instruction code: Operation:

LD	X0	Loading in contact A of X0
CNT	**C20 K100**	Set value in counter C20 as K100

Mnemonic	Function	Program steps	Controllers
DCNT	32-bit counter	1	ES | EX | SS | SA | SX | SC | EH | SV

	C-K	C200 ~ C255, K-2,147,483,648 ~ K2,147,483,647
Operand	C-D	C200 ~ C255, D0 ~ D9999

Explanations:

1. DCNT is the instruction for enabling the 32-bit high-speed counters C200 ~ C255.

2. For general purpose addition/subtraction counters C200 ~ C234, when DCNT goes from Off to On, the present value in the counter will pulse 1 (counting up) or minus 1 (counting down) according to the modes set in special M1200 ~ M1235.

3. For high-speed addition/subtraction counters C235 ~ C255, when the high-speed counting pulse input goes from Off to On, the counting will start its execution. For the input terminals (X0 ~ X17) and counting methods

(counting up/down) of the high-speed counter, see Chapter 2.7 Numbering and Function of Counter [C] for more details.

4. When DCNT is Off, the counting will stop, but the existing present value in the counter will not be cleared. To clear the present value and the contact, you have to use the instruction RST C2XX. Use externally designated input points to clear the present values and contacts of high-speed addition/subtraction counters C235 ~ C255.

Program Example:
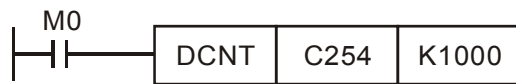
Ladder diagram: | Instruction code: | Operation:

LD	M0	Loading in contact A of M0
DCNT	**C254 K1000**	Set value of counter C254 as K1,000

Mnemonic	Function	Program steps	Controllers
MC / MCR	Master control Start/Reset	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	N0 ~ N7

Explanations:

1. MC is the main-control start instruction. When MC instruction is executed, the execution of instructions between MC and MCR will not be interrupted. When MC instruction is Off, the actions of the instructions between MC and MCR are:

Instruction type	Explanation
General purpose timer	Present value = 0 Coil is Off, No action for the contact
Accumulative timer	Coil is Off, present value and contact stay intact
Subroutine timer	Present value = 0 Coil is Off, No action for the contact
Counter	Coil is Off, present value and contact stay intact
Coils driven by OUT instruction	All Off
Devices driven by SET and 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 the instructions between MC and MCR.

2. MCR is the main-control end instruction that is placed in the end of the main-control program. There should not be any contact instructions prior to MCR instruction.

3. MC-MCR main-control program instructions support the nested program structure (max. 8 layers) and please use the instruction in the order N0 ~ N7.

Program Example:

Ladder diagram:

X0 ─┤├─	MC	N0
X1 ─┤├─	(Y0)	
X2 ─┤├─ ⬇	MC	N1
X3 ─┤├─	(Y1)	
⬇	MCR	N1
⬇	MCR	N0
X10 ─┤├─ ⬇	MC	N0
X11 ─┤├─	(Y10)	
⬇	MCR	N0
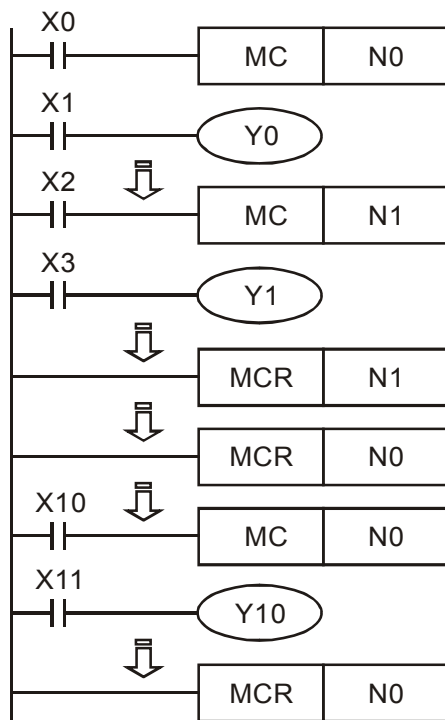
Instruction code: Operation:

LD	X0	Loading in A contact of X0
MC	**N0**	Enabling N0 common series connection contact
LD	X1	Loading in A contact of X1
OUT	Y0	Driving Y0 coil
:		
LD	X2	Loading in A contact of X2
MC	**N1**	Enabling N1 common series connection contact
LD	X3	Loading in A contact of X3
OUT	Y1	Driving Y1 coil
:		
MCR	**N1**	Disabling N1 common series connection contact
:		
MCR	**N0**	Disabling N0 common series connection contact
:		
LD	X10	Loading in A contact of X10
MC	**N0**	Enabling N0 common series connection contact
LD	X11	Loading in A contact of X11
OUT	Y10	Driving Y10 coil
:		
MCR	**N0**	Disabling N0 common series connection contact

Mnemonic	Function	Program steps	Controllers							
			ES	EX	SS	SA	SX	SC	EH	SV
LDP	Rising-edge detection operation	1								

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The method of using LDP is the same as using LD, but the actions of the two instructions differ. LDP saves the current content and store the detected status of rising-edge to the accumulative register.

Program Example:

Ladder diagram:

X0 X1 ─┤↑├─┤├─	(Y1)	

Instruction code: Operation:

LDP	**X0**	Starting X0 rising-edge detection
AND	X1	Series connecting A contact of X1
OUT	Y1	Driving Y1 coil

Remarks:

1.   See the specification of each model for the range of operands.

2.   If the status of a designated rising-edge is On before the PLC is powered, the contact of the rising-edge will be TRUE after PLC is powered.

Mnemonic	Function	Program steps	Controllers
LDF	Falling-edge detection operation	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The method of using LDF is the same as using LD, but the actions of the two instructions differ. LDF saves the current content and store the detected status of falling-edge to the accumulative register.

Program Example:

Ladder diagram:

X0   X1

Y1

Instruction code:		Operation:
LDF	**X0**	Starting X0 falling-edge detection
AND	X1	Series connecting A contact of X1
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
ANDP	Rising-edge series connection	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

ANDP instruction is used in the series connection of the contacts' rising-edge detection.

Program Example:

Ladder diagram:

X0   X1

Y1

Instruction code:		Operation:
LD	X0	Loading in A contact of X0
ANDP	**X1**	X1 rising-edge detection in series connection
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
ANDF	Falling-edge series connection	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

ANDF instruction is used in the series connection of the contacts' falling-edge detection.

[**Program Example:**

Ladder diagram:

X0   X1

Y1

Instruction code:		Operation:
LD	X0	Loading in A contact of X0
ANDF	**X1**	X1 falling-edge detection in series connection
OUT	Y1	Drive Y1 coil

Mnemonic	Function	Program steps	Controllers
ORP	Rising-edge parallel connection	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The ORP instructions are used in the parallel connection of the contact's rising-edge detection.

Program Example:

Ladder diagram:

```
      X0
  ┤├──────────( Y1 )
  │   X1
  └──┤↑├
```

Instruction code:		Operation:
LD	X0	Loading in A contact of X0
ORP	**X1**	X1 rising-edge detection in parallel connection
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
ORF	Falling-edge parallel connection	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	✓	✓	✓	✓	✓	✓	-

Explanations:

The ORP instructions are used in the parallel connection of the contact's falling-edge detection.

Program Example:

Ladder diagram:

```
      X0
  ┤├──────────( Y1 )
  │   X1
  └──┤↓├
```

Instruction code:		Operation:
LD	X0	Loading in A contact of X0
ORF	**X1**	X1 falling-edge detection in parallel connection
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers
PLS	Rising-edge output	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	-	✓	✓	-	-	-	-

Explanations:

When X0 goes from Off to On (rising-edge trigger), PLS instruction will be executed and **S** will send out pulses for once of 1 scan time.

Program Example:

Ladder diagram:

```
X0
─┤├─────┬──[ PLS    M0 ]
M0       │
─┤├──────┴──[ SET    Y0 ]
```

Instruction code:		Operation:
LD	X0	Loading in A contact of X0
PLS	**M0**	M0 rising-edge output
LD	M0	Loading in contact A of M0
SET	Y0	Y0 latched (On)

Timing Diagram:

X0 ‾‾|‾‾|_____|‾‾|___

M0 ‾‾|‾| 1 scan time |‾|____

Y0 ___|‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾
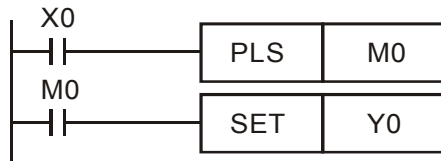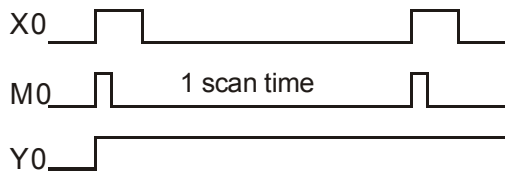
Mnemonic	Function	Program steps	Controllers							
			ES	EX	SS	SA	SX	SC	EH	SV
PLF	Falling-edge output	1								

Operand	X0 ~ X377	Y0 ~ Y377	M0 ~ M4095	S0 ~ S1023	T0 ~ T255	C0 ~ C255	D0 ~ D9999
	-	✓	✓	-	-	-	-

Explanations:

When X0 goes from On to Off (falling-edge trigger), PLF instruction will be executed and **S** will send out pulses for once of 1 scan time.

Program Example:

Ladder diagram:

```
X0
─┤├─────┬──[ PLF    M0 ]
M0       │
─┤├──────┴──[ SET    Y0 ]
```

Instruction code:		Operation:
LD	X0	Loading in A contact of X0
PLF	**M0**	M0 falling-edge output
LD	M0	Loading in contact A of M0
SET	Y0	Y0 latched (On)

Timing Diagram:

X0 ___|‾‾|_____|‾‾|___

M0 _____|‾| 1 scan time |‾|___

Y0 _____|‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾
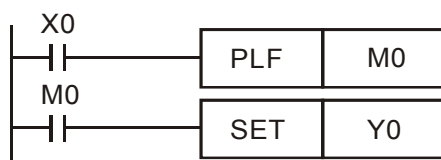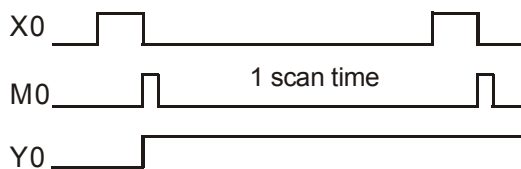
Mnemonic	Function	Program steps	Controllers							
			ES	EX	SS	SA	SX	SC	EH	SV
END	Program End	1								

Operand	N/A

Explanations:

END instruction has to be placed in the end of a ladder diagram or instruction program. PLC will start to scan from address 0 to END instruction and return to address 0 to restart the scan.

Mnemonic	Function	Program steps	Controllers								
			ES	EX	SS	SA	SX	SC	EH	SV
NOP	No operation	1									

Operand	N/A

Explanations:

NOP instruction does not conduct any operations in the program; therefore, after the execution of NOP, the existing logical operation result will be kept. If you want to delete a certain instruction without altering the length of the program, you can use NOP instruction.

Program Example:

Ladder diagram:

NOP instruction will be omitted in the ladder diagram

X0
├─┤├──[NOP]──(Y1)
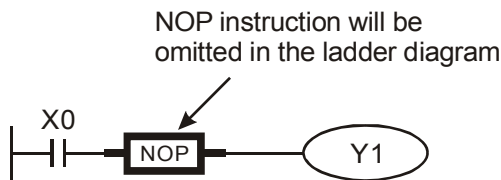
Instruction code: Operation:

LD   X0   Loading in B contact of X0
NOP   No operation
OUT   Y1   Driving Y1 coil

Mnemonic	Function	Program steps	Controllers								
			ES	EX	SS	SA	SX	SC	EH	SV
INV	Inverting Operation	1									

Operand	N/A

Explanations:

The logical operation result before INV instruction will be inverted and stored in the accumulative register.

Program Example:

Ladder diagram:

X0
├─┤├──/──(Y1)

Instruction code: Operation:

LD   X0   Loading in A contact of X0
INV   Inverting the operation result
OUT   Y1   Driving Y1 coil

Mnemonic	Function	Program steps	Controllers								
			ES	EX	SS	SA	SX	SC	EH	SV
P	Pointer	1									

Operand	P0 ~ P255

Explanations:

Pointer P is used in API 00 CJ and API 01 CALL instructions. The use of P does not need to start from No. 0, and the No. of P cannot be repeated; otherwise, unexpected errors may occur.

Program Example:

Ladder diagram:
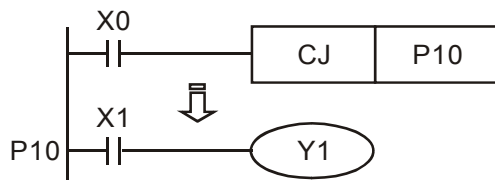
Instruction code:

		Operation:
LD	X0	Loading in A contact of X0
CJ	P10	From instruction CJ to P10
:		
P10		Pointer P10
LD	X1	Loading in A contact of X1
OUT	Y1	Driving Y1 coil

Mnemonic	Function	Program steps	Controllers								
I	Interruption program marker (I)	1	ES	EX	SS	SA	SX	SC	EH	SV

Operand	I00□, I10□, I20□, I30□, I40□, I50□, I6□□, I7□□, I8□□
	I010, I020, I030, I040, I050, I060, I110, I120, I130, I140, I150, I160, I170, I180

Explanations:

A 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. See Chapter 2.9 for pointers of all DVP series PLCs.

Program Example:

Ladder diagram:
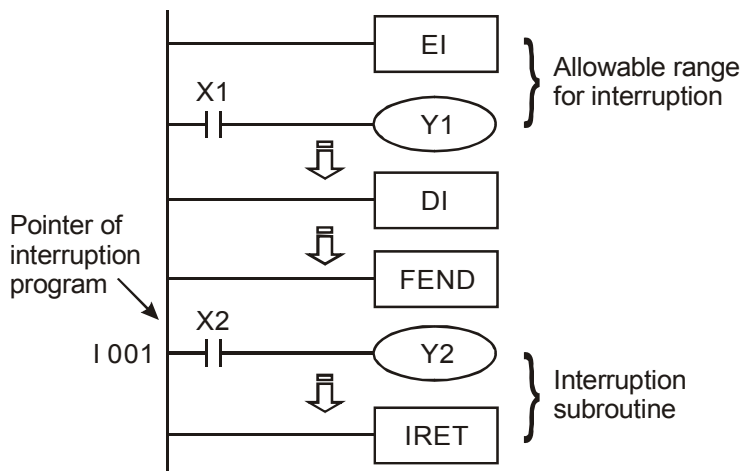
Instruction code: Operation:

EI		Enabling interruption
LD	X1	Loading A contact of X1
OUT	Y1	Driving Y1 coil
:		
DI		Disabling interruption
:		
FEND		Main program ends
I001		Interruption pointer
LD	X2	Loading in A contact of X2
OUT	Y2	Driving Y2 coil
:		
IRET		Interruption return

4.1 Step Ladder Instructions [STL], [RET]

Mnemonic	Function	Program steps	Controllers
STL	Step Transition Ladder Start	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	S0 ~ S1023

Explanations:

STL Sn constructs a step. When STL instruction appears in the program, the program will enter a step ladder diagram status controlled by steps. The initial status has to start from S0 ~ S9. RET instruction indicates the end of a step ladder diagram starting from S0 ~ S9 and the bus returns to a normal ladder diagram instruction. SFC uses the step ladder diagram composed of STL/RET to complete the action of a circuit. The No. of S cannot be repeated.

Mnemonic	Function	Program steps	Controllers
RET	Step Transition Ladder Return	1	ES | EX | SS | SA | SX | SC | EH | SV

Operand	N/A

Explanations:

RET indicates the end of a step. There has to be a RET instruction in the end of a series of steps. One PLC program can be written in maximum 10 steps (S0 ~ S9) and every step should end with a RET.

Program Example:
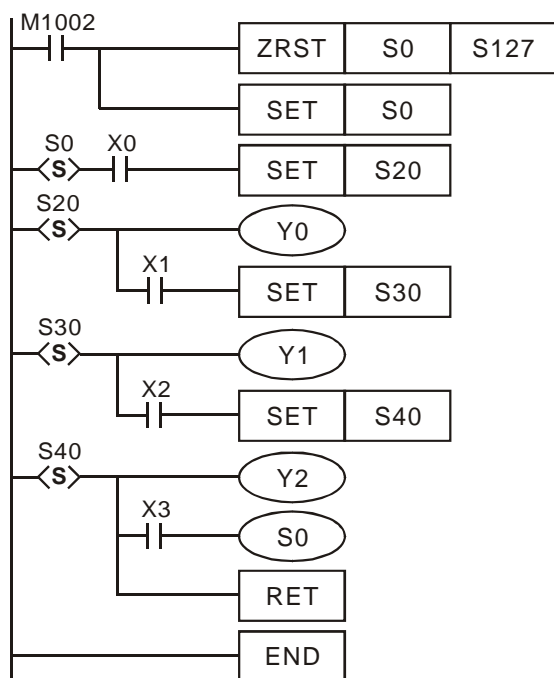
Ladder diagram: SFC:
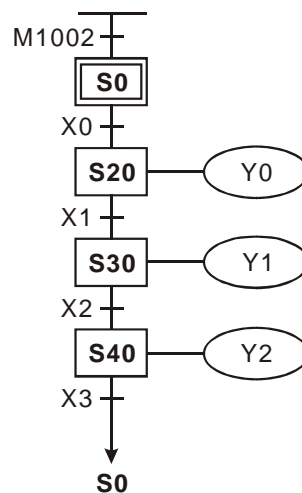
4.2   Sequential Function Chart (SFC)

In automation, we always need electric control to work with mechanical control for an automation control. The sequential control can be divided into several orderly steps (or stages). Each step has its actions that should be completed and the transition from one step to another normally requires some criteria. The action of the last step finishes when all criteria is true and the beginning of the next step will clear the actions of the last step. This is the concept of designing a sequential function chart (SFC).

Features:

1. No sequential design is required for constant step actions, and PLC will automatically execute the interlocking and dual outputs among all status. An easy sequential design is the only thing required to make the machine work normally. 2. The actions in SFC are easy to understand, adjust for a trial operation, detect the errors and maintain. 3. SFC is a type of diagram editing. The structure of a SFC looks like a flow chart. Every No. of the step relay S inside the PLC represents a step, equal to every processing procedure in a flow chart. When the current procedure is completed, the program will move to the next step according to the set transition criteria. Therefore, you can repeat the cycle and obtain the result you desire. 4. See the SFC chart in the right hand side: The initial step S0 transfers to a general purpose step S21 by making the status transition condition X0 condition true. S21 transfer to S22 or jumps to S24 by making X1 or X2 true. In step S25, X6 will be true and the chart will return to S0 to complete a cycle. The cycle and be repeated to reach a cyclic control.	SFC:

5. Next are some basic icons for drawing SFC in WPLSoft SFC editor.
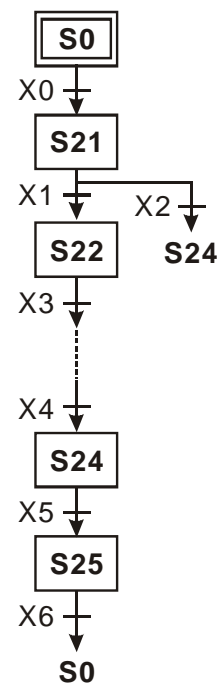
LAD	Ladder diagram mode. The icon indicates that the internal editing program is a general ladder diagram, not a step ladder program.
▣	Initial step in SFC. Applicable for S0 ~ S9.
▢	General step. Applicable for S10 ~ S1023.
⤴	Step jumps. Used for a step to jump to another non-adjacent step. (Jumping up/down to non-adjacent steps in the same sequence, returning to initial step, or jumping among different sequences.)
✚	Transition condition between steps.
⊢⊤	Alternative divergence. The same step transfers to a corresponding step by different transition condition.
⊢⊥	Alternative convergence. More than 2 steps transfer to the same step by transition condition.
⊨⊤	Simultaneous divergence. The same step transfers to more than 2 steps by the same transition condition.
⊨⊥	Simultaneous convergence. More than 2 steps transfer to the same step by a single transition condition.

4.3 How does a Step Ladder Instruction Work?

STL instruction is used for designing the syntax of a sequential function chart (SFC), making the program designing similar to drawing a flow chart and allowing a more explicit and readable program. From the figure in the left hand side below, we can see very clearly the sequence to be designed, and we can convert the sequence into the step ladder diagram in the right hand side.

RET instruction has to be written at the end of every step sequence, representing the end of a sequence. There can be more than one step sequence in a program. Therefore, we have to write in RET at the end of every step sequence. There is no limitation on the times of using RET which is used together with S0 ~ S9.

If there is no RET instruction at the end of a step sequence, errors will be detected by WPL editor.
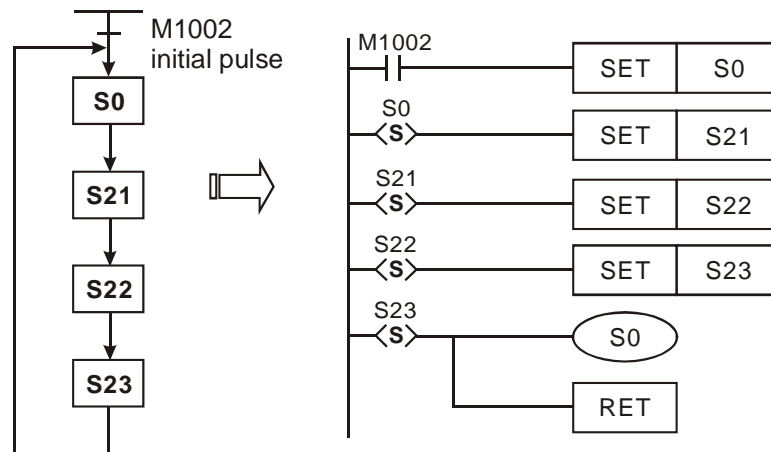
1. Actions of Step Ladder:

A step ladder is composed of many steps and every step controls an action in the sequence. The step ladder has to:

a) Drive the output coil

b) Designate the transition condition

c) Designate which step will take over the control from the current step

Example:

When X0=On, S20 = On, S10 = Off.
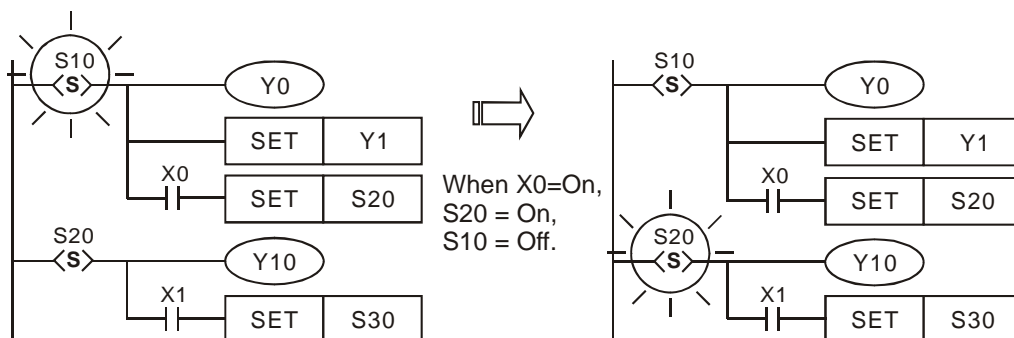
Explanation:

When S10 = On, Y0 and Y1 will be On. When X0 = On, S20 will be On and Y10 will be On. When S10 = Off, Y0 will be Off and Y1 will be On.

2. Timing Diagram of Step Ladder:

When the status contact Sn = On, the circuit will be activated. When Sn = Off, the circuit will be disactivated. The actions will delay for 1 scan time.
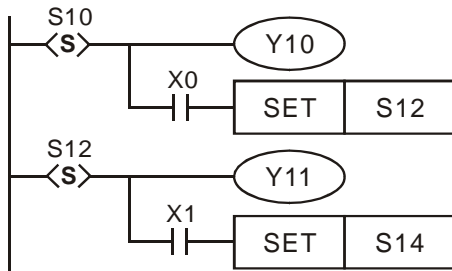
Executing the timing diagram below. After the status of S10 and S12 are transferred (taking place simultaneously), and after a delay of 1 scan time, Y10 will be Off and Y11 will be On. There will not be overlapping outputs.
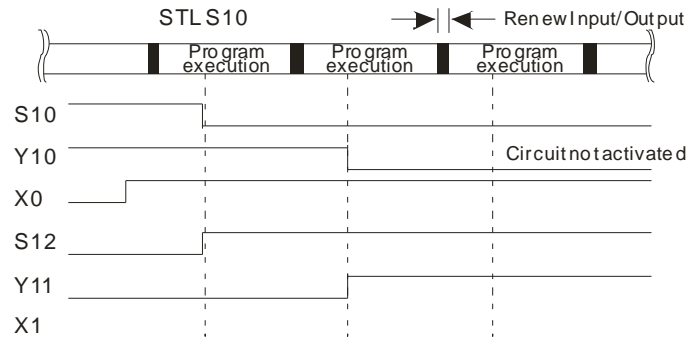
3. Repeated Use of Output Coil:

    a) You can use output coils of the same No. in different steps.

    b) See the diagram in the right. There can be the same output device (Y0) among different statuses. Y0 will be On when S10 or S20 is On. Such as right diagram, there is the same output device Y0 in the different state. No matter S10 or S20 is On, Y0 will be On.

    c) Y0 will be Off when S10 is transferring to S20. After S20 is On, Y0 will output again. Therefore in this case, Y0 will be On when S10 or S20 is On.

    d) Normally in a ladder diagram, avoid repeated use of an output coil. 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.
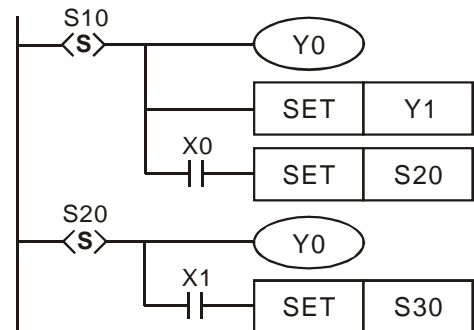
4. Repeated Use of Timer:

The timers in EH/EH2/SV series MPU are the same as general
output points and can be repeatedly used in different steps.
This is one of the features of the step ladder diagram.
However, in a general ladder diagram, it is better not be
repeatedly use the output coil. Also avoid using the No. of the
output coil used by a step after the step ladder diagram returns
to a general ladder diagram.
Note: See the figure in the right. The timers in ES/EX/SS/SA/
SX/SC series MPU can be used repeatedly in non-adjacent
steps.
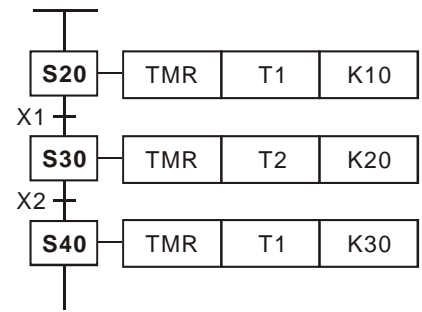
5. Transfer of Step:

SET Sn and OUT Sn instructions are used to enable (or transfer to) another step. When the control power is
shifted to another step, the status of the previous step S and the action of the output point will be cleared. Due to
that there can be many step control sequences (i.e. the step ladder diagram starting with S0 ~ S9) co-existing in
the program. The transfer of a step can take place in the same step sequence, or be transferred to different step
sequence. Therefore, there are some slight differences regarding how to use SET Sn and OUT Sn. See the
explanations below.

SET Sn   Used for driving the next
step in the same
sequence. After the
transition, all output from
the previous status will be
cleared.

When SET S12 instruction is executed,
S10 will transfer to S12, and S10 and all
its outputs (Y10) will be cleared.
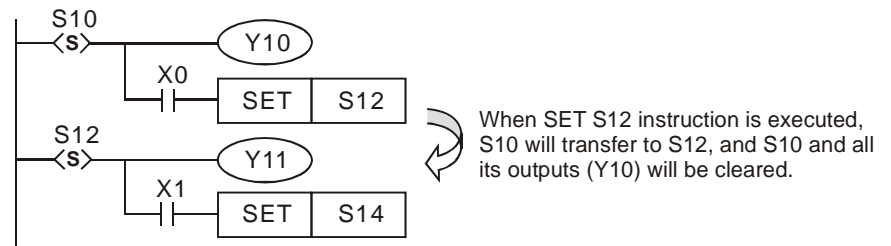
OUT Sn   Used for returning to the initial step in the same step sequence. Also for jumping up/down to
non-adjacent steps in the same sequence, or separating steps in different sequences. After the
transition, all output from the previous status will be cleared.

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.
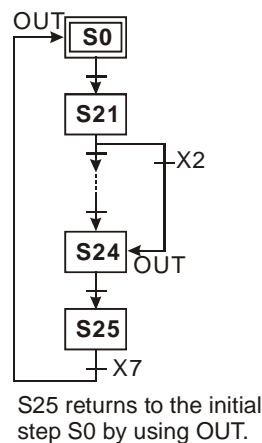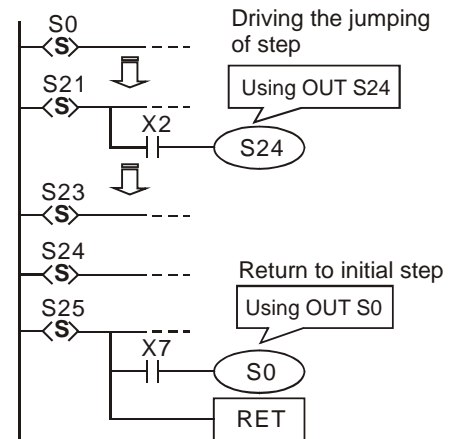
Ladder diagram:

③ Separating steps in different sequences.

SFC:
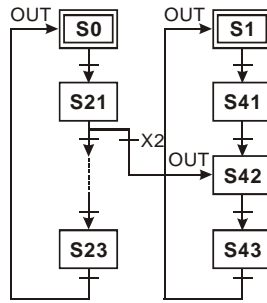
Ladder diagram:

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.
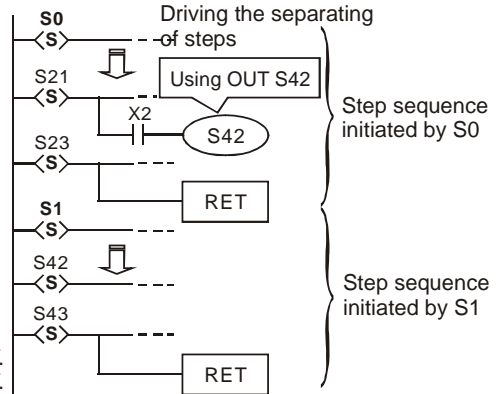
6. Cautions for Driving Output Point:

See the figure below. After the step point and once LD or LDI instructions are written into the second line, the bus will not be able to connect directly to the output coil, and errors will occur in the compilation of the ladder diagram. You have to correct the diagram into the diagram in the right hand side for a correct compilation.
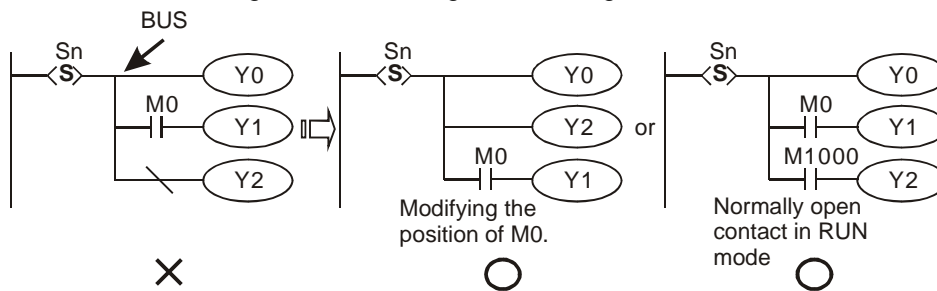
7. Restrictions on Using Some Instructions:

The program of every step is the same as a general ladder diagram, in which you can use all kinds of series/parallel circuits or instructions. However, there are restrictions on some of the instructions.

Basic instructions applicable in a step

Instruction / Step		LD/LDI/LDP/LDF AND/ANI/ANDP/ANDF OR/ORI/ORP/ORF INV/OUT/SET/RST	ANB/ORB MPS/MRD/MPP	MC/MCR
Initial step/general step		Yes	Yes	No
Divergence/ convergence step	General output	Yes	Yes	No
	Step transfer	Yes	No	No

- DO NOT use MC/MCR instruction in the step.
- DO NOT use STL instruction in a general subroutine or interruption subroutine.
- You can still use CJ instruction in STL instruction, but this will make the actions more complicated. We do not recommend you do so.

■ The position of MPS/MRD/MPP instruction:

Ladder diagram:
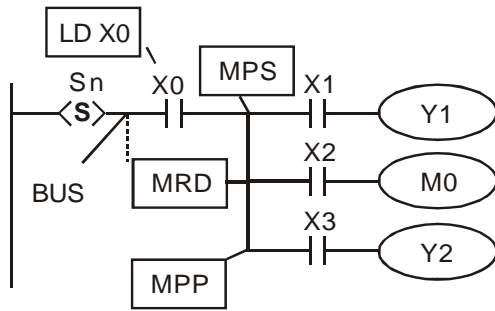
Instruction code:

```
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.

8. Other Points to Note:

The instruction used for transferring the step (SET S☐ or OUT S☐) can only be executed after all the relevant outputs and actions in the current status are completed. See the figure below. The executed results by the PLC are the same, but if there are many conditions or actions in S10, it is recommended that you modify the diagram in the left hand side into the diagram in the right hand side. SET S20 is only executed after all relevant outputs and actions are completed, which is a more explicit sequence.
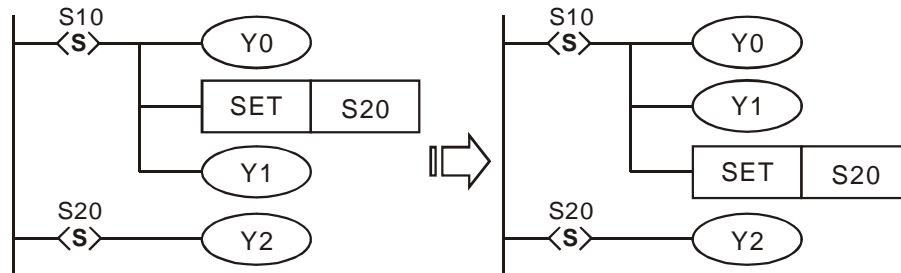
Make sure to add RET instruction after STL at the end of the step ladder diagram.
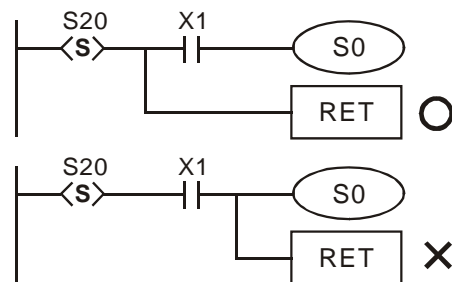
4.4   Things to Note for Designing a Step Ladder Program

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 end a complete sequence with RET instruction.

2. If STL instruction is not in use, step S can be 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: There is only one sequence without alternative divergence, alternative convergence, simultaneous divergence and simultaneous convergence in a program.

Complicated single sequence: There is only one sequence with alternative divergence, alternative convergence, simultaneous divergence and simultaneous convergence in a program.

Multiple sequences: There are more than one sequence in a program, maximum 10 sequences, S0 ~ S9.

5. Separation of sequence: Multiple sequences are allowed to be written into the step ladder diagram.

a) See the diagram in the right hand side. There are two sequences S0 and S1. The program writes in S0 ~ S30 first and S1 ~ S43 next.

b) You can designate a step in the sequence to jump to any step in another sequence.

c) When the condition below S21 is true, the sequence will jump to step S42 in sequence S1, which is called "separating the step".
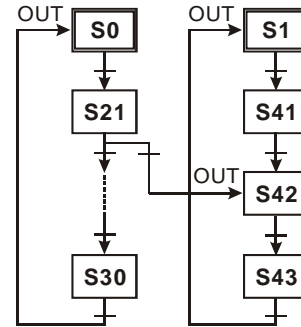
6. Restrictions on diverging sequence: See 4.5 for example

a) You can use maximum 8 diverged steps in a divergence sequence.

b) You can use maximum 16 loops in multiple divergence sequences or in simultaneous sequences combined into one sequence.

c) You can designate a step in the sequence to jump to any step in another sequence.

7. Reset of the step and the inhibiting output:

a) Use ZRST instruction to reset a step to be Off.

b) Make M1034 = On to inhibit output Y.

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 re-powered, the status before the power-off will be recovered and the execution will resume. Please be aware of the area for the latched steps.

9. Special auxiliary relays and special registers: See 4.6refer to chapter 4.6 IST instruction for more details.

Device No.	Function
M1040	Disabling step. Disabling all the shifting of steps when On.
M1041	Starting step. Flag for IST instruction.
M1042	Enabling pulses. 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. Flag for IST instruction.

Device No.	Function
M1046	Setting STL status as On. On when any of the steps is On.
M1047	Enabling STL monitoring
D1040	On status of step No. 1
D1041	On status of step No. 2
D1042	On status of step No. 3
D1043	On status of step No. 4
D1044	On status of step No. 5
D1045	On status of step No. 6
D1046	On status of step No. 7
D1047	On status of step No. 8

4.5   Types of Sequences

Single Sequence: The basic type of sequences

The first step in a step ladder diagram is called the initial step, which can be S0 ~ S9. The steps following the initial step are general steps, which can be S10 ~ S1023. If you are using IST instruction, S10 ~ S19 will become the steps for zero return.

a) _Single sequence without divergence and convergence_

After a sequence is completed, the control power on the steps will be given to the initial step.
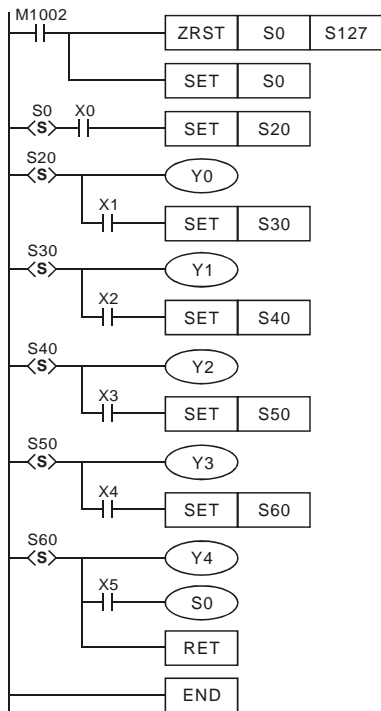
Step ladder diagram   SFC:
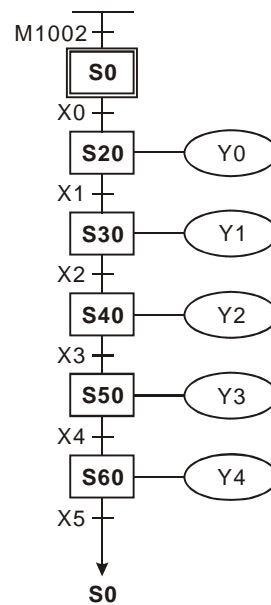
b) *Jumping Sequence*

1. The control power over the step is transferred to a certain step on top.

2. The control power over the step is transferred to the step in another sequence.
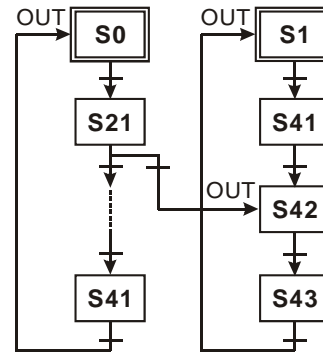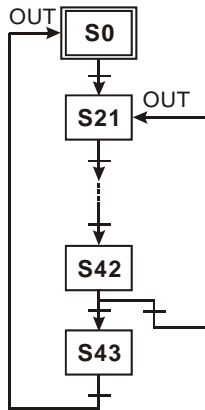
c) *Reset Sequence*

See the diagram in the right hand side. When the condition at S50 is true, S50 will be reset and the sequence will be completed at this time.
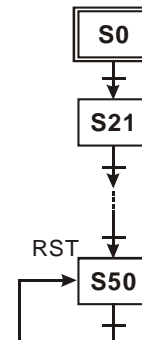
Complicated Single Sequence:

Including simultaneous divergence, alternative divergence, simultaneous convergence and alternative convergence.

a) *Structure of simultaneous divergence*

When the condition at the current step is true, the step can be transferred to many steps. See the diagrams below. When X0 = On, S20 will be simultaneously transferred to S21, S22, S23 and S24.
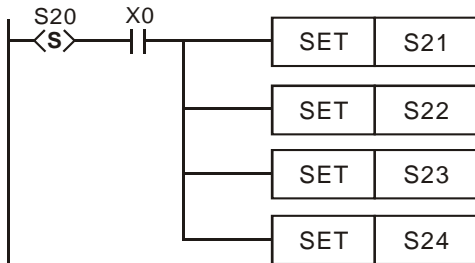
Ladder diagram: SFC:
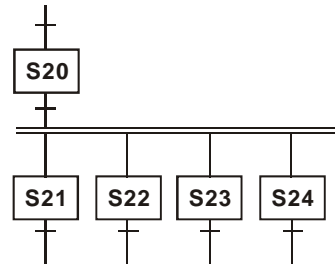
b) *Structure of alternative divergence*

When the individual condition at the current status is true, the step will be transferred to another individual step. See the diagrams below. 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.
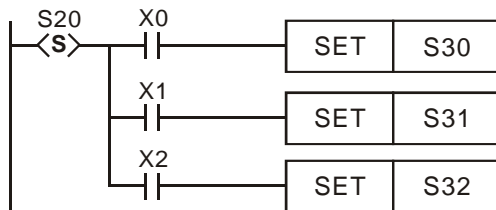
Ladder diagram: SFC:
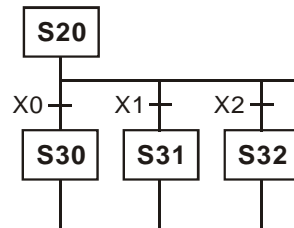
c) *Structure of the simultaneous convergence*

See the ladder diagram below. A continuous STL instruction represents a simultaneous convergence. When the condition is true after a continuous output, the step will be transferred to the next step. In the simultaneous convergence, only when several conditions are true will the transfer be allowed.
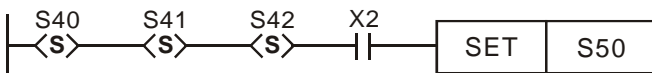
Ladder diagram: SFC:
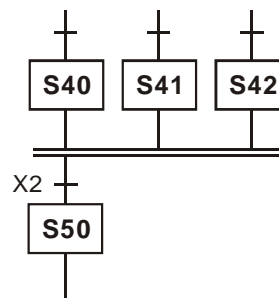
d) *Structure of alternative convergence*

See the diagrams below. Depending on the condition of the input signal of which of S30, S40 and S50 becomes true first, the first one will be first transferred to S60.

Ladder diagram:

```
S30  X0
┤S├──┤├────┤ SET │ S60 │

S40  X1
┤S├──┤├────┤ SET │ S60 │

S50  X2
┤S├──┤├────┤ SET │ S60 │
```

SFC:

```
 │      │      │
┌────┐ ┌────┐ ┌────┐
│S30 │ │S40 │ │S50 │
└────┘ └────┘ └────┘
X0─┤   X1─┤   X2─┤
     ┌────┐
     │S60 │
     └────┘
       │
```
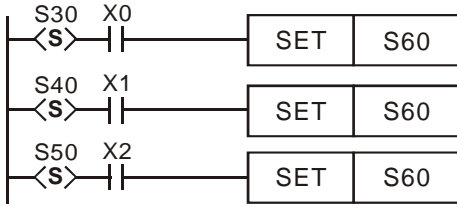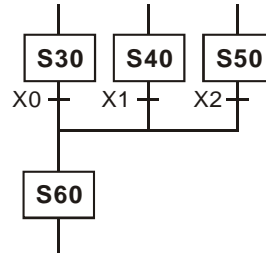
Examples of alternative divergence & alternative convergence:

Ladder diagram:

```
M1002
┤├─────┬──┤ ZRST │ S0  │ S127 │
       └──┤ SET  │ S1  │

S1   X0
┤S├──┤├───┤ SET  │ S20 │

S20
┤S├───────( Y0 )
    X1
    ┤├────┤ SET  │ S30 │
    X4
    ┤├────┤ SET  │ S31 │
    X7
    ┤├────┤ SET  │ S32 │

S30
┤S├───────( Y1 )
    X2
    ┤├────┤ SET  │ S40 │

S40
┤S├───────( Y2 )
    X3
    ┤├────┤ SET  │ S50 │

S31
┤S├───────( Y3 )
    X5
    ┤├────┤ SET  │ S41 │

S41
┤S├───────( Y4 )
    X6
    ┤├────┤ SET  │ S50 │

S32
┤S├───────( Y5 )
    X10
    ┤├────┤ SET  │ S42 │

S42
┤S├───────( Y6 )
    X11
    ┤├────┤ SET  │ S50 │

S50
┤S├───────┤ TMR │ T1 │ K10 │
    T1
    ┤├────┤ SET  │ S60 │

S60
┤S├───────( Y7 )
    X12
    ┤├────( S1 )

          │ RET │

          │ END │
```

SFC:

```
M1002 ─┤
      ┌────┐
      │ S1 │
      └────┘
X0 ─┤
      ┌────┐
      │S20 │──( Y0 )
      └────┘
  ┌──────┬──────────────┬──────────────┐
X1─┤    X4─┤           X7─┤
┌────┐    ┌────┐         ┌────┐
│S30 │─(Y1)│S31 │─(Y3)   │S32 │─(Y5)
└────┘    └────┘         └────┘
X2─┤    X5─┤           X10─┤
┌────┐    ┌────┐         ┌────┐
│S40 │─(Y2)│S41 │─(Y4)   │S42 │─(Y6)
└────┘    └────┘         └────┘
X3─┤    X6─┤           X11─┤
  └──────┴──────────────┴──────────────┘
      ┌────┐
      │S50 │──┤ TMR │ T1 │ K10 │
      └────┘
T1 ─┤
      ┌────┐
      │S60 │──( Y7 )
      └────┘
X12 ─┤
        ↓
      ┌────┐
      │ S1 │
      └────┘
```
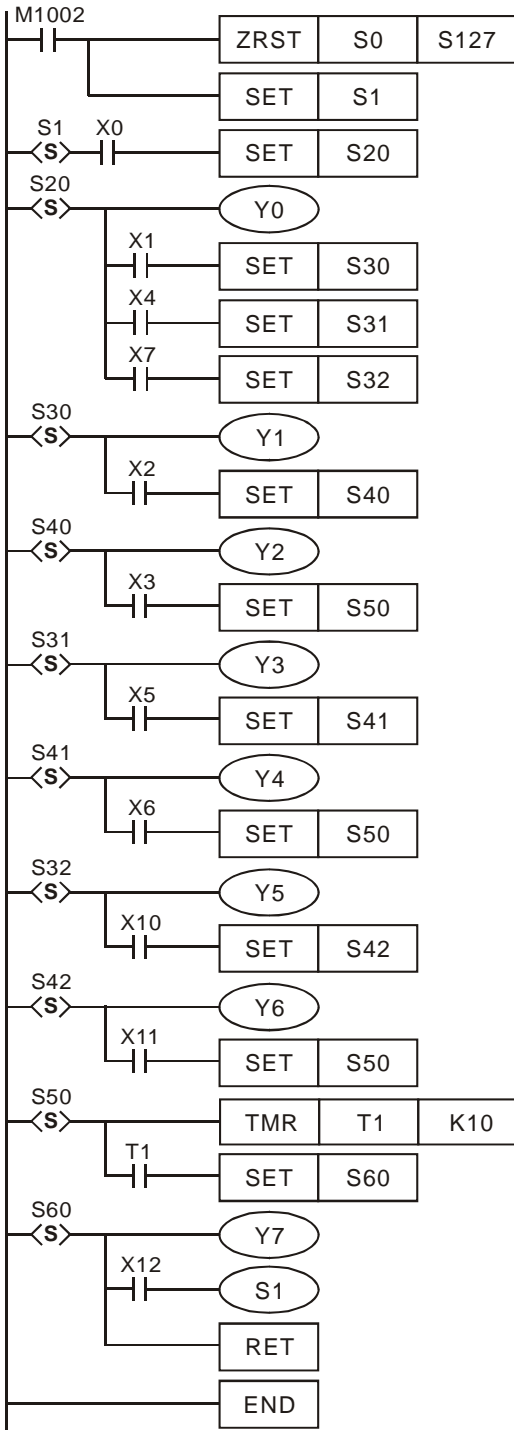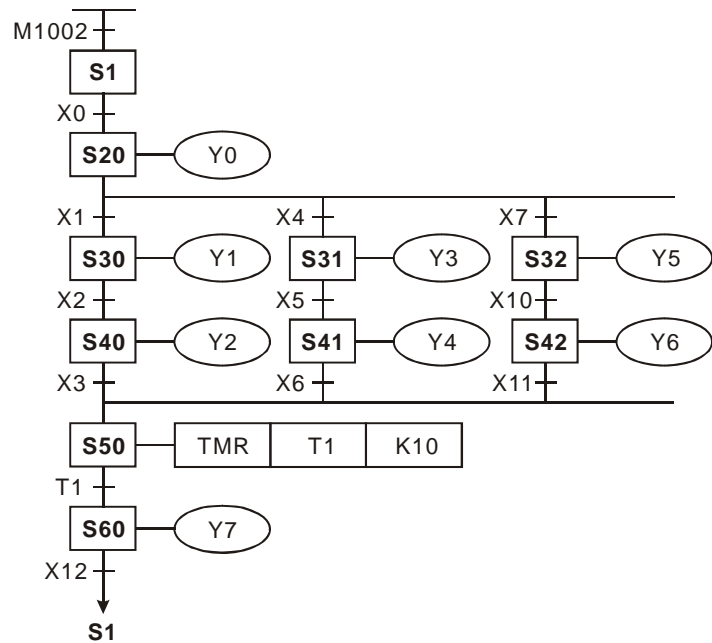
Examples of simultaneous divergence & simultaneous convergence:

Ladder diagram:
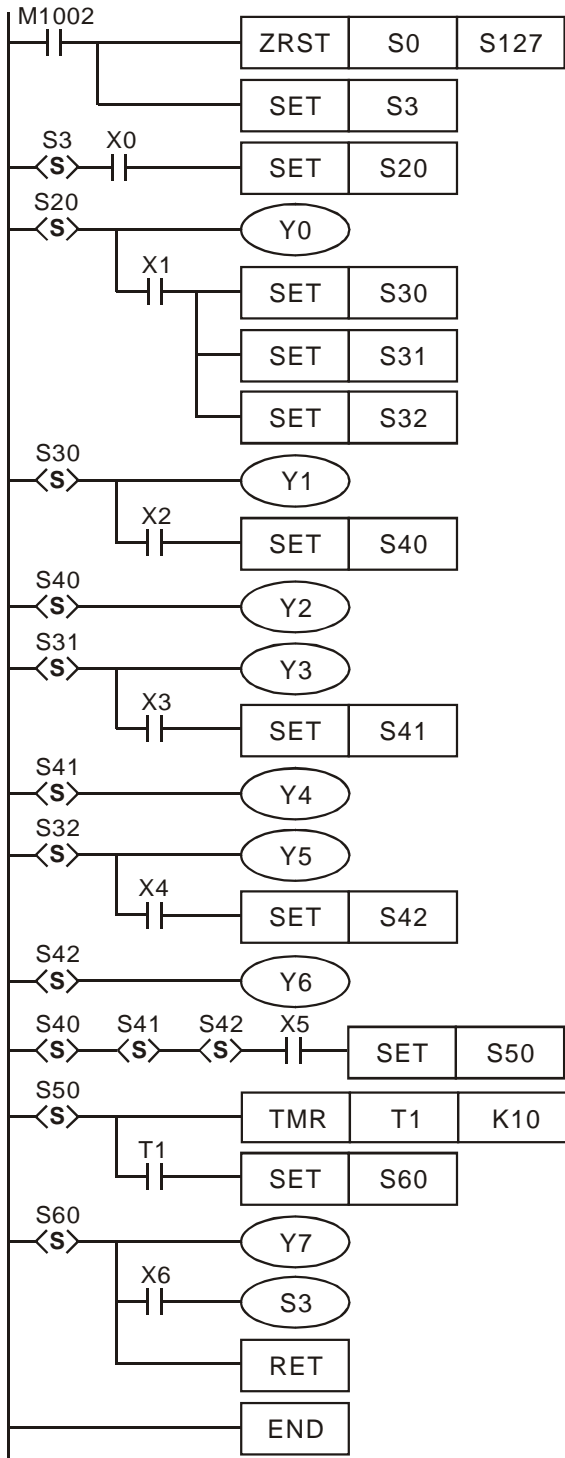
SFC:
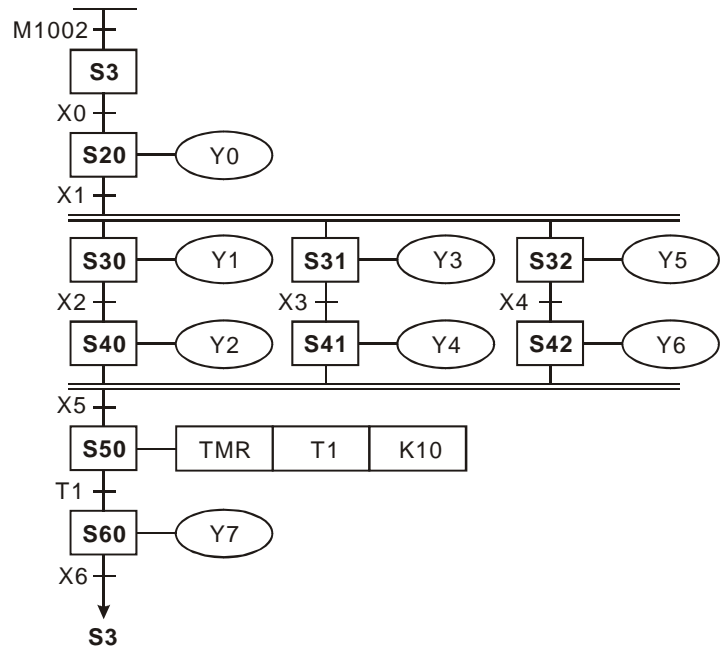
Example of the simultaneous divergence & alternative convergence:

Ladder diagram:

M1002		
ZRST	S0	S127
SET	S4	

| S4 | X0 | | |
| SET | S20 | |

S20
- Y0
- X1
  - SET S30
  - SET S31
  - SET S32

S30
- Y1
- X2
  - SET S40

S40
- Y2
- X3
  - SET S50

S31
- Y3
- X4
  - SET S41

S41
- Y4
- X5
  - SET S50

S32
- Y5
- X6
  - SET S42

S42
- Y6
- X7
  - SET S50

S50
- TMR T1 K10
- T1
  - SET S60

S60
- Y7
- X6
  - S4
- RET
- END

SFC:
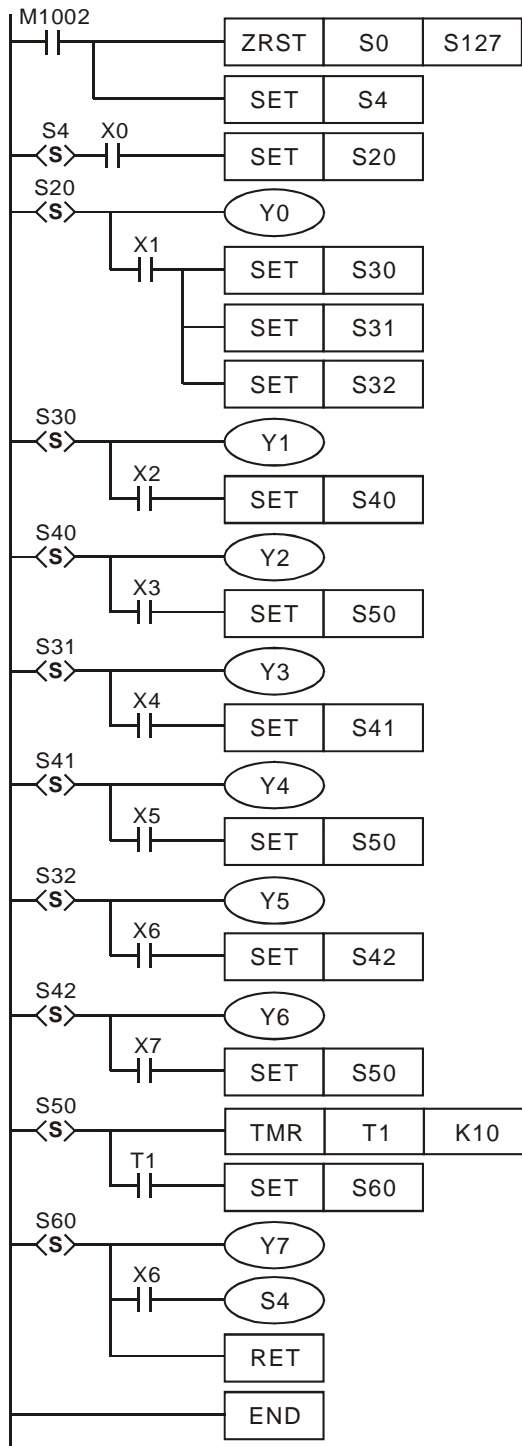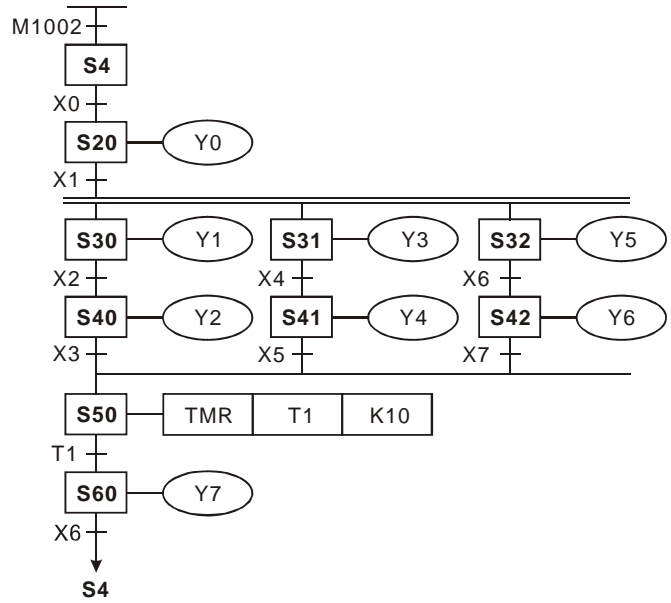
Combination Example 1:

(Including alternative divergence/convergence and simultaneous divergence/convergence)

Ladder diagram:

SFC:
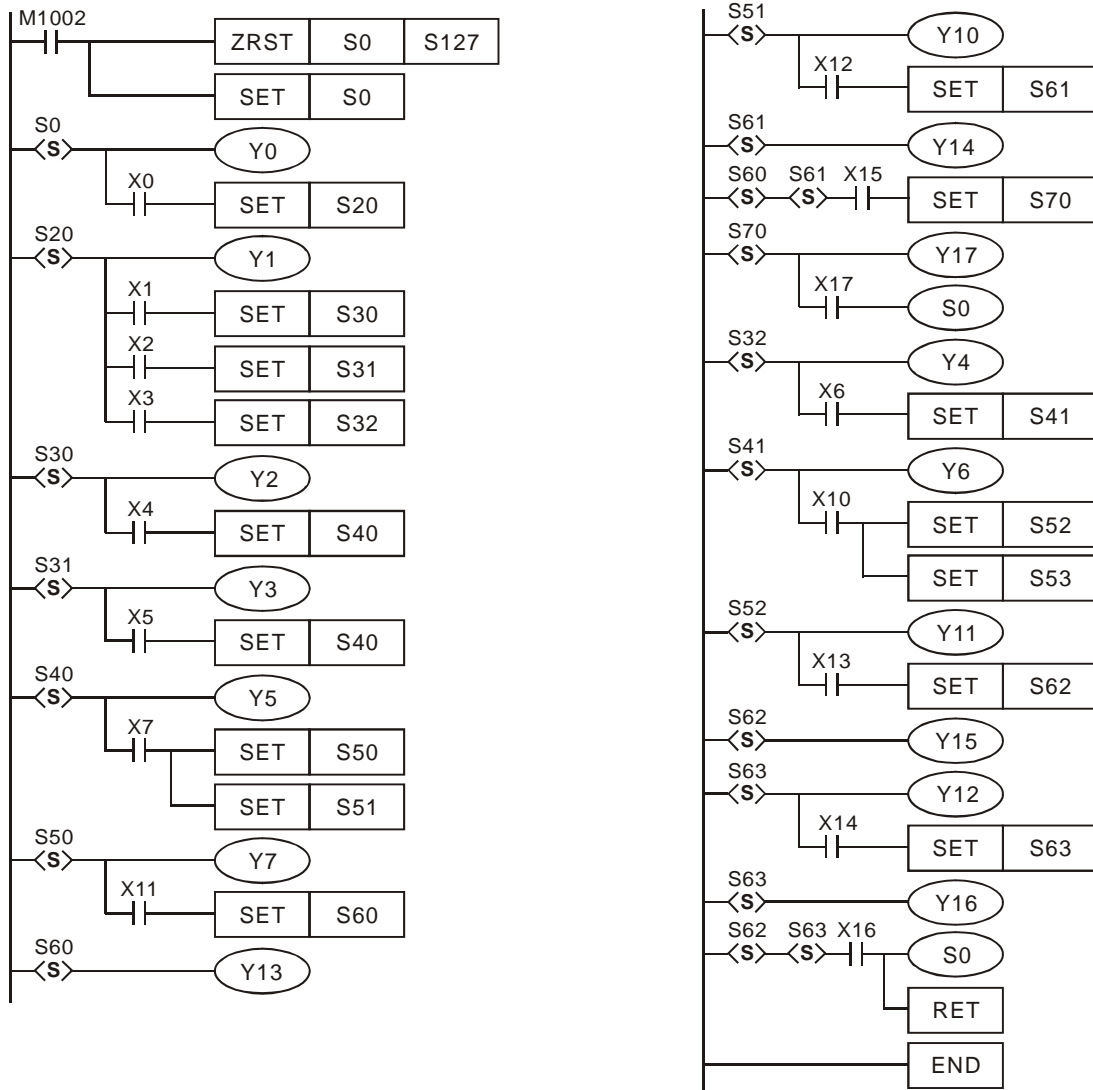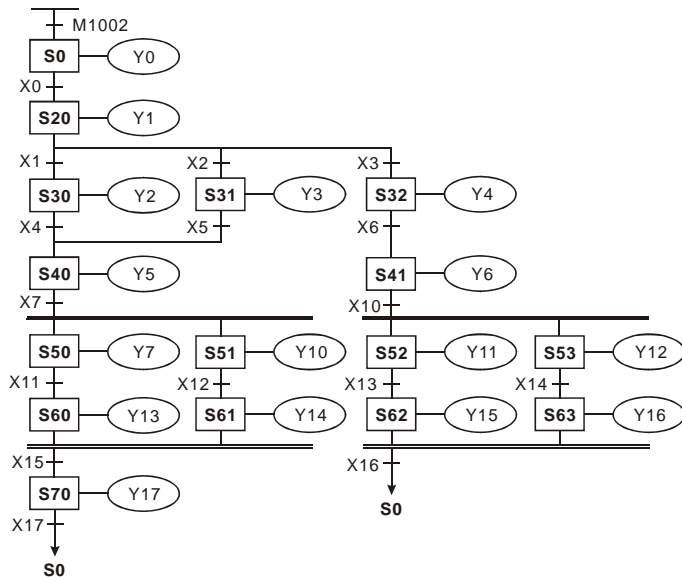
Combination Example 2:

(Including alternative divergence/convergence and simultaneous divergence/convergence)
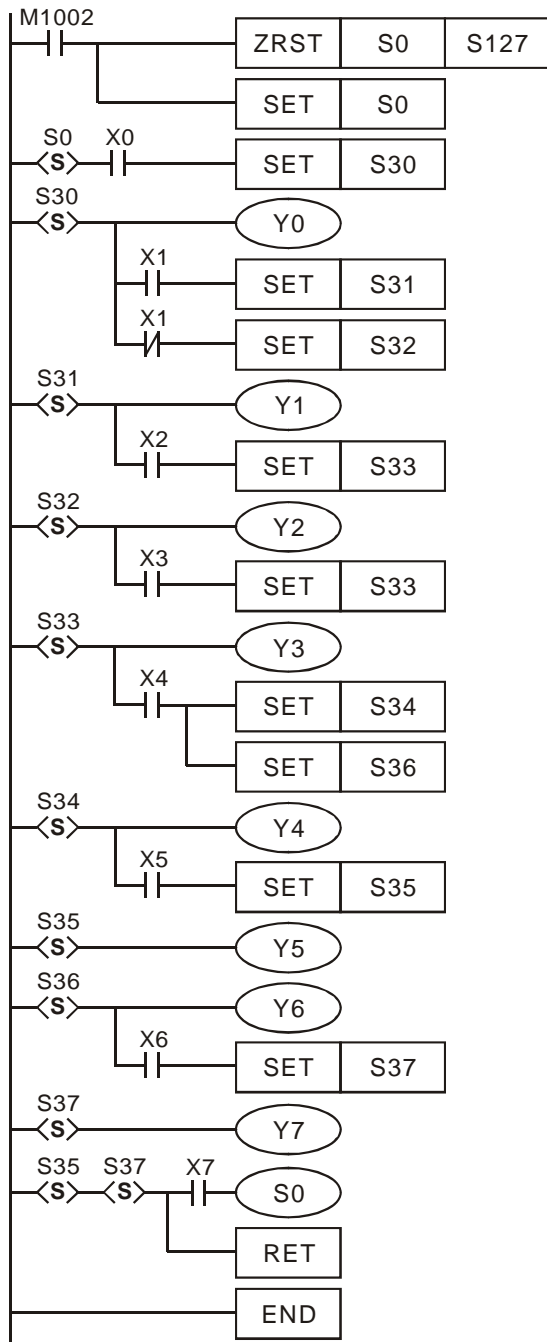
Ladder diagram: SFC:
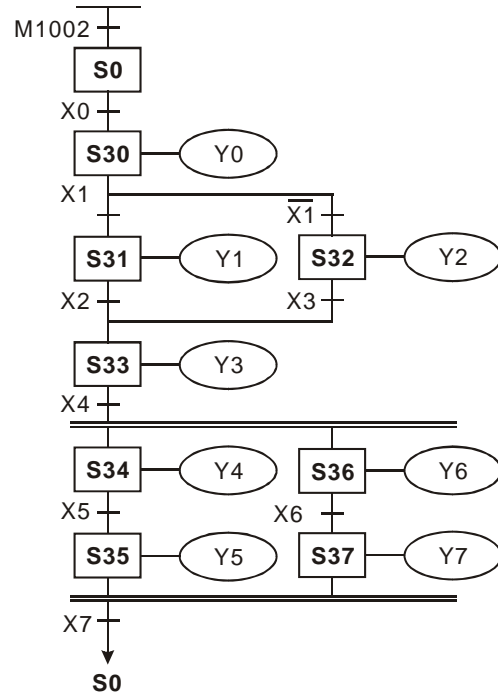
Restrictions on Divergence Sequence:

1. You can use maximum 8 divergence steps in a divergence sequence. As the diagram below, there are maximum 8 diverged steps S30 ~ S37 after step S20.

2. You can use maximum 16 loops in multiple divergence sequences or in simultaneous sequences combined into one sequence. 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. You can designate a step in the sequence to jump to any step in another sequence.

4.6 IST Instruction

API	Mnemonic	Operands	Function	Controllers
60	IST	S D_1 D_2	Initial State	ES/EX/SS SA/SX/SC EH/SV

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	*	*	*													
D₁				*												
D₂				*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device in the designated operation mode   **D₁**: The smallest No. of designated step in auto mode

D₂: The biggest No. of designated step in auto mode

Explanations:

1. **S** will occupy 8 consecutive points.

2. Range of **D₁** and **D₂**: for SA/SX/SC/EH/EH2/SV S20 ~ S899; for ES/EX/SS S20 ~ S127; **D₂** > **D₁**.

3. See the specifications of each model for their range of use.

4. IST instruction can only be used once in the program.

5. Flags: M1040 ~ M1047. See remarks for more details.

6. IST instruction is a handy instruction specifically for the initial status of step ladder control procedure to accommodate special auxiliary relay.

Program Example 1:

1. Use of IST instruction

```
 M1000
├──┤├──┤  IST  │  X10  │  S20  │  S60  │
```

  S   X10: Individual operation   X14: Continuous operation
        X11: Zero return   X15: Zero return enabled switch
        X12: Step operation   X16: Start switch
        X13: One cycle operation   X17: Stop switch

2. When IST instruction is being executed, the following special auxiliary relays will switch automatically.

       M1040: Operation forbidden   S0: Initiates manual operation
       M1041: Operation starts   S1: Initiates zero return
       M1042: Pulse output enabled   S2: Initiates auto operation
       M1047: STL monitor enabled

3. S10 ~ S19 are for zero return and cannot be used as general steps. When S0 ~ S9 are in use, S0 ~ S2 represent manual operation mode, zero return mode and auto operation mode. Therefore, in the program, you have to write the circuit of the three steps in advance.

4. When switched to S1 (zero return) mode, any On in S10 ~ S19 will result in no zero return.

5. When switched to S2 (auto operation) mode, any On of the S in $D_1 \sim D_2$ or M1043 = On will result in no auto operation.

Program Example 2:

1. Robot arm control (by IST instruction):

     a) Motion request: Separate the big ball and small ball and move them to different boxes. Configure the control panel for the control.

     b) Motions of the robot arm: descending, clipping ball, ascending, right shifting, releasing ball, ascending, left shifting.

     c) I/O devices:

2. Operation modes:

Manual operation: Turn On/Off of the load by a single button.

Zero return: Press the zero return button to automatically zero-return the machine.

Auto operation:

     a) Single step operation: Press "auto start" button for every one step forward.

     b) One cycle operation: Press "auto start" button at the zero point. After a cycle of auto operation, the operation will stops at the zero point. Press "auto stop" button in the middle of the operation to stop the operation and press "auto start" to restart the operation. The operation will resume until it meets the zero point.

     c) Continuous operation: Press "auto start" button at the zero point to resume the operation. Press "auto stop" to operate until it meets the zero point.

3. The control panel:

a) Ball size sensor X0.

b) Robot arm: left limit X1, big ball right limit X2, small ball right limit X3, upper limit X4, lower limit X5.

c) Robot arm: ascending Y0, descending Y1, right shifting Y2, left shifting Y3, clipping Y4.

Start Circuit

Manual Operation Mode

Zero Return Mode

SFC:

Ladder Diagram:

S1 X15 SET	S10	Enter zero return mode
S10 RST	Y4	Clipping released
RST	Y1	Descending stops
Y0		Robot arm ascends to upper limit (X4 On)
X4 SET	S11	
S11 RST	Y2	Right shifting stops
Y3		Robot arm left shifting to left limit (X1 On)
X1 SET	S12	
S12 SET	M1043	Enable zero return completed flag
RST	S12	Zero return operation completed

Auto Operation Modes

SFC:

S2

M1041
M1044

S20 — Y1

X5 / X0 → S30 — SET Y4 / TMR T0 K30

X5 / X̄0 → S40 — SET Y4 / TMR T1 K30

T0 → S31 —X4— Y0

T1 → S41 —X4— Y0

X4 → S32 —X2— Y2

X4 → S42 —X3— Y2

X2 / X3 → S50 —X5— Y1

X5 → S60 — RST Y4 / TMR T2 K30

T2 → S70 —X4— Y0

X4 → S80 —X1— Y3

X1 → S2

Ladder Diagram:

S2 M1041 M1044 —⟨S⟩—┤├——┤├	SET	S20	Enter auto operation mode

S20 —⟨S⟩— (Y1) Robot arm descending

X5 X0 ┤├—┤├ SET S30

X5 X0 ┤├—┤/ SET S40

S30 —⟨S⟩— SET Y4 Clipping tightly

TMR T0 K30

T0 ┤├ SET S31

S31 X4 —⟨S⟩—┤/ (Y0) Robot arm ascending to upper limit (X4 On)

X4 ┤├ SET S32

S32 X2 —⟨S⟩—┤/ (Y2) Right shifting

X2 ┤├ SET S50

S40 —⟨S⟩— SET Y4 Clipping tightly

TMR T1 K30

T1 ┤├ SET S41

S41 X4 —⟨S⟩—┤/ (Y0) Robot arm ascending to upper limit (X4 On)

X4 ┤├ SET S42

S42 X3 —⟨S⟩—┤/ (Y2) Right shifting

X3 ┤├ SET S50

S50 X5 —⟨S⟩—┤/ (Y1) Robot arm descending

X5 ┤├ SET S60

S60 —⟨S⟩— RST Y4 Clipping released

TMR T2 K30

T2 ┤├ SET S70

S70 X4 —⟨S⟩—┤/ (Y0) Robot arm ascending to upper limit (X4 On)

X4 ┤├ SET S80

S80 X1 —⟨S⟩—┤/ (Y3) Robot arm left shifting to left limit (X1 On)

X1 ┤├ (S2)

RET

END

MEMO

5.1 List of Instructions

Category	API	Mnemonic		P instruction	Function	Applicable to			STEPS		Page
		16-bit	32-bit			ES	SA	EH	16-bit	32-bit	
Loop Control	00	CJ	-	✓	Conditional Jump	✓	✓	✓	3	-	6-1
	01	CALL	-	✓	Call Subroutine	✓	✓	✓	3	-	6-5
	02	SRET	-	-	Subroutine Return	✓	✓	✓	1	-	6-5
	03	IRET	-	-	Interrupt Return	✓	✓	✓	1	-	6-8
	04	EI	-	-	Enable Interrupts	✓	✓	✓	1	-	6-8
	05	DI	-	-	Disable Interrupts	✓	✓	✓	1	-	6-8
	06	FEND	-	-	The End of The Main Program (First End)	✓	✓	✓	1	-	6-12
	07	WDT	-	✓	Watchdog Timer Refresh	✓	✓	✓	1	-	6-14
	08	FOR	-	-	Start of a FOR-NEXT loop	✓	✓	✓	3	-	6-15
	09	NEXT	-	-	End of a FOR-NEXT loop	✓	✓	✓	1	-	6-15
Transmission Comparison	10	CMP	DCMP	✓	Compare	✓	✓	✓	7	13	6-18
	11	ZCP	DZCP	✓	Zone Compare	✓	✓	✓	9	17	6-19
	12	MOV	DMOV	✓	Move	✓	✓	✓	5	9	6-20
	13	SMOV	-	✓	Shift Move	–	✓	✓	11	-	6-21
	14	CML	DCML	✓	Compliment	✓	✓	✓	5	9	6-23
	15	BMOV	-	✓	Block Move	✓	✓	✓	7	–	6-24
	16	FMOV	DFMOV	✓	Fill Move	✓	✓	✓	7	13	6-26
	17	XCH	DXCH	✓	Exchange	✓	✓	✓	5	9	6-27
	18	BCD	DBCD	✓	Binary Coded Decimal	✓	✓	✓	5	9	6-29
	19	BIN	DBIN	✓	Binary	✓	✓	✓	5	9	6-30
Four Arithmetic Operation	20	ADD	DADD	✓	Addition	✓	✓	✓	7	13	6-32
	21	SUB	DSUB	✓	Subtraction	✓	✓	✓	7	13	6-34
	22	MUL	DMUL	✓	Multiplication	✓	✓	✓	7	13	6-35
	23	DIV	DDIV	✓	Division	✓	✓	✓	7	13	6-37
	24	INC	DINC	✓	Increment	✓	✓	✓	3	5	6-39
	25	DEC	DDEC	✓	Decrement	✓	✓	✓	3	5	6-40
	26	WAND	DAND	✓	Logical Word AND	✓	✓	✓	7	13	6-41
	27	WOR	DOR	✓	Logical Word OR	✓	✓	✓	7	13	6-42
	28	WXOR	DXOR	✓	Logical Exclusive OR	✓	✓	✓	7	13	6-43
	29	NEG	DNEG	✓	2's Complement (Negative)	✓	✓	✓	3	5	6-44
Rotation & Displacement	30	ROR	DROR	✓	Rotation Right	✓	✓	✓	5	9	6-46
	31	ROL	DROL	✓	Rotation Left	✓	✓	✓	5	9	6-47
	32	RCR	DRCR	✓	Rotation Right with Carry	✓	✓	✓	5	9	6-48
	33	RCL	DRCL	✓	Rotation Left with Carry	✓	✓	✓	5	9	6-49
	34	SFTR	-	✓	Bit Shift Right	✓	✓	✓	9	-	6-50
	35	SFTL	-	✓	Bit Shift Left	✓	✓	✓	9	-	6-51
	36	WSFR	-	✓	Word Shift Right	-	✓	✓	9	-	6-52
	37	WSFL	-	✓	Word Shift Left	-	✓	✓	9	-	6-54
	38	SFWR	-	✓	Shift Register Write	-	✓	✓	7	-	6-55
	39	SFRD	-	✓	Shift Register Read	-	✓	✓	7	-	6-56
Data Processing	40	ZRST	-	✓	Zero Reset	✓	✓	✓	5	-	6-57
	41	DECO	-	✓	Decode	✓	✓	✓	7	-	6-59
	42	ENCO	-	✓	Encode	✓	✓	✓	7	-	6-61
	43	SUM	DSUM	✓	Sum of Active Bits	✓	✓	✓	5	9	6-63
	44	BON	DBON	✓	Check Specified Bit Status	✓	✓	✓	7	13	6-64
	45	MEAN	DMEAN	✓	Mean	✓	✓	✓	7	13	6-65
	46	ANS	-	-	Timed Annunciator Set	-	✓	✓	7	-	6-66
	47	ANR	-	✓	Annunciator Reset	-	✓	✓	1	-	6-66
	48	SQR	DSQR	✓	Square Root	✓	✓	✓	5	9	6-72
	49	FLT	DFLT	✓	Floating Point	✓	✓	✓	5	9	6-70
High Speed Processing	50	REF	-	✓	Refresh	✓	✓	✓	5	-	7-1
	51	REFF	-	✓	Refresh and Filter Adjust	-	✓	✓	3	-	7-2
	52	MTR	-	-	Input Matrix	-	✓	✓	9	-	7-3
	53	-	DHSCS	-	High Speed Counter Set	✓	✓	✓	-	13	7-5
	54	-	DHSCR	-	High Speed Counter Reset	✓	✓	✓	–	13	7-15

Category	API	Mnemonic 16-bit	Mnemonic 32-bit	P instruction	Function	ES	SA	EH	STEPS 16-bit	STEPS 32-bit	Page
High Speed Processing	55	-	DHSZ	-	High Speed Zone Compare	-	✓	✓	-	17	7-17
	56	SPD	-	-	Speed Detection	✓	✓	✓	7	-	7-24
	57	PLSY	DPLSY	-	Pulse Y Output	✓	✓	✓	7	13	7-26
	58	PWM	-	-	Pulse Width Modulation	✓	✓	✓	7	-	7-32
	59	PLSR	DPLSR	-	Pulse Ramp	✓	✓	✓	9	17	7-35
Handy Instructions	60	IST	-	-	Initial State	✓	✓	✓	7	-	7-39
	61	SER	DSER	✓	Search a Data Stack	-	✓	✓	9	17	7-45
	62	ABSD	DABSD	-	Absolute Drum Sequencer	-	✓	✓	9	17	7-46
	63	INCD	-	-	Incremental Drum Sequencer	-	✓	✓	9	-	7-48
	64	TTMR	-	-	Teaching Timer	-	✓	✓	5	-	7-50
	65	STMR	-	-	Special Timer	-	✓	✓	7	-	7-52
	66	ALT	-	✓	Alternate State	✓	✓	✓	3	-	7-54
	67	RAMP	-	-	Ramp Variable Value	-	✓	✓	9	-	7-55
	69	SORT	-	-	Sort Tabulated Data	-	✓	✓	11	-	7-57
Display of External Settings	70	TKY	DTKY	-	Ten Key Input	-	✓	✓	7	13	7-59
	71	HKY	DHKY	-	Hexadecimal Key Input	-	✓	✓	9	17	7-61
	72	DSW	-	-	Digital Switch	-	✓	✓	9	-	7-64
	73	SEGD	-	✓	Seven Segment Decoder	✓	✓	✓	5	-	7-66
	74	SEGL	-	-	Seven Segment with Latch	✓	✓	✓	7	-	7-67
	75	ARWS	-	-	Arrow Switch	-	✓	✓	9	-	7-70
	76	ASC	-	-	ASCII Code Conversion	-	✓	✓	11	-	7-72
	77	PR	-	-	Print (ASCII Code Output)	-	✓	✓	5	-	7-73
Serial I/O	78	FROM	DFROM	✓	Read CR Data in Special Modules	✓	✓	✓	9	17	7-75
	79	TO	DTO	✓	Write CR Data into Special Modules	✓	✓	✓	9	17	7-76
	80	RS	-	-	Serial Communication Instruction	✓	✓	✓	9	-	7-80
	81	PRUN	DPRUN	✓	Parallel Run	-	✓	✓	5	9	7-93
	82	ASCI	-	✓	Converts Hex to ASCII	✓	✓	✓	7	-	7-94
	83	HEX	-	✓	Converts ASCII to Hex	✓	✓	✓	7	-	7-98
	84	CCD	-	✓	Check Code	-	✓	✓	7	-	7-101
	85	VRRD	-	✓	Volume Read	-	✓	✓	5	-	7-103
	86	VRSC	-	✓	Volume Scale	-	✓	✓	5	-	7-105
	87	ABS	DABS	✓	Absolute Value	✓	✓	✓	3	5	7-106
	88	PID	DPID	-	PID Control Loop	✓	✓	✓	9	17	7-107
Basic Instructions	89	PLS	-	-	Rising-edge Output	✓	✓	✓	3	-	3-13
	90	LDP	-	-	Rising-edge Detection Operation	✓	✓	✓	3	-	3-11
	91	LDF	-	-	Falling-edge Detection Operation	✓	✓	✓	3	-	3-12
	92	ANDP	-	-	Rising-edge Series Connection	✓	✓	✓	3	-	3-12
	93	ANDF	-	-	Falling-edge Series Connection	✓	✓	✓	3	-	3-12
	94	ORP	-	-	Rising-edge Parallel Connection	✓	✓	✓	3	-	3-13
	95	ORF	-	-	Falling-edge Parallel Connection	✓	✓	✓	3	-	3-13
	96	TMR	-	-	16-bit Timer	✓	✓	✓	4	-	3-8
	97	CNT	DCNT	-	16-bit / 32-bit Counter	✓	✓	✓	4	6	3-9
	98	INV	-	-	Inverting Operation	✓	✓	✓	1	-	3-15
	99	PLF	-	-	Falling-edge Output	✓	✓	✓	3	-	3-14
Communication	100	MODRD	-	-	Read Modbus Data	✓	✓	✓	7	-	8-1
	101	MODWR	-	-	Write Modbus Data	✓	✓	✓	7	-	8-6
	102	FWD	-	-	Forward Running of VFD-A	✓	✓	✓	7	-	8-11
	103	REV	-	-	Reverse Running of VFD-A	✓	✓	✓	7	-	8-11
	104	STOP	-	-	Stop VFD-A	✓	✓	✓	7	-	8-11
	105	RDST	-	-	Read VFD-A Status	✓	✓	✓	5	-	8-14
	106	RSTEF	-	-	Reset Abnormal VFD-A	✓	✓	✓	5	-	8-16
	107	LRC	-	✓	Checksum LRC Mode	✓	✓	✓	7	-	8-17
	108	CRC	-	✓	Checksum CRC Mode	✓	✓	✓	7	-	8-19
	109	SWRD	-	✓	Read Digital Switch	-	-	✓	3	-	8-22

Category	API	Mnemonic		P instruction	Function	Applicable to			STEPS		Page
		16-bit	32-bit			ES	SA	EH	16-bit	32-bit	
Floating Point Operation	110	–	DECMP	✓	Floating Point Compare	✓	✓	✓	-	13	8-23
	111	–	DEZCP	✓	Floating Point Zone Compare	✓	✓	✓	-	17	8-24
	112	-	DMOVR	✓	Move Floating Point Data	✓	✓	✓	-	9	8-25
	116	-	DRAD	✓	Angle → Radian	-	✓	✓	-	9	8-26
	117	-	DDEG	✓	Radian → Angle	-	✓	✓	-	9	8-27
	118	-	DEBCD	✓	Float to Scientific Conversion	✓	✓	✓	-	9	8-28
	119	-	DEBIN	✓	Scientific to Float Conversion	✓	✓	✓	-	9	8-29
	120	-	DEADD	✓	Floating Point Addition	✓	✓	✓	-	13	8-31
	121	-	DESUB	✓	Floating Point Subtraction	✓	✓	✓	-	13	8-32
	122	-	DEMUL	✓	Floating Point Multiplication	✓	✓	✓	-	13	8-33
	123	-	DEDIV	✓	Floating Point Division	✓	✓	✓	-	13	8-34
	124	-	DEXP	✓	Exponent of Binary Floating Point	✓	✓	✓	-	9	8-35
	125	-	DLN	✓	Natural Logarithm of Binary Floating Point	✓	✓	✓	-	9	8-36
	126	-	DLOG	✓	Logarithm of Binary Floating Point	✓	✓	✓	-	13	8-37
	127	-	DESQR	✓	Floating Point Square Root	✓	✓	✓	-	9	8-39
	128	-	DPOW	✓	Floating Point Power Operation	✓	✓	✓	-	13	8-40
	129	INT	DINT	✓	Float to Integer	✓	✓	✓	5	9	8-42
	130	-	DSIN	✓	Sine	✓	✓	✓	-	9	8-43
	131	-	DCOS	✓	Cosine	✓	✓	✓	-	9	8-45
	132	-	DTAN	✓	Tangent	✓	✓	✓	-	9	8-47
	133	-	DASIN	✓	Arc Sine	-	✓	✓	-	9	8-49
	134	-	DACOS	✓	Arc Cosine	-	✓	✓	-	9	8-50
	135	-	DATAN	✓	Arc Tangent	-	✓	✓	-	9	8-51
	136	-	DSINH	✓	Hyperbolic Sine	-	-	✓	-	9	8-52
	137	-	DCOSH	✓	Hyperbolic Cosine	-	-	✓	-	9	8-53
	138	-	DTANH	✓	Hyperbolic Tangent	-	-	✓	-	9	8-54
Others	143	DELAY	-	✓	Delay Instruction	-	✓	✓	3	-	8-55
	144	GPWM	-	-	General PWM Output	-	✓	✓	7	-	8-57
	145	FTC	-	-	Fuzzy Temperature Control	-	✓	✓	9	-	8-58
	146	CVM	-	-	Valve Control (*)	-	-	✓	7	-	8-63
	147	SWAP	DSWAP	✓	Byte Swap	✓	✓	✓	3	5	8-66
	148	MEMR	DMEMR	✓	Read File Register	-	✓	✓	7	13	8-67
	149	MEMW	DMEMW	✓	Write File Register	-	✓	✓	7	13	8-68
	150	MODRW	-	-	Read/Write MODBUS Data	✓	✓	✓	11	-	9-1
	151	PWD	-	-	Detection of Input Pulse Width	-	-	✓	5	-	9-10
	152	RTMU	-	-	Start of the Measurement of Execution Time of I Interruption	-	-	✓	5	-	9-11
	153	RTMD	-	-	End of the Measurement of the Execution Time of I Interruption	-	-	✓	3	-	9-11
	154	RAND	-	✓	Random Number	-	✓	✓	7	-	9-13
Position Control	155	-	DABSR	-	Read the Absolute Position from a Servo Motor	-	✓	✓	7	13	9-14
	156	ZRN	DZRN	-	Zero Return	-	-	✓	9	17	9-19
	157	PLSV	DPLSV	-	Adjustable Speed Pulse Output	-	-	✓	7	13	9-23
	158	DRVI	DDRVI	-	Drive to Increment	-	-	✓	9	17	9-24
	159	DRVA	DDRVA	-	Drive to Absolute	-	-	✓	9	17	9-30
Real Time Calendar	160	TCMP	-	✓	Time Compare	-	✓	✓	11	-	9-39
	161	TZCP	-	✓	Time Zone Compare	-	✓	✓	9	-	9-40
	162	TADD	-	✓	Time Addition	-	✓	✓	7	-	9-41
	163	TSUB	-	✓	Time Subtraction	-	✓	✓	7	-	9-42
	166	TRD	-	✓	Time Read	-	✓	✓	3	-	9-43
	167	TWR	-	✓	Time Write	-	✓	✓	3	-	9-45
	169	HOUR	DHOUR	-	Hour Meter	-	✓	✓	7	13	9-47
	170	GRY	DGRY	✓	BIN → Gray Code	-	✓	✓	5	9	9-49
	171	GBIN	DGBIN	✓	Gray Code → BIN	-	✓	✓	5	9	9-50

| Category | API | Mnemonic | | P instruction | Function | Applicable to | | | STEPS | | Page |
		16-bit	32-bit			ES	SA	EH	16-bit	32-bit	
Floating Point Operation	172	-	DADDR	✓	Floating Point Addition	✓	✓	✓	-	13	9-51
	173	-	DSUBR	✓	Floating Point Subtraction	✓	✓	✓	-	13	9-53
	174	-	DMULR	✓	Floating Point Multiplication	✓	✓	✓	-	13	9-55
	175	-	DDIVR	✓	Floating Point Division	✓	✓	✓	-	13	9-57
Matrix	180	MAND	-	✓	Matrix 'AND' Operation	-	✓	✓	9	-	9-59
	181	MOR	-	✓	Matrix 'OR' Operation	-	✓	✓	9	-	9-61
	182	MXOR	-	✓	Matrix 'XOR' Operation	-	✓	✓	9	-	9-62
	183	MXNR	-	✓	Matrix 'XNR' Operation	-	✓	✓	9	-	9-63
	184	MINV	-	✓	Matrix Inverse Operation	-	✓	✓	7	-	9-64
	185	MCMP	-	✓	Matrix Compare	-	✓	✓	9	-	9-65
	186	MBRD	-	✓	Read Matrix Bit	-	✓	✓	7	-	9-67
	187	MBWR	-	✓	Write Matrix Bit	-	✓	✓	7	-	9-69
	188	MBS	-	✓	Matrix Bit Displacement	-	✓	✓	7	-	9-71
	189	MBR	-	✓	Matrix Bit Rotation	-	✓	✓	7	-	9-73
	190	MBC	-	✓	Matrix Bit Status Counting	-	✓	✓	7	-	9-75
Positioning Instruction	191	-	DPPMR	-	2-Axis Relative Point to Point Motion (*)	-	-	✓	-	17	9-76
	192	-	DPPMA	-	2-Axis Absolute Point to Point Motion (*)	-	-	✓	-	17	9-79
	193	-	DCIMR	-	2-Axis Relative Position Arc Interpolation (*)	-	-	✓	-	17	9-81
	194	-	DCIMA	-	2-Axis Absolute Position Arc Interpolation (*)	-	-	–	-	17	9-86
	195	-	DPTPO	-	Single-Axis Pulse Output by Table (*)	-	-	–	-	13	9-91
	196	HST	-	✓	High Speed Timer	-	-	✓	3	–	9-93
	197	-	DCLLM	-	Close Loop Position Control (*)	-	-	✓	-	17	9-95
	202	SCAL	-	✓	Proportional Value Calculation	✓	✓	✓	9	-	10-1
	203	SCLP	-	✓	Parameter Proportional Value Calculation	✓	✓	✓	9	-	10-3
Contact Type Logic Operation	215	LD&	DLD&	-	S_1 & S_2	-	✓	✓	5	9	10-7
	216	LD|	DLD|	-	S_1 | S_2	-	✓	✓	5	9	10-7
	217	LD^	DLD^	-	S_1 ^ S_2	-	✓	✓	5	9	10-7
	218	AND&	DAND&	-	S_1 & S_2	-	✓	✓	5	9	10-8
	219	AND|	DAND|	-	S_1 | S_2	-	✓	✓	5	9	10-8
	220	AND^	DAND^	-	S_1 ^ S_2	-	✓	✓	5	9	10-8
	221	OR&	DOR&	-	S_1 & S_2	-	✓	✓	5	9	10-9
	222	OR|	DOR|	-	S_1 | S_2	-	✓	✓	5	9	10-9
	223	OR^	DOR^	-	S_1 ^ S_2	-	✓	✓	5	9	10-9
Contact Type Comparison Instruction	224	LD=	DLD=	-	$S_1 = S_2$	✓	✓	✓	5	9	10-10
	225	LD>	DLD>	-	$S_1 > S_2$	✓	✓	✓	5	9	10-10
	226	LD<	DLD<	-	$S_1 < S_2$	✓	✓	✓	5	9	10-10
	228	LD<>	DLD<>	-	$S_1 \neq S_2$	✓	✓	✓	5	9	10-10
	229	LD<=	DLD<=	-	$S_1 \leq S_2$	✓	✓	✓	5	9	10-10
	230	LD>=	DLD>=	-	$S_1 \geq S_2$	✓	✓	✓	5	9	10-10
	232	AND=	DAND=	-	$S_1 = S_2$	✓	✓	✓	5	9	10-11
	233	AND>	DAND>	-	$S_1 > S_2$	✓	✓	✓	5	9	10-11
	234	AND<	DAND<	-	$S_1 < S_2$	✓	✓	✓	5	9	10-11
	236	AND<>	DAND<>	-	$S_1 \neq S_2$	✓	✓	✓	5	9	10-11
	237	AND<=	DAND<=	-	$S_1 \leq S_2$	✓	✓	✓	5	9	10-11
	238	AND>=	DAND>=	-	$S_1 \geq S_2$	✓	✓	✓	5	9	10-11
	240	OR=	DOR=	-	$S_1 = S_2$	✓	✓	✓	5	9	10-12
	241	OR>	DOR>	-	$S_1 > S_2$	✓	✓	✓	5	9	10-12
	242	OR<	DOR<	-	$S_1 < S_2$	✓	✓	✓	5	9	10-12
	244	OR<>	DOR<>	-	$S_1 \neq S_2$	✓	✓	✓	5	9	10-12
	245	OR<=	DOR<=	-	$S_1 \leq S_2$	✓	✓	✓	5	9	10-12
	246	OR>=	DOR>=	-	$S_1 \geq S_2$	✓	✓	✓	5	9	10-12

*Instructions marked with * are available in EH2/SV series MPU.*

Note:
1. For applicable models, ES includes ES/EX/SS; SA includes SA/SX/SC; EH includes EH/EH2/SV.
2. ES/EX/SS series MPU does not support pulse execution type instructions (P instruction).

5.2 Composition of Application Instruction

■ An application instruction has two parts: the instruction and operands.

Instruction: The function of the instruction

Operands: Devices for processing the operations of the instruction

The instruction part of an application instruction usually occupies 1 step, and one operand occupies 2 or 4 steps depending on the instruction is a 16-bit or 32-bit one.

■ Format of an application instruction:

API	Mnemonic		Operands	Function		Controllers		
41	DECO	P	(S) (D) (n)	Decode		ES /EX /SS	SA /SX /SC	EH /S V

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								
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV
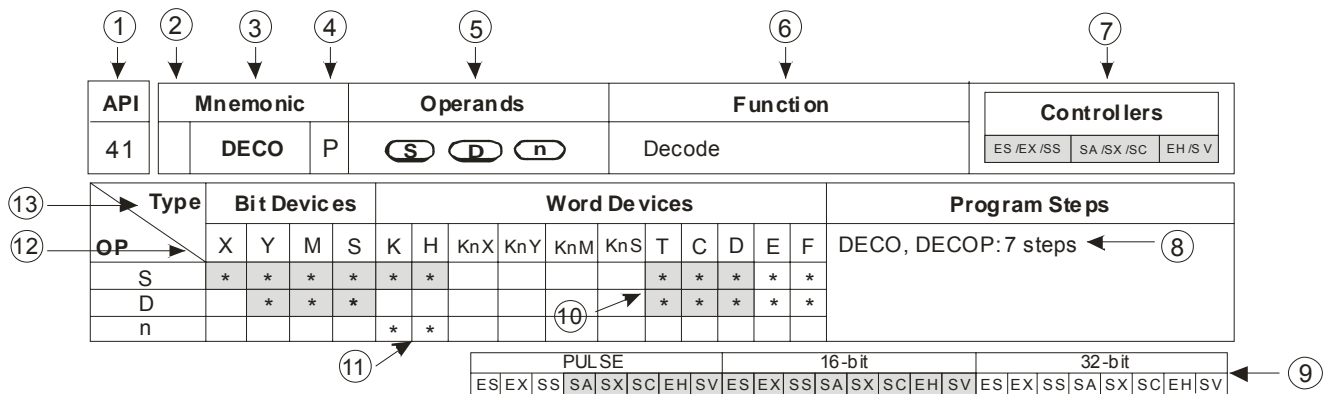
① API No.

② Indication of if there is a 16-bit or 32-bit instruction. If there is a 32-bit instruction, the column will be marked with "D".

③ Mnemonic of the application instruction

④ Indication of if there is a pulse execution type instruction. If there is a pulse instruction, the column will be marked with "P".

⑤ Operands

⑥ Function of the application instruction

⑦ DVP-PLC applicable to the application instruction. ES includes ES/EX/SS, SA includes SA/SX/SC, EH includes EH/EH2/SV.

⑧ Steps occupied by the 16-bit/32-bit/pulse execution instruction

⑨ DVP-PLC applicable to the pulse/16-bit/32-bit instruction

⑩ Column marked with * and in grey refers to E, F index register modification is applicable.

⑪ Column marked with * is the device applicable for the operand

⑫ Device name

⑬ Device type

■ Input of application instruction:

Some application instructions are only composed of the instruction part (mnemonic), e.g. EI, DI, WDT…. Most application instructions are composed of the instruction part and many operands.

The application instructions for DVP-PLC are represented as API 00 ~ API 246. Every application instruction has its own mnemonic. For example, the mnemonic of API 12 is MOV. If you are using the ladder diagram editing software (WPLSoft) to input API 12 into the program, you only have to enter "MOV". If you are using the handheld programming panel (HPP) to input API 12 into the program, you will have to enter the API No. "12".

Different application instructions designate different operands. Take MOV instruction for example:
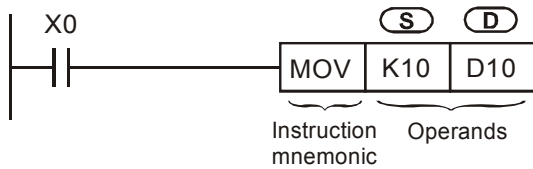
MOV instruction is to move the operand designated in **S** to the operand designated in **D**.

S	Source operand: If there are more than 1 source operands, they will be represented as S_1, S_2, ….
D	Destination operand: If there are more than 1 destination operands, they will be represented as D_1, D_2, ….
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, ….	

■ Length of operand (16-bit instruction or 32-bit instruction)

Depending on the contents in the operand, the length of an operand can be 16-bit or 32-bit. Therefore, a 16-bit instruction is for processing 16-bit operands, and 32-bit instruction is for processing 32-bit operands. The 32-bit instruction is indicated by adding a "D" before the 16-bit instruction.

16-bit MOV instruction
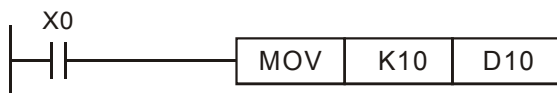
When X0 = On, K10 will be sent to D10.

32-bit DMOV instruction

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

■ Continuous execution instruction and pulse execution instruction

Continuous execution and pulse execution are the two types of execution for an application instruction. Due to that the execution time required will be shorter when the instruction is not executer, the pulse execution instructions are used more to shorten the scan period. Instructions marked with a "P" following the mnemonic are pulse execution instruction. Some instructions are mostly used as pulse execution type, e.g. INC, DEC, the kind of displacement instructions.

Pulse execution instruction

```
  X0
──┤ ├──────────┤ MOVP │ D10 │ D12 │
```

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

```
  X1
──┤ ├──────────┤ MOV │ D10 │ D12 │
```
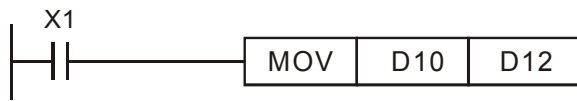
In every scan period when X1 = On, MOV instruction will be executed once.

In the two figures, when X0, X1 = Off, the instruction will not be executed, and the content in operand **D** will remain unchanged.

■ Designation of operands

1. Bit devices X, Y, M, and S can be combined into word device, storing values and data for operaions 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 is usually in 16 bits, i.e. of the length of 1 register D. A designated 32-bit data register refers to 2 consecutive register Ds.
4. If an operand of a 32-bit instruction designates D0, the 32-bit data register composed of (D1, D0) will be occupied. D1 is the higher 16 bits; D0 is the lower 16 bits. The same rule also apply to timer T, 16-bit timers and 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.

■ Format of operand

1. X, Y, M, and S can only On/Off a single point and are defined as bit devices.
2. 16-bit (or 32-bit) devices T, C, D, and registers E, F are defined as word devices.
3. You can place Kn (n = 1 refers to 4 bits. For 16-bit instruction, n = K1 ~ K4; for 32-bit instruction, n = K1 ~ K8) before bit devices X, Y, M and S to make it a word device for performing word-device operations. For example, K1M0 refers to 8 bits, M0 ~ M7.

```
  X0
──┤ ├──────────┤ MOV │ K2M0 │ D10 │
```
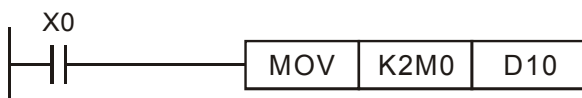
When X0 = On, the contents in M0 ~ M7 will be moved to bit0 ~ 7 in D10 and bit8 ~ 15 will be set to "0".

■ Data processing of word devices combined from bit devices

16-bit instruction	
Designated value: K-32,768 ~ K32,767	
Values for designated K1 ~ K4	
K1 (4 bits)	0 ~ 15
K2 (8 bits)	0 ~ 255
K3 (12 bits)	0 ~ 4,095
K4 (16 bits)	-32,768 ~ +32,767

32-bit instruction	
Designated value: K-2,147,483,648 ~ K2,147,483,647	
Values for designated K1 ~ K8	
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

   a) The flags listed below are for indicating the operational result of the application instruction.

      M1020: zero flag   M1022: carry flag

      M1021: borrow flag   M1029: execution of instruction is completed

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

   b) Example of M1029

   When the contact of DSW (Digital Switch) instruction is On, 4 output points will automatically act in cycle at the frequency of 0.1 second in order to read the set value of the digital switch. If the contact goes Off during the execution, the action will be disable. When it is On again, the disabled action will be re-executed. If you do not wish the action to be disabled, you can take the circuit below as a reference.
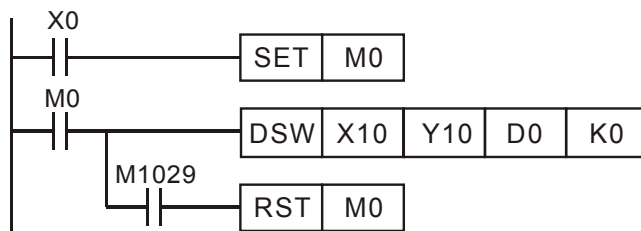
When X0 = On, DSW will be enabled.

When X0 = Off, M0 will be Off only when DSW completes a cycle and M1029 = On.

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.

Device	Explanation
M1067 D1067 D1069	When operational errors occur, M1067 will be On. D1067 displays the error code. D1069 displays the step where the error occurs. Other errors occurring will update the contents in D1067 and D1069. M1067 will be Off when the error is eliminated.
M1068 D1068	When operational errors occur, M1068 will be On. D1068 displays the step where the error occurs. Other errors occurring wil not update the content in D1068. You have to use RST instruction to reset M1068 to Off; otherwise M1068 will keep being On.

3. Flags for expanding functions

Some application instructions can use some special flags to expand their functions or complete special functions. For example, the communication instruction RS can use M1161 to switch between 8-bit and 16-bit transmission mode.

■ Times of using instructions

There are limitation on the times of using some instructions in the program. However, you can use index register modification in the operands to expand the functions of the instruction.

1. Can be used only once in the program:

API 58 PWM (ES series MPU)	API 60 IST (ES/SA/EH series MPU)
API 74 SEGL (ES series MPU)	API 155 DABSR (SC/EH series MPU)

2. Can be used only twice in the program:

API 57 PLSY (ES series MPU)	API 59 PLSR (ES series MPU)
API 74 SEGL (EH series MPU)	API 77 PR (SA/EH series MPU)

3. Can be used only 4 times in the program:

API 169 HOUR (SA series MPU)

4. Can be used only 8 times in the program:

API 64 TTMR (SA series MPU)

5. API 53 DHSCS and API 54 DHSCR together can be used only maximum 4 times in the program (ES series MPU).

6. API 53 DHSCS, API 54 DHSCR, and API 55 DHSZ together can be used only maximum 6 times in the program (SA series MPU).

■ There is no limitation on the times of using the instructions listed below, but there are limitations on the times of executing the same instruction at the same time.

1. Instructions which can be executed only once: API 52 MTR (SA/EH), API 56 SPD (ES/SA/EH), API 69 SORT (SA/EH), API 70 TKY (SA/EH), API 71 HKY (SA/EH), API 72 DSW (SA), API 74 SEGL (SA), API 75 ARWS, API 80 RS (ES/SA/EH), API 100 MODRD (ES/SA/EH), API 101 MODWR (ES/SA/EH), API 102 FWD (ES/SA/EH),

API 103 REV (ES/SA/EH), API 104 STOP (ES/SA/EH), API 105 RDST (ES/SA/EH), API 106 RSTEF (ES/SA/EH), API 150 MODRW (ES/SA/EH), API 151 PWD (EH).

2. Instructions which can be executed only twice: API 57 PLSY (EH), API 58 PWM (SA/EH), API 59 PLSR (SA/EH), API 72 DSW (EH).

3. Instructions which can be executed only 4 times: API 57 PLSY (EH2/SV), API 58 PWM (EH2/SV), API 169 HOUR (EH).

4. Instructions which can be executed only 8 times: API 64 TTMR (EH).

5. In SA series MPU, there is on limitation on the times of using the high-speed output instructions PLSY, PWM and PLSR, bit only one high-speed output instruction will be enabled in every scan.

6. In EH series MPU, there is no limitation on the times of using hardware high-speed counter instructions DHSCS, DHSCR and DHSZ, but when the three instructions are enabled at the same time, DHSCS will occupy 1 memory unit, DHSCR 1 memory unit, and DHSZ 2 memory units. The total memeory units occupied by the three instructions cannot be more than 8 units. If there are more than 8 memory units occupied, the PLC system will execute the instruction that is first scanned and enabled and ignore the rest.

5.3 Handling of Numeric Values

■ Devices only with On/Off status are called bit devices, e.g. X, Y, M and S. Devices used exclusively for storing numeric values are called word devices, e.g. T, C, D, E and F. Bit device plus a specific bit device (place a digit before the bit device in Kn) can be used in the operand of an application instruction in the form of numeric value.

■ n = K1 ~ K4 for a 16-bit value; n = K1 ~ K8 for a 32-bit value. For example, K2M0 refers to a 8-bit value composed of M0 ~ M7.
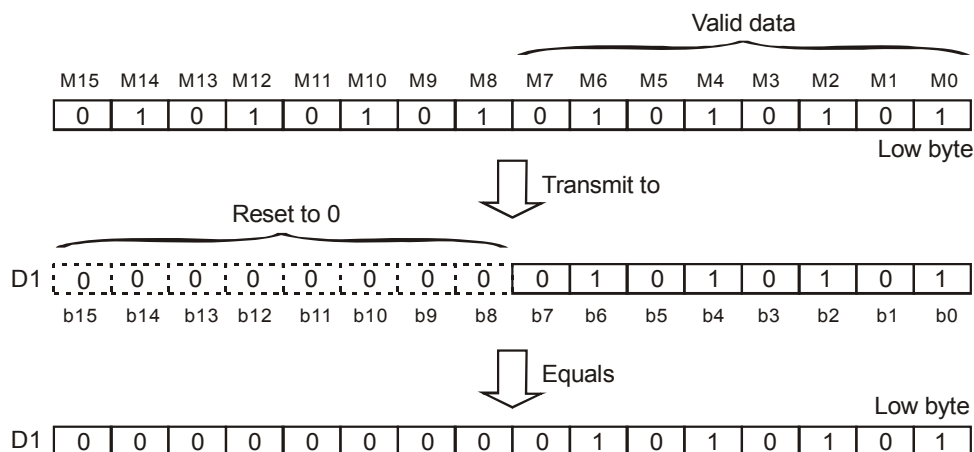
■ K1M0, K2M0, and K3M0 are transmitted to 16-bit registers and the vacant high bits will be filled in "0". The same rule applied to when K1M0, K2M0, K3M0, K4M0, K5M0, K6M0, and K7M0 are transmitted to 32-bit registers and the vacant high bits will be filled in "0".

■ In the 16-bit (or 32-bit) operation, if the contents of the operand are designated as bit devices K1 ~ K3 (or K4 ~ K7), the vacant high bits will be regarded as "0". Therefore, the operation is a positive-value one.

```
      M0
      ┤├─────────[ BIN │ K2X4 │ D0 ]
```
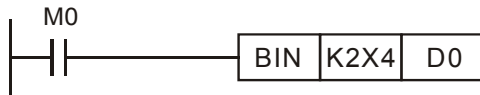
The BCD value composed of X4 ~ X13 will be converted to BIN value and sent to D0.

▪ You can choose any No. for bit devices, but please make the 1s digit of X and Y "0", e.g. X0, X10, X20, …Y0, Y10…, and the 1s digit of M and S "8's multiple" ("0" is still the best choice), e.g. M0, M10, M20….

▪ Designating continuous device No.

Take data register D for example, continuous D refers to D0, D1, D2, D3, D4….

For bit devices with specifically designated digit, continuous No. refers to:

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

Please follow the No. in the table and do not skip No. in case confusion may occur. In addition, if you use K4Y0 in the 32-bit operation, the higher 16 bits will be regarded as "0". For 32-bit data, please use K8Y0.

The operations in DVP-PLC are conducted in BIN integers. When the integer performs division, e.g. 40 ÷ 3 = 13 and the remainder is 1. When the integer performs square root operations, the decimal point will be left out. Use decimal point operation instructions to obtain the decimal point.

Application instructions revelant to decimal point:

API 49 (FLT)	API 110 (D ECMP)	API 111 (D EZCP)	API 112 (D MOVR)
API 116 (D RAD)	API 117 (D DEG)	API 118 (D EBCD)	API 119 (D EBIN)
API 120 (D EADD)	API 121 (D ESUB)	API 122 (D EMUL)	API 123 (D EDIV)
API 124 (D EXP)	API 125 (D LN)	API 126 (D LOG)	API 127 (D ESQR)
API 128 (D POW)	API 129 (INT)	API 130 (D SIN)	API 131 (D COS)
API 132 (D TAN)	API 133 (D ASIN)	API 134 (D ACOS)	API 135 (D ATAN)
API 136 (D SINH)	API 137 (D COSH)	API 138 (D TANH)	API 172 (D ADDR)
API 173 (D SUBR)	API 174 (D MULR)	API 175 (D DIVR)	

Binary Floating Point

DVP-PLC represents floating points in 32 bits, following the IEEE754 standard:
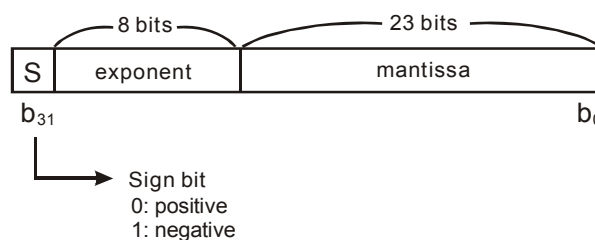
$(-1)^{S} \times 2^{E-B} \times 1.M$, in which B = 127

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

Example 1: Representing "23" in 32-bit floating point

  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: $\because$ E – B = 4 $\rightarrow$ E – 127 = 4 $\therefore$ E = 131 = 10000011_{2}

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

      0 100**0001**1 011**1000**0000**000**0000**0000**$_{2}$ = $41B80000_{16}$

Example 2: Representing "-23.0" in 32-bit floating point

  The steps required are the same as those in Example 1. The only difference is you have to alter the sign bit into "1".

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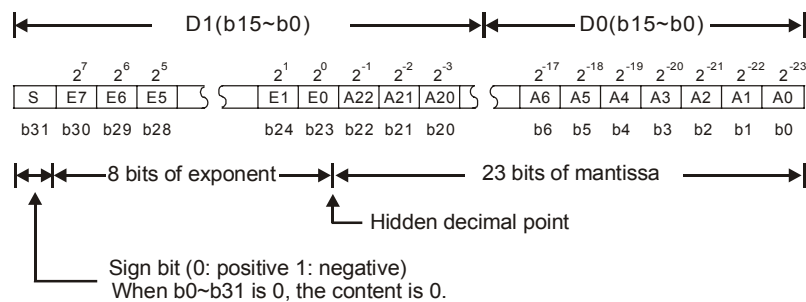
Decimal Floating Point

- Since the binary floating point are not very user-friendly, we can convert it into a decimal floating point for use. Please be noted that the decimal point operation in DVP-PLC is still in binary floating point.

- The decimal floating point is represented by 2 continuous registers. The register of smaller No. is for the constant while the register of bigger No. is for the exponent.

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

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

    Constant D0 = $\pm 1,000 \sim \pm 9,999$

    Exponent D1 = $-41 \sim +35$

The constant 100 does not exist in D0 due to 100 is represented as $1,000 \times 10^{-1}$. The range of decimal floating point is $\pm 1175 \times 10^{-41} \sim \pm 3402 \times 10^{+35}$.

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

D EBCD: Converting binary floating point to decimal floating point

D EBIN: Converting decimal floating point to binary floating point

▪ Zero flag (M1020), carry flag (M1021), carry flag (M1022) and the floating point operation instructions:

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.

5.4   E, F Index Register Modification

The index registers are 16-it registers. There are 2 points of E, F in ES/EX/SS, 8 points E0 ~ E3 and F0 ~ F3 in SA/SX/SC, and 16 points E0 ~ E7 and F0 ~ F7 in EH/EH2/SV series MPU.
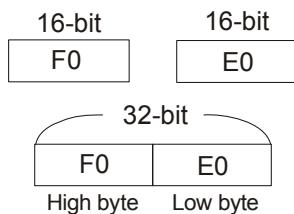
▪ E and F index registers are 16-bit data registers, can be read and written.

▪ 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)

See the diagram in the left hand side. E, F index register modification refers to the content in the operand changes with the contents in E and F.

For example, E0 = 8 and K20E0 represents constant K28 (20 + 8). When the condition is true, constant K28 will be transmitted to register D24.
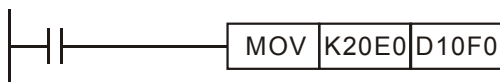
E0 = 8   F0 = 14
20 + 8 = 28   10 + 14 = 24
Transmission   K28 → D24

MOV K20E0 D10F0

Devices modifiable in ES/EX/SS series MPU: P, X, Y, M, S, KnX, KnY, KnM, KnS, T, C, D.

Devices modifiable in SA/SX/SC series MPU: P, X, Y, M, S, KnX, KnY, KnM, KnS, T, C, D

Devices modifiable in EH/EH2/SV series MPU: P, I, X, Y, M, S, K, H, KnX, KnY, KnM, KnS, T, C, D

E and F can modify the devices listed above but cannot modify themselves and Kn. 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 E, F, you have to select a 16-bit register, i.e. you can designate E or F. To modify constant K and H in a 32-bit instruction, you have to select a 32-bit register, i.e. you have to designate E.

When you use the instruction mode in WPLSoft to modify constant K and H, you have to use @, e.g. "MOV K10@E0 D0F0"

5.5   Instruction Index

Sorted by alphabetic order & API No.

Category	API	Mnemonic		P Instruction	Function	Applicable to			Page
		16-bit	32-bit			ES	SA	EH	
A	20	ADD	DADD	✓	Addition	✓	✓	✓	6-32
	46	ANS	-	-	Timed Annunciator Set	-	✓	✓	6-66
	47	ANR	-	✓	Annunciator Reset	-	✓	✓	6-66
	62	ABSD	DABSD	-	Absolute Drum Sequencer	-	✓	✓	7-46
	66	ALT	-	✓	Alternate State	✓	✓	✓	7-54
	75	ARWS	-	-	Arrow Switch	-	✓	✓	7-70
	76	ASC	-	-	ASCII Code Conversion	-	✓	✓	7-72
	82	ASCI	-	✓	Converts Hex to ASCII	✓	✓	✓	7-94
	87	ABS	DABS	✓	Absolute Value	✓	✓	✓	7-106
	133	-	DASIN	✓	Arc Sine	-	✓	✓	8-49
	134	-	DACOS	✓	Arc Cosine	-	✓	✓	8-50
	135	-	DATAN	✓	Arc Tangent	-	✓	✓	8-51
	155	-	DABSR	-	Read the Absolute Position from a Servo Motor	-	✓	✓	9-14
	172	-	DADDR	✓	Floating Point Addition	✓	✓	✓	9-51
	218	AND&	DAND&	-	S_1 & S_2	-	✓	✓	10-8
	219	AND|	DAND|	-	$S_1 \mid S_2$	-	✓	✓	10-8
	220	AND^	DAND^	-	S_1 ^ S_2	-	✓	✓	10-8
	232	AND=	DAND=	-	$S_1 = S_2$	✓	✓	✓	10-11
	233	AND>	DAND>	-	$S_1 > S_2$	✓	✓	✓	10-11
	234	AND<	DAND<	-	$S_1 < S_2$	✓	✓	✓	10-11
	236	AND<>	DAND<>	-	$S_1 \neq S_2$	✓	✓	✓	10-11
	237	AND<=	DAND<=	-	$S_1 \leqq S_2$	✓	✓	✓	10-11
	238	AND>=	DAND>=	-	$S_1 \geqq S_2$	✓	✓	✓	10-11
B	15	BMOV	-	✓	Block Move	✓	✓	✓	6-24
	18	BCD	DBCD	✓	Binary Coded Decimal	✓	✓	✓	6-29
	19	BIN	DBIN	✓	Binary	✓	✓	✓	6-30
	44	BON	DBON	✓	Check Specified Bit Status	✓	✓	✓	6-64
C	00	CJ	-	✓	Conditional Jump	✓	✓	✓	6-1
	01	CALL	-	✓	Call Subroutine	✓	✓	✓	6-5
	10	CMP	DCMP	✓	Compare	✓	✓	✓	6-18
	14	CML	DCML	✓	Compliment	✓	✓	✓	6-23
	84	CCD	-	✓	Check Code	-	✓	✓	7-101
	108	CRC	-	✓	Checksum CRC Mode	-	✓	✓	8-19
	131	-	DCOS	✓	Cosine	✓	✓	✓	8-45
	137	-	DCOSH	✓	Hyperbolic Cosine	-	-	✓	8-53
	146	CVM	-	-	Valve Control (*)	-	-	✓	8-63
	193	-	DCIMR	-	2-Axis Relative Position Arc Interpolation (*)	-	-	✓	9-81
	194	-	DCIMA	-	2-Axis Absolute Position Arc Interpolation (*)	-	-	✓	9-86
	197	-	DCLLM	-	Close Loop Position Control (*)	-	-	✓	9-95
D	05	DI	-	-	Disable Interrupts	✓	✓	✓	6-8
	23	DIV	DDIV	✓	Division	✓	✓	✓	6-37
	25	DEC	DDEC	✓	Decrement	✓	✓	✓	6-40
	41	DECO	-	✓	Decode	✓	✓	✓	6-59
	72	DSW	-	-	Digital Switch	-	✓	✓	7-64
	117	-	DDEG	✓	Radian → Angle	-	✓	✓	8-27
	143	DELAY	-	✓	Delay Instruction	-	✓	✓	8-55

Category	API	Mnemonic 16-bit	Mnemonic 32-bit	P Instruction	Function	ES	SA	EH	Page
D	158	DRVI	DDRVI	-	Drive to Increment	-	-	✓	9-24
	159	DRVA	DDRVA	-	Drive to Absolute	-	-	✓	9-30
	175	-	DDIVR	✓	Floating Point Division	✓	✓	✓	9-57
E	04	EI	-	-	Enable Interrupts	✓	✓	✓	6-8
	42	ENCO	-	✓	Encode	✓	✓	✓	6-61
	110	-	DECMP	✓	Floating Point Compare	✓	✓	✓	8-23
	111	-	DEZCP	✓	Floating Point Zone Compare	✓	✓	✓	8-24
	118	-	DEBCD	✓	Float to Scientific Conversion	✓	✓	✓	8-28
	119	-	DEBIN	✓	Scientific to Float Conversion	✓	✓	✓	8-29
	120	-	DEADD	✓	Floating Point Addition	✓	✓	✓	8-31
	121	-	DESUB	✓	Floating Point Subtraction	✓	✓	✓	8-32
	122	-	DEMUL	✓	Floating Point Multiplication	✓	✓	✓	8-33
	123	-	DEDIV	✓	Floating Point Division	✓	✓	✓	8-34
	124	-	DEXP	✓	Exponent of Binary Floating Point	✓	✓	✓	8-35
	127	-	DESQR	✓	Floating Point Square Root	✓	✓	✓	8-39
F	06	FEND	-	-	The End of the Main Program (First End)	✓	✓	✓	6-12
	08	FOR	-	-	Start of a FOR-NEXT Loop	✓	✓	✓	6-15
	16	FMOV	DFMOV	✓	Fill Move	✓	✓	✓	6-26
	49	FLT	DFLT	✓	Floating Point	✓	✓	✓	6-70
	78	FROM	DFROM	✓	Read CR Data in Special Modules	✓	✓	✓	7-75
	102	FWD	-	-	Forward Running of VFD-A	✓	✓	✓	8-11
	145	FTC	-	-	Fuzzy Temperature Control	-	✓	✓	8-58
G	144	GPWM	-	-	General PWM Output	-	✓	✓	8-57
	170	GRY	DGRY	✓	BIN → Gray Code	-	✓	✓	9-49
	171	GBIN	DGBIN	✓	Gray Code → BIN	-	✓	✓	9-50
H	53	-	DHSCS	-	High Speed Counter Set	✓	✓	✓	7-5
	54	-	DHSCR	-	High Speed Counter Reset	✓	✓	✓	7-15
	55	-	DHSZ	-	High Speed Zone Compare	-	✓	✓	7-17
	71	HKY	DHKY	-	Hexadecimal Key Input	-	✓	✓	7-61
	83	HEX	-	✓	Converts ASCII to Hex	✓	✓	✓	7-98
	169	HOUR	DHOUR	-	Hour Meter	-	✓	✓	9-47
	196	HST	-	✓	High Speed Timer	-	-	✓	9-93
I	03	IRET	-	-	Interrupt Return	✓	✓	✓	6-8
	24	INC	DINC	✓	Increment	✓	✓	✓	6-39
	60	IST	-	-	Initial State	✓	✓	✓	7-39
	63	INCD	-	-	Increment Drum Sequencer	-	✓	✓	7-48
	129	INT	DINT	✓	Float to Integer	✓	✓	✓	8-42
L	107	LRC	-	✓	Checksum LRC Mode	✓	✓	✓	8-17
	125	-	DLN	✓	Natural Logarithm of Binary Floating Point	✓	✓	✓	8-36
	126	-	DLOG	✓	Logarithm of Binary Floating Point	✓	✓	✓	8-37
	215	LD&	DLD&	-	S_1 & S_2	-	✓	✓	10-7
	216	LD|	DLD|	-	S_1 | S_2	-	✓	✓	10-7
	217	LD^	DLD^	-	S_1 ^ S_2	-	✓	✓	10-7
	224	LD=	DLD=	-	$S_1 = S_2$	✓	✓	✓	10-10
	225	LD>	DLD>	-	$S_1 > S_2$	✓	✓	✓	10-10
	226	LD<	DLD<	-	$S_1 < S_2$	✓	✓	✓	10-10
	228	LD<>	DLD<>	-	$S_1 \neq S_2$	✓	✓	✓	10-10

Category	API	Mnemonic		P Instruction	Function	Applicable to			Page
		16-bit	32-bit			ES	SA	EH	
L	229	LD<=	DLD<=	-	$S_1 \leqq S_2$	✓	✓	✓	10-10
	230	LD>=	DLD>=	-	$S_1 \geqq S_2$	✓	✓	✓	10-10
M	12	MOV	DMOV	✓	Move	✓	✓	✓	6-20
	22	MUL	DMUL	✓	Multiplication	✓	✓	✓	6-35
	45	MEAN	DMEAN	✓	Mean	✓	✓	✓	6-65
	52	MTR	-	-	Input Matrix	-	✓	✓	7-3
	100	MODRD	-	-	Read Modbus Data	✓	✓	✓	8-1
	101	MODWR	-	-	Write Modbus Data	✓	✓	✓	8-6
	112	-	DMOVR	✓	Move Floating Point Data	✓	✓	✓	8-25
	148	MEMR	DMEMR	✓	Read File Register	-	✓	✓	8-67
	149	MEMW	DMEMW	✓	Write File Register	-	✓	✓	8-68
	150	MODRW	-	-	Read/Write MODBUS Data	✓	✓	✓	9-1
	174	-	DMULR	✓	Floating Point Multiplication	✓	✓	✓	9-55
	180	MAND	-	✓	Matrix 'AND' Operation	-	✓	✓	9-59
	181	MOR	-	✓	Matrix 'OR' Operation	-	✓	✓	9-61
	182	MXOR	-	✓	Matrix 'XOR' Operation	-	✓	✓	9-62
	183	MXNR	-	✓	Matrix 'NOR' Operation	-	✓	✓	9-63
	184	MINV	-	✓	Matrix Inverse Operation	-	✓	✓	9-64
	185	MCMP	-	✓	Matrix Compare	-	✓	✓	9-65
	186	MBRD	-	✓	Read Matrix Bit	-	✓	✓	9-67
	187	MBWR	-	✓	Write Matrix Bit	-	✓	✓	9-69
	188	MBS	-	✓	Matrix Bit Displacement	-	✓	✓	9-71
	189	MBR	-	✓	Matrix Bit Rotation	-	✓	✓	9-73
	190	MBC	-	✓	Matrix Bit Status Counting	-	✓	✓	9-75
N	09	NEXT	-	-	End of a FOR-NEXT Loop	✓	✓	✓	6-15
	29	NEG	DNEG	✓	2's Complement (Negative)	✓	✓	✓	6-44
O	221	OR&	DOR&	-	S_1 & S_2	-	✓	✓	10-9
	222	OR|	DOR|	-	$S_1 \mid S_2$	-	✓	✓	10-9
	223	OR^	DOR^	-	$S_1 ^\wedge S_2$	-	✓	✓	10-9
	240	OR=	DOR=	-	$S_1 = S_2$	✓	✓	✓	10-12
	241	OR>	DOR>	-	$S_1 > S_2$	✓	✓	✓	10-12
	242	OR<	DOR<	-	$S_1 < S_2$	✓	✓	✓	10-12
	244	OR<>	DOR<>	-	$S_1 \neq S_2$	✓	✓	✓	10-12
	245	OR<=	DOR<=	-	$S_1 \leqq S_2$	✓	✓	✓	10-12
	246	OR>=	DOR>=	-	$S_1 \geqq S_2$	✓	✓	✓	10-12
P	57	PLSY	DPLSY	-	Pulse Y Output	✓	✓	✓	7-26
	58	PWM	-	-	Pulse Width Modulation	✓	✓	✓	7-32
	59	PLSR	DPLSR	-	Pulse Ramp	✓	✓	✓	7-35
	77	PR	-	-	Print (ASCII Code Output)	-	✓	✓	7-73
	81	PRUN	DPRUN	✓	Parallel Run	-	✓	✓	7-93
	88	PID	DPID	-	PID Control Loop	✓	✓	✓	7-107
	128	-	DPOW	✓	Floating Point Power Operation	✓	✓	✓	8-40
	151	PWD	-	-	Detection of Input Pulse Width	-	-	✓	9-10
	157	PLSV	DPLSV	-	Adjustable Speed Pulse Output	-	-	✓	9-23
	191	-	DPPMR	-	2-Axis Relative Point to Point Motion (*)	-	-	✓	9-76
	192	-	DPPMA	-	2-Axis Absolute Point to Point Motion (*)	-	-	✓	9-79
	195	-	DPTPO	-	Single-Axis Pulse Output by Table (*)	-	-	✓	9-91
R	30	ROR	DROR	✓	Rotation Right	✓	✓	✓	6-46
	31	ROL	DROL	✓	Rotation Left	✓	✓	✓	6-47

Category	API	Mnemonic		P Instruction	Function	Applicable to			Page
		16-bit	32-bit			ES	SA	EH	
R	32	RCR	DRCR	✓	Rotation Right with Carry	✓	✓	✓	6-48
	33	RCL	DRCL	✓	Rotation Left with Carry	✓	✓	✓	6-49
	50	REF	-	✓	Refresh	✓	✓	✓	7-1
	51	REFF	-	✓	Refresh and Filter Adjust	-	✓	✓	7-2
	67	RAMP	-	-	Ramp Variable Value	-	✓	✓	7-55
	80	RS	-	-	Serial Communication Instruction	✓	✓	✓	7-80
	103	REV	-	-	Reverse Running of VFD-A	✓	✓	✓	8-11
	105	RDST	-	-	Read VFD-A Status	✓	✓	✓	8-14
	106	RSTEF	-	-	Reset Abnormal VFD-A	✓	✓	✓	8-16
	116	-	DRAD	✓	Angle → Radian	-	✓	✓	8-26
	152	RTMU	-	-	Start of the Measurement of Execution Time of I Interruption	-	-	✓	9-11
	153	RTMD	-	-	End of the Measurement of the Execution Time of I Interruption	-	-	✓	9-11
	154	RAND	✓	-	Random Number	-	✓	✓	9-13
S	02	SRET	-	-	Subroutine Return	✓	✓	✓	6-5
	13	SMOV	-	✓	Shift Move	-	✓	✓	6-21
	21	SUB	DSUB	✓	Subtraction	✓	✓	✓	6-34
	34	SFTR	-	✓	Bit Shift Right	✓	✓	✓	6-50
	35	SFTL	-	✓	Bit Shift Left	✓	✓	✓	6-51
	38	SFWR	-	✓	Shift Register Write	-	✓	✓	6-55
	39	SFRD	-	✓	Shift Register Read	-	✓	✓	6-56
	43	SUM	DSUM	✓	Sum of Active Bits	✓	✓	✓	6-63
	48	SQR	DSQR	✓	Square Root	✓	✓	✓	6-72
	56	SPD	-	-	Speed Detection	✓	✓	✓	7-24
	61	SER	DSER	✓	Search a Data Stack	-	✓	✓	7-45
	65	STMR	-	-	Special Timer	-	✓	✓	7-52
	69	SORT	-	-	Sort Tabulated Data	-	✓	✓	7-57
	73	SEGD	-	✓	Seven Segment Decoder	✓	✓	✓	7-66
	74	SEGL	-	-	Seven Segment with Latch	✓	✓	✓	7-67
	104	STOP	-	-	Stop VFD-A	✓	✓	✓	8-11
	109	SWRD	-	✓	Read Digital Switch	-	-	✓	8-22
	130	-	DSIN	✓	Sine	✓	✓	✓	8-43
	136	-	DSINH	✓	Hyperbolic Sine	-	-	✓	8-52
	147	SWAP	DSWAP	✓	Byte Swap	✓	✓	✓	8-66
	173	-	DSUBR	✓	Floating Point Subtraction	✓	✓	✓	9-53
	202	SCAL	-	✓	Proportional Value Calculation	✓	✓	✓	10-1
	203	SCLP	-	✓	Parameter Proportional Value Calculation	✓	✓	✓	10-3
T	64	TTMR	-	-	Teaching Timer	-	✓	✓	7-50
	70	TKY	DTKY	-	Ten Key Input	-	✓	✓	7-59
	79	TO	DTO	✓	Write CR Data into Special Modules	✓	✓	✓	7-76
	132	-	DTAN	✓	Tangent	✓	✓	✓	8-47
	138	-	DTANH	✓	Hyperbolic Tangent	-	-	✓	8-54
	160	TCMP	-	✓	Time Compare	-	✓	✓	9-39
	161	TZCP	-	✓	Time Zone Compare	-	✓	✓	9-40
	162	TADD	-	✓	Time Addition	-	✓	✓	9-41
	163	TSUB	-	✓	Time Subtraction	-	✓	✓	9-42
	166	TRD	-	✓	Time Read	-	✓	✓	9-43

Category	API	Mnemonic		P Instruction	Function	Applicable to			Page
		16-bit	32-bit			ES	SA	EH	
T	167	TWR	-	✓	Time Write	-	✓	✓	9-45
V	85	VRRD	-	✓	Volume Read	-	✓	✓	7-103
	86	VRSC	-	✓	Volume Scale	-	✓	✓	7-105
W	07	WDT	-	✓	Watchdog Timer Refresh	✓	✓	✓	6-14
	26	WAND	DAND	✓	Logical Word AND	✓	✓	✓	6-41
	27	WOR	DOR	✓	Logical Word OR	✓	✓	✓	6-42
	28	WXOR	DXOR	✓	Logical Exclusive OR	✓	✓	✓	6-43
	36	WSFR	-	✓	Word Shift Right	-	✓	✓	6-52
	37	WSFL	-	✓	Word Shift Left	-	✓	✓	6-54
X	17	XCH	DXCH	✓	Exchange	✓	✓	✓	6-27
Z	11	ZCP	DZCP	✓	Zone Compare	✓	✓	✓	6-19
	40	ZRST	-	✓	Zero Reset	✓	✓	✓	6-57
	156	ZRN	DZRN	-	Zero Return	-	-	✓	9-19

*Instructions marked with * are available in EH2/SV series MPU.*

Note:

1. For applicable models, ES includes ES/EX/SS; SA includes SA/SX/SC; EH includes EH/EH2/SV.

2. ES/EX/SS series MPU does not support pulse execution type instructions (P instruction).

API	Mnemonic		Operands	Function	Controllers
00	CJ	P	Ⓢ	Conditional Jump	ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: The destination pointer of conditional jump

Explanations:

1. Operand **S** can designate P

2. P can be modified by index register E, F

3. In ES/EX/SS series models: Operand S can designate P0 ~ P63

4. In SA/SX/SC/EH/EH2/SV series models: Operand S can designate P0 ~ P255

5. When the user does not wish 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.

6. 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.

7. CJ instruction can designate the same pointer P repeatedly. However, CJ and CALL cannot designate the same pointer P; otherwise an error will occur.

8. Actions of all devices while conditional jumping is being executed.

   a) Y, M and S remain their previous status before the conditional jump takes place.

   b) Timer 10ms and 100ms 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) The ordinary counters stop executing.

   f) If the "reset instruction" of the timer is executed before the conditional jump, the device will still be in the reset status while conditional jumping is being executed.

   g) Ordinary application instructions are not executed.

   h) The application instructions that are being executed, i.e. API 53 DHSCS, API 54 DHSCR, API 55 DHSZ, API 56 SPD, API 57 PLSY, API 58 PWM, API 59 PLSR, API 157 PLSV, API 158 DRVI, API 159 DRVA, continue being executed.

Program Example 1:

1. When X0 = On, the program automatically jumps from address 0 to N (the designated label P1) and keeps its execution. The addresses between 0 and N will not be executed.

2. When X0 = Off, as an ordinary program, the program keeps on executing from address 0. CJ instruction will not be executed at this time.
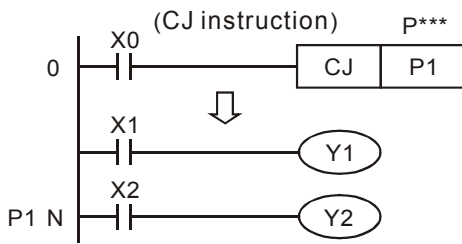
Program Example 2:

1. CJ instruction can be used in the following 5 conditions between MC and MCR instructions.

  a) Without MC ~ MCR.

  b) From without MC to within MC. Valid in the loop P1 as shown in the figure below.

  c) In the same level N, inside of MC~MCR.

  d) From within MC to without MCR.

  e) Jumping from this MC ~ MCR to another MC ~ MCR[1].

2. Actions in ES/EX/SS series models V4.7 (and below): When CJ instruction is used between MC and MCR, it can only be applied without MC ~ MCR or in the same N layer of MC ~ MCR. Jumping from this MC ~ MCR to another MC ~ MCR will result in errors, i.e. a) and c) as stated above can ensure correct actions; others will cause errors.

3. When MC instruction is executed, PLC will push the status of the switch contact into the self-defined stack in PLC. The stack will be controlled by the PLC, and the user cannot change it. When MCR instruction is executed, PLC will obtain the previous status of the switch contact from the top layer of the stack. Under the conditions as stated in b), d) and e), the times of pushing-in and obtaining stack may be different. In this case, the maximum stack available to be pushed in is 8 and the obtaining of stacks cannot resume once the stack becomes empty. Thus, when using CALL or CJ instructions, the user has to be aware of the pushing-in and obtaining of stacks.

---
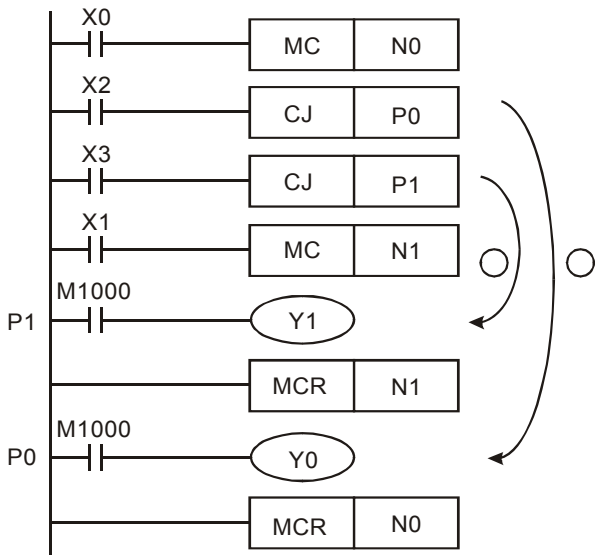
[1] This function is only available in ES/EX/SS series models V4.9 (and above) and SA/SX/SC/EH/EH2/SV series models.

Program Example 3:

1. The states of each device

Device	Contact state before CJ is executed	Contact state when CJ is being executed	Output coil state when CJ is being executed
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 ES/SA/EH	M4 Off	M4 Off→On	Timer T0 is not enabled.
	M4 On	M4 On→Off	Timer T0 immediately stops and is latched. M0 On→Off, T0 is reset as 0.
1ms, 10ms, 100ms Timer [*2] (accumulative) SA/EH	M6 Off	M6 Off→On	Timer T240 is not enabled.
	M6 On	M6 On→Off	Once the timer function is enabled and when met with CJ instruction, all accumulative timers will stop timing and stay latched. M0 On→Off. T240 remains unchanged.
C0 ~ C234 [*3]	M7, M10 Off	M10 On/Off trigger	Counter does not count.
	M7 Off, M10 On/Off trigger	M10 On/Off trigger	Counter C0 stops counting and stays latched. After M0 goes Off, C0 resumes its counting.
Application instruction	M11 Off	M11 Off→On	Application instructions are not executed.
	M11 On	M11 On→Off	The skipped application instructions are not executed, but API 53 ~ 59, API 157 ~ 159 keep being executed.

*1: Y1 is a dual output. When M0 is Off, M1 will control Y1. When M0 is On, M12 will control Y1.

*2: When the timers (T192 ~ T199, applicable in SA/EH series MPU) used by a subroutine re driven and encounter the execution of CJ instruction, the timing will resume. After the timing target is reached, the output contact of the timer will be On.

*3: When the high-speed counters (C235 ~ C255) are driven and encounter the execution of CJ instruction, the counting will resume, as well as the action of the output points.

2. Y1 is a dual output. When M0 = Off, Y1 is controlled by M1. When M0 = On, Y1 is controlled by M12.

```
      M0
      ─┤├──────────┤ CJ  │ P0  │
      M1
      ─┤├──────────( Y1 )
      M2
      ─┤├──────────( M20 )
      M17
      ─┤├──────────┤ CJ  │ P0  │
      M3
      ─┤├──────────( S1 )
      M4
      ─┤├──────────┤ TMR │ T0   │ K10  │
      M5
      ─┤├──────────┤ RST │ T127 │
      M6
      ─┤├──────────┤ TMR │ T127 │ K1000 │
      M7
      ─┤├──────────┤ RST │ C0  │
      M10
      ─┤├──────────┤ CNT │ C0  │ K20  │
      M11
      ─┤├──────────┤ MOV │ K3  │ D0   │
      M0
 P0   ─┤/├─────────┤ CJ  │ P63 │
      M12
      ─┤├──────────( Y1 )
      M13
 P63  ─┤├──────────┤ RST │ T127 │
                   ├ RST │ C0  │
                   └ RST │ D0  │
      ──────────────┤ END │
```
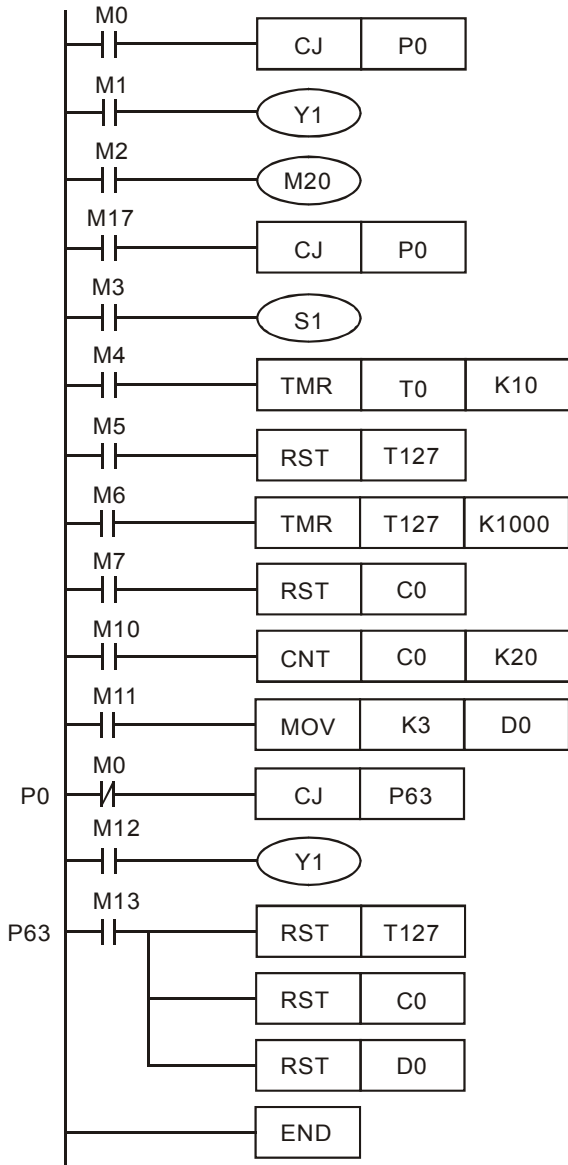
API	Mnemonic		Operands	Function	Controllers
01	CALL	P	Ⓢ	Call Subroutine	ES/EX/SS SA/SX/SC EH/SV

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

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: The pointer of call subroutine.

Explanations:

1. Operand S can designate P.

2. P can be modified by index register E, F.

3. In ES/EX/SS series models: Operand S can designate P0 ~ P63.

4. In SA/SX/SC/EH/EH2/SV series models: Operand S can designate P0 ~ P255.

5. Edit the subroutine designated by the pointer after FEND instruction.

6. The number of pointer P, when used by CALL, cannot be the same as the number designated by CJ instruction.

7. If only CALL instruction is in use, it can call subroutines of the same pointer number with no limit on times.

8. Subroutine can be nested for 5 levels including the initial CALL instruction. (If entering the sixth level, the subroutine won't be executed.)

API	Mnemonic	Function	Controllers
02	SRET	Subroutine Return	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A	Automatically returns to the step immediately following the CALL instruction which activated the subroutine	SRET: 1 steps

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. No operand. No contact to drive the instruction is required.

2. The subroutine will return to main program by SRET after the termination of subroutine and execute the sequence program located at the next step to the CALL instruction.

Program Example 1:

When X0 = On, CALL instruction is executed and the program jumps to the subroutine designated by P2. When SRET instruction is executed, the program returns to address 24 and continues its execution.
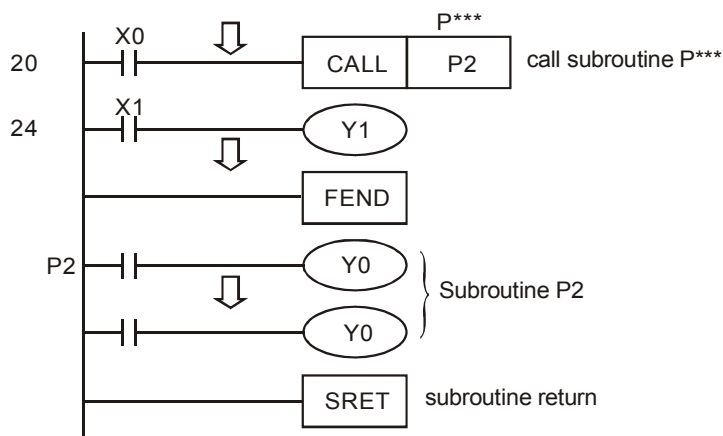
Program Example 2:

1. When X10 goes from Off to On, its rising-edge trigger executes CALL P10 instruction and the program jumps to the subroutine designated by P10.

2. When X11 is On, CALL P11 is executed and the program jumps to the subroutine designated by P11.

3. When X12 is On, CALL P12 is executed and the program jumps to the subroutine designated by P12.

4. When X13 is On, CALL P13 is executed and the program jumps to the subroutine designated by P13.

5. When X14 is On, CALL P14 is executed and the program jumps to the subroutine designated by P14. When SRET is executed, the program returns to the previous P※ subroutine and continues its execution.

6. After SRET instruction is executed in P10 subroutine, returning to the main program.

```
     X0
     ┤├──────┌─────────────┐              P12   X2
             │ INC  │  D0  │                    ┤/├──────┌──────────────┐
             └─────────────┘                            │ INC  │  D30  │
             ┌─────────────┐                            └──────────────┘
             │    Y0       │                            ┌──────────────┐
             └─────────────┘                            │    Y10      │
     X10                              Main              └──────────────┘
     ┤↑├──────┌─────────────┐        Program    X13
             │ CALL │  P10 │                     ┤├──────┌──────────────┐    subroutine
             └─────────────┘                            │ CALL │  P13  │
     X0                                                 └──────────────┘
     ┤├──────┌─────────────┐                    X2
             │ INC  │  D1  │                     ┤/├──────┌──────────────┐
             └─────────────┘                            │ INC  │  D31  │
             ┌─────────────┐                            └──────────────┘
             │    Y1       │                            ┌──────────────┐
             └─────────────┘                            │    Y11      │
             ┌─────────────┐                            └──────────────┘
             │   FEND      │                            ┌──────────────┐
             └─────────────┘                            │   SRET      │
                                                        └──────────────┘
          X2
P10       ┤/├──────┌─────────────┐         P13   X2
                  │ INC  │  D10 │                ┤/├──────┌──────────────┐
                  └─────────────┘                        │ INC  │  D40  │
                  ┌─────────────┐                        └──────────────┘
                  │    Y4       │                        ┌──────────────┐
                  └─────────────┘                        │    Y12      │
          X11                                            └──────────────┘
          ┤├──────┌─────────────┐         subroutine   X14
                 │ CALL │  P11 │                        ┤├──────┌──────────────┐   subroutine
                 └─────────────┘                               │ CALL │  P14  │
          X2                                                   └──────────────┘
          ┤/├──────┌─────────────┐                     X2
                  │ INC  │  D11 │                       ┤/├──────┌──────────────┐
                  └─────────────┘                              │ INC  │  D41  │
                  ┌─────────────┐                              └──────────────┘
                  │    Y5       │                              ┌──────────────┐
                  └─────────────┘                              │    Y13      │
                  ┌─────────────┐                              └──────────────┘
                  │   SRET      │                              ┌──────────────┐
                  └─────────────┘                              │   SRET      │
          X2                                                   └──────────────┘
P11       ┤/├──────┌─────────────┐         P14   X2
                  │ INC  │  D20 │                ┤/├──────┌──────────────┐
                  └─────────────┘                        │ INC  │  D50  │
                  ┌─────────────┐                        └──────────────┘
                  │    Y6       │                        ┌──────────────┐
                  └─────────────┘                        │    Y14      │
          X12                                            └──────────────┘
          ┤├──────┌─────────────┐                        ┌──────────────┐   subroutine
                 │ CALL │  P12 │                         │   SRET      │
                 └─────────────┘         subroutine      └──────────────┘
          X2                                             ┌──────────────┐
          ┤├──────┌─────────────┐                        │   END       │
                 │ INC  │  D21 │                         └──────────────┘
                 └─────────────┘
                 ┌─────────────┐
                 │    Y7       │
                 └─────────────┘
                 ┌─────────────┐
                 │   SRET      │
                 └─────────────┘
```
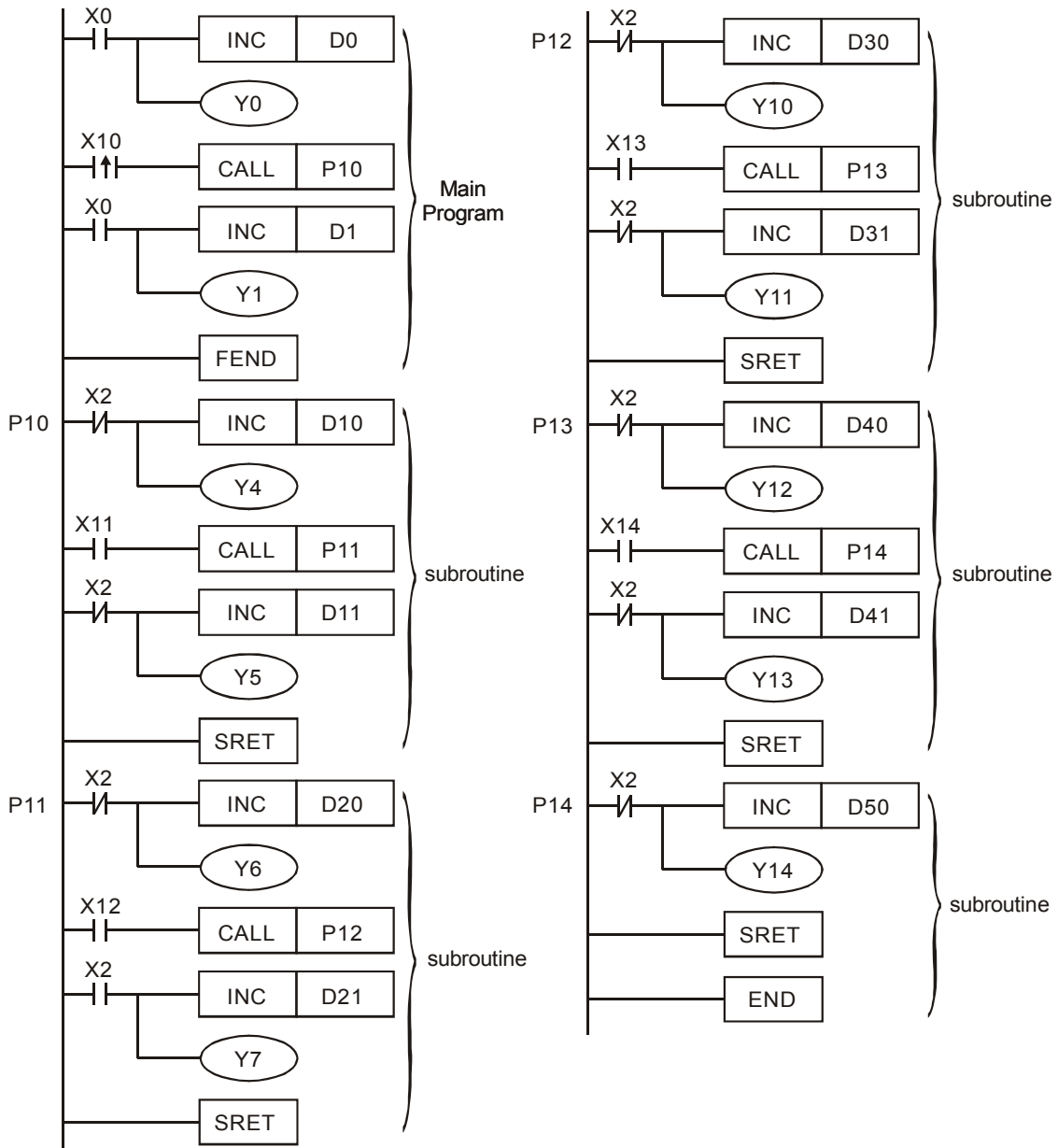
API	Mnemonic	Function	Controllers
03	IRET	Interrupt Return	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A	IRET ends the processing of an interruption subroutine and returns to the execution of the main program.	IRET: 1 steps

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. No operand. No contact to drive the instruction is required.

2. Interruption return refers to interrupt the subroutine.

3. After the interruption is over, returning to the main program from IRET to execute the next instruction where the program was interrupted.

API	Mnemonic	Function	Controllers
04	EI	Enable Interrupts	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A	See more details of the explanation on this instruction in DI (Disable Interruption) instruction.	EI: 1 steps

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. No operand. No contact to drive the instruction is required.

2. The pulse width of the interruption signal should be >200us.

3. See DI instruction for the range of the No. of I for all models.

4. See DI instruction for more details about M1050 ~ M1059, M1280 ~ M1299.

API	Mnemonic	Function	Controllers
05	DI	Disable Interrupts	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A	When the special auxiliary relay M1050 ~ M1059, M1280 ~ M1299 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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. No operand. No contact to drive the instruction is required.

2. EI instruction allows interrupting subroutine in the program, e.g. external interruption, timed interruption, and high-speed counter interruption.

3. In the program, using interruption subroutine between EI and DI instruction is allowed. However, you can choose not to use DI instruction if there is no interruption-disabling section in the program.

4. When M1050 ~ M1059 are the special auxiliary relays to drive disabling interruption in ES/SA, or M1280 ~ M1299 are the special auxiliary relays to drive disabling interruption in EH/EH2/SV, the corresponding interruptions will not be executed even in the area allowed for interruptions.

5. Pointer for interruption (I) must be placed after FEND instruction.

6. Other interruptions are not allowed during the execution of interruption subroutine.

7. When many interruptions occur, the priority is given to the firstly executed interruption. If several interruptions occur simultaneously, the priority is given to the interruption with the smaller pointer No.

8. The interruption request occurring between DI and EI instructions that cannot be executed immediately will be memorized and will be executed in the area allowed for interruption.

9. The time interruptions in ES/SA will not be memorized.

10. When using the interruption pointer, DO NOT repeatedly use the high-speed counter driven by the same X input contact.

11. When immediate I/O is required during the interruption, write REF instruction in the program to update the status of I/O.

Program Example:

During the operation of PLC, when the program scans to the area between EI and DI instructions and X1 = Off→On or X2 = Off→On, interruption subroutine A or B will be executed. When the subroutine executes to IRET, the program will return to the main program and resumes its execution.
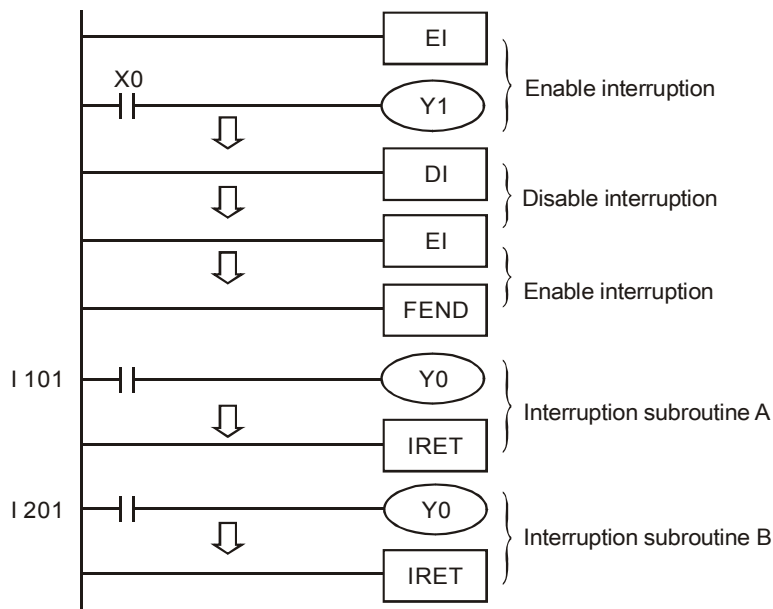
Remarks:

1. No. of interruption pointer I in ES/EX/SS:

a) External interruptions: (I001, X0), (I101, X1), (I201, X2), (I301, X3) 4 points[2].

b) Time interruptions: I6☐☐, 1 point (☐☐ = 10 ~ 99, time base = 1ms) (support V5.7 and above)

---

[2] Input points occupied by external interruptions cannot be used for inputs of high-speed counters; otherwise grammar check errors may occur when the program is written in PLC.

c) Communication interruption for receiving specific words (I150) (support V5.7 and above)

2. No. of interruption pointer I in SA/SX/SC:

   a) External interruptions: (I001, X0), (I101, X1), (I201, X2), (I301, X3), (I401, X4), (I501, X5) 6 points.

   b) Time interruptions: I6☐☐, I7☐☐ 2 points. (☐☐ = 1 ~ 99ms, time base = 1ms)

   c) High-speed counter interruptions: I010, I020, I030, I040 4 points. (used with API 53 DHSCS instruction to generate interruption signals)

   d) Communication interruption for receiving specific words .(I150)

   e) The order for execution of interruption pointer I: high-speed counter interruption, external interruption, time interruption and communication interruption for receiving specific words.

   f) Among the following 6 interruption No., (I001, I010), (I101, I020), (I201, I030), (I301, I040), (I401, I050), (I501, I060), the program allows the user to use only one of the two numbers in a pair. If the user uses the two numbers in the pair, grammar check errors may occur when the program is written into PLC.

3. No. of interruption pointer I in EH/EH2/SV:

   a) External interruptions: (I00☐, X0), (I10☐, X1), (I20☐, X2), (I30☐, X3), (I40☐, X4), (I50☐, X5) 6 points. (☐ = 0 designates interruption in falling-edge, ☐ = 1 designates interruption in rising-edge)

   b) Time interruptions: I6☐☐, I7☐☐, 2 points. (☐☐ = 1~99ms, time base = 1ms)

      I8☐☐ 1 point. (☐☐ = 1 ~ 99ms, time base = 0.1ms)

   c) High-speed counter interruptions: I010, I020, I030, I040, 1050, 1060 6 points. (used with API 53 DHSCS instruction to generate interruption signals)

   d) When pulse output interruptions I110, I120 (triggered when pulse output is finished), I130, I140 (triggered when the first pulse output starts) are executed, the currently executed program is interrupted and jumps to the designated interruption subroutine.

   e) Communication interruption: I150, I160, I170

   f) Frequency measurement card interruption: I180

   g) The order for execution of interruption pointer I: external interruption, time interruption, high-speed counter interruption, pulse interruption, communication interruption and frequency measurement card interruption.

4. "Disable interruption" flags in ES/EX/SS:

Flag	Function
M1050	Disable external interruption I001
M1051	Disable external interruption I101
M1052	Disable external interruption I201
M1053	Disable external interruption I301
M1056	Disable time interruption I6☐☐

5. "Disable interruption" flags in SA/SX/SC:

Flag	Function
M1050	Disable external interruption I001
M1051	Disable external interruption I101
M1052	Disable external interruption I201
M1053	Disable external interruption I301
M1054	Disable external interruption I401
M1055	Disable external interruption I501

M1056	Disable time interruption I6□□
M1057	Disable time interruption I7□□
M1059	Disable high-speed counter interruption I010 ~ I060

6. "Disable interruption" flags in EH/EH2/SV:

Flag	Function
M1280	Disable external interruption I00□
M1281	Disable external interruption I10□
M1282	Disable external interruption I20□
M1283	Disable external interruption I30□
M1284	Disable external interruption I40□
M1285	Disable external interruption I50□
M1286	Disable time interruption I6□□
M1287	Disable time interruption I7□□
M1288	Disable time interruption I8□□
M1289	Disable high-speed counter interruption I010
M1290	Disable high-speed counter interruption I020
M1291	Disable high-speed counter interruption I030
M1292	Disable high-speed counter interruption I040
M1293	Disable high-speed counter interruption I050
M1294	Disable high-speed counter interruption I060
M1295	Disable pulse output interruption I110
M1296	Disable pulse output interruption I120
M1297	Disable pulse output interruption I130
M1298	Disable pulse output interruption I140
M1299	Disable communication interruption I150
M1300	Disable communication interruption I160
M1301	Disable communication interruption I170
M1302	Disable frequency measurement card interruption I180
M1340	Generate interruption I110 after CH0 pulse is sent
M1341	Generate interruption I120 after CH1 pulse is sent
M1342	Generate interruption I130 when CH0 pulse is being sent
M1343	Generate interruption I140 when CH1 pulse is being sent

API	Mnemonic	Function	Controllers
06	FEND	The End of The Main Program (First End)	ES/EX/SS SA/SX/SC EH/SV

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

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. This instruction denotes the end of the main program. It has the same function as that of END instruction when being executed by PLC.

2. CALL must be written after FEND instruction and add SRET instruction in the end of its subroutine. Interruption program has to be written after FEND instruction and IRET must be added in the end of the service program.

3. If several FEND instructions are in use, place the subroutine and interruption service programs between the final FEND and END instruction.

4. After CALL instruction is executed, executing FEND before SRET will result in errors in the program.

5. After FOR instruction is executed, executing FEND before NEXT will result in errors in the program.

CJ Instruction Program Flow:

CALL Instruction Program Flow:
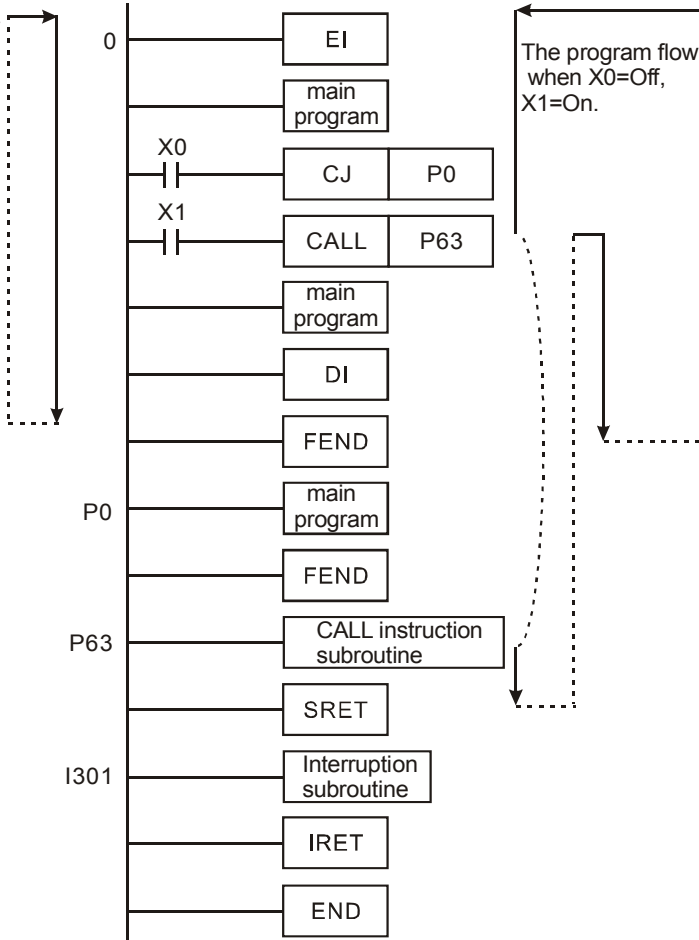
API	Mnemonic		Function	Controllers	
07	WDT	P	Watchdog Timer Refresh	ES/EX/SS	SA/SX/SC	EH/SV

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

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Explanations:

1. No operand.

2. The watchdog timer in DVP series PLCs is used for monitoring the operation of the PLC system.

3. WDT instruction can be used to reset Watch Dog Timer. If the PLC scan time (from step 0 to END or when FEND instruction is executed) exceeds 200ms, PLC ERROR LED will flash. The user will have to turn off PLC and back On again. PLC will determine RUN/STOP status by RUN/STOP switch. If there is no RUN/STOP switch, PLC will return to STOP status automatically.

4. When to use WDT:

   a) When errors occur in the PLC system.

   b) When the executing time of the program is too long, resulting in the scan time being larger than the content in D1000, the user can improve the problem by the following two methods.

   ■   Using WDT instruction
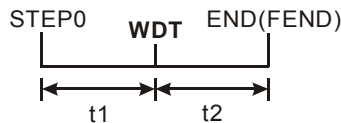
   ■   Using the set value in D1000 (default value: 200ms) to change the time for watchdog.

Program Example:

Assume the scan time of the program is 300ms, divide the program into two parts and place WDT instruction in the middle of the two parts, making scan time of the first half and second half of the program being less than 200ms.
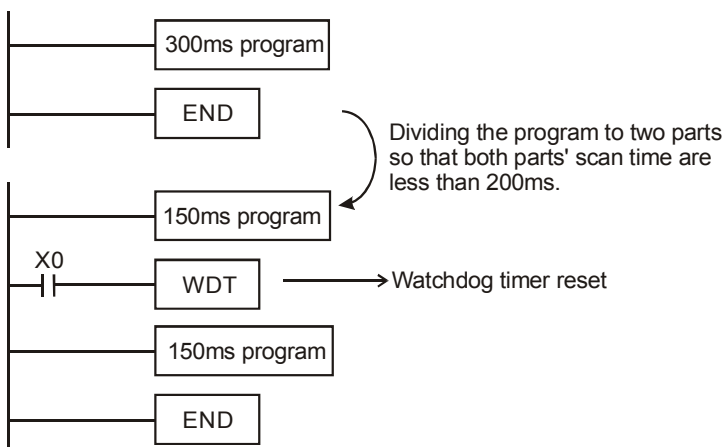
API	Mnemonic	Operands	Function	Controllers
08	FOR	Ⓢ	Start of a FOR-NEXT Loop	ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: The number of repeated nested loops

Explanations:

1. No contact to drive the instruction is required.
2. See the specifications of each model for their range of use.

API	Mnemonic	Function	Controllers
09	NEXT	End of a FOR-NEXT Loop	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A		NEXT: 1 steps

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. No operand. No contact to drive the instruction is required.
2. FOR instruction indicates FOR ~ NEXT loops executing back and forth N times before escaping for the next execution.
3. N = K1 ~ K32,767. N is regarded as K1 when N ≤ 1.
4. When FOR~NEXT loops are not executed, the user can use the CJ instruction to escape the loops.
5. Error will occur when
  a) NEXT instruction is before FOR instruction.
  b) FOR instruction exists but NEXT instruction does not exist.
  c) There is NEXT instruction after FEND or END instruction.
  d) The number of instructions between FOR ~ NEXT differs.
6. 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.
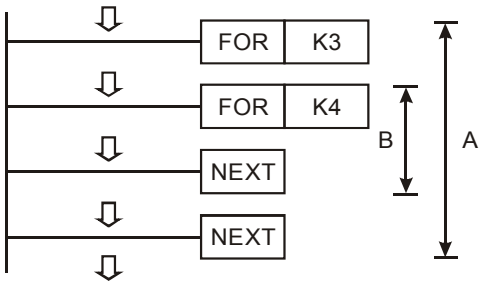
Program Example 2:

When X7 = Off, PLC will execute the program between FOR ~ NEXT. When X7 = On, CJ instruction jumps to P6 and avoids executing the programs between FOR ~ NEXT.
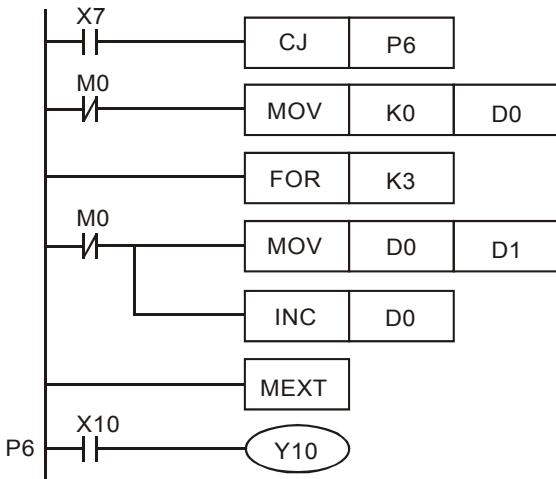
Program Example 3:

When the programs between FOR ~ NEXT are not to be executed, the user can adopt CJ instruction for a jumping. When the most inner FOR ~ NEXT loop is in the status of X1 = On, CJ instruction executes jumping to P0 and skips the execution on P0.

```
X0
─┤/├─────────────── [ TMR    T0    K10 ]

                    [ FOR   K4X100 ]

X0
─┤/├─────────────── [ INC    D0 ]

                    [ FOR    K2 ]

X0
─┤/├─────────────── [ INC    D1 ]

                    [ FOR    K3 ]

X0
─┤/├─────────────── [ INC    D2 ]

                    [ FOR    K4 ]

X0
─┤/├────────┬────── [ WDT ]
            │
            └────── [ INC    D3 ]

X1
─┤ ├─────────────── [ CJ     P0 ]

                    [ FOR    K5 ]

X0
─┤/├─────────────── [ INC    D4 ]

                    [ NEXT ]

P0                  [ NEXT ]

                    [ NEXT ]

                    [ NEXT ]

                    [ NEXT ]

                    [ END ]
```

API		Mnemonic		Operands			Function	Controllers
10	D	CMP	P	S_1 S_2 D			Compare	ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Comparison Value 1   **S₂**: Comparison Value 2   **D**: Comparison result

Explanations:

1.   If **S₁** and **S₂** are used in device F, only 16-bit instruction is applicable.

2.   Operand **D** occupies 3 consecutive devices.

3.   See the specifications of each model for their range of use.

4.   The contents in **S₁** and **S₂** are compared and the result will be stored in **D**.

5.   The two comparison values are compared algebraically and the two values are signed binary values. When b15 = 1 in 16-bit instruction or b31 = 1 in 32-bit instruction, the comparison will regard the value as negative binary values.

Program Example:

1.   Designate device Y0, and operand D automatically occupies Y0, Y1, and Y2.

2.   When X10 = On, CMP instruction will be executed and one of Y0, Y1, and Y2 will be On. When X10 = Off, CMP instruction will not be executed and Y0, Y1, and Y2 remain their status before X10 = Off.

3.   If the user need to obtain a comparison result with ≥ ≤, and ≠, make a series parallel connection between Y0 ~ Y2.
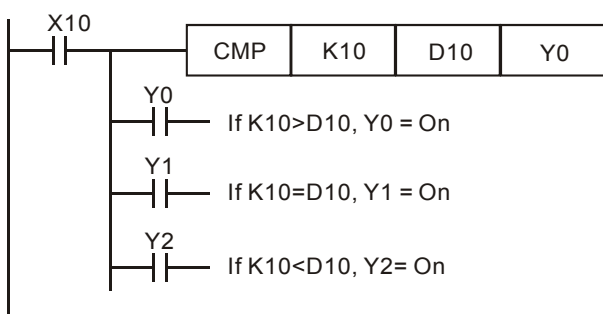
```
   X10
   ─┤├───────┌─────┬─────┬─────┬─────┐
             │ CMP │ K10 │ D10 │ Y0  │
             └─────┴─────┴─────┴─────┘
   Y0
   ─┤├─────── If K10>D10, Y0 = On

   Y1
   ─┤├─────── If K10=D10, Y1 = On

   Y2
   ─┤├─────── If K10<D10, Y2= On
```

4.   To clear the comparison result, use RST or ZRST instruction.

```
   X10                                X10
   ─┤/├──┬──┌─────┬─────┐             ─┤/├──┌──────┬─────┬─────┐
         │  │ RST │ M0  │                   │ ZRST │ M0  │ M2  │
         │  └─────┴─────┘                   └──────┴─────┴─────┘
         │  ┌─────┬─────┐
         ├──│ RST │ M1  │
         │  └─────┴─────┘
         │  ┌─────┬─────┐
         └──│ RST │ M2  │
            └─────┴─────┘
```
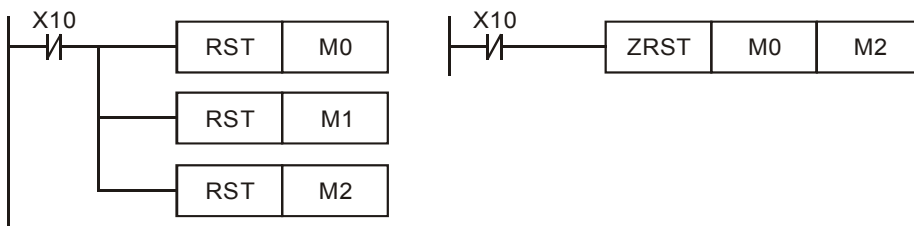
API		Mnemonic		Operands				Function	Controllers
11	D	ZCP	P	S_1 S_2 S D				Zone Compare	ES/EX/SS SA/SX/SC EH/SV

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
S_1					*	*	*	*	*	*	*	*	*	*	*	DZCP, DZCPP: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
S					*	*	*	*	*	*	*	*	*	*	*	
D		*	*	*												

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Lower bound of zone comparison   **S₂**: Upper bound of zone comparison   **S**: Comparison value

D: Comparison result

Explanations:

1. If **S₁**, **S₂** and **S** are used in device F, only 16-bit instruction is applicable.

2. The content in **S₁** should be smaller than the content in **S₂**.

3. Operand **D** occupies 3 consecutive devices.

4. See the specifications of each model for their range of use.

5. **S** is compared with its **S₁**, **S₂** and the result is stored in **D**.

6. When **S₁** > **S₂**, the instruction performs comparison by using **S₁** as the lower/upper bound.

7. The two comparison values are compared algebraically and the two values are signed binary values. When b15 = 1 in 16-bit instruction or b31 = 1 in 32-bit instruction, the comparison will regard the value as negative binary values.

Program Example:

1. Designate device M0, and operand D automatically occupies M0, M1 and M2.

2. When X0 = On, ZCP instruction will be executed and one of M0, M1, and M2 will be On. When X0 = Off, ZCP instruction will not be executed and M0, M1, and M2 remain their status before X0 = Off.

```
   X0
───┤├──────┬── ZCP │ K10 │ K100 │ C10 │ M0
           │
   M0      │
───┤├──────── If C10 < K10, M0 = On
           │
   M1      │
───┤├──────── If K10 ≤ C10 ≤ K100, M1 = On
           │
   M2      │
───┤├──────── If C10 > K100, M2 = On
```

3. To clear the comparison result, use RST or ZRST instruction.

```
   X0                         X0
───┤/├────┬── RST │ M0      ───┤/├──── ZRST │ M0 │ M2
          │
          ├── RST │ M1
          │
          └── RST │ M2
```

API		Mnemonic		Operands		Function			Controllers
12	D	MOV	P	Ⓢ Ⓓ		Move			ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. 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.

4. If the operation result refers to a 32-bit output, (i.e. application instruction MUL and so on), and the user needs to move the present value in the 32-bit high-speed counter, DMOV instruction has to be adopted.

Program Example:

1. MOV instruction has to be adopted in the moving of 16-bit data.

  a) When X0 = Off, the content in D10 will remain unchanged. If X0 = On, the value K10 will be moved to D10 data register.

  b) When X1 = Off, the content in D10 will remain unchanged. If X1 = On, the present value T0 will be moved to D10 data register.

2. DMOV instruction has to be adopted in the moving of 32-bit data.

    When X2 = Off, the content in (D31, D30) and (D41, D40) will remain unchanged. If X2 = On, the present value of (D21, D20) will be sent to (D31, D30) data register. Meanwhile, the present value of C235 will be moved to (D41, D40) data register.

X0	MOV	K10	D0
X1	MOV	T0	D10
X2	DMOV	D20	D30
	DMOV	C235	D40
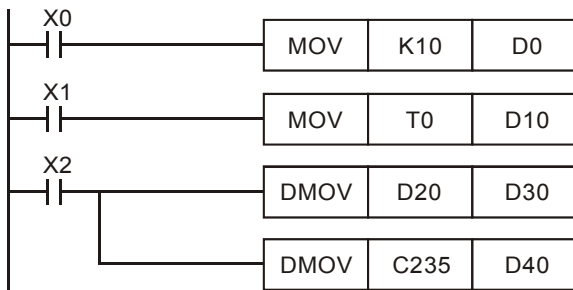
API	Mnemonic		Operands	Function	Controllers
13	SMOV	P	Ⓢ ⓜ₁ ⓜ₂ Ⓓ Ⓝ	Shift Move	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices										Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SMOV, SMOVP: 11 steps
S							*	*	*	*	*	*	*	*	*	
m₁					*	*										
m₂					*	*										
D								*	*	*	*	*	*	*	*	
n					*	*										

	PULSE									16-bit									32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source of data   **m₁**: Start digit to be moved of the source data   **m₂**: Number of digits (nibbles) to be moved of the source data   **D**: Destination device   **n**: Start digit of the destination position for the moved digits

Explanations:

1. This instruction is able to re-allocate or combine data. When the instruction is executed, **m₂** digits of contents starting from digit **m₁** (from high digit to low digit) of **S** will be sent to **m₂** digits starting from digit **n** (from high digit to low digit) of **D**.

2. Range: m_1 = 1 ~ 4; m_2 = 1 ~ m_1; n = m_2 ~ 4

3. See the specifications of each model for their range of use.

4. M1168 is designated by SMOV working mode. When M1168 = On, the program is in BIN mode. When M1168 = Off, the program is in BCD mode.

Program Example 1:

1. When M1168 = Off (in BCD mode) and X0 = On, the 4th (thousand) and 3rd (hundred) digit of the decimal value in D10 start to move to the 3rd (hundred) and 2nd (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 will determine an operation error and will not execute the instruction. M1067, M1068 = On and D1067 records the error code OE18 (hex).
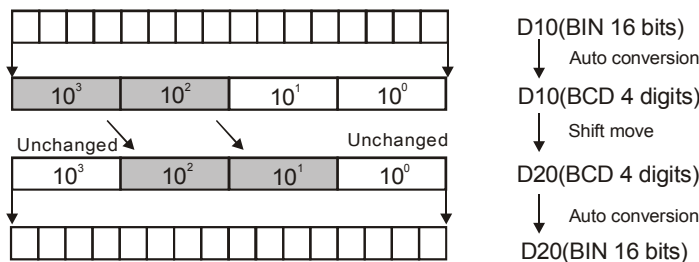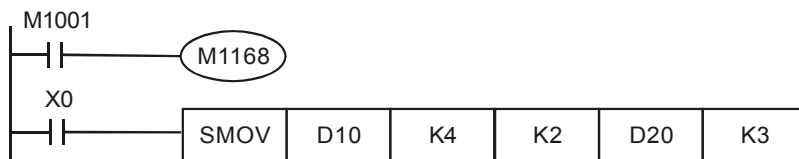
Before the execution, assume D10 = K1234 and D20 = K5678. After the execution, D10 will remain unchanged and D20 will become K5128.

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).
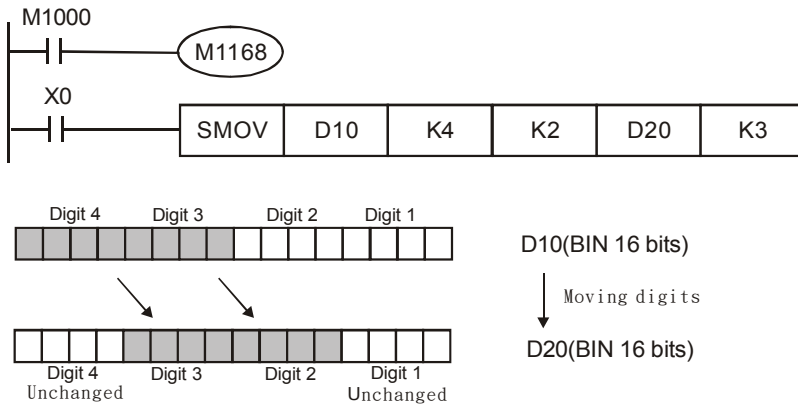
Before the execution, assume D10 = H1234 and D20 = H5678. After the execution, D10 will remain unchanged and D20 will become H5128.

Program Example 3:

1. This instruction can be used to combine the DIP switches connected to the input terminals with interrupted No.
2. Move the 2nd right digit of the DIP switch to the 2nd right digit of D2, and the 1st left digit of the DIP switch to the 1st right digit of D1.
3. Use SMOV instruction to move the 1st digit of D1 to the 3rd digit of D2 and combine the two DIP switches into one.
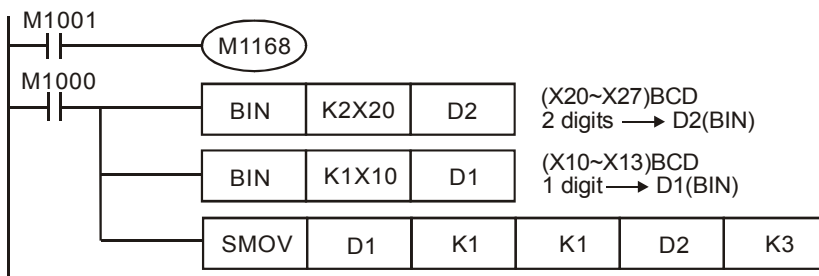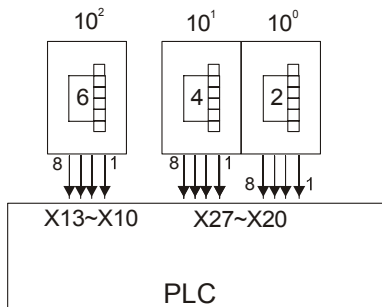
API		Mnemonic		Operands		Function		Controllers
14	D	CML	P	Ⓢ Ⓓ		Compliment		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. This instruction can be used for phase-reversed output.

4. Reverse the phase (0→1, 1→0) of all the contents in **S** and send the contents to **D**. Given that the content is a constant K, K will be automatically converted into a BIN value.

Program Example 1:

1. When X10 = On, b0 ~ b3 in D1 will be phase-reversed and send to Y0 ~ Y3.
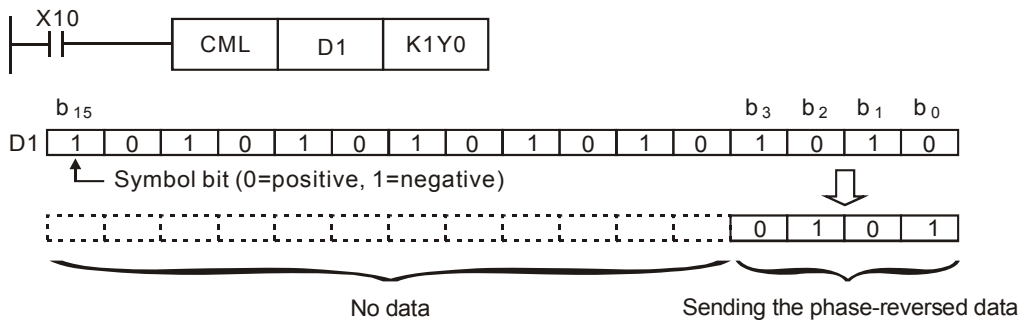
Program Example 2:

The loop below can also adopt CML instruction (see right below).
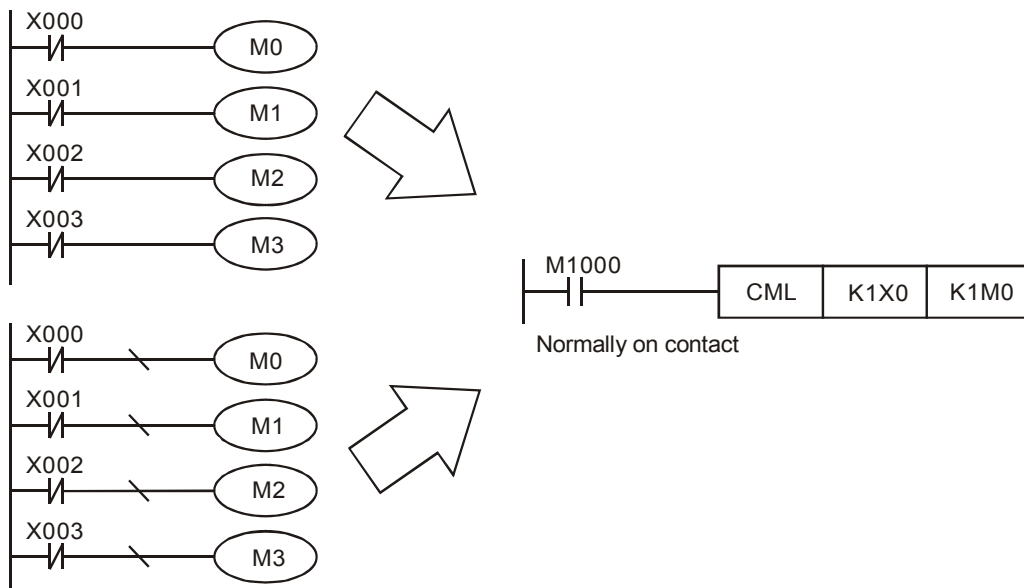
API	Mnemonic		Operands			Function		Controllers
15	BMOV	P	**S** **D** **n**			Block Move		ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. Range of **n**: 1 ~ 512

2. See the specifications of each model for their range of use.

3. 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.

Program Example 1:

When X10 = On, the contents in registers D0 ~ D3 will be moved to the 4 registers D20 ~ D23.
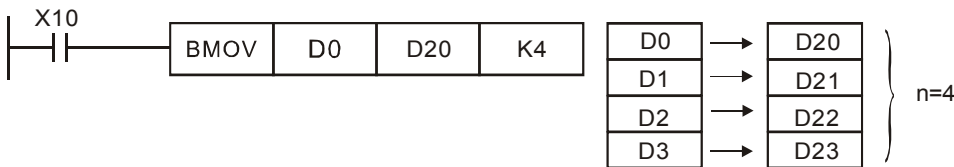
Program Example 2:

1. 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.

2. ES/EX/SS do not support the use of KnX, KnY, KnM, KnS and E, F index register modification.
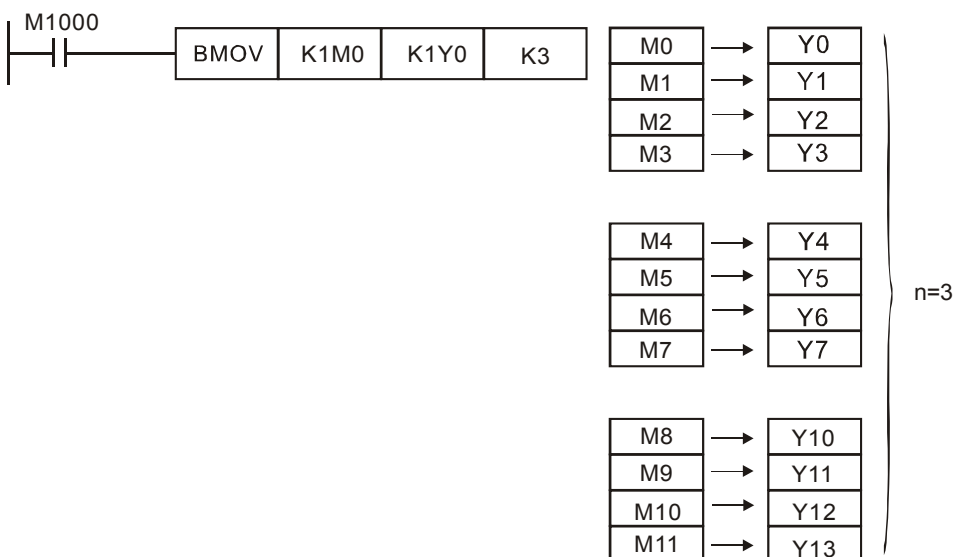
Program Example 3:

To avoid coincidence of the device numbers to be moved designated by the two operands and cause confusion, please be aware of the arrangement on the designated device numbers.

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

X10				
┤ ├	BMOV	D20	D19	K3

D20	①	D19
D21	②	D20
D22	③	D21
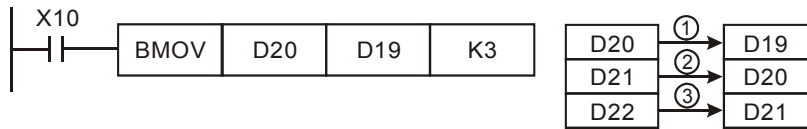
2. In EH/EH2/SV, when **S** < **D**, the instruction is processed following the order ①→②→③

X11				
┤ ├	BMOV	D10	D11	K3

D10	③	D11
D11	②	D12
D12	①	D13
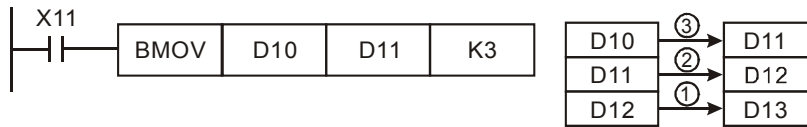
3. In ESEX/SS/SA/SX/SC, when **S** < **D**, avoid the number difference of "1" and the instruction is processed following the order ③→②→①. If the devices have the number difference of "1", the contents in D11 ~ D13 will all be the content in D10.

X11				
┤ ├	BMOV	D10	D11	K3

D10	③	D11
D11	②	D12
D12	①	D13
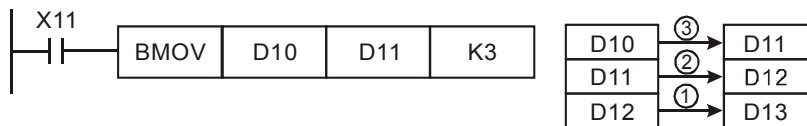
API	Mnemonic			Operands			Function		Controllers
16	D	FMOV	P	**S** **D** **n**			Fill Move		ES/EX/SS SA/SX/SC EH/SV

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 steps
D								*	*	*	*	*	*	*		
n					*	*										

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

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

Explanations:

1. If **S** is used in device F, only 16-bit instruction is applicable.

2. Range of **n**: 1~ 512 (16-bit instruction); 1 ~ 256 (32-bit instruction).

3. See the specifications of each model for their range of use.

4. 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.

5. ES/EX/SS do not support the use of KnX, KnY, KnM, KnS and E, F index register modification.

Program Example:

When X10 = On, K10 will be moved to the 5 consecutive registers starting from D10.

```
 X10
 ─┤ ├──────┬─────┬─────┬─────┬─────┐
            │FMOV │ K10 │ D10 │ K5  │
            └─────┴─────┴─────┴─────┘
```

```
 ┌─────┐
 │ K10 ├─┬── K10  ⟶  ┌──────┐
 └─────┘ │           │ D10  │
         ├── K10  ⟶  ┌──────┐
         │           │ D11  │
         ├── K10  ⟶  ┌──────┐   n=5
         │           │ D12  │
         ├── K10  ⟶  ┌──────┐
         │           │ D13  │
         └── K10  ⟶  ┌──────┐
                     │ D14  │
```
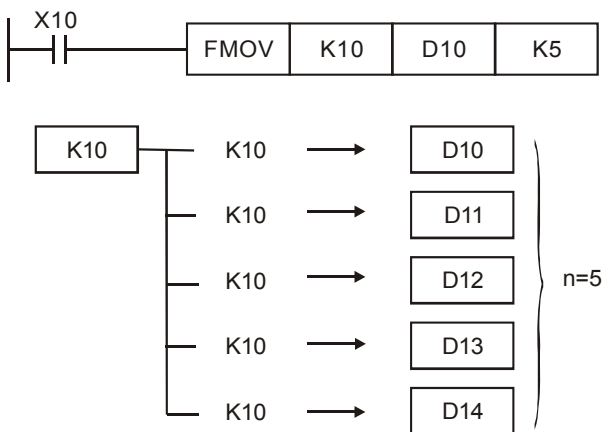
API	Mnemonic			Operands		Function	Controllers
17	D	XCH	P	Ⓓ₁ Ⓓ₂		Exchange	ES/EX/SS SA/SX/SC EH/SV

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
D₁								*	*	*	*	*	*	*	*	
D₂								*	*	*	*	*	*	*	*	DXCH, DXCHP: 9 steps

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D₁: Data to be exchanged 1   **D₂**: Data to be exchanged 2

Explanations:

1.　If **D₁** and **D₂** are used in device F, only 16-bit instruction is applicable.

2.　See the specifications of each model for their range of use.

3.　The contents in the devices designated by **D₁** and **D₂** will exchange.

4.　Flag: M1303 (designated by XCH working mode).

Program Example 1:

When X0 = Off→On, the contents in D20 and D40 exchange with each other.
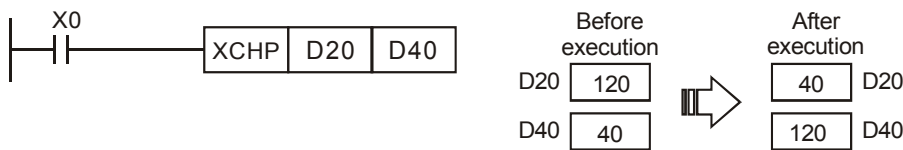
Program Example 2:

When X0 = Off → On, the contents in D100 and D200 exchange with each other.

Remarks:

1.　ES/EX/SS do not support M1303.

2.　As a 16-bit instruction, when the devices designated by **D₁** and **D₂** are the same and M1303 = On, the upper and lower 8 bits of the designated devices exchange with each other.

3.　As a 32-bit instruction, when the devices designated by **D₁** and **D₂** are the same and M1303 = On, the upper and lower 16 bits in the individual designated device exchange with each other.

4.　When X0 = On and M1303 = On, the 16-bit contents in D100 and those in D101 will exchange with each other.

```
 X0
─┤ ├──────( M1303 )
        │
        ├──┤DXCHP│ D100 │ D100 │
```
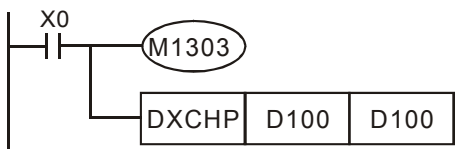
	Before execution		After execution	
D100L	9	⇨	8	D100L
D100H	20		40	D100H
D101L	8	⇨	9	D101L
D101H	40		20	D101H

API		Mnemonic		Operands		Function		Controllers
18	D	BCD	P	Ⓢ	Ⓓ	Binary Coded Decimal		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. Flags: M1067 (operation error); M1068 (operation error); D1067 (error code)

4. The content in **S** (BIN value) is converted into BCD value and stored in **D**.

5. 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).

6. The four arithmetic operations and applications in PLC and the execution of INC and DEC instructions are performed in BIN format. Therefore, if the user needs to see the decimal value display, simply use this instruction to convert the BIN value into BCD value.

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).

X0			
┤├	BCD	D10	K1Y0
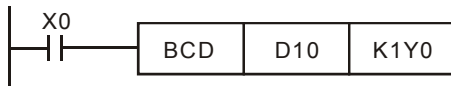
2. When D10 = 001E (hex) = 0030 (decimal), the execution result will be: Y0 ~ Y3 = 0000(BIN).

API		Mnemonic		Operands		Function		Controllers
19	D	BIN	P	S D		Binary		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices									Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. Flags: M1067 (operation error); M1068 (operation error); D1067 (error code)

4. The content in **S** (BCD value) is converted into BIN value and stored in **D**.

5. Valid range of **S** : BCD (0 ~ 9,999), DBCD (0 ~ 99,999,999)

6. Provided the content in S is not a BCD value (in hex and any one of its digits does not fall in the range of 0 ~ 9), an operation error will occur. M1067, M1068 = On and D1067 records the error code 0E18 (hex).

7. Constant K and H will automatically be converted into BIN format. Thus, they do not need to adopt this instruction.

Program Example:

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

```
  X0
──┤├──┤ BIN │ K1M0 │ D10 │
```

Remarks:

Explanations on BCD and BIN instructions:

1. 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.

2. When PLC needs to display its stored data by a 7-segment display in BCD format, BCD instruction has to be first adopted to convert the data into BCD value and send the data to the 7-segment display.

3. When X0 = On, the BCD value of K4X0 is converted into BIN value and sent it to D100. The BIN value of D100 will then be converted into BCD value and sent to K4Y20.

```
  X0
──┤├──┬─┤ BIN │ K4X0 │ D100 │
      │
      └─┤ BCD │ D100 │ K4Y20 │
```

4-digit DIP switch in BCD format

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.

4-digit 7-segment display in BCD format
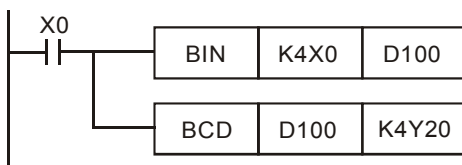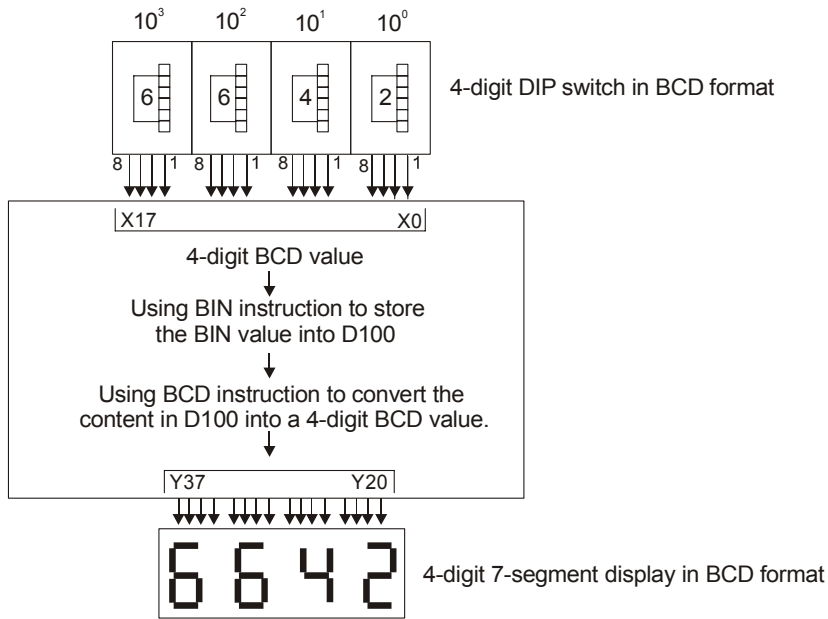
API		Mnemonic		Operands			Function		Controllers
20	D	ADD	P	S_1 S_2 D			Addition		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. If **S_1**, **S_2** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

4. This instruction adds **S_1** and **S_2** in BIN format and store the result in **D**.

5. The highest bit is symbolic bit 0 (+) and 1 (-), which is suitable for algebraic addition, e.g. $3 + (-9) = -6$.

6. Flag changes in binary addition

   In 16-bit BIN addition,

   a) If the operation result $=$ 0, zero flag M1020 = On.

   b) If the operation result $<$ -32,768, borrow flag M1021 = On.

   c) If the operation result $>$ 32,767, carry flag M1022 = On.

   In 32-bit BIN addition,

   a) If the operation result $=$ 0, zero flag M1020 = On.

   b) If the operation result $<$ -2,147,483,648, borrow flag M1021 = On.

   c) If the operation result $>$ 2,147,483,647, carry flag M1022 = On.

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.

X0	ADD	D0	D10	D20
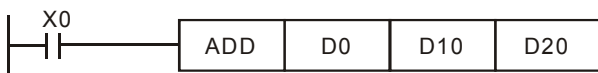
Program Example 2:

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 16-bit data; D31, D41 and D51 are high 16-bit data.

X10	DADD	D30	D40	D50
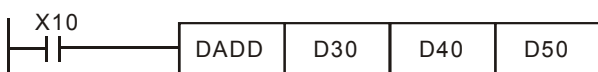
(D31, D30) $+$ (D41, D40) $=$ (D51, D50)

Remarks:

Flags and the positive/negative sign of the values:

16-bit :

| Zero flag | Zero flag | Zero flag |

-2、-1、0、-32,768 ← − − -1、0、1 → + → 32,767、0、1、2

Borrow flag   The highest bit of the data = 1 (negative)   The highest bit of the data = 0 (positive)   Carry flag

32-bit :

| Zero flag | Zero flag | Zero flag |

-2、-1、0、-2,147,483,648 ← − -1、0、1 → + → 2,147,483,647、0、1、2

Borrow flag   The highest bit of the data = 1 (negative)   The highest bit of the data = 0 (positive)   Carry flag
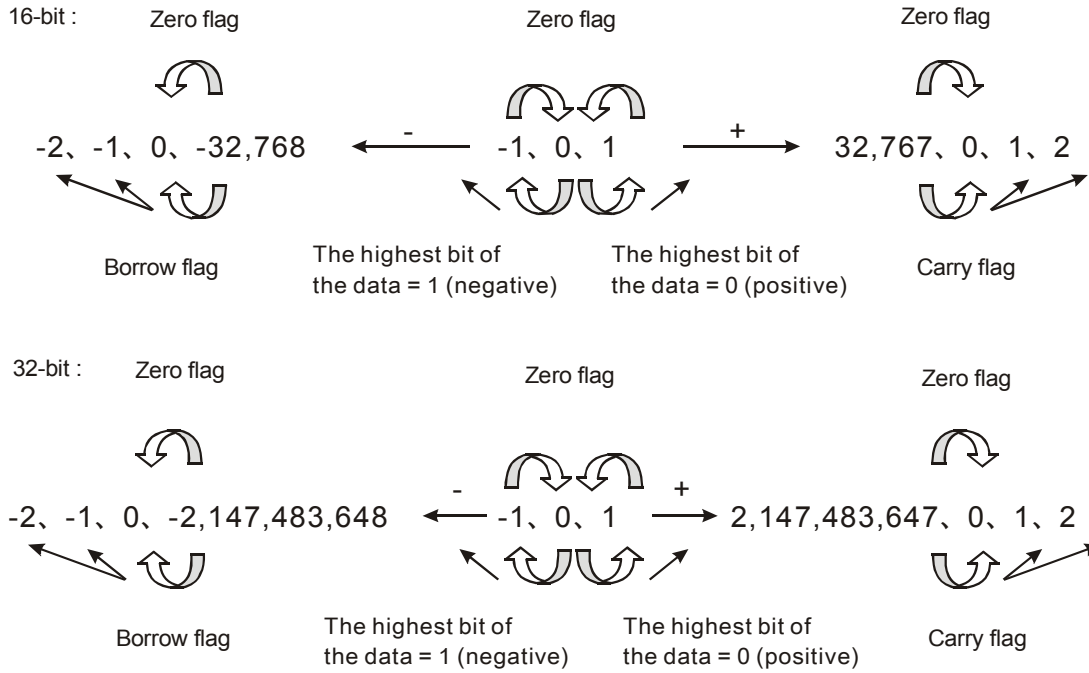
API	Mnemonic			Operands			Function		Controllers
21	D	SUB	P	S_1 S_2 D			Subtraction		ES/EX/SS SA/SX/SC EH/SV

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

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Minuend   **S₂**: Subtrahend   **D**: Remainder

Explanations:

1. If **S₁**, **S₂** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

4. This instruction subtracts **S₁** and **S₂** in BIN format and stores the result in **D**.

5. The highest bit is symbolic bit 0 (+) and 1 (-), which is suitable for algebraic subtraction.

6. Flag changes in binary subtraction

   In 16-bit instruction:

   a) If the operation result $=$ 0, zero flag M1020 = On.

   b) If the operation result $<$ –32,768, borrow flag M1021 = On.

   c) If the operation result $>$ 32,767, carry flag M1022 = On.

   In 32-bit instruction:

   a) If the operation result $=$ 0, zero flag M1020 = On.

   b) If the operation result $<$ –2,147,483,648, borrow flag M1021 = On.

   c) If the operation result $>$ 2,147,483,647, carry flag M1022 = On.

7. For flag operations of SUB instruction and the positive/negative sign of the value, see the explanations in ADD instruction on the previous page.

Program Example 1:

In 16-bit BIN subtraction:

When X0 = On, the content in D0 will minus the content in D10 and the remainder will be stored in D20.

X0				
┤├	SUB	D0	D10	D20
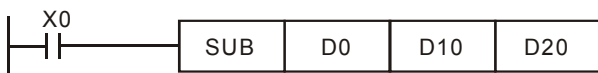
Program Example 2:

In 32-bit BIN subtraction:

When X10 = On, the content in (D31, D30) will minus the content in (D41, D40) and the remainder will be stored in (D51, D50). D30, D40 and D50 are low 16-bit data; D31, D41 and D51 are high 16-bit data.

X10				
┤├	DSUB	D30	D40	D50
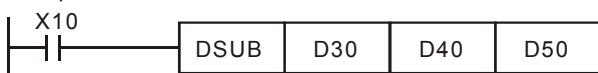
(D31, D30) − (D41, D40) = (D51, D50)

API	Mnemonic			Operands			Function		Controllers
22	D	MUL	P	S₁ S₂ D			Multiplication		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MUL, DMULP: 7 steps
S₁					*	*	*	*	*	*	*	*	*	*	*	DMUL, DMULP: 13 steps
S₂					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*		

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Multiplicand   **S₂**: Multiplicator   **D**: Product

Explanations:

1. If **S₁** and **S₂** are used in device F, only 16-bit instruction is applicable.

2. If **D** is used in device E, only 16-bit instruction is applicable.

3. In 16-bit instruction, **D** occupies 2 consecutive devices.

4. In 32-bit instruction, **D** occupies 4 consecutive devices.

5. See the specifications of each model for their range of use.

6. This instruction multiplies **S₁** by **S₂** in BIN format and stores the result in **D**. Be careful with the positive/negative signs of **S₁**, **S₂** and **D** when doing 16-bit and 32-bit operations.

7. In 16-bit BIN multiplication,
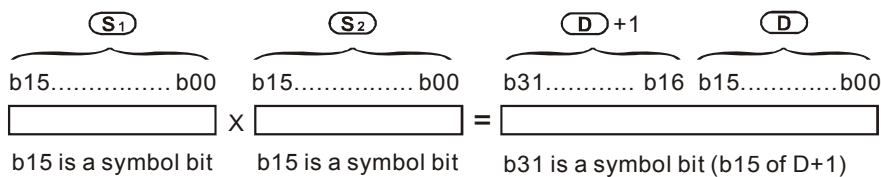
S₁   b15................ b00   b15 is a symbol bit
S₂   b15................ b00   b15 is a symbol bit
D+1 b31............ b16 D b15..............b00   b31 is a symbol bit (b15 of D+1)

Symbol bit = 0 refers to a positive value.
Symbol bit = 1 refers to a negative value.

When D serves as a bit device, it can designate K1 ~ K4 and construct a 16-bit result, occupying consecutive 2 groups of 16-bit data. ES/EX/SS only stores low 16-bit data.

8. 32-bit BIN multiplication,
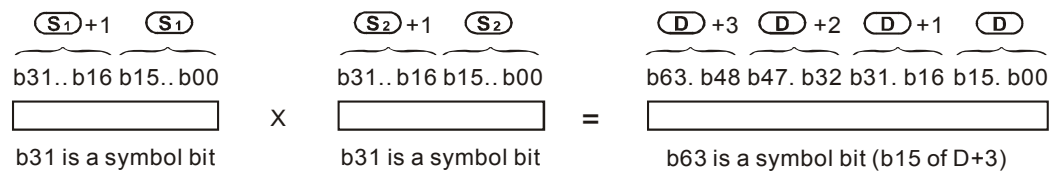
S₁+1 b31..b16 S₁ b15..b00   b31 is a symbol bit
S₂+1 b31..b16 S₂ b15..b00   b31 is a symbol bit
D+3 b63.b48 D+2 b47.b32 D+1 b31.b16 D b15.b00   b63 is a symbol bit (b15 of D+3)

Symbol bit = 0 refers to a positive value.
Symbol bit = 1 refers to a negative value.

When D serves as a bit device, it can designate K1 ~ K8 and construct a 32-bit result, occupying consecutive 2 groups of 32-bit data.

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 the most left bit indicates the positive/negative status of the result value.

X0			
MUL	D0	D10	D20
MUL	D0	D10	K8M0

API		Mnemonic		Operands			Function		Controllers
23	D	DIV	P	Ⓢ₁ Ⓢ₂ Ⓓ			Division		ES/EX/SS SA/SX/SC EH/SV

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
S₁					*	*	*	*	*	*	*	*	*	*		DDIV, DDIVP: 13 steps
S₂					*	*	*	*	*	*	*	*	*	*		
D							*	*	*	*	*	*	*	*		

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Dividend   **S₂**: Divisor   **D**: Quotient and remainder

Explanations:

1. If **S₁** and **S₂** are used in device F, only 16-bit instruction is applicable.

2. If **D** is used in device E, only 16-bit instruction is applicable.

3. In 16-bit instruction, **D** occupies 2 consecutive devices.

4. In 32-bit instruction, **D** occupies 4 consecutive devices.

5. See the specifications of each model for their range of use.

6. This instruction divides **S₁** and **S₂** in BIN format and stores the result in **D**. Be careful with the positive/negative signs of **S₁**, **S₂** and **D** when doing 16-bit and 32-bit operations.

7. 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).

8. In 16-bit BIN division,
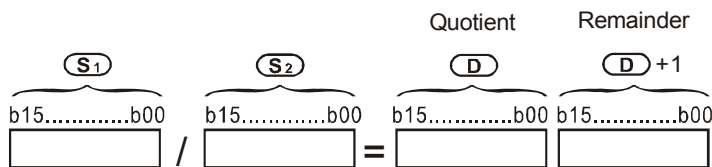
When D serves as a bit device, it can designate K1 ~ K4 and construct a 16-bit result, occupying consecutive 2 groups of 16-bit data and bringing forth the quotient and remainder. ES/EX/SS is able to bring forth only quotient without the remainder.

9. In 32-bit BIN division,
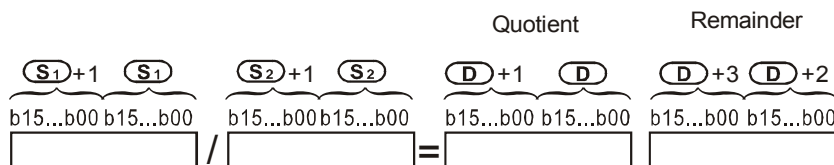
When D serves as a bit device, it can designate K1 ~ K8 and construct a 32-bit result, occupying consecutive 2 groups of 32-bit data and bringing forth the quotient and remainder.

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 highest bit indicates the positive/negative status of the result value.

API		Mnemonic		Operands	Function		Controllers
24	D	INC	P	Ⓓ	Increment		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Destination device

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. If the instruction is not a pulse execution one, the content in the designated device D will plus "1" in every scan period whenever the instruction is executed.

4. This instruction adopts pulse execution instructions (INCP, DINCP).

5. In 16-bit operation, 32,767 pluses 1 and obtains -32,768. In 32-bit operation, 2,147,483,647 pluses 1 and obtains -2,147,483,648.

6. The operation results will not affect M1020 ~ M1022.

Program Example:

When X0 = Off→On, the content in D0 pluses 1 automatically.

```
      X0
   ├──┤ ├──────┤ INCP │ D0 │
```
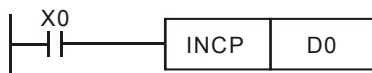
API		Mnemonic		Operands	Function		Controllers
25	D	DEC	P	Ⓓ	Decrement		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices									Program Steps
OP	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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Destination device

Explanations:

1.   If **D** is used in device F, only 16-bit instruction is applicable.

2.   See the specifications of each model for their range of use.

3.   If the instruction is not a pulse execution one, the content in the designated device D will minus "1" in every scan period whenever the instruction is executed.

4.   This instruction adopts pulse execution instructions (DECP, DDECP).

5.   In 16-bit operation, -32,768 minuses 1 and obtains 32,767. In 32-bit operation, -2,147,483,648 minuses 1 and obtains 2,147,483,647.

6.   The operation results will not affect M1020 ~ M1022.

Program Example:

When X0 = Off→On, the content in D0 minuses 1 automatically.

```
     X0
  ───┤ ├───    ┌──────┬──────┐
               │ DECP │  D0  │
               └──────┴──────┘
```
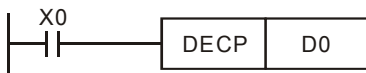
API	Mnemonic			Operands			Function	Controllers
26	W / D	AND	P	S_1 S_2 D			Logical Word AND	ES/EX/SS SA/SX/SC EH/SV

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					*	*	*	*	*	*	*	*	*	*	*	DAND, DANDP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D							*	*	*	*	*	*	*	*	*	

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Source data device 1   **S₂**: Source data device 2   **D**: Operation result

Explanations:

1. If **S₁**, **S₂** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. This instruction conducts logical AND operation of **S₁** and **S₂** and stores the result in **D**.

4. Operation rule: The corresponding bit of the operation result in **D** will be "0" if any of the bits in **S₁** or **S₂** is "0".

Program Example 1:

When X0 = On, the 16-bit D0 and D2 will perform WAND, logical AND operation, and the result will be stored in D4.
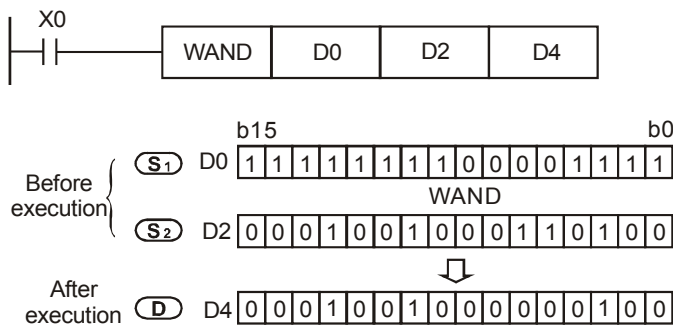
Program Example 2:

When X1 = On, the 32-bit (D11, D10) and (D21, D20) will perform DAND, logical AND operation, and the result will be stored in (D41, D40).
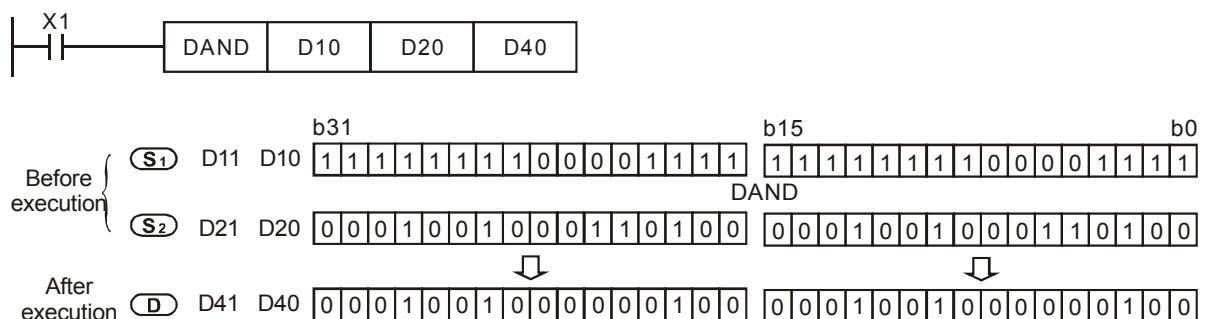
API		Mnemonic		Operands		Function		Controllers
27	W D	OR	P	S_1 S_2 D		Logical Word OR		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WOR, WORP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DOR, DORP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D							*	*	*	*	*	*	*	*	*	

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Source data device 1   **S₂**: Source data device 2   **D**: Operation result

Explanations:

1. If **S₁**, **S₂** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. This instruction conducts logical OR operation of **S₁** and **S₂** and stores the result in **D**.

4. Operation rule: The corresponding bit of the operation result in **D** will be "1" if any of the bits in **S₁** or **S₂** is "1".

Program Example 1:

When X0 = On, the 16-bit D0 and D2 will perform WOR, logical OR operation, and the result will be stored in D4.
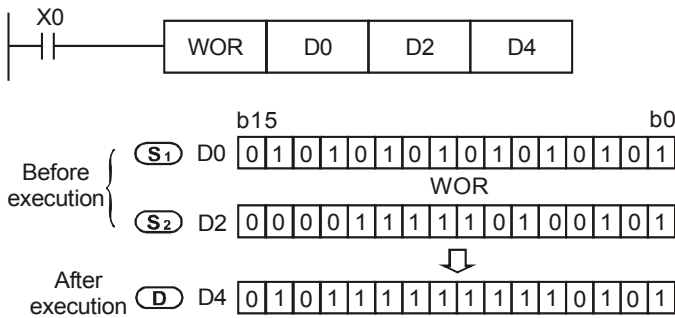
Program Example 2:

When X1 = On, the 32-bit (D11, D10) and (D21, D20) will perform DOR, logical OR operation, and the result will be stored in (D41, D40).
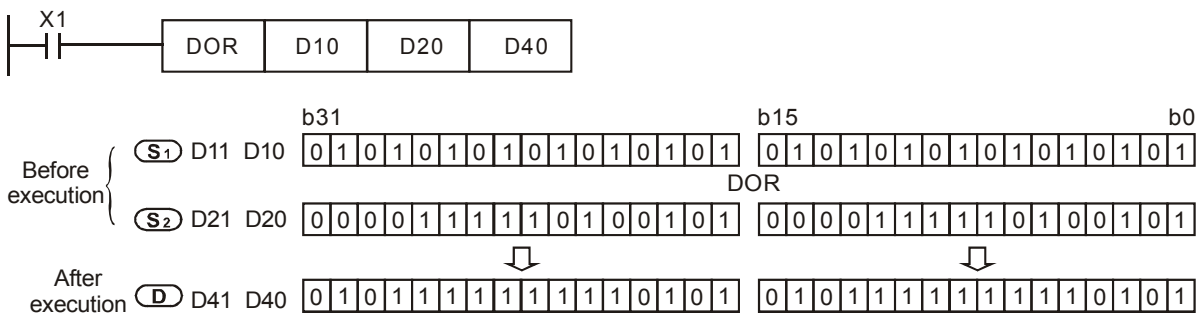
API		Mnemonic		Operands		Function	Controllers
28	W D	XOR	P	S_1 S_2 D		Logical Exclusive OR	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WXOR, WXORP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DXOR, DXORP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D							*	*	*	*	*	*	*	*	*	

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Source data device 1   **S₂**: Source data device 2   **D**: Operation result

Explanations:

1.   If **S₁**, **S₂** and **D** are used in device F, only 16-bit instruction is applicable.

2.   See the specifications of each model for their range of use.

3.   This instruction conducts logical XOR operation of **S₁** and **S₂** and stores the result in **D**.

4.   Operation rule: If the bits in **S₁** and **S₂** are the same, the corresponding bit of the operation result in **D** will be "0"; if the bits in **S₁** and **S₂** are different, the corresponding bit of the operation result in **D** will be "1".

Program Example 1:

When X0 = On, the 16-bit D0 and D2 will perform WXOR, logical XOR operation, and the result will be stored in D4.
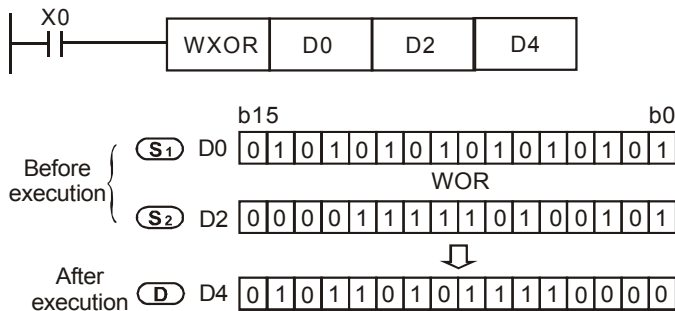
Program Example 2:

When X1 = On, the 32-bit (D11, D10) and (D21, D20) will perform DXOR, logical XOR operation, and the result will be stored in (D41, D40).
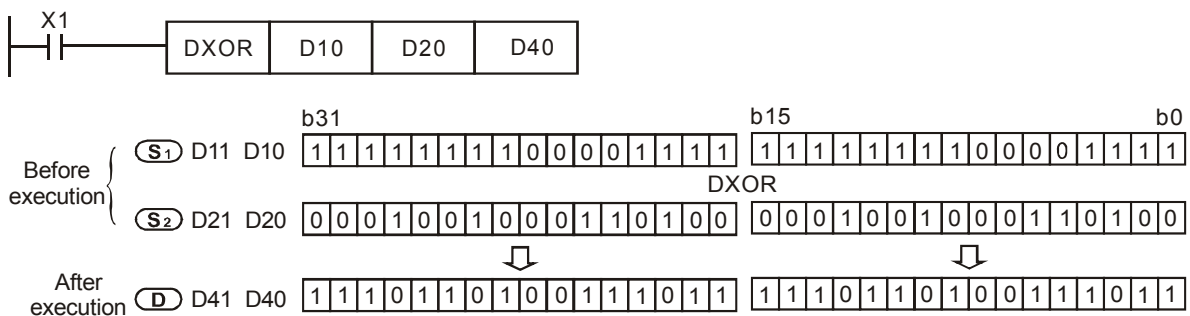
API		Mnemonic		Operands	Function	Controllers
29	D	NEG	P	$\overline{D}$	2's Complement (Negative)	ES/EX/SS SA/SX/SC EH/SV

Type	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
OP																
D								*	*	*	*	*	*	*	*	DNEG, DNEGP: 5 steps

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device to store 2's complement

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. This instruction converts a negative BIN value into an absolute value.

4. This instruction adopts pulse execution instructions (NEGP, DNEGP).

Program Example 1:

When X0 = Off→On, the phase of every bit of the content in D10 will be reversed (0→1, 1→0) and pluses 1. The result will then be stored in D10.

```
  X0
 ─┤├─────[ NEGP │ D10 ]
```

Program Example 2:

Obtaining the absolute value of a negative value:

a) When the 15th bit of D0 is "1", M0 = On. (D0 is a negative value).

b) When M0 = Off→On, NEG instruction will obtain 2's complement of D0 and further its absolute value.

```
  M1000
 ─┤├─────[ BON │ D0 │ M0 │ K15 ]
Normal ON contact

  M0
 ─┤├─────[ NEGP │ D0 ]
```
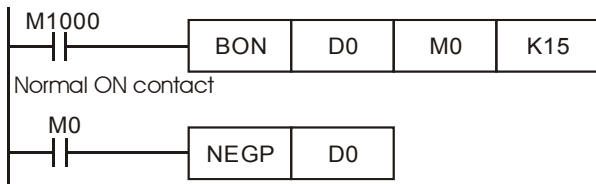
Program Example 3:

Obtaining the absolute value by 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.
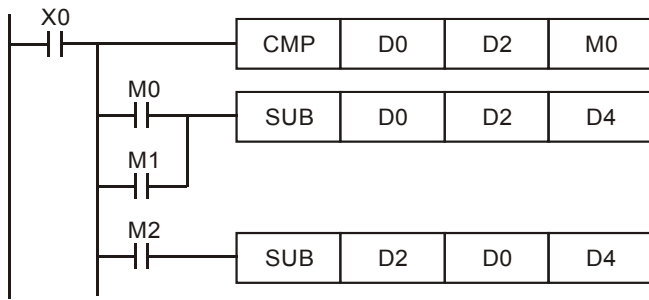
Remarks:

Negative value and its absolute value

a) The sign of a value is indicated by the highest (most left) bit in the register. 0 indicates that the value is a positive one and 1 indicates that the value is a negative one.

b) NEG instruction is able to convert a negative value into its absolute value.

(D0=2)
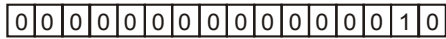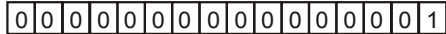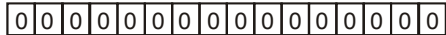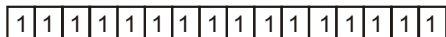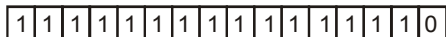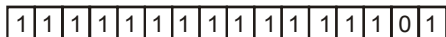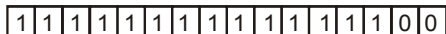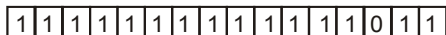
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

(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
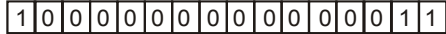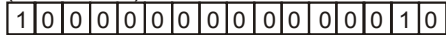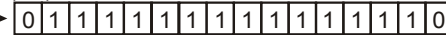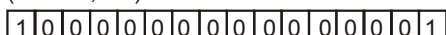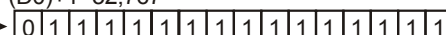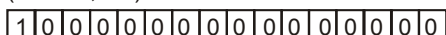
Max. absolute value is 32,767

API		Mnemonic		Operands		Function	Controllers
30	D	ROR	P	$\overline{D}$ $\overline{n}$		Rotation Right	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device to be rotated   **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.

3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)

4. See the specifications of each model for their range of use.

5. Flag: M1022 (carry flag)

6. This instruction rotates the device content designated by **D** to the right for **n** bits.

7. This instruction adopts pulse execution instructions (RORP, DRORP).

Program Example:

When X0 = Off→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.

DVP-PLC Application Manual
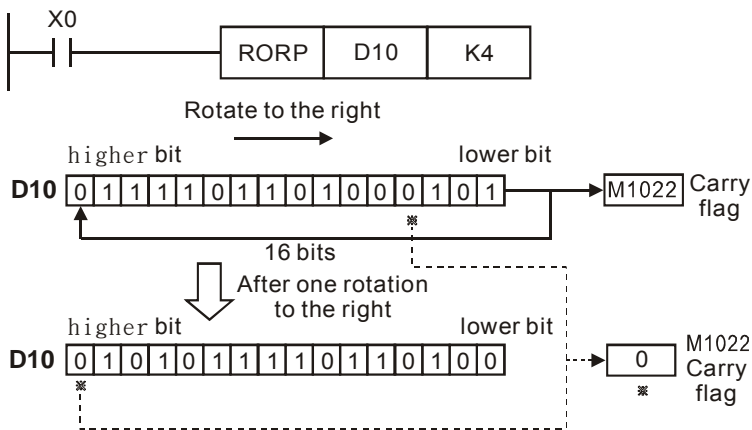
API		Mnemonic		Operands	Function	Controllers
31	D	ROL	P	D n	Rotation Left	ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device to be rotated   **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.

3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)

4. See the specifications of each model for their range of use.

5. Flag: M1022 (carry flag)

6. This instruction rotates the device content designated by **D** to the left for **n** bits.

7. This instruction adopts pulse execution instructions (ROLP, DROLP).

Program Example:

When X0 = Off→On, 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.
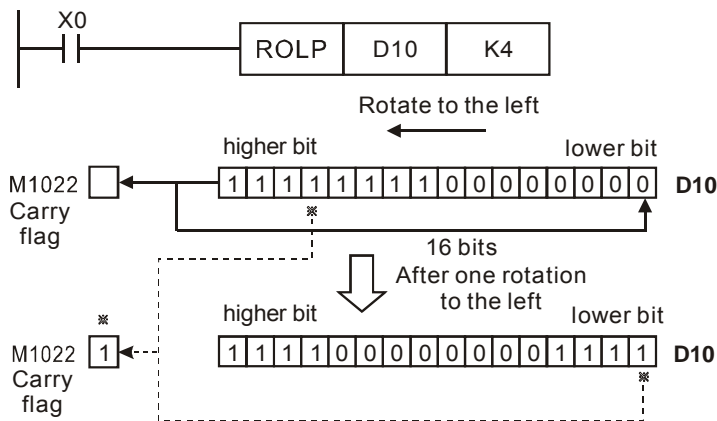
API		Mnemonic		Operands		Function	Controllers
32	D	RCR	P	⬭D ⬭n		Rotation Right with Carry	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RCR, RCRP: 5 steps
D								*	*	*	*	*	*	*	*	DRCR, DRCRP: 9 steps
n					*	*										

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device to be rotated   **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.

3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)

4. See the specifications of each model for their range of use.

5. Flag: M1022 (carry flag)

6. This instruction rotates the device content designated by **D** together with carry flag M1022 to the right for **n** bits.

7. This instruction adopts pulse execution instructions (RCRP, DRCRP).

Program Example:

When X0 = Off→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 sent to carry flag M1022.
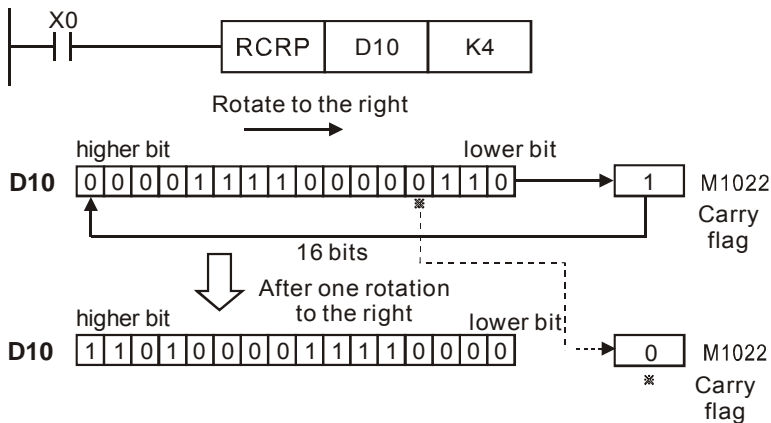
API		Mnemonic		Operands	Function	Controllers
33	D	RCL	P	D n	Rotation Left with Carry	ES/EX/SS SA/SX/SC EH/SV

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							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Device to be rotated   **n**: Number of bits to be rotated in 1 rotation

Explanations:

1.   If **D** is used in device F, only 16-bit instruction is applicable.

2.   If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.

3.   Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)

4.   See the specifications of each model for their range of use.

5.   Flag: M1022 (carry flag)

6.   This instruction rotates the device content designated by **D** together with carry flag M1022 to the left for **n** bits.

7.   This instruction adopts pulse execution instructions (RCLP, DRCLP).

Program Example :

When X0 = Off→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.
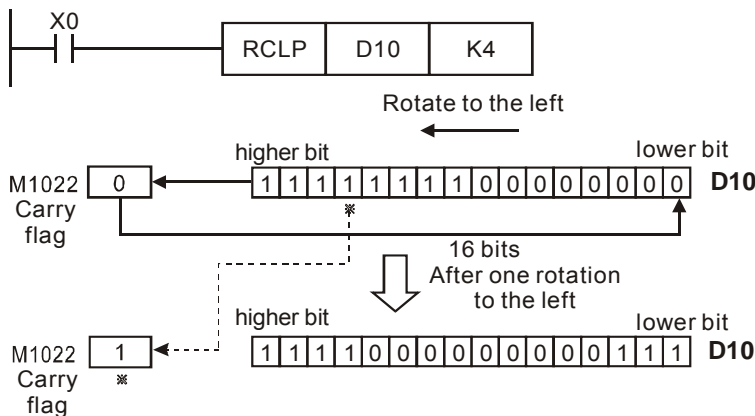
API	Mnemonic		Operands				Function	Controllers
34	SFTR	P	**S**	**D**	**n₁**	**n₂**	Bit Shift Right	ES/EX/SS SA/SX/SC EH/SV

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		*	*	*												
n₁					*	*										
n₂					*	*										

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start No. of the shifted device   **D**: Start No. of the device to be shifted   **n₁**: Length of data to be shifted

n₂: Number of bits to be shifted in 1 shift

Explanations:

1. Range of **n₁**: 1~ 1,024

2. Range of **n₂**: 1 ~ **n₁**

3. In ES/EX/SS, $1 \le n_2 \le n_1 \le 512$

4. ES/EX/SS series MPU does not support E, F index register modification.

5. See the specifications of each model for their range of use.

6. This instruction shifts the bit device of **n₁** bits (desired length for shifted register) starting from **D** to the right for **n₂** bits. **S** is shifted into **D** for **n₂** bits to supplement empty bits.

7. This instruction adopts pulse execution instructions (SFTRP).

Program Example:

1. When X0 = Off→On, M0 ~M15 will form 16 bits and shifts to the right (4 bits as a group).

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
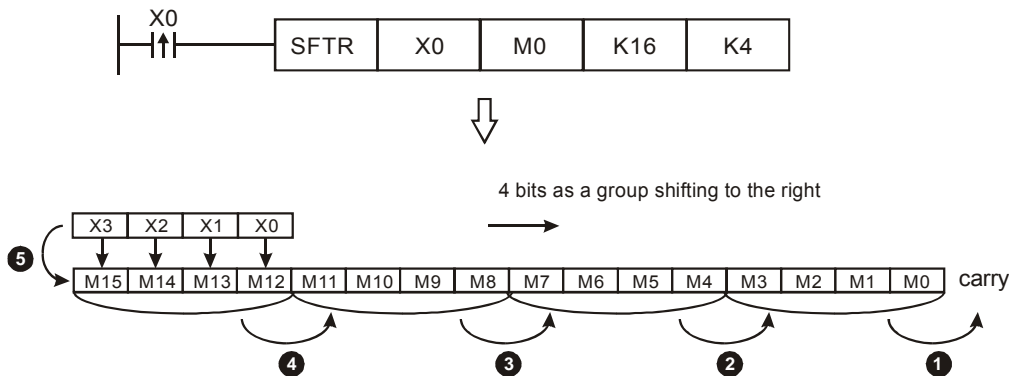
API		Mnemonic		Operands				Function	Controllers
35		SFTL	P	Ⓢ	Ⓓ	Ⓝ₁	Ⓝ₂	Bit Shift Left	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		SFTL, SFTLP: 9 steps
S	*	*	*	*													
D		*	*	*													
n₁					*	*											
n₂					*	*											

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start No. of the shifted device   **D**: Start No. of the device to be shifted   **n₁**: Length of data to be shifted

n₂: Number of bits to be shifted in 1 shift

Explanations:

1. Range of **n₁**: 1~ 1,024

2. Range of **n₂**: 1 ~ **n₁**

3. In ES/EX/SS, $1 \le n_2 \le n_1 \le 512$

4. ES/EX/SS series MPU does not support E, F index register modification.

5. See the specifications of each model for their range of use.

6. This instruction shifts the bit device of **n₁** bits (desired length for shifted register) starting from **D** to the left for **n₂** bits. **S** is shifted into **D** for **n₂** bits to supplement empty bits.

7. This instruction adopts pulse execution instructions (SFTLP).

Program Example:

1. When X0 = Off→On, M0 ~M15 will form 16 bits and shifts to the left (4 bits as a group).

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

4 bits as a group shifting to the left

API	Mnemonic		Operands				Function	Controllers
36	WSFR	P	S D n₁ n₂				Word Shift Left	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices									Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WSFR, WSFRP: 9 steps
S							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n₁					*	*										
n₂					*	*										

	PULSE							16-bit							32-bit						
	ES	EX	SS	SA	SX	SC	EH	ES	EX	SS	SA	SX	SC	EH	ES	EX	SS	SA	SX	SC	EH

Operands:

S: Start No. of the shifted device   **D**: Start No. of the device to be shifted   **n₁**: Length of data to be shifted

n₂: Number of words to be shifted in 1 shift

Explanations:

1. 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.

2. Provided the devices designated by **S** and **D** belong to K_n type, the number of digits of K_n has to be the same.

3. Range of **n₁**: 1~ 512

4. Range of **n₂**: 1 ~ **n₁**

5. See the specifications of each model for their range of use.

6. This instruction shifts the stack data of **n₁** words starting from **D** to the right for **n₂** words. **S** is shifted into **D** for **n₂** words to supplement empty words.

7. This instruction adopts pulse execution instructions (WSFRP)

Program Example 1:

1. When X0 = Off→On, the 16 register stack data composed of D20 ~ D35 will shift to the right for 4 registers.

2. The figure below illustrates the right shift of the words 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
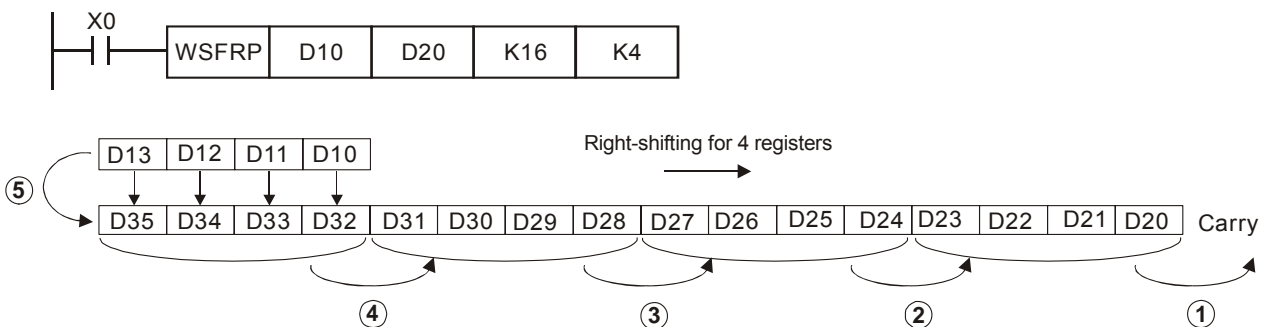
Program Example 2:

1.   When X0 = Off→On, the bit register stack data composed of Y10 ~ Y27 will shift to the right for 2 digits.

2.   The figure below illustrates the right shift of the words in one scan.

❶ Y17 ~ Y10 → carry

❷ Y27 ~ Y20 → Y17 ~ Y10

❸ X27 ~ X20 → Y27 ~ Y20   completed
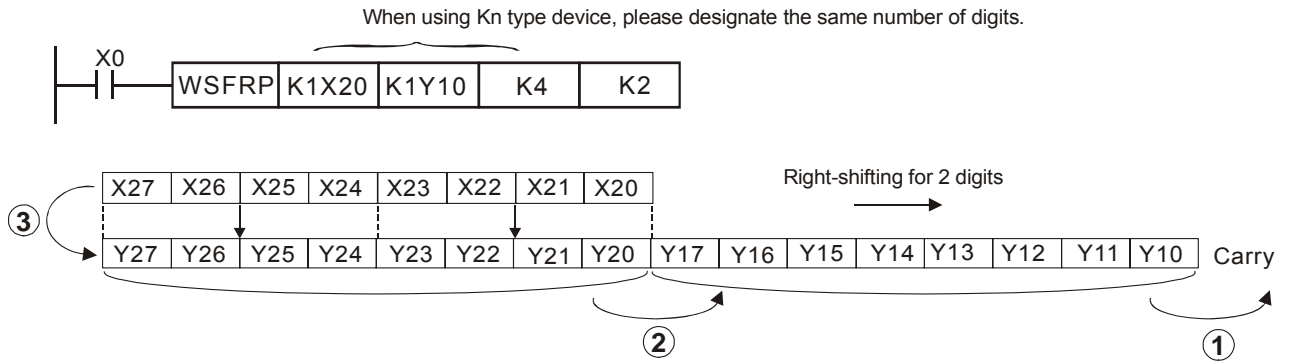
API	Mnemonic		Operands	Function	Controllers
37	WSFL	P	Ⓢ Ⓓ (n₁) (n₂)	Word Shift Left	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	WSFL, WSFLP: 9 steps
S							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n₁					*	*										
n₂					*	*										

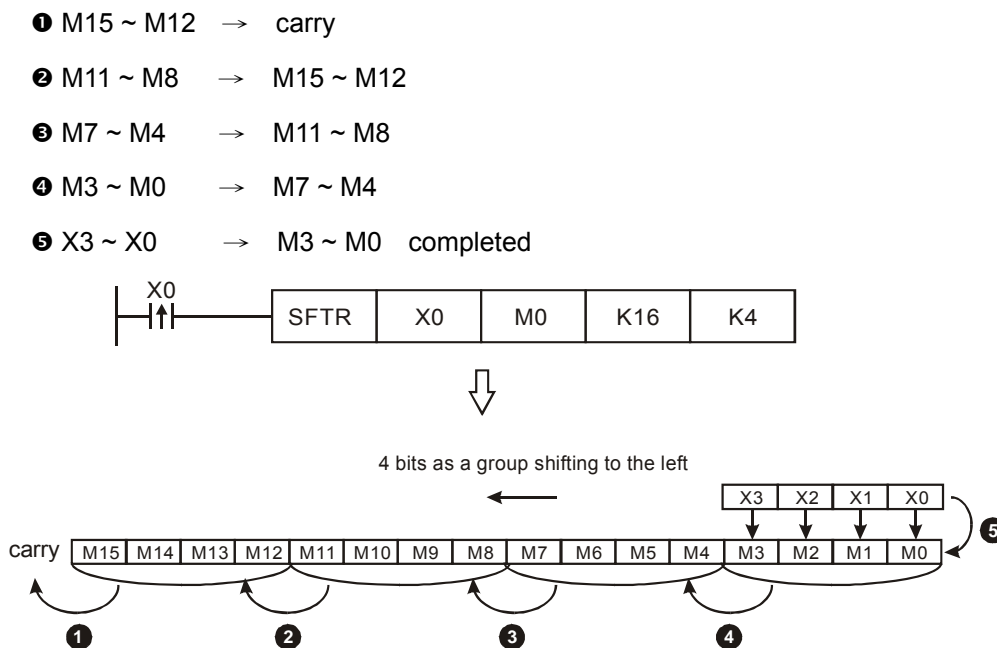|	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start No. of the shifted device **D**: Start No. of the device to be shifted **n₁**: Length of data to be shifted

n₂: Number of words to be shifted in 1 shift

Explanations:

1. 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.

2. Provided the devices designated by **S** and **D** belong to K_n type, the number of digits of K_n has to be the same.

3. Range of **n₁**: 1~ 512

4. Range of **n₂**: 1 ~ **n₁**

5. See the specifications of each model for their range of use.

6. This instruction shifts the stack data of **n₁** words starting from **D** to the left for **n₂** words. **S** is shifted into **D** for **n₂** words to supplement empty words.

7. This instruction adopts pulse execution instructions (WSFLP)

Program Example:

1. When X0 = Off→On, the 16 register stack data composed of D20 ~ D35 will shift to the left for 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
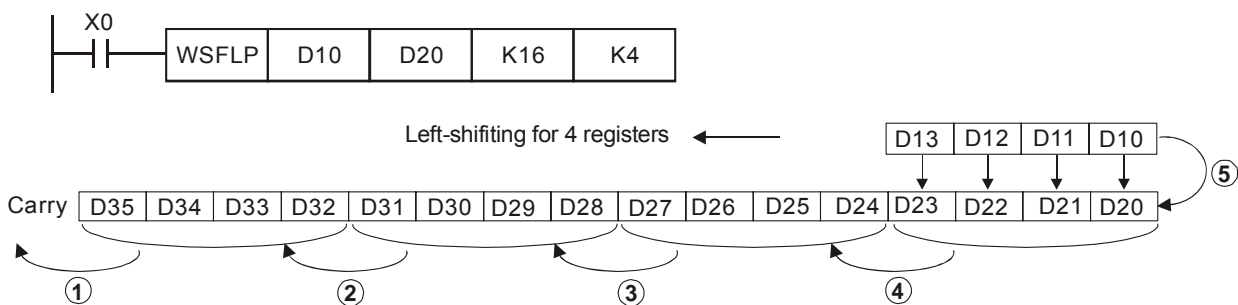
API		Mnemonic		Operands			Function	Controllers
38		SFWR	P	Ⓢ Ⓓ ⓝ			Shift Register Write	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Device of stack data written in   **D**: Start No. of stack data   **n**: Length of stack data

Explanations:

1. Range of **n**: 2 ~ 512

2. See the specifications of each model for their range of use.

3. Flag: M1022 (carry flag)

4. The stack data of **n** words starting from **D** are defined as "first-in, first-out" stack data and designate the first device as the pointer. When the instruction is executed, the content in the pointer pluses 1, and the content in the device designated by **S** will be written into the designated location in the "first-in, first-out" stack data designated by the pointer. When the content in the pointer exceeds **n** - 1, this instruction will not process any new value written in and the carry flag M1022 = On.

5. This instruction adopts pulse execution instructions (SFWRP)

Program Example:

1. Pointer D0 is reset as 0. When X0 = Off→On, the content in D20 will be sent to D1 and the content in pointer D0 becomes 1. After the content in D20 is changed, make X0 = Off→On again, and the content in D2 will be sent to D2 and the content in D0 becomes 2.

2. The figure below illustrates the shift and writing in 1~2 execution of the instruction.

    ❶ The content in D20 is sent to D1.

    ❷ The content in pointer D0 becomes 1.

X10			
⊢⊢	RST	D0	Reset the content of D0 as 0 in advance

X0				
⊢↑⊢	SFWRP	D20	D0	K10

```
                      n = 10 points
┌─────┐   ┌────┬────┬────┬────┬────┬────┬────┬────┬────┬────┐
│ D20 │   │ D9 │ D8 │ D7 │ D6 │ D5 │ D4 │ D3 │ D2 │ D1 │ D0 │
└─────┘   └────┴────┴────┴────┴────┴────┴────┴────┴────┴────┘
                                        ↑    ↑    ↑   pointer
                                 D0 =   3    2    1
```
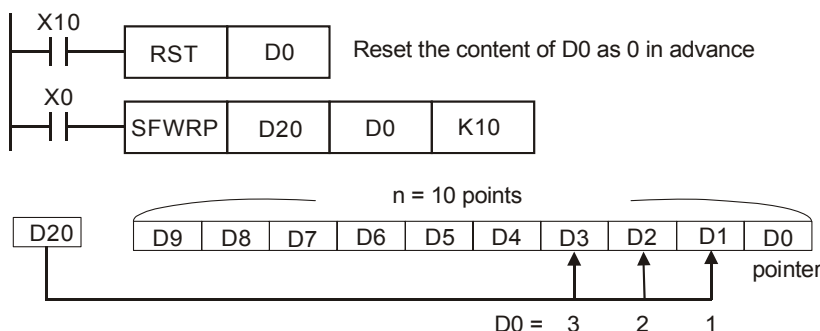
Remarks:

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	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices													Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start No. of stack data   **D**: Device of stack data read out   **n**: Length of stack data

Explanations:

1. Range of **n**: 2 ~ 512

2. See the specifications of each model for their range of use.

3. Flag: M1020 (zero flag)

4. The stack data of **n** words starting from **S** are defined as "first-in, first-out" stack data and designate the first device as the pointer. When the instruction is executed, the content in the pointer minuses 1, and the content in the device designated by **S** will be written into the designated location in the "first-in, first-out" stack data designated by the pointer. When the content in the pointer equals 0, this instruction will not process any new value written in and the zero flag M1020 = On.

5. This instruction adopts pulse execution instructions (SFRDP)

Program Example:

1. When X0 = Off→On, the content in D1 will be sent to D21 and D9~D2 will shift to the right for 1 register (content in D9 remains unchanged) and the content in D0 minus 1.

2. The figure below illustrates the shift and reading in 1~3 execution of the instruction.

   ❶ The content in D1 is sent to D21.

   ❷ D9 ~ D2 shift to the right for 1 register.

   ❸ The content in D0 minuses 1.

```
  X0
──┤├──┬──────┬──────┬──────┬──────┐
      │ SFRDP│  D0  │ D21  │ K10  │
      └──────┴──────┴──────┴──────┘
```

n = 10 points

| D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | | D21 |

pointer

data read
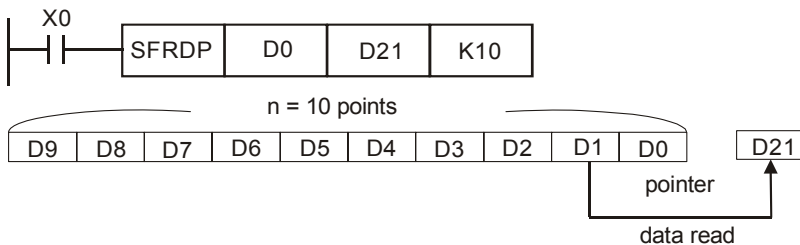
Remarks:

This instruction can be used together with API 38 SFWR for the reading/writing of "first-in, first-out" stack data.

API		Mnemonic		Operands		Function		Controllers		
40		ZRST	P	D_1 D_2		Zero Reset		ES/EX/SS	SA/SX/SC	EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D_1: Start device of the range to be reset   **D_2**: End device of the range to be reset

Explanations:

1.   No. of operand $D_1 \leq$ No. of operand D_2.

2. D_1 and D_2 have to designate devices of the same type.

3.   ES/EX/SS series MPU does not support E, F index register modification.

4.   See the specifications of each model for their range of use.

5.   When the instruction is executed, area from **D_1** to **D_2** will be cleared.

6.   In ES/EX/SS, 16-bit counter and 32-bit counter cannot use ZRST instruction together.

7.   In SA/EH, 16-bit counter and 32-bit counter can use ZRST instruction together.

8.   When **D_1** > **D_2**, only operands designated by **D_2** will be reset.

Program Example:

1.   When X0 = On, auxiliary relays M300 ~ M399 will be reset to Off.

2.   When X1 = On, 16 counters C0 ~ C127 will all be reset (writing in 0; contact and coil being reset to Off).

3.   When X10 = On, timers T0 ~ T127 will all be reset (writing in 0; contact and coil being reset to Off).

4.   When X2 = On, steps S0 ~ S127 will be reset to Off.

5.   When X3 = On, data registers D0 ~ D100 will be reset to 0.

6.   When X4 = On, 32-bit counters C235 ~ C254 will all be reset. (writing in 0; contact and coil being reset to Off)

```
  X0
  ─┤├──────   ZRST │ M300 │ M399
  X1
  ─┤├──────   ZRST │ C0   │ C127
  X10
  ─┤├──────   ZRST │ T0   │ T127
  X2
  ─┤├──────   ZRST │ S0   │ S127
  X3
  ─┤├──────   ZRST │ D0   │ D100
  X4
  ─┤├──────   ZRST │ C235 │ C254
```
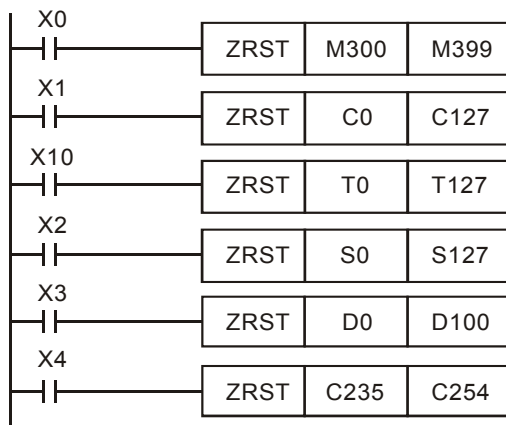
Remarks:

1.   Devices, e.g. bit devices Y, M, S and word devices T, C, D, can use RST instruction.

2.   API 16 FMOV instruction is also to send K0 to word devices T, C, D or bit registers KnY, KnM, KnS for reset.

```
   X0
   ┤├──────┌─────┬─────┐
   │       │ RST │ M0  │
   │       └─────┴─────┘
   │       ┌─────┬─────┐
   │       │ RST │ T0  │
   │       └─────┴─────┘
   │       ┌─────┬─────┐
   │       │ RST │ Y0  │
   │       └─────┴─────┘
   │       ┌──────┬────┬─────┬────┐
   │       │ FMOV │ K0 │ D10 │ K5 │
   │       └──────┴────┴─────┴────┘
```

API	Mnemonic		Operands	Function	Controllers
41	DECO	P	Ⓢ Ⓓ Ⓝ	Decode	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices										Program Steps
OP	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
	ES EX SS **SA SX SC EH SV**	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device to be decoded   **D**: Device for storing the decoded result   **n**: Length of decoded bits

Explanations:

1. Range of **n** when **D** is a bit device: 1 ~ 8

2. Range of **n** when **D** is a word device: 1 ~ 4

3. ES/EX/SS series MPU does not support E, F index register modification.

4. See the specifications of each model for their range of use.

5. The lower "**n**" bits of **S** are decoded and the results of "2^n" bits are stored in **D**.

6. This instruction adopts pulse execution instructions (DECOP)

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. When **n** = 8, the maximum points to decode is 2^8 = 256 points. (Please be aware of the storage range of the devices after the decoding and do not use the devices repeatedly.)

3. When X10 = Off→On, this instruction will decode the content in X0 ~ X2 to M100 ~ M107.

4. When the source of data is 1 + 2 = 3, set M103, the 3rd bit starting from M100, as 1.

5. After the execution of this instruction is completed and X10 turns to Off, the content that has been decoded and output keeps acting.
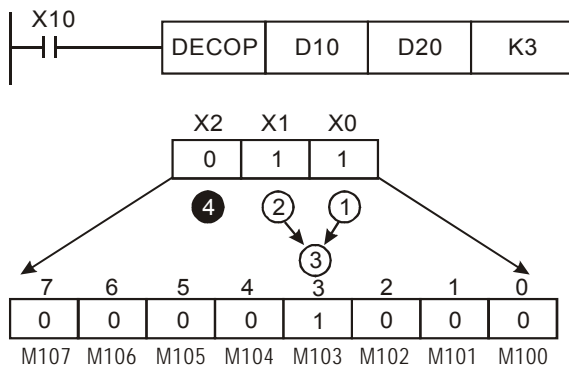
Program Example 2:

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 maximum points to decode is 2^4 = 16 points.

3. When X10 = Off→On, this instruction will decode b2 ~ b0 in D10 to b7 ~ b0 in D20. b15 ~ b8 that have not been used in D20 will all become 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 of this instruction is completed and X10 turns to Off, the content that has been decoded and output keeps acting.
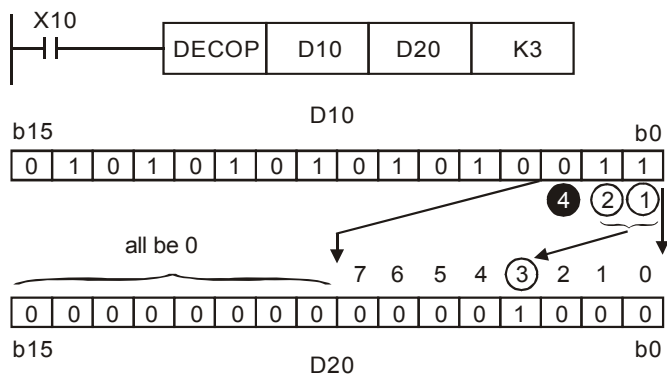
API		Mnemonic		Operands	Function		Controllers
42		ENCO	P	$\underset{\text{S}}{\bigcirc}$ $\underset{\text{D}}{\bigcirc}$ $\underset{\text{n}}{\bigcirc}$	Encode		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ENCO, ENCOP: 7 steps
S	*	*	*	*							*	*	*	*	*	
D											*	*	*	*	*	
n					*	*										

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

**Operands:**

S: Source device to be encoded   **D**: Device for storing the encoded result   **n**: Length of encoded bits

Explanations:

1.   Range of **n** when **S** is a bit device: 1 ~ 8

2.   Range of **n** when **S** is a word device: 1 ~ 4

3.   ES/EX/SS series MPU does not support E, F index register modification.

4.   See the specifications of each model for their range of use.

5.   The lower "2^n" bits of **S** are encoded and the result is stored in **D**.

6.   If several bits of **S** are 1, the first bit that is 1 will be processed orderly from high bit to low bit.

7.   If no bits of S is 1, M1067, M1068 = On and D1067 records the error code 0E1A (hex).

8.   This instruction adopts pulse execution instructions (ENCOP)

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.   When **n** = 8, the maximum points to encode is 2^8 = 256 points.

3.   When X10 = Off→On, this instruction will encode the 2^3 bits data (M0 ~ M7) and store the result in the lower 3 bits (b2 ~ b0) of D0. b15 ~ b3 that have not been used in D0 will all become 0.

4.   After the execution of this instruction is completed and X10 turns to Off, the content in D remains unchanged.
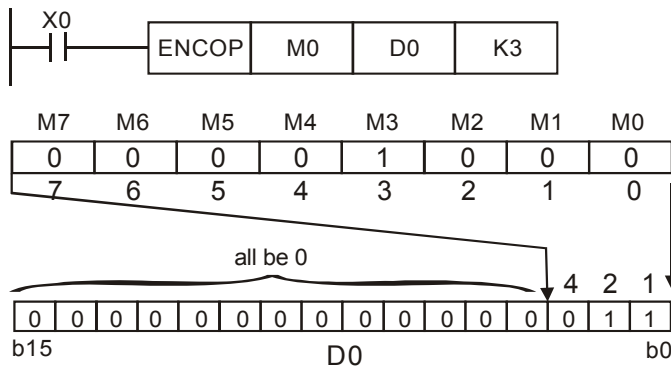
Program Example 2:

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 maximum points to decode is 2^4 = 16 points.

3.   When X10 = Off→On, this instruction will encode 2^3 bits (b0 ~ b7) in D10 and stores the result in the lower 3 bits (b2 ~ b0) of D20. b15 ~ b3 that have not been used in D20 will all become 0. b8 ~ b15 of D10 are invalid data.

4.   After the execution of this instruction is completed and X10 turns to Off, the content in D remains unchanged.
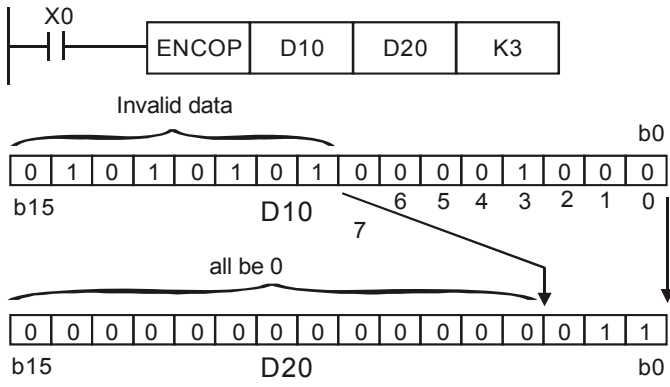
API		Mnemonic		Operands		Function		Controllers
43	D	SUM	P	Ⓢ Ⓓ		Sum of Active Bits		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SUM, SUMP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	DSUM, DSUMP: 9 steps
D								*	*	*	*	*	*	*	*	

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source device   **D**: Destination device for storing counted value

Explanations:

1.   If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2.   See the specifications of each model for their range of use.

3.   Flag: M1020 (zero flag)

4.   Among the bits of **S**, the total of bits whose content is "1" will be stored in **D.**

5.   When all the 16 bits of **S** are "0", zero flag M1020 = On.

6.   When 32- instruction is in use, **D** will occupy 2 registers.

Program Example:

When X10 = On, among the 16 bits of D0, the total of bits whose content is "1" will be stored in D2.

X10		
─┤├─	SUM	D0	D2

0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0		K3

D0   **D2**

API		Mnemonic		Operands			Function	Controllers
44	D	BON	P	S D n			Check Specified Bit Status	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	BON, BONP: 7 steps
S					*	*	*	*	*	*	*	*	*	*	*	DBON, DBONP: 13 steps
D		*	*	*												
n					*	*					*	*	*	*	*	

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source device   **D**: Device for storing check result   **n**: Bits specified for check

Explanations:

1. If **S** is used in device F, only 16-bit instruction is applicable.

2. Range of **n**: 0 ~ 15 (16-bit instruction); 0 ~ 31 (32-bit instruction)

3. See the specifications of each model for their range of use.

4. When the **n**th bit of **S** is "1", D = On; when the **n**th bit of **S** is "0", D = Off.

Program Example:

1. When X0 = On, assume the 15th bit of D0 is "1", and M0 = On. Assume the 15th bit of D0 is "0", and M0 = Off.

2. When X0 goes Off, M0 will remains in its previous status.

```
 X0
─┤├─────┬──────┬──────┬──────┬──────┐
         │ BON  │  D0  │  M0  │ K15  │
         └──────┴──────┴──────┴──────┘
```

```
 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
```
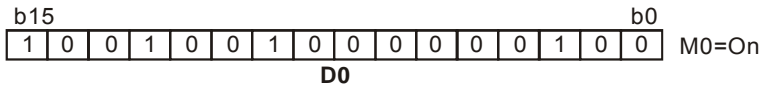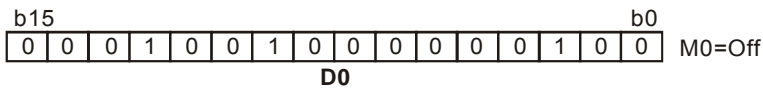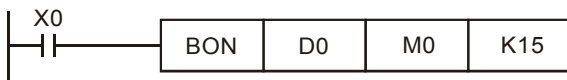
API		Mnemonic		Operands			Function	Controllers
45	D	MEAN	P	Ⓢ Ⓓ ⓝ			Mean	ES/EX/SS SA/SX/SC EH/SV

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 steps
D								*	*	*	*	*	*	*	*	
n					*	*	*	*	*	*	*	*	*	*	*

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device to obtain mean value   **D**: Destination device for storing mean value   **n**: The number of consecutive source devices used

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. Range of **n**: 1 ~ 64

3. In ES/EX/SS series models: Operand S cannot designate KnX, KnY, KnM, KnS.

4. ES/EX/SS series MPU does not support E, F index register modification.

5. See the specifications of each model for their range of use.

6. After the content of **n** devices starting from **S** are added up, the mean value of the result will be stored in **D**.

7. Remainders in the operation will be left out.

8. Provided the No. of designated device exceeds its normal range, only the No. within the normal range can be processed.

9. If **n** falls without the range of 1 ~ 64, PLC will determine it as an "instruction operation error".

Program Example:

When X10 = On, the contents in 3 (n = 3) registers starting from D0 will be summed and then divided by 3. The obtained mean value will be stored in D10 and the remainder will be left out.


```
   X10
────┤├────┤ MEAN │ D0 │ D10 │ K3 │

   (D0+D1+D2)/3 ───→ D10
```

```
 D0  │ K100 │ ⎫
                ⎬
 D1  │ K113 │ ⎬ ───→  D10 │ K112 │
                ⎬       Remainder = 3, left out
 D2  │ K125 │ ⎭
```
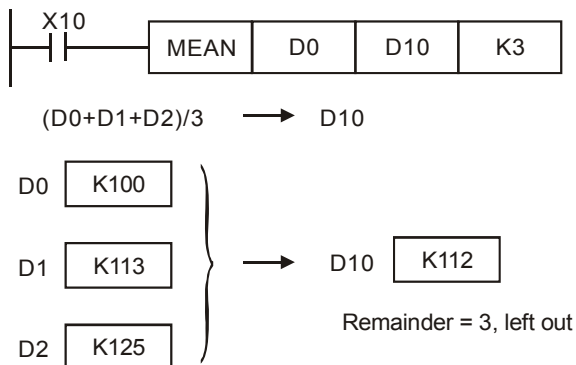
API	Mnemonic	Operands			Function	Controllers
46	ANS	Ⓢ	ⓜ	Ⓓ	Timed Annunciator Set	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Timer for monitoring annunciator   **m**: Time setting   **D**: Annunciator device

Explanations:

1. Range of **S**: for SA/SX/SC T0 ~ T191; for EH/EH2/SV T0 ~ T199.

2. **m** can designate K1 ~ K32,767 (unit: 100ms)

3. Range of **D**: for SA/SX/SC S896 ~ S1023; for EH/EH2/SV S900 ~ S1023.

4. See the specifications of each model for their range of use.

5. Flags: M1048 (annunciator in action); M1049 (valid monitoring)

6. This instruction is used for enabling the annunciator.

Program Example:

If X3 = On for more than 5 seconds, annuniciator point S999 = On. Even X3 goes Off afterwards, S999 will still keep On. However, T10 will be reset to Off and the present value = 0.

X3	ANS	T10	K50	S999
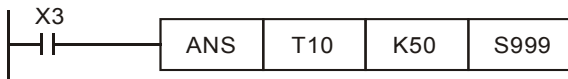
API	Mnemonic		Operands	Function	Controllers
47	ANR	P		Annunciator Reset	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A		ANR, ANRP: 1 steps

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Explanations:

1. No operand.

2. This instruction is used for resetting the annunciator.

3. When more than one annuciators are On, the annunciator of smaller number will be reset.

4. This instruction adopts pulse execution instructions (ANRP)

Program Example:

1. If X10 and X11 = On at the same time for more than 2 seconds, annuniciator point S910 = On. Even X10 and X11 go Off afterwards, S910 will still keep On. However, T10 will be reset to Off and the present value = 0.

2. When X10 and X11 are On at the same time for less than 2 seconds, the present value of T10 will be reset to 0.

3. When X3 goes from Off to On,

   S896 ~ S1023 in SA/SX/SX are able to reset the annunciators in action.

   S900 ~ S1023 in EH/EH2/SV are able to reset the annunciators in action.

4. When X3 goes from Off to On again, the annuniator with secondary smaller No. will be reset.

```
 X10   X11
──┤├────┤├────┤ ANS │ T10 │ K20 │ S910 │

 X3
──┤├──────────┤ ANRP │
```
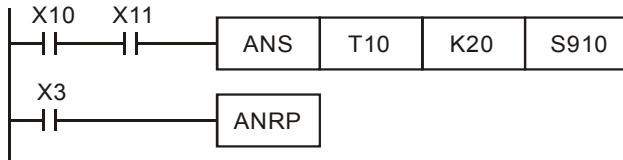
Remarks:

1. Flag:

   a) M1048 (annunciator in action): When M1049 = On, any of the annunciators among S896 ~ S1023 in SA/SX/SC or S900 ~ S1023 in EH/EH2/SV starts output, M1048 will be On.

   b) M1049 (valid monitoring): When M1049 = On, D1049 will automatically display the annuciator of the smallest number in action.

2. Application of annunciators:

   I/O point configuration:

X0: Forward switch	Y0: Forward	S910: Forward annunciator
X1: Backward switch	Y1: Backward	S920: Backward annunciator
X2: Front position switch	Y2: Annunciator indicator	
X3: Back position switch		
X4: Annunciatro reset button		

```
 M1000
──┤├──────────────────( M1049 )

 Y0    X2
──┤├────┤/├───┤ ANS │ T0 │ K100 │ S910 │

 Y1    X3
──┤├────┤/├───┤ ANS │ T1 │ K200 │ S920 │

 X0    X2
──┤├──┬─┤/├───────────( Y0 )
 Y0  │
──┤├──┘

 X1    X3
──┤├──┬─┤/├───────────( Y1 )
 Y1  │
──┤├──┘

 M1048
──┤├──────────────────( Y2 )

 X4
──┤├──────────┤ ANRP │
```
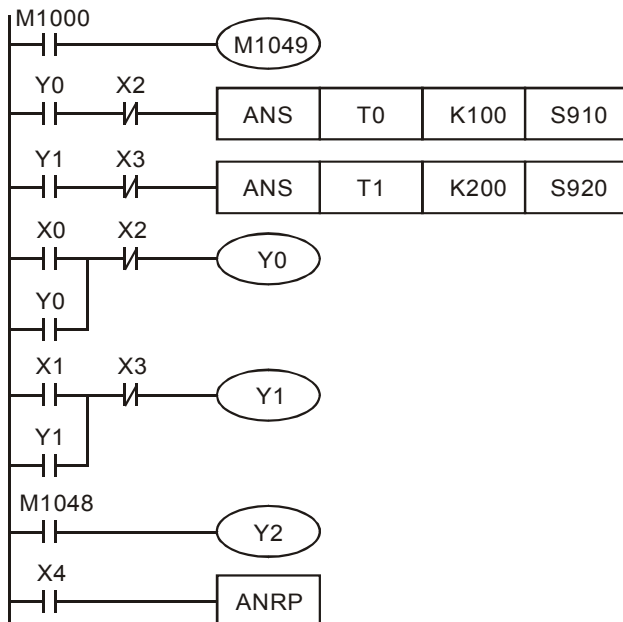
1. M1048 and D1049 are valid only when M1049 = On.

2. When Y0 = On for more than 10 seconds and the device fails to reach the frong position X2, S910= On.

3. When Y1 = On for more than 10 seconds and the device fails to reach the back position X3, S920= On.

4. When backward switch X1 = On and backward device Y1 = On, Y1 will go Off only when the device reaches the back position switch X3.

5. Y2 will be On when any annunciator is enabled. Whenever X4 is on, 1 annunciator in action will be reset. The reset starts from the annunciator with the smallest No.

API		Mnemonic		Operands		Function		Controllers
48	D	SQR	P	Ⓢ Ⓓ		Square Root		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device   **D**: Device for storing the result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1067 (instruction operation error)

3. This instruction performs a square root operation on **S** and stores the result in **D**.

4. **S** can only be a positive value. If **S** is negative, PLC will regard it as an "instruction operation error" and will not execute this instruction. M1067 and M1068 = On and D1067 records the error code 0E1B (hex).

5. The operation result **D** should be integer only, and the decimal will be left out. Borrow flag M1021 = On.

6. When the operation result **D** = 0, zero flag M1020 = On.

Program Example:

When X10 = On, the instruction performs a square root on D0 and stores the result in D12.

```
   X10
 ──┤ ├──        ┌──────┬──────┬──────┐
                │ SQR  │  D0  │ D12  │
                └──────┴──────┴──────┘

   √D0  ──▶  D12
```

API	Mnemonic			Operands		Function		Controllers	
49	D	FLT	P	S	D	Convert BIN integer to binary floating point		ES/EX/SS	SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps	
OP	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													*				

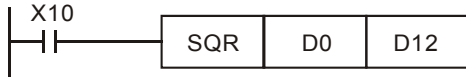| PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source device for conversion   **D**: Device for storing the conversion result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1081 (FLT instruction function switch); M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. **D** will occupy 2 consecutive devices

4. When M1081 is Off, BIN integer is converted into binary floating point value. At this time, **S** of the 16-bit instruction, FLT, occupies 1 register and **D** occupies 2 registers.

   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 the conversion result is 0, zero flag M1020 = On.

5. When M1081 is On, binary floating point value is converted into BIN integer (digits after decimal point are left out). At this time, **S** of the 16-bit instruction, FLT, occupies 2 registers and **D** occupies 1 register (action same as that of INT instruction).

   a) If the conversion result exceeds the range of BIN integer available 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 any digits is left out during the conversion, borrow flag M1021 = On.

   c) If **S** = 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 X10 = On, D0 (BIN integer) is converted into D13 and D12 (binary floating point value).

3. When X11 = On, D1 and D0 (BIN integer) are converted into D21 and D20 (binary floating point value).

4. If D0 = K10, X10 will be On. The 32-bit value of the converted floating point will be H41200000 and stored in 32-bit register D12 (D13).

5. If 32-bit register D0 (D1) = K100,000, X11 will be On. The 32-bit value of the converted floating point will be H47C35000 and stored in 32-bit register D20 (D21).

```
M1002
 ┤├        | RST  | M1081 |

 X10
 ┤├        | FLT  | D0 | D12 |

 X11
 ┤├        | DFLT | D0 | D20 |
```
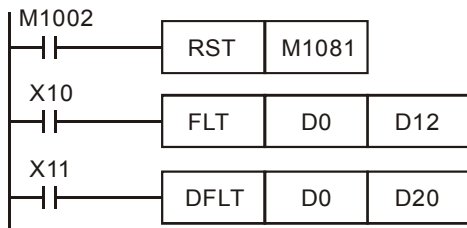
Program Example 2:

1. When M1081 = On, the binary floating point value is converted into BIN integer (the decimal is left out).

2. When X10 = On, D0 and D1 (binary floating point value) are converted into D12 (BIN integer). If D0 (D1) = H47C35000, the floating point value will be presented as 100,000. Due to that the value is larger than the value presentable by the 16-bit register D12, the result will be D12 = K32, 767 and M1022 = On.

3. When X11 = On, D1 and D0 (binary floating point value) are converted into D21 and D20 (BIN integer). If D0 (D1) = H47C35000, the floating point value will be presented as 100,000. The result will be stored in the 32-bit register D20 (D21).

```
M1002
 ┤├        | SET  | M1081 |

 X10
 ┤├        | FLT  | D0 | D12 |

 X11
 ┤├        | DFLT | D0 | D20 |
```
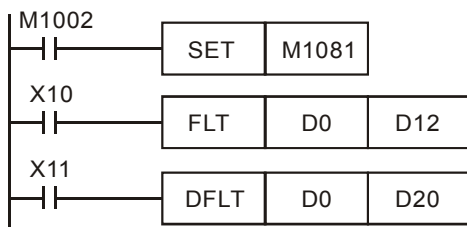
Program Example 3:

Please use this instruction to complete the following operation.

```
     M1000
     ┤ ├───┬──────┌──────┬──────┬──────┐
          ①      │ FLT  │ D10  │ D100 │
                 └──────┴──────┴──────┘
          ②      ┌──────┬──────┬──────┐
                 │ BIN  │ K2X0 │ D200 │
                 └──────┴──────┴──────┘
          ③      ┌──────┬──────┬──────┐
                 │ FLT  │ D200 │ D202 │
                 └──────┴──────┴──────┘
          ④      ┌──────┬──────┬──────┬──────┐
                 │ DEDIV│ K615 │ K10  │ D300 │
                 └──────┴──────┴──────┴──────┘
          ⑤      ┌──────┬──────┬──────┬──────┐
                 │ DEDIV│ D100 │ D202 │ D400 │
                 └──────┴──────┴──────┴──────┘
          ⑥      ┌──────┬──────┬──────┬──────┐
                 │ DEMUL│ D400 │ D300 │ D20  │
                 └──────┴──────┴──────┴──────┘
          ⑦      ┌──────┬──────┬──────┐
                 │ DEBCD│ D20  │ D30  │
                 └──────┴──────┴──────┘
          ⑧      ┌──────┬──────┬──────┐
                 │ DINT │ D20  │ D40  │
                 └──────┴──────┴──────┘
```
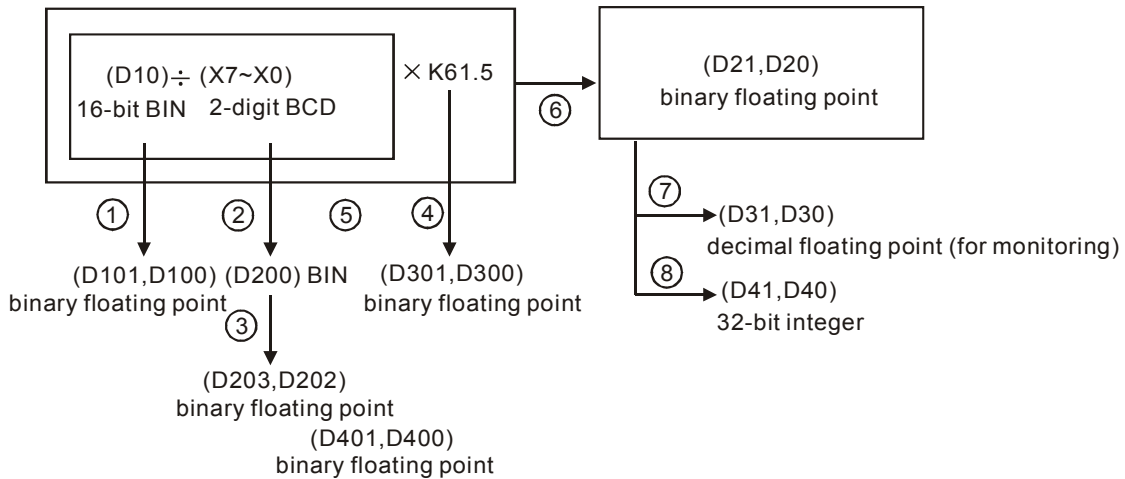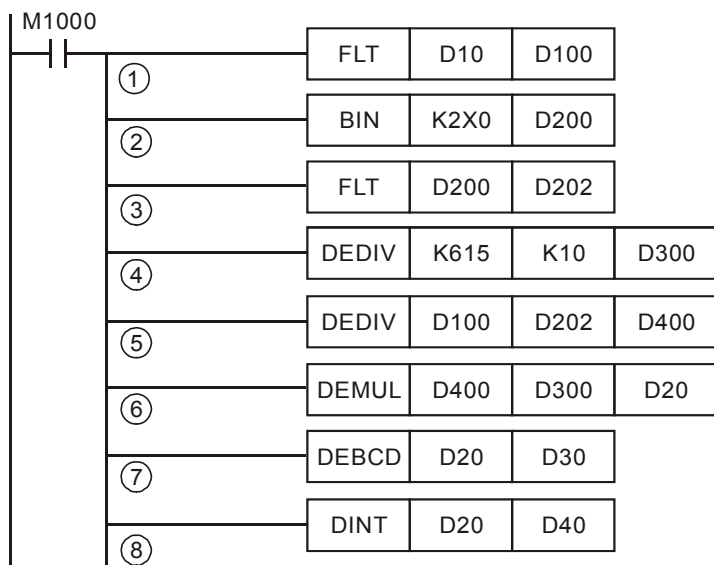
① D10 (BIN integer) is converted to D101 and D102 (binary floating point value).

② X7 ~ X0 (BCD value) are converted to D200 (BIN value).

③ D200 (BIN integer) is converted to D203 and D202 (binary floating point value).

   4The result of K615 ÷ K10 is stored in D301 and D300 (binary floating point value).

⑤ The result of binary decimal division (D101, D100) ÷ (D203, D202) is stored in D401 and D400 (binary floating point value).

⑥ The result of binary decimal multiplication (D401, D400) × (D301, D300) is stored in D21 and D20 (binary floating point value).

⑦ D21 and D20 (binary floating point value) are converted to D31 and D30 (decimal floating point value).

⑧ D21 and D20 (binary floating point value) are converted to D41 and D40 (BIN integer).

API		Mnemonic		Operands		Function		Controllers
50		REF	P	Ⓓ ⓝ		Refresh		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		REF, REFP: 5 steps
D	*	*															
n					*	*											

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Start device to be I/O refreshed   **n**: Number of items to be I/O refreshed

Explanations:

1. **D** must designate X0, X10, Y0, Y10…the points whose 1s digit is "0". See remarks for more details.

2. Range of **n**: 8 ~ 256 (has to be the multiple of 8).

3. See the specifications of each model for their range of use.

4. The status of all PLC input/output terminals will be updated after the program scans to END. When the program starts to scan, the status of the external input terminal is read and stored into the memory of the input point. The output terminal will send the content in the output memory to the output device after END instruction is executed. Therefore, this instruction is applicable when the latest input/output data are needed for the operation.

Program Example 1:

When X0 = On, PLC will read the status of input points X0 ~ X17 immediately and refresh the input signals without any input delay.

```
   X0
───┤├─────┬──────┬──────┬──────┐
          │ REF  │  X0  │ K16  │
          └──────┴──────┴──────┘
```

Program Example 2:

When X0 = On, the 8 output signal from Y0 ~ Y7 will be sent to output terminals and refreshed without having to wait for the END instruction for output.

```
   X0
───┤├─────┬──────┬──────┬──────┐
          │ REF  │  Y0  │  K8  │
          └──────┴──────┴──────┘
```

Remarks:

The instruction only process the I/O points X0 ~ X17 and Y0 ~ Y17 of ES/EX/SS/SA/SX/SC series MPU, namely **n** = K8 or **n** = K16.

API		Mnemonic		Operands	Function		Controllers	
51		REFF	P	**n**	Refresh and Filter Adjust		ES/EX/SS	SA/SX/SC EH/SV

Type	Bit Devices				Word Devices										Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

n: Response time (unit: ms)

Explanation:

1. Range of **n**: for SA/SX/SC, **n** = K0 ~ K20; for EH/EH2/SV, **n** = K0 ~ K60.

2. To avoid interferences, X0 ~ X17 of EH/EH2/SV series MPU and X0 ~ X7 of SA/SX/SC series MPU are equipped with digital filters on output terminals. Digital filters adjust the response time by REFF instruction. This instruction sets up **n** directly in D1020 (adjusting the response time of X0 ~ X7) and D1021 (adjusting the response time of X10 ~ X17).

3. Rules for adjusting the reponse time of the filter at X0 ~ X17:

   a) When the power of PLC turns from Off to On or the END instruction is being executed, the response time will be determined upon the contents in D1020 and D1021.

   b) You can use MOV instruction in the program to move the time values to D1020 and D1021 and make adjustments in the next scan.

   c) You can use REFF instruction to change the response time during the execution of the program. The changed response time will be move to D1020 and D1021 and you can make adjustments in the next scan.

Program Example:

1. When the power of PLC turns from Off to On, the response time of X0 ~ X17 will be determined by the contents in D1020 and D1021.

2. When X20 = On, REFF K5 will be executed and the response time will be changed to 5ms for the adjustment in the next scan.

3. When X20 = Off, the REFF K20 will be executed and the response time will be changed to 20ms for the adjustment in the next scan.
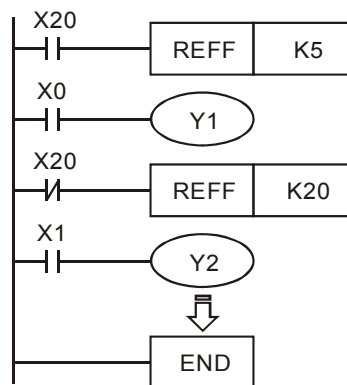
Remarks:

When inserting an interruption subroutine in the program or using the high speed counter or API 56 SPD instruction, the corresponding signals at the input terminals will not delay and has nothing to do with this instruction.

API	Mnemonic	Operands	Function	Controllers
52	MTR	S D₁ D₂ n	Input Matrix	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MTR: 9 steps
S	*															
D₁		*														
D₂		*	*	*												
n					*	*										

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device of matrix input   **D₁**: Start device of matrix output   **D₂**: Corresponding start device for matrix scan

n: Number of arrays in matrix scan

Explanations:

1. **S** must designate X0, X10…the X points whose 1st digit is "0" and occupies 8 consecutive points.

2. **D₁** must designate Y0, Y10…the Y points whose 1st digit is "0" and occupies n consecutive points.

3. **D₂** must designate Y0, M0. S0…the Y, M, S points whose 1st digit is "0".

4. Range of **n**: 2 ~ 8.

5. See the specifications of each model for their range of use.

6. Flag: M1029 (execution of the instruction is completed).

7. **S** is the start device No. of all input terminals connected to the matrix. Once **S** is designated, the 8 points following the No. will be the input terminals in the matrix.

8. **D₁** designate the start device No. of transistor output Y in the matrix scan.

9. This instruction occupies continuous 8 input devices starting from **S**. **n** external output terminals starting from **D₁** read the 8 switches of **n** arrays by matrix scan, obtaining 8 × **n** multiple-matrix input points. The status of scanned switches will be stored in the devices starting from **D₂**.

10. Maximum 8 input switches can be parallelly connected in 8 arrays and obtaining 64 input points (8 × 8 = 64).

11. When the 8-point 8-array matrix inputs are in use, the reading time of each array is approximately 25ms, totaling the reading of 8 arrays 200ms, i.e. the input signals with On/Off speed of over 200ms are not applicable in a matrix input.

12. The drive contact of this instruction uses normally On contact M1000.

13. Whenever this instruction finishes a matrix scan, M1029 will be On for one scan period.

14. There is no limitation on the number of times using the instruction, but only one instruction can be executed in a period of time.

Program Example:

1. When PLC RUN, MRT instruction will start to be executed. The statuses of the external 2 arrays of 16 switches will be read in order and stored in the internal relays M10 ~ M17, M20 ~ M27.

```
  M1000
───┤├───┤  MTR  │  X40  │  Y40  │  M10  │  K2  │
```
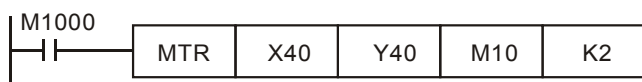
2. The figure below illustrates the external wiring of the 2-array matrix input loop constructed by X40 ~ X47 and Y40 ~ Y41. The 16 switches correponds to the internal relays M10 ~ M17, M20 ~ M27. Should be used with MTR instruction.
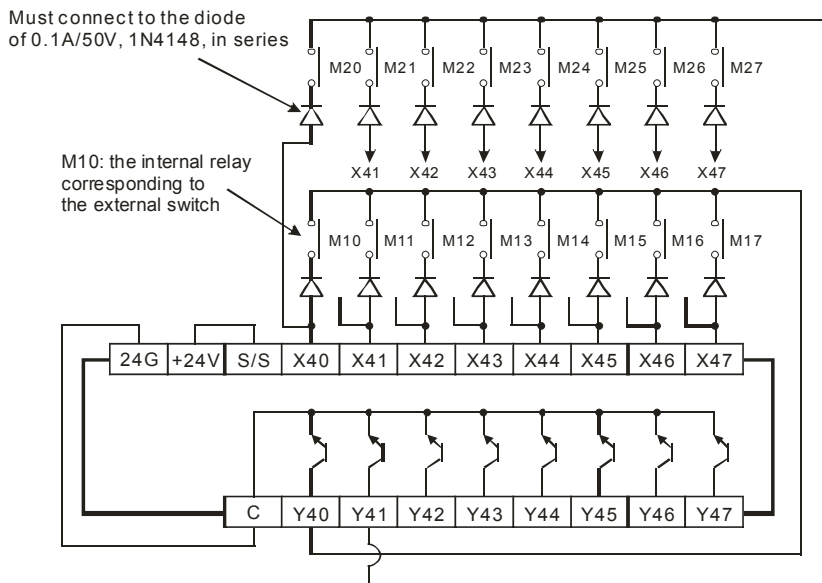
3. See the figure above. The 8 points starting from X40 start to perform a matrix scan from Y40 ~ Y41 (n = 2). D_2 designates that the start device No. of the read results is M10, indicating that the first array is read to M10 ~ M17 and the second array is read to M20 ~ M27.

API	Mnemonic		Operands	Function		Controllers
53	D	HSCS	S_1 S_2 D	High Speed Counter Set		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DHSCS: 13 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2												*				
D		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Comparative value   **S₂**: No. of high speed counter   **D**: Comparison result

Explanations:

1. **S₂** has to designate the No. of high speed counters C235 ~ C255. See remarks for more details.

2. **D** can designate I0☐0; ☐ = 1 ~ 6. ES series MPU does not support this.

3. **D** of ES and SA series MPU does not support E, F index register modification.

4. See the specifications of each model for their range of use.

5. Flags: M1289 ~ M1294 are interruption disability of the high speed counters in EH/EH2/SV series MPU. See Program Example 3 for more details.

6. The high speed counter inputs counting pulses from the corresponding external input terminals X0 ~ X17 by inserting an interruption. When the high speed counter designated in **S₂** pluses 1 or minuses 1, DHSCS instruction will perform a comparison immediately. When the present value in the high speed counter equals the comparative value designated in **S₁**, device designated in **D** will turn On. Even the afterward comparison results are unequal, the device will still be On.

7. If the devices specified as the device **D** are Y0 ~ Y17, when the compare value and the present value of the high-speed counter are equal, the comparison result will immediately output to the external inputs Y0 ~ Y17, and other Y devices will be affected by the scan cycle. However, M, S devices are immediate output and will not be affected by the scan cycle.

Program Example 1:

After PLC RUN and M0 = On, DHSCS instruction will be executed. When the present value in C235 changes from 99 to 100 or 101 to 100, Y10 will be On constantly.

M1000		DCNT	C235	K1000	
M0		DHSCS	K100	C235	Y10	← On immediately
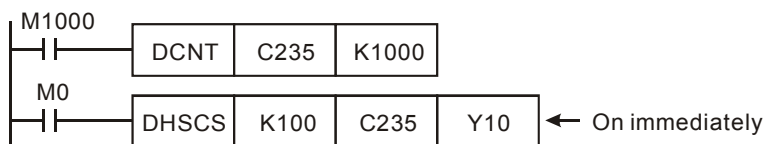
Program Example 2:

Differences between Y output of DHSCS instruction and general Y output:

a) When the present value in C249 changes from 99 to 100 or 101 to 100, Y10 outputs immediately to the external output point by interruption and has nothing to do with the PLC scan time. However, the time will still be delayed by the relay (10ms) or transistor (10us) of the output module.

b) When the present value in C249 changes from 99 to 100, the drive contact of C249 will be On immediately. When the execution arrives at SET Y17, Y17 will still be affected by the scan time and will output after END instruction.

```
M1000
──┤├──┬──[ DCNT │ C249 │ K100 ]
      │
      ├──[ DHSCS │ K100 │ C249 │ Y10 ]  ← On immediately
C249
──┤├─────[ SET │ Y17 ]
```
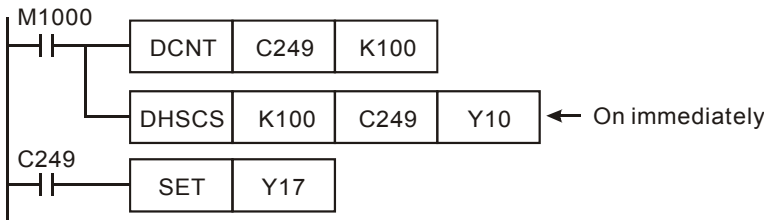
Program Example 3:

1. High speed counter interruption:

   a) Operand **D** of DHSCS instruction can designate I0□0, □ = 1 ~ 6, as the timing of interruption when the counting reaches its target.

   b) ES/EX/SS series MPU does not support high speed counter interruption.

   c) SA/SX/SC series MPU supports high speed counter interruption. However, when DHSCS instruction designates an I interruption, the designated high speed counter cannot be used in DHSCS, DHSCR, DHSZ instructions. Misuse of high speed counter will result in error.

   d) For SA/SX/SC series MPU, when the counting reaches the target, the interruption will occur. X0 is the counter for counting input and the interruption No. is I010 (1 phase 2 inputs and A-B phase counter No. C246 ~ C254 can only designate I010). X1 designates I020; X2 designates I030; X3 designates I040; X4 designates I050; X5 designates I060, totaling 6 points.

   e) When the present value in C251 changes from 99 to 100 or 101 to 100, the program will jump to I010 and execute the interruption service subroutine.

```
                 ┌──────┐
─────────────────┤  EI  │
                 └──────┘
   M1000
   ──┤├──┬──[ DCNT │ C251 │ K1000 ]
         │
         ├──[ DHSCS │ K100 │ C251 │ I010 ]
         ▽
                 ┌──────┐
─────────────────┤ FEND │
                 └──────┘
   M1000
I010 ──┤├────────( Y1 )
         ▽
                 ┌──────┐
                 │ IRET │
                 └──────┘
                 ┌──────┐
─────────────────┤ END  │
                 └──────┘
```
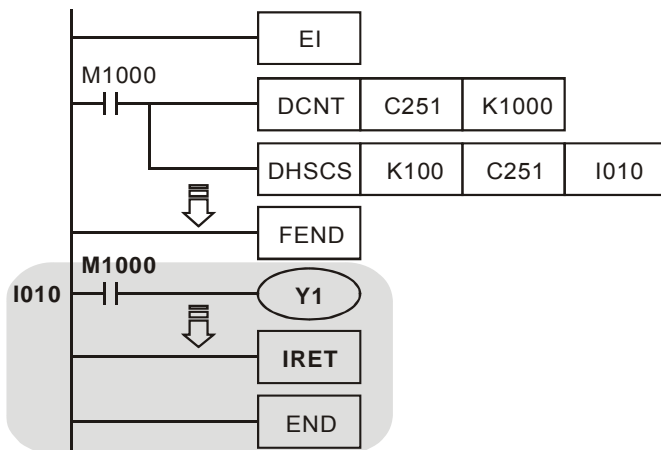
2. In SA/SX/SC series MPU, M1059 is "I010 ~ I060 high speed counter interruption forbidden" flag.

3. In EH/EH2/SV series MPU, M1289 ~ M1294 are the respectively for I010 ~ I060 "high speed counter interruption forbidden flags", i.e. when M1294 = On, I060 interruption will be forbidden.

Interruption pointer I No.	Interruption forbidden flag	Interruption pointer I No.	Interruption forbidden flag
I010	M1289	I040	M1292
I020	M1290	I050	M1293
I030	M1291	I060	M1294

Remarks:

1. The output contact of the high speed counter and the comparative outputs of API 53 DHSCS, API 34 DHSCR and API 55 DHSZ instructions only perform comparison and contact outputs when there is a counting input. When using data operation instructions, e.g. DADD, DMOV, for changing the present value in the high speed counter or making the present value equals the set value, there will not be comparisons or comparative outputs because there is no counting inputs.

2. High speed counters supported by ES/EX/SS series MPU (total bandwidth: 20KHz):

Type / Input	1-phase 1 input							1-phase 2 inputs			2-phase 2 inputs		
	C235	C236	C237	C238	C241	C242	C244	C246	C247	C249	C251	C252	C254
X0	U/D				U/D		U/D	U	U	U	A	A	A
X1		U/D			R		R	D	D	D	B	B	B
X2			U/D			U/D			R	R		R	R
X3				U/D		R	S			S			S

U: Progressively increasing input   A: A phase input   S: Input started

D: Progressively decreasing input   B: B phase input   R: Input cleared

a) Input points X0 and X1 can be planned as counters of higher speed (1 phase input can reach 20KHz). However, the total counting frequency of the two input points has to be smaller or equal 20KHz. Provided the input is a 2-phas input signal, the counting frequency will be approximately 4KHz. The frequency of the input points X2 and X3 (1-phase) can reach 10KHz.

b) For ES/EX/SS series MPU, the uses of DHSCS instructio with DHSCR instruction cannot be more than 4 times.

3. High speed counters supported by SA/SX series MPU (total bandwidth: 40KHz):

Type / Input	1-phase 1 input									1-phase 2 inputs			2-phase 2 inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C244	C246	C247	C249	C251	C252	C253	C254
X0	U/D						U/D		U/D	U	U	U	A	A	B	A
X1		U/D					R		R	D	D	D	B	B	A	B
X2			U/D					U/D			R	R		R		R
X3				U/D				R	S			S				S
X4					U/D											
X5						U/D										

U: Progressively increasing input   A: A phase input   S: Input started

D: Progressively decreasing input   B: B phase input   R: Input cleared

a) Input points X0 and X1 for 1-phase input can reach a frequency of 20KHz and X2 ~ X5 can reach 10KHz. 2-phase input (X0, X1) C251, C252 and C254 can reach a frequency of 4KHz and C253 reach 25KHz (only supports 4 times frequency counting).

b) Functions of the input point X5:

    i) When M1260 = Off, C240 is the general U/D high speed counuter.

    ii) When M1260 = On and C240 is enabled by DCNT instruction, X5 will be the shared reset signal for C235 ~ C239. The counter C240 will still receive the counting input signals from X5.

4. High speed counters supported by SC series MPU (total bandwidth: 130KHz):

Type / Input	1-phase 1 input											1-phase 2 inputs				2-phase inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245	C246	C247	C249	C250	C251	C252	C254	C255
X0	U/D						U/D			U/D		U	U	U		A	A	A	
X1		U/D					R			R		D	D	D		B	B	B	
X2			U/D					U/D					R	R			R	R	
X3				U/D				R		S				S				S	
X4					U/D														
X5						U/D													
X10									U/D						U				A
X11											U/D				D				B

U: Progressively increasing input   A: A phase input   S: Input started

D: Progressively decreasing input   B: B phase input   R: Input cleared

a) The functions of the high speed counters of input points X0 ~ X5 are the same of those in SA/SX series MPU.

b) The input points of 1-phase input X10 (C243), X11 (C245) and (X10, X11) C250 can reach a frequency of 100KHz. The total bandwidth of X10 ~ X11 is 130KHz. C255 of the 2-phase input (X10, X11) can reach a frequency of 50KHz.

c) For SA/SX/SC series MPU, the uses of DHSCS instruction with DHSCR instruction cannot be more than 6 times and the uses of DHSZ instruction cannot be more than 6 times as well. When DHSCR instruction designates I interruption, the designated high speed counter cannot be used in other DHSCS, DHSCR and DHSZ instructions.

d) The functions of X10 ~ X11 high speed counters in SC series MPU:

   i) When X10 and X11 are set as 1-phase 1 input or 1-phase 2 inputs. The maximum frequency of a single phase can reach 100KHz. When they are set as 2-phase 2 inputs, the maximum frequency can reach 50KHz.

   ii) X10 and X11 can select rising-edge counting mode or falling-edge counting mode. X10 is set by D1166 and X11 is set by D1167. K0: rising-edge counting. K1: falling-edge counting. K2: rising/falling edge counting (only supports X10).

   iii) The counting up and down of C243 are determined by the On/Off of M1243. The counting up and down of C245 are determined by the On/Off of M1245. Rising-edge and falling-edge counting are not able to take place at the same time. The rising-edge trigger and falling-edge trigger of C250 are determined by the content (K0 or K1) of D1166. C255 can only be used in a 4 times frequency counting and you can only select rising-edge trigger.

   iv) When C243 or C245 is in use, you will not be able to use C250 or C255, and vice versa.

   v) High speed counter and high speed comparator:
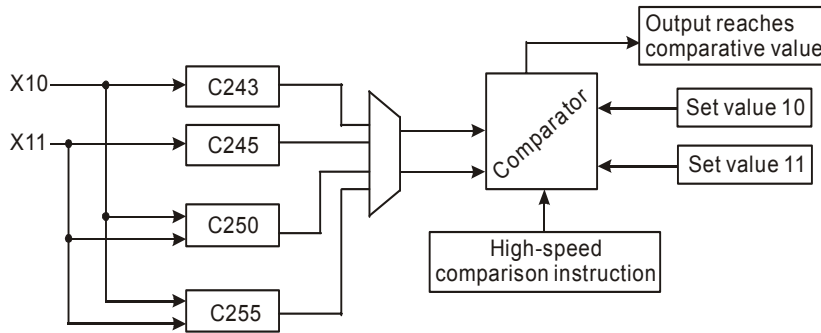
vi) Explanations on high speed counter and high speed comparator:

(1) When DHSCS and DHSCR instructions use the high speed counter (C243/C245/C250/C255), they can only use the set values of 2 groups of high speed comparative instructions. Assume that a group of comparative instruction DHSCS D0 C243 Y10 is already in use, you can only set another group DHSCR D2 C243 Y10 or DHSCS D4 C245 Y10.

(2) When DHSZ instruction use the high speed counter (C243/C245/C250/C255), it can only use the set value of a group of comparators.

(3) The number of set values in a high-speed comparative instruction offered in SA/SX series MPU will not decrease becasus of the addition of the new high speed counters.

(4) If the high-speed comparative instruction DHSCS requires a high-speed reponse output, it is suggested that you use Y10 or Y11 for the output. If you use other general devices for the output, there will be delay of 1 scan period. For example, when in I0x0 interruption, C234 will correspond to I020, C245 to I040 and C250/C255 to I060.

(5) The high-speed comparative instruction DHSCR can clear output devices and counter devices, but only the counters used by the same instruction, e.g. DHSCR K10 C243 C243. This function can only applied in the four special high speed counters C243, C245, C250 and C255.

e) Counting modes:

i) The 2-phase 2 inputs counting mode of the high speed counters in ES/EX/SS (V5.5 and above) and SA/SX/SC series MPU is set by special D1022 with normal frequency, double frequency and 4 times frequency modes. The contents in D1022 will be loaded in in the first scan when PLC is switched from STOP to RUN.

Device No.	Function
D1022	Setting up the multiplied frequency of the counter
D1022 = K1	Normal frequency mode selected
D1022 = K2 or 0	Double frequency mode selected (default)
D1022 = K4	4 times frequency mode selected

ii) Multiplied frequency mode (↑↓ indicates the occurrence of counting)

Counting mode		Counting wave pattern
2-phase 2 inputs	1 (Normal frequency)	
	2 (Double frequency)	
	4 (4 times frequency)	
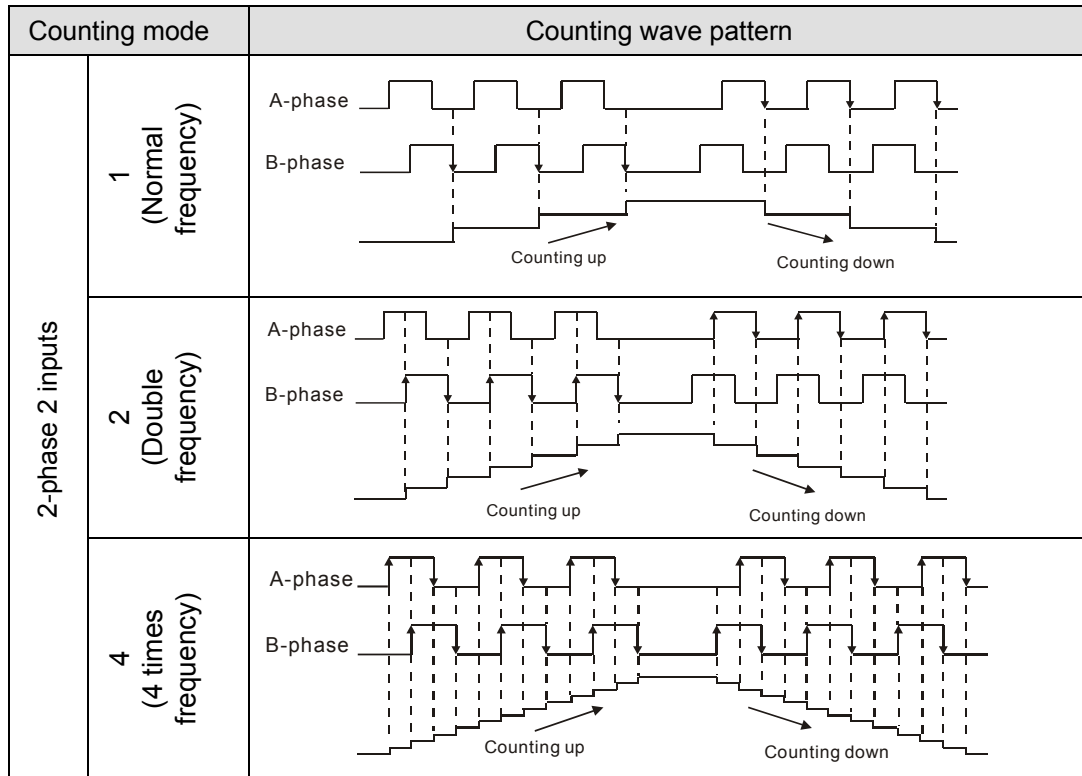
5. EH/EH2/SV series MPU supports high speed counters. C235 ~ C240 are program-interruption 1-phase high speed counter with a total bandwidth of 20KHz, can be used alone with a counting frequency of up to 10KHz. C241 ~ C254 are hardware high speed counter (HHSC). There are four HHSC in EH/EH2/SV series MPU, HHSC0 ~ 3. The pulse input frequency of HHSC0 and HHSC1 can reach 200KHz and that of HHSC2 and HHSC3 can reach 20KHz (1 phase or A-B phase). The pulse input frequency of HHSC0 ~ 3 of 40EH2 series MPU can reach 200KHz, among which:

C241, C246 and C251 share HHSC0

C242, C247 and C252 share HHSC1

C243, C248 and C253 share HHSC2

C244, C249 and C254 share HHSC3

a) Every HHSC can only be designated to one counter by DCNT instruction.

b) There are three counting modes in every HHSC (see the table below):

   i) 1-phase 1 input refers to "pulse/direction" mode.

   ii) 1-phase 2 inputs refers to "clockwise/counterclockwise (CW/CCW)" mode.

   iii) 2-phase 2 inputs refers to "A-B phase" mode.

Counter type	Program-interruption high speed counter						Hardware high speed counter											
Type	1-phase 1 input						1-phase 1 input				1-phase 2 inputs				2-phase 2 inputs			
Input	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C246	C247	C248	C249	C251	C252	C253	C254
X0	U/D						U/D				U				A			
X1		U/D									D				B			
X2			U/D				R				R				R			
X3				U/D			S				S				S			
X4					U/D			U/D				U				A		
X5						U/D						D				B		
X6								R				R				R		
X7								S				S				S		
X10									U/D				U				A	
X11													D				B	
X12									R				R				R	
X13									S				S				S	
X14										U/D				U				A
X15														D				B
X16										R				R				R
X17										S				S				S

U: Progressively increasing input	A: A phase input	S: Input started
B: Progressively decreasing input	B: B phase input	R: Input cleared

c) In EH/EH2/SV series MPU, there is no limitation on the times of using the hardware high speed counter related instructions, DHSCS, DHSCR and DHSZ. However, when these instructions are enabled at the same time, there will be some limitations. DHSCS instruction will occupy 1 group of settings, DHSCR 1 group of settings and DHSZ 2 groups of settings. There three instructions cannot occupy 8 groups of settings in total; otherwise the system will ignore the instructions which are not the first scanned and enabled.

d) System structure of the hardware high speed counters:

   i) HHSC0 ~ 3 have reset signals and start signals from external inputs. Settings in M1272, M1274, M1276 and M1278 are reset signals of HHSC0, HHSC1, HHSC2 and HHSC3. Settings in M1273, M1275, M1277 and M1279 are start signals of HHSC0, HHSC1, HHSC2 and HHSC3.

   ii) If the external control signal inputs of R and S are not in use, you can set M1264/M1266/M1268/M1270 and M1265/M1267/M1269/M1271 as True and disable the input signals. The corresponding external inputs can be used again as general input points (see the figure below).

   iii) When special M is used as a high speed counter, the inputs controlled by START and RESET will be affected by the scan time.

HHSC0	HHSC1	HHSC2	HHSC3	
X0	X4	X10	X14	Counting pulses (U/D, U, A)

HHSC0	HHSC1	HHSC2	HHSC3	
X1	X5	X11	X15	Counting pulses (D, B)

HHSC0	HHSC1	HHSC2	HHSC3	
D1225	D1226	D1227	D1228	Select counting modes

Set values 1 ~ 4 indicate
Mode 1 ~ 4 (1 ~ 4 times frequency)

HHSC0	HHSC1	HHSC2	HHSC3	
C241	C242	C243	C244	U/D mode setup flag
M1241	M1242	M1243	M1244	

HHSC0	HHSC1	HHSC2	HHSC3	
X2	X6	X12	X16	Reset signal R
M1264	M1266	M1268	M1270	
M1272	M1274	M1276	M1278	

HHSC0	HHSC1	HHSC2	HHSC3	
X3	X7	X13	X17	Start signal S
M1265	M1267	M1269	M1271	
M1273	M1275	M1277	M1279	

Present value in counter: HHSC0, HHSC1, HHSC2, HHSC3 → Comparator

Counting reaches set value — 8 set values

DHSCS occupies 1 group of set values
DHSCR occupies 1 group of set values
DHSCZ occupies 2 groups of set values

Counting up/down monitoring flag:

HHSC0	M1246	M1251
HHSC1	M1247	M1252
HHSC2	M1248	M1253
HHSC3	M1249	M1254

Output reaches comparative value for outputs

High-speed comparative instruction	Output reaches comparative value
DHSCS	SET/RESET
DHSCR	I010 ~ I060 clear the present value
DHSCZ	

Interruption forbidden flag:

I010	M1289
I020	M1290
I030	M1291
I040	M1292
I050	M1293
I060	M1294

e) Counting modes:

Special D1225 ~ D1228 are for setting up different counting modes of the hardware high speed counters (HHSC0 ~ 3) in EH/EH2/SV series MPU. There are normal ~ 4 times frequency for the counting and the default setting is double frequency.

Counting modes		Wave pattern	
Type	Set value in special D	Counting up(+1)	Counting down(-1)
1-phase 1 input	1 (Normal frequency)	U/D, U/D FLAG	
	2 (Double frequency)	U/D, U/D FLAG	
1-phase 2 inputs	1 (Normal frequency)	U, D	
	2 (Double frequency)	U, D	
2-phase 2 inputs	1 (Normal frequency)	A, B	
	2 (Double frequency)	A, B	
	3 (Triple frequency)	A, B	
	4 (4 times frequency)	A, B	

f) Special registers for relevant flags and settings of high speed counters:

Flag	Function
M1150	DHSZ instruction in multiple set values comparison mode
M1151	The execution of DHSZ multiple set values comparison mode is completed.
M1152	Set DHSZ instruction as frequency control mode
M1153	DHSZ frequency control mode has been executed.
M1235 ~ M1245	Designating the counting direction of high speed counters C235 ~ C245 When M12☐☐ = Off, C2☐☐ will perform a counting up. When M12☐☐ = On, C2☐☐ will perform a counting down.
M1246 ~ M1255	Monitor the counting direction of high speed counters C246 ~ C255 When M12☐☐ = Off, C2☐☐ will perform a counting up. When M12☐☐ = On, C2☐☐ will perform a counting down.
M1260	X5 as the reset input signal of all high speed counters
M1261	High-speed comparison flag for DHSCR instruction
M1264	Disable the external control signal input point of HHSC0 reset signal point (R)
M1265	Disable the external control signal input point of HHSC0 start signal point (S)
M1266	Disable the external control signal input point of HHSC1 reset signal point (R)
M1267	Disable the external control signal input point of HHSC1 start signal point (S)
M1268	Disable the external control signal input point of HHSC2 reset signal point (R)
M1269	Disable the external control signal input point of HHSC2 start signal point (S)
M1270	Disable the external control signal input point of HHSC3 reset signal point (R)
M1271	Disable the external control signal input point of HHSC3 start signal point (S)
M1272	Internal control signal input point of HHSC0 reset signal point (R)
M1273	Internal control signal input point of HHSC0 start signal point (S)
M1274	Internal control signal input point of HHSC1 reset signal point (R)
M1275	Internal control signal input point of HHSC1 start signal point (S)
M1276	Internal control signal input point of HHSC2 reset signal point (R)
M1277	Internal control signal input point of HHSC2 start signal point (S)
M1278	Internal control signal input point of HHSC3 reset signal point (R)
M1279	Internal control signal input point of HHSC3 start signal point (S)
M1289	High speed counter I010 interruption forbidden
M1290	High speed counter I020 interruption forbidden
M1291	High speed counter I030 interruption forbidden
M1292	High speed counter I040 interruption forbidden
M1293	High speed counter I050 interruption forbidden
M1294	High speed counter I060 interruption forbidden
M1312	C235 Start input point control
M1313	C236 Start input point control

Flag	Function
M1314	C237 Start input point control
M1315	C238 Start input point control
M1316	C239 Start input point control
M1317	C240 Start input point control
M1320	C235 Reset input point control
M1321	C236 Reset input point control
M1322	C237 Reset input point control
M1323	C238 Reset input point control
M1324	C239 Reset input point control
M1325	C240 Reset input point control
M1328	Enable Start/Reset of C235
M1329	Enable Start/Reset of C236
M1330	Enable Start/Reset of C237
M1331	Enable Start/Reset of C238
M1332	Enable Start/Reset of C239
M1333	Enable Start/Reset of C240

Special D	Function
D1022	Multiplied frequency of A-B phase counters for ES/SA series MPU
D1150	Table counting register for DHSZ multiple set values comparison mode
D1151	Register for DHSZ instruction frequency control mode (counting by table)
D1152 (low word) D1153 (high word)	In frequency control mode, DHSZ reads the upper and lower limits in the table counting register D1153 and D1152.
D1166	Switching between rising/falling edge counting modes of X10 (for SC series MPU only)
D1167	Switching between rising/falling edge counting modes of X11 (for SC series MPU only)
D1225	The counting mode of the 1st group counters (C241, C246, C251)
D1226	The counting mode of the 2nd group counters (C242, C247, C252)
D1227	The counting mode of the 3rd group counters (C243, C248, C253)
D1228	The counting mode of the 4th group counters (C244, C249, C254)
D1225 ~ D1228	Counting modes of HHSC0 ~ HHSC3 in EH/EH2/SV series MPU (default = 2) 1: Normal frequency counting mode 2: Double frequency counting mode 3: Triple frequency counting mode 4: 4 times frequency counting mode

API	Mnemonic		Operands	Function		Controllers
54	D	HSCR	S_1 S_2 D	High Speed Counter Reset		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DHSCR: 13 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2												*				
D		*	*	*								*				

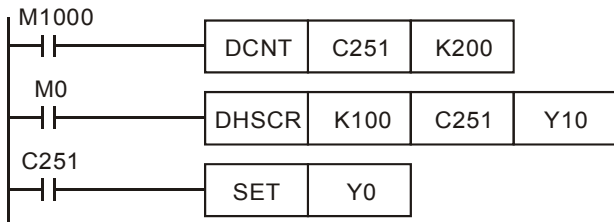|	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Comparative value   **S_2**: No. of high speed counter   **D**: Comparison result

Explanations:

1. **S_2** has to designate the No. of high speed counters C235 ~ C255. See remarks of API 53 DHSCS for more details.

2. **D** of EH/EH2/SV series MPU can designate the No. of high speed counters C241 ~ C254 that are the same as the counters designated by **S_2**.

3. **D** of SC series MPU can designate the No. of high speed counters C243, C245, C250 and C255 that are the same as the counters designated by **S_2**.

4. **D** of ES/EX/SS/SA/SX series MPU does not support device C.

5. See the specifications of each model for their range of use.

6. Flags: M1150 ~ M1333; see remarks of API 53 DHSCS for more details. ES/EX/SS/SA/SX/SC series MPU does not support M1261 (high speed counter external reset mode designation); see remarks for more details.

7. The high speed counter inputs counting pulses from the corresponding external input terminals X0 ~ X17 by inserting an interruption. When the No. of high-speed counter designated in **S_2** "+1" or "-1", DHSCR will perform a comparison immediately. When the present value in the high speed counter equals the comparative value designated in **S_1**, the device designated in **D** will turn Off and even the afterward comparison results are unequal, the device will still be Off.

8. If the devices designated in **D** are Y0 ~ Y17, when the comparative value equals the present value in the high speed counter, the comparison result will immediately output to the external output terminals Y0 ~ Y17 (and clear the designated Y output) and the rest of Y devices will be affected by the scan cycle. Devices M and S act immediately without being affected by the scan cycle.

Program Example 1:

1. When M0 = On and the present value in the high speed counter C251 changes from 99 to 100 or 101 to 100, Y10 will be cleared and Off.

2. When the present value in the high speed counter C251 changes from 199 to 200, the contact of C251 will be On and make Y0 = On. However, the program scan time will delay the output.

3. Y10 will immediately reset the status when the counting reaches its target. **D** is also able to designate high speed counters of the same No. See Program Example 2.

```
    M1000
    ┤├──────────┌──────┬──────┬──────┐
                │ DCNT │ C251 │ K200 │
                └──────┴──────┴──────┘
    M0
    ┤├──────────┌───────┬──────┬──────┬──────┐
                │ DHSCR │ K100 │ C251 │ Y10  │
                └───────┴──────┴──────┴──────┘
    C251
    ┤├──────────┌──────┬──────┐
                │ SET  │  Y0  │
                └──────┴──────┘
```
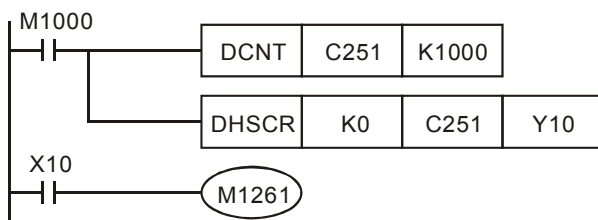
Program Example 2:

When DHSCR instruction designates the same high speed counter, and the present value in the high speed counter C251 changes from 999 to 1,000 or 1,001 to 1,000, C251 will be reset to Off.

```
    M1000
    ┤├──┬───────┌──────┬──────┬──────┐
        │       │ DCNT │ C251 │ K200 │
        │       └──────┴──────┴──────┘
        │       ┌───────┬───────┬──────┬──────┐
        └───────│ DHSCR │ K1000 │ C251 │ C251 │
                └───────┴───────┴──────┴──────┘
```
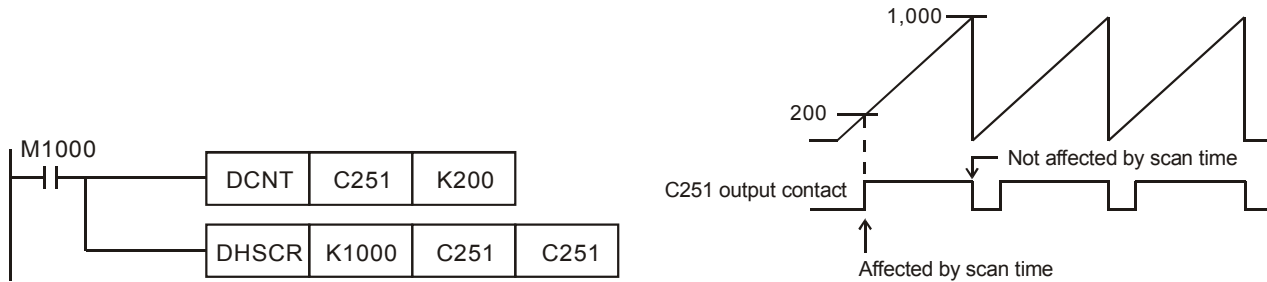

Remarks:

1. DVP all series MPU support high speec counters. For the limitation on the use of instructions, see remarks of API 53 DHSCS for more details.

2. M1261 of EH/EH2/SV series MPU designates the external reset modes of the high speed counter. Some high speec counters have input points for external reset; therefore, when the input point is On, the present value in the corresponding high speed counter will be cleared to 0 and the output contact will be Off. If you wish the reset to be executed immediately by the external output, you have to set M1261 to be On.

3. M1261 can only be used in the hardware high speed counter C241 ~ C255.

4. Example:

    a) X2 is the input point for external reset of C251.

    b) Assume Y10 = On.

    c) When M1261 = Off and X2 = On, the present value in C251 will be cleared to 0 and the contact of C251 will be Off. When DHSCR instruction is executed, there will be no counting input and the comparison result will not output. The external output will not execute the reset; therefore Y10 = On will remain unchanged.

    d) When M1261 = On and X2 = On, the present value in C251 will be cleared to 0 and the contact of C251 will be Off. When DHSCR instruction is executed, there will be no counting input but the comparison result will output. Therefore, Y10 will be reset.

```
    M1000
    ┤├──┬───────┌──────┬──────┬───────┐
        │       │ DCNT │ C251 │ K1000 │
        │       └──────┴──────┴───────┘
        │       ┌───────┬──────┬──────┬──────┐
        └───────│ DHSCR │  K0  │ C251 │ Y10  │
                └───────┴──────┴──────┴──────┘
    X10
    ┤├──────────( M1261 )
```

API	Mnemonic		Operands	Function	Controllers
55	D	HSZ	S₁ S₂ S D	High Speed Zone Compare	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DHSZ: 17 steps
S₁					*	*	*	*	*	*	*	*	*	*		
S₂					*	*	*	*	*	*	*	*	*	*		
S												*				
D		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

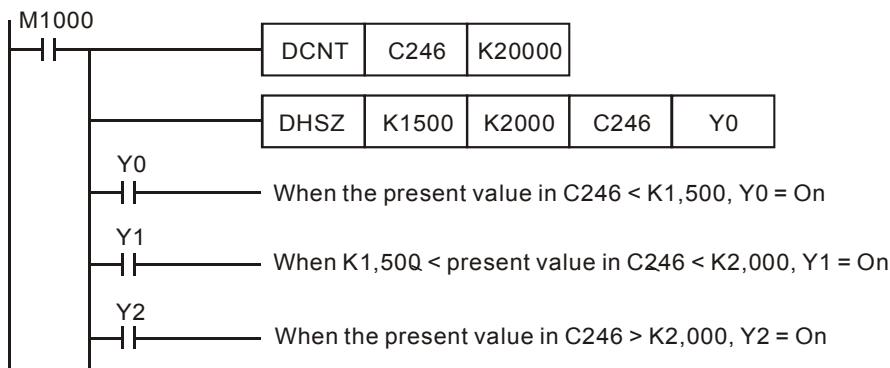**Operands:**

S₁: Lower bound of the comparison zone   **S₂**: Upper bound of the comparison zone   **S**: No. of high speed counter   **D**: Comparison result

Explanations:

1. **S₁** has to be euqal to or smaller than **S₂**. ($S_1 ≤ S_2$)

2. When **S₁** > **S₂**, the instruction will perform a comparison by using **S₁** as the upper bound and **S₂** as the lower bound.

3. **S** has to designate high speed counters C235 ~ C255, See remarks of API 53 DHSCS for more details.

4. **D** will occupy 3 consecutive devices.

5. Flags: M1150 ~ M1333; see remarks of API 53 DHSCS for more details. M1150, M1151 DHSZ executing multiple points comparison mode; see Program Example 3 for more details; SA/SX/SC series MPU does not support. M1152, M1153 DHSZ as frequency control mode; see Program Example 4 for more details; SA/SX/SC series MPU does not support.

6. The output will not be affected by the scan time.

7. The zone comparisons and outputs are all processed by inserting interruptions.

Program Example 1:

1. Designate device Y0 and Y0 ~ Y2 will be automatically occupied.

2. When DHSZ instruction is being executed and the counting of the high speed counter C246 reaches upper and lower bounds, one of Y0 ~ Y2 will be On

```
  M1000
  ─┤├──┬──────┤ DCNT │ C246 │ K20000 │
       │
       ├──────┤ DHSZ │ K1500 │ K2000 │ C246 │ Y0 │
       │
   Y0  │
  ─┤├──┼────── When the present value in C246 < K1,500, Y0 = On
       │
   Y1  │
  ─┤├──┼────── When K1,50Q < present value in C246 < K2,000, Y1 = On
       │
   Y2  │
  ─┤├──┴────── When the present value in C246 > K2,000, Y2 = On
```

Program Example 2:

1. Use DHSZ instruction for high/low speed stop control. C251 is an A-B phase high speed counter and DHSZ only performs comparison output when there is a C251 counting pulse input. Therefore, even when the present value in the counter is 0, Y10 will not be On.

2. When X10 = On, DHSZ will require that Y10 has to be On when the present value in the counter ≤ K2,000. To solve this requirement, you can execute DZCPP instruction when the program was first RUN and compare C251 with K2,000. When the present value in the counter ≤ K2,000, Y10 will be On. DZCPP instruction is a pulse execution instruction and will only be executed once with Y10 being kept On.

3. When the drive contact X10 = Off, Y10 ~ Y12 will be reset to Off.
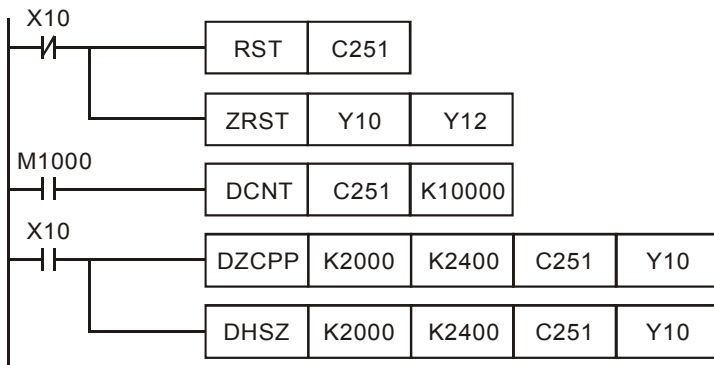
4. The timing diagram
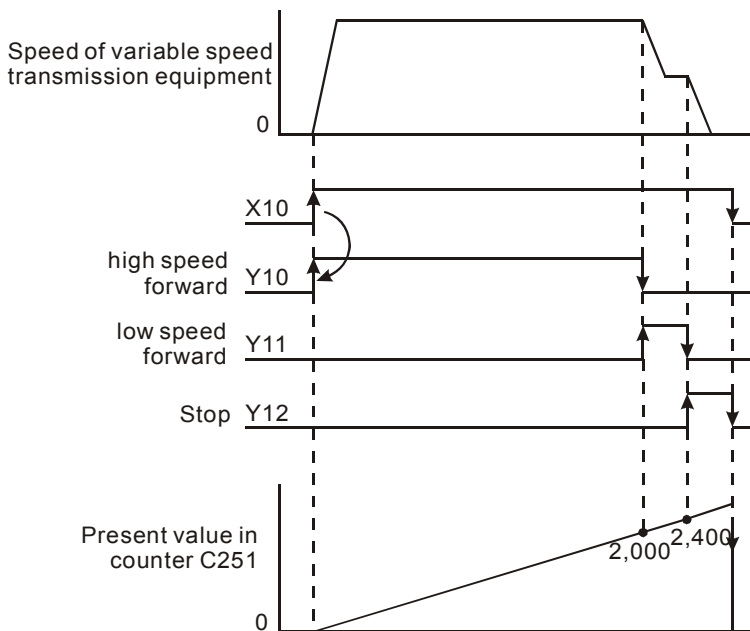
Program Example 3:

1. Program Example 3 is only applicable to EH/EH2/SV series MPU.

2. The multiple set values comparison mode: If **D** of DHSZ instruction designates a special auxiliary relay M1150, the instruction will be able to compare (output) the present value in the high speed counter with many set values.

3. In this mode,
    - **S₁**: start device in the comparison table. S_1 can only designate data register D and can be modified by E and F. Once this mode is enabled, S_1 will not be changed even the E and F has been changed.
    - **S₂**: number of group data to be compared. S_2 can only designate K1 ~ K255 or H1 ~ HFF and can be modified by E and F. Once this mode is enabled, S_2 cannot be changed. If S_2 is not within its range, error code 01EA (hex) will display and the instruction will not be executed.
   - **S**: No. of high speed counter (designated as C241 ~ C254).
    - **D**: Designated mode (can only be M1150)

4. The No. of start register designated in S_1 and the number of rows (groups) designated in S_2 construct a comparison table. Please enter the set values in every register in the table before executing the instruction.

5. When the present value in the counter C251 designated in **S** equals the set values in D1 and D0, the Y output designated by D2 will be reset to Off (D3 = K0) or On (D3 = K1) and be kept. Output Y will be processed as an interruption. No. of Y output pointss are in decimal (range: 0 ~ 255). If the No. falls without the range, SET/RESET will not be enabled when the comparison reaches its target.

6. When this mode is enabled, PLC will first acquire the set values in D0 and D1 as the target value for the first comparison section. At the same time, the index value displayed in D1150 will be 0, indicating that PLC performs the comparison based on the group 0 data.

7. When the group 0 data in the table have been compared, PLC will first execute the Y output set in group 0 data and determine if the comparison reaches the target number of groups. If the comparison reaches the target, M1151 will be On; if the comparison has not reached the final group, the content in D1150 will plus 1 and continue the comprison for the next group.

8. M1151 is the flag for the completion of one execution of the table, can be Off by the user. Or when the next comparion cycle takes place and the group 0 data has been compared, PLC will automatically reset the flag.

9. When the drive contact of the instruction X10 goes Off, the execution of the instruction will be interrupted and the content in D1150 (table counting register) will be reset to 0. However, the On/Off status of all outputs will be remained.

10. When the instruction is being executed, all set values in the comparison table will be regarded as valid values only when the scan arrives at END instruction for the first time.

11. This mode can only be used once in the program.

12. This mode can only be used on the hardware high speed counters C241 ~ C254.

13. When in this mode, the frequency of the input counting pulses cannot exceed 50KHz or the neighboring two groups of comparative values cannot differ by 1; otherwise there will not be enough time for the PLC to react and result in errors.

X10					
⊣├─	DHSZ	D0	K4	C251	M1150
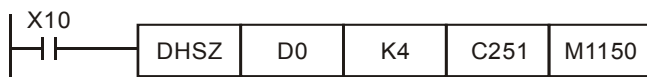
The comparison table:

32-bit data for comparison		No. of Y output	On/Off indication	Table counting register D1150
High word	Low word			
D1 (K0)	D0 (K100)	D2 (K10)	D3 (K1)	0
D5 (K0)	D4 (K200)	D6 (K11)	D7 (K1)	1
D9 (K0)	D8 (K300)	D10 (K10)	D11 (K0)	2
D13 (K0)	D12 (K400)	D14 (K11)	D15 (K0)	3
		K10: Y10	K0: Off	0→1→2→3→0
		K11: Y11	K1: On	Cyclic scan
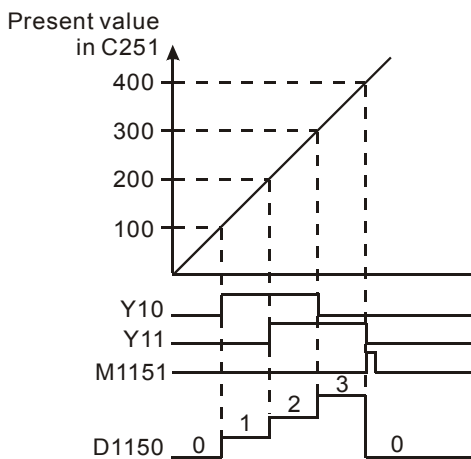
14. Special registers for flags and relevant settings:

Flag	Function
M1150	DHSZ instruction in multiple set values comparison mode
M1151	The execution of DHSZ multiple set values comparison mode is completed.

Special D	Function
D1150	Table counting register for DHSZ multiple set values comparison mode

Program Example 4:

1. Program Example 4 is only applicable to EH/EH2/SV series MPU.

2. DHSZ and DPLSY instructions are combined for frequency control. If **D** of DHSZ instruction is a special auxiliary relay M1152, the present value in the counter will be able to control the pulse output frequency of DPLSY instruction.

3. In this mode,
   - **S₁**: start device in the comparison table. **S₁** can only designate data register D and can be modified by E and F. Once this mode is enabled, **S₁** will not be changed even the E and F has been changed.
   - **S₂**: number of group data to be compared. **S₂** can only designate K1 ~ K255 or H1 ~ HFF and can be modified by E and F. Once this mode is enabled, **S₂** cannot be changed. If **S₂** is not within its range, error code 01EA (hex) will display and the instruction will not be executed.
   - **S**: No. of high speed counter (designated as C241 ~ C254).

- **D**: Designated mode (can only be M1152)

4. This mode can only be used once. For EH/EH2/SV series MPU, this mode can only be used in the hardware high speed counter C241 ~ C254. Please enter the set values in every register in the table before executing the instruction.

5. When this mode is enabled, PLC will first acquire the set values in D0 and D1 as the target value for the first comparison section. At the same time, the index value displayed in D1152 will be 0, indicating that PLC performs the comparison based on the group 0 data.

6. When the group 0 data in the table have been compared, PLC will first execute at the frequency set in group 0 data (D2, D3) and copy the data to D1152 and D1153, determining if the comparison reaches the target number of groups. If the comparison reaches the target, M1153 will be On; if the comparison has not reached the final group, the content in D1151 will plus 1 and continue the comprison for the next group.

7. M1153 is the flag for the completion of one execution of the table, can be Off by the user. Or when the next comparion cycle takes place and the group 0 data has been compared, PLC will automatically reset the flag.

8. If you wish to use this mode with PLSY instruction, please preset the value in D1152.

9. If you wish to stop the execution at the last row, please set the value in the last row K0.

10. When the drive contact of the instruction X10 goes Off, the execution of the instruction will be interrupted and the content in D1151 (table counting register) will be reset to 0.

11. When in this mode, the frequency of the input counting pulses cannot exceed 50KHz or the neighboring two groups of comparative values cannot differ by 1; otherwise there will not be enough time for the PLC to react and result in errors.

```
 X10
 ─┤├──┬──── DHSZ │ D0 │ K5 │ C251 │ M1152
      │
      ├──── PLS  │ M0
 M0   │
 ─┤/├──── DPLSY │ D1152 │ K0 │ Y0
```
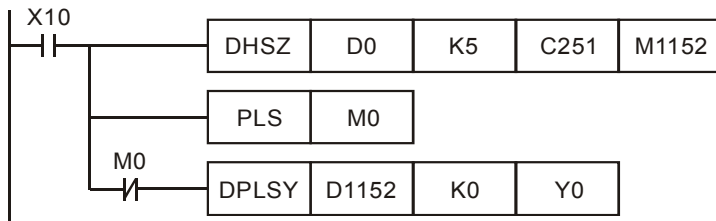
The comparison table:

32-bit data for comparison		Pulse output frequency 0 ~ 200KHz		Table counting register D1151
High word	Low word			
D1 (K0)	D0 (K0)	D3, D2	(K5,000)	0
D5 (K0)	D4 (K100)	D7, D6	(K10,000)	1
D9 (K0)	D8 (K200)	D11, D10	(K15,000)	2
D13 (K0)	D12 (K300)	D15, D14	(K6,000)	3
D17 (K0)	D16 (K400)	D19, D18	(K0)	4
				0→1→2→3→4 Cyclic scan
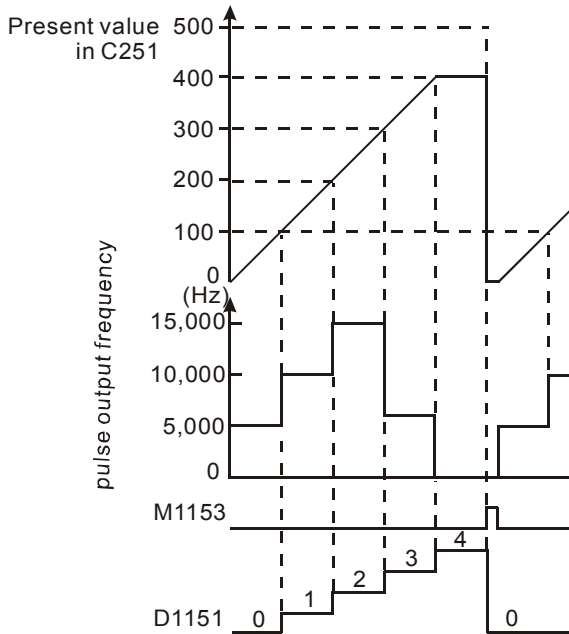
12. Special registers for flags and relevant settings:

Flag	Function
M1152	DHSZ instruction in frequency control mode
M1153	The execution of DHSZ frequency control mode is completed.

Special D	Function
D1151	Table counting register for DHSZ multiple set values comparison mode
D1152 (low word) D1153 (high word)	In frequency control mode, DHSZ reads the upper and lower limits in the table counting register D1153 and D1152.
D1336 (low word) D1337 (high word)	Current number of pulses output by DPLSY instruction

13. The complete program:

DMOVP	K200	D8		
DMOVP	K300	D12		
DMOVP	K400	D16		
DHSZ	D0	K5	C251	M1152
PLS	M0			

M0
DPLSY	D1152	K0	Y0
	Frequency	Number of pulses	Output point
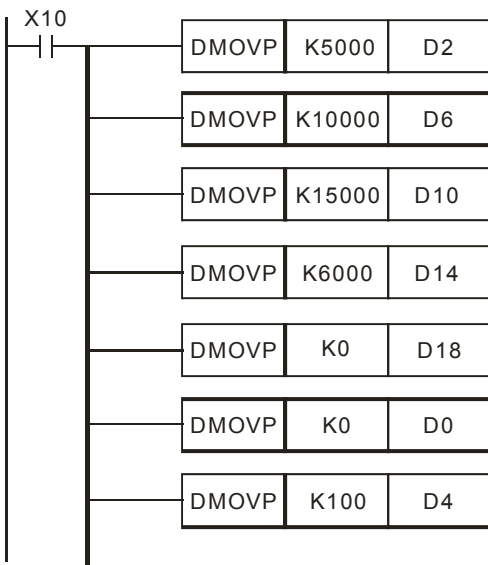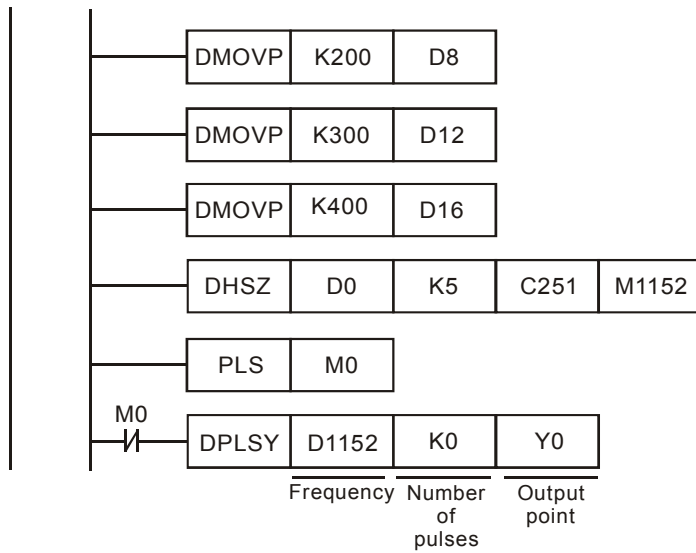
14. During the execution of DHSZ instruction, do not modify the set values in the comparison table.

15. The designated data will be arranged into the the above program diagram when the program executes to END instruction. Therefore, PLSY instruction has to be executed after DHSZ instruction has been executed once.

API	Mnemonic	Operands	Function	Controllers
56	SPD	S_1 S_2 D	Speed Detection	ES/EX/SS SA/SX/SC EH/SV

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

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: External pulse input terminal   **S₂**: Pulse receiving time (ms)   **D**: Detected result

Explanations:

1. See the specifications of each model for their range of use.

2. Flag: M1100 (SPD instruction performs sampling for one time)

3. External pulse input terminals designated in **S₁** for all series MPU:

MPU \ Input	ES/EX/SS (V5.7and above)	SA/SX/SC	EH/EH2/SV
Available input points	X1, X2	X0, X1, X2	X0 ~ X3

4. For SA/SX (V1.4 and above) series MPU and SC (V1.2 and above) series MPU, the new X0 and X1 can be used together with A-B phase input points. When "A ahead of B" detection result is a positive value and "B ahead of A" detection result is a negative value, the multiplied frequency of the counter can be set by D1022.

5. The received number of pulses of the input terminal designated in **S₁** is calculated within the time (in ms) designated in **S₂**. The result is stored in the register designated in **D**.

6. **D** will occupy 5 consecutive devices. **D** + 1 and **D** are the detected value obtained from the previous pulses; **D** +3 and **D** + 2 are the current accumulated number of values; **D** + 4 is the counting time remaining (max. 32,767ms).

7. Pulse frequency detection for all series:

MPU	ES/EX/SS (V5.7 and above)	SA/SX/SC	EH/EH2/SV
Max. frequency	X1 (20KHz), X2 (10KHz)	X0/X1 A-B phase input (4KHz) X1 (30KHz), X2 (10KHz)	X0/X1 (100KHz) X2/X3 (10KHz)

8. This instruction is mainly used for obtaining a proportional value of rotation speed. The result **D** and rotation speed will be in proportion. The following equation is for obtaining the rotation speed of motor.

$$N = \frac{60(D0)}{nt} \times 10^3 \ (rpm)$$

    N: Rotation speed
    n: The number of pulses produced per rotation
    t: Detecting time designated in **S₂** (ms)

9. The X input point designated by this instruction cannot be used again as the pulse input terminal of the high speed counter or as an external interruption signal.

10. When M1036 in SC (V1.4 and above) series MPU is enabled, SPD instruction can detect the speeds at X0 ~ X5

at the same time with a total bandwidth of 40KHz. See 2.11 for more details for how to use M1036.

11. There is no limitation on the times of using this instruction in the program, but only one instruction will be executed at a time.

12. When SPD instruction is enabled and M1100 = On, SPD instruction will perform a sampling at the moment when M1100 goes from Off to On and stop the sampling. If you wish to resume the sampling, you have to turn Off M1100 and re-enable SPD instruction.

Program Example:

1. When X7 = On, D2 will calculate the high-speed pulses input by X1 and stop the calculation automatically after 1,000ms. The result will be stored in D0.

2. When the 1,000ms counting is completed, D2 will be cleared to 0. When X7 is On again, D2 will start the calculation again.
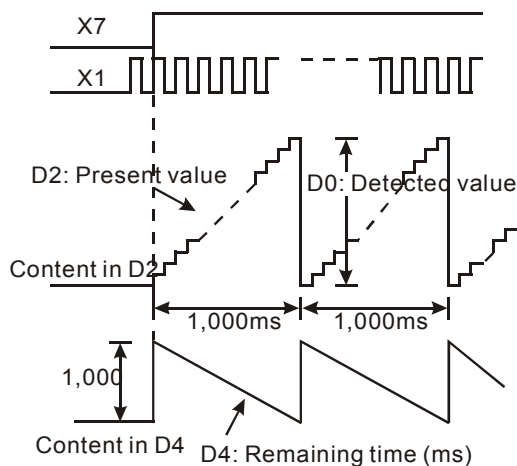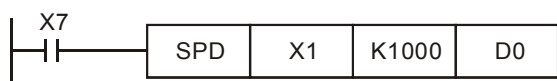
Remarks:

1. When ES/EX/SS (V5.7 and above) and SA/SX/SC series MPU use X1 or X2, the relevant high speed counters or external interruptions I101 and I201 cannot be used.

2. For SC (V1.4 and above) series MPU, when M1036 is enabled, the speed of X0 ~ X5 can be detected at the same time.

API	Mnemonic		Operands	Function	Controllers
57	D	PLSY	S_1 S_2 D	Pulse Y Output	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Pulse output frequency   **S_2**: Number of output pulses   **D**: Pulse output device (please use transistor output module)

Explanations:

1. The program of ES/EX/SS series MPU can use PLSY instruction two times but cannot designate the same Y device.

2. Flags: M1010 ~ M1345. See remarks for more details.

3. **S_1** designates the pulse output frequency. With M1133 ~ M1135 and D1133, Y0 of SA/SX series MPU is able to output pulses at 50KHz. See 2.11 for M1133 ~ M1135 and D1133.

   Range of output frequency for all series:

MPU	ES/EX/SS	SA/SX	SC	EH	EH2/SV
Frequency range	Y0: 0 ~10KHz Y1: 0 ~10KHz	Y0: 0 ~ 32KHz Y1: 0 ~10KHz	Y0: 0 ~ 30KHz Y1: 0 ~ 30KHz Y10: 77 ~ 100KHz Y11: 77 ~ 100KHz	Y0: 1 ~ 200KHz Y2: 1 ~ 200KHz	Y0: 0 ~ 200KHz Y2: 0 ~ 200KHz Y4: 0 ~ 200KHz Y6: 0 ~ 200KHz

4. **S_2** designates the number of output pulses. The 16-bit instruction can designate 1 ~ 32,767 pulses and the 32-bit instruction can designate 1 ~ 2,147,483,647 pulses.

   Number of continuous pulses for all series:

MPU	ES/EX/SS/SA/SX/SC	SC	EH/EH2/SV
How to designate continuous pulses	M1010 (Y0) On M1023 (Y1) On	M1010 (Y0) On M1023 (Y1) On The number of output pulses designated for Y10 and Y11 is set to K0.	The number of output pulses designated for Y0, Y2, Y4 and Y6 is set to K0

5. For EH/EH2/SV series MPU, when the number of output pulses is set to 0, there will be continuous pulse output with no limitation on the number of pulses. For ES/EX/SS/SA/SX/SC series MPU, you have to make M1010 (Y0) or M1023 (Y1) On to allow a continuous pulse output with no limitation on the number of pulses.

6. For the pulse output device designated in D, EH series MPU can designate Y0 and Y2, EH2/SV series MPU can designate Y0, Y2, Y4 and Y6, ES/EX/SS/SA/SX series MPU can designate Y0 and Y1, SC series MPU can designate Y0, Y1, Y10 and Y11. (SC V1.2 and above series MPU supports Y10 and Y11).

7. EH series MPU has two groups of A-B phase pulse output from CH0 (Y0, Y1) and CH1 (Y2, Y3); EH2/SV series MPU has four groups of A-B phase pulse output from CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See 2.3 and remarks for how to set up.

8. When PLSY instruction is executed, it will designate the number of output pulses (S_2) output from the output device (D) at a pulse output frequency (S_1).

9. When PLSY instruction is used in the program, its outputs cannot be the same as those in API 58 PWM and API 59 PLSR.

10. Pulse output completed flags for all series:

MPU	ES/EX/SS SA/SX/SC		SC		EH/EH2/SV		EH2/SV	
Output device	Y0	Y1	Y10	Y11	Y0	Y2	Y4	Y6
Flag	M1029	M1030	M1102	M1103	M1029	M1030	M1036	M1037

11. For ES/EX/SS/SA/SX/SC/EH series MPU, when PLSY and DPLSY instruction is disabled, the pulse output completed flags will all be Off automatically.

12. For EH2/SV series MPU, when PLSY and DPLSY instruction is disabled, the user will have to reset the pulse output completed flags.

13. The user has to reset the pulse output completed flags after the pulse output is completed.

14. After PLSY instruction starts to be executed, Y will start a pulse output. Modifying S_2 at this moment will not affect the current output. If you wish to modify the number of output pulses, you have to first stop the execution of PLSY instruction and modify the number.

15. S_1 can be modified when the program executes to PLSY instruction.

16. Off time : On time of the pulse output = 1 : 1.

17. When the program executes to PLSY instruction, the current number of output pulses will be stored in the special data registers D1336 ~ D1339. See remarks for more details.

18. For SA/EH series MPU, there is no limitation on the times using this instruction. For SA/SX/SC/EH series MPU, the program allows two instructions being executed at the same time. For EH2/SV series MPU, the program allows four instructions being executed at the same time.

Program Example:

1. When X0 = On, there will be 200 pulses output from Y0 at 1KHz. When the pulse output is completed, M1029 will be On and Y10 will be On.

2. When X0 = Off, the pulse output from Y0 will stop immediately. When X0 is On again, the output will start again ffrom the first pulse.
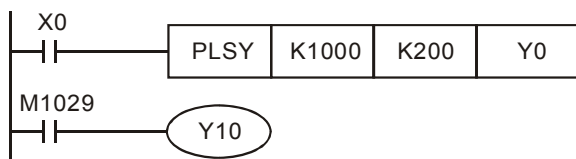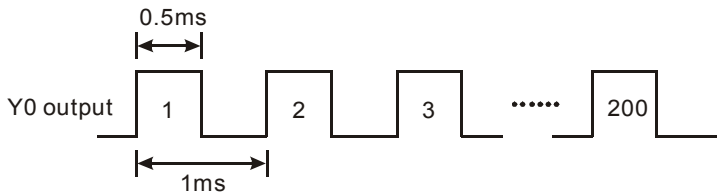
Remarks:

1. Flags and special registers for ES/EX/SS series MPU:

   M1010: When On, Y0 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y0 will be decided by S_2.

   M1023: When On, Y1 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y1 will be decided by S_2.

   M1029: On when Y0 pulse output is completed.

   M1030: On when Y1 pulse output is completed.

   M1078: Y0 output pauses.

   M1079: Y1 output pauses.

   D1030: Low word of the current number of output pulses from Y0

   D1031: High word of the current number of output pulses from Y0

   D1032: Low word of the current number of output pulses from Y1

   D1033: High word of the current number of output pulses from Y1

2. Flags and special registers for SA/SX/SC series MPU:

   M1010: (SA/SX/SC) When On, Y0 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y0 will be decided by S_2.

   M1023: (SA/SX/SC) When On, Y1 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y1 will be decided by S_2.

   M1029: (SA/SX/SC) On when Y0 pulse output is completed.

   M1030: (SA/SX/SC) On when Y1 pulse output is completed.

   M1078: (SA/SX/SC) Y0 output pauses.

   M1079: (SA/SX/SC) Y1 output pauses.

   M1102: (SC) On when Y10 pulse output is completed.

   M1103: (SC) On when Y11 pulse output is completed.

   D1030: (SA/SX/SC) Low word of the current number of output pulses from Y0

   D1031: (SA/SX/SC) High word of the current number of output pulses from Y0

   D1032: (SA/SX/SC) Low word of the current number of output pulses from Y1

   D1033: (SA/SX/SC) High word of the current number of output pulses from Y1

   D1348: (SC) Low word of the current number of output pulses from Y10

   D1349: (SC) High word of the current number of output pulses from Y10

   D1350: (SC) Low word of the current number of output pulses from Y11

   D1351: (SC) High word of the current number of output pulses from Y11

3. Flags and special registers for EH/EH2/SV series MPU:

    M1010: (EH/EH2/SV) When On, CH0, CH1, CH2 and CH3 will output pulses at END instruction. Off when the output starts.

    M1029: (EH/EH2/SV) On when CH0 pulse output is completed.

    M1030: (EH/EH2/SV) On when CH1 pulse output is completed.

    M1036: (EH2/SV) On when CH2 pulse output is completed.

    M1037: (EH2/SV) On when CH3 pulse output is completed.

    M1334: (EH/EH2/SV) CH0 pulse output pauses.

    M1335: (EH/EH2/SV) CH1 pulse output pauses.

    M1520: (EH2/SV) CH2 pulse output pauses.

    M1521: (EH2/SV) CH3 pulse output pauses.

    M1336: (EH/EH2/SV) CH0 pulse output has been sent.

    M1337: (EH/EH2/SV) CH1 pulse output has been sent.

    M1522: (EH2/SV) CH2 pulse output has been sent.

    M1523: (EH2/SV) CH3 pulse output has been sent.

    M1338: (EH/EH2/SV) CH0 offset pulses enabled.

    M1339: (EH/EH2/SV) CH1 offset pulses enabled.

    M1340: (EH/EH2/SV) I110 interruption occurs after CH0 pulse output is completed.

    M1341: (EH/EH2/SV) I120 interruption after occurs CH1 pulse output is completed.

    M1342: (EH/EH2/SV) I130 interruption occurs when CH0 pulse output is sending.

    M1343: (EH/EH2/SV) I140 interruption occurs when CH0 pulse output is sending.

    M1344: (EH/EH2/SV) CH0 pulse compensation enabled.

    M1345: (EH/EH2/SV) CH1 pulse compensation enabled.

    M1347: (EH/EH2/SV) CH0 pulse output reset flag

    M1348: (EH/EH2/SV) CH1 pulse output reset flag

    M1524: (EH2/SV) CH2 pulse output reset flag

    M1525: (EH2/SV) CH3 pulse output reset flag

    D1220: (EH/EH2/SV) Phase setting of CH0 (Y0, Y1): D1220 determines the phase by the last two bits; other bits are invalid.

    1. K0: Y0 output
    2. K1: Y0, Y1 AB-phase output; A ahead of B.
    3. K2: Y0, Y1 AB-phase output; B ahead of A.
    4. K3: Y1 output

    D1221: (EH/EH2/SV) Phase setting of CH1 (Y2, Y3): D1221 determines the phase by the last two bits; other bits are invalid.

    1. K0: Y2 output
    2. K1: Y2, Y3 AB-phase output; A ahead of B.
    3. K2: Y2, Y3 AB-phase output; B ahead of A.
    4. K3: Y3 output

D1229:   (EH2/SV) Phase setting of CH2 (Y4, Y5): D1229 determines the phase by the last two bits; other

bits are invalid.

1. K0: Y4 output
2. K1: Y4, Y5 AB-phase output; A ahead of B.
3. K2: Y4, Y5 AB-phase output; B ahead of A.
4. K3: Y5 output

D1230:   (EH2/SV) Phase setting of CH3 (Y6, Y7): D1230 determines the phase by the last two bits; other

bits are invalid.

1. K0: Y6 output
2. K1: Y6, Y7 AB-phase output; A ahead of B.
3. K2: Y6, Y7 AB-phase output; B ahead of A.
4. K3: Y7 output

D1328:   (EH/EH2/SV) Low word of the number of CH0 offset pulses

D1329:   (EH/EH2/SV) High word of the number of CH0 offset pulses

D1330:   (EH/EH2/SV) Low word of the number of CH1 offset pulses

D1331:   (EH/EH2/SV) High word of the number of CH1 offset pulses

D1332:   (EH/EH2/SV) Low word of the number of remaining pulses at CH0

D1333:   (EH/EH2/SV) High word of the number of remaining pulses at CH0

D1334:   (EH/EH2/SV) Low word of the number of remaining pulses at CH1

D1335:   (EH/EH2/SV) High word of the number of remaining pulses at CH1

D1336:   (EH/EH2/SV) Low word of the current number of output pulses at CH0

D1337:   (EH/EH2/SV) High word of the current number of output pulses at CH0

D1338:   (EH/EH2/SV) Low word of the current number of output pulses at CH1

D1339:   (EH/EH2/SV) High word of the current number of output pulses at CH1

D1375:   (EH2/SV) Low word of the current number of output pulses at CH2

D1376:   (EH2/SV) High word of the current number of output pulses at CH2

D1377:   (EH2/SV) Low word of the current number of output pulses at CH3

D1378:   (EH2/SV) High word of the current number of output pulses at CH3

D1344:   (EH/EH2/SV) Low word of the number of compensation pulses at CH0

D1345:   (EH/EH2/SV) High word of the number of compensation pulses at CH0

D1346:   (EH/EH2/SV) Low word of the number of compensation pulses at CH1

D1347:   (EH/EH2/SV) High word of the number of compensation pulses at CH1

4. When there are many high speed output instructions (PLSY, PWM, PLSR) for Y0 output in a program, PLC will only execute the settings and outputs of the instruction that is first enabled.

5. More explanations on M1347 and M1348:

If M1347 and M1348 is enabled, and when the execution of PLSY instruction has been completed, M1347/M1348 will be reset automatically, i.e. you do not have to turn the status of the drive contact from Off to On before PLSY instruction and when PLC scans to the instruction (assume the drive contact of the instruction is True), there will still be pulse output. PLC detects the status of M1347 and M1348 when END instruction is being executed. Therefore, when the pulse output is completed and if PLSY instruction is a continuous execution one, there will be a scan time of delay in the next string of pulse output.

Program Example 1:

```
                    ┌─────────┐
                    │   EI    │
                    └─────────┘
                    ┌─────────┐
                    │  FEND   │
                    └─────────┘
      M1000         ┌───────┬───────┬───────┬───────┐
I 001  ─┤ ├──┬──────│ DPLSY │ K1000 │ K1000 │  Y0   │
             │      └───────┴───────┴───────┴───────┘
             │      ┌───────┬───────┐
             └──────│  SET  │ M1347 │
                    └───────┴───────┘
                    ┌─────────┐
                    │  IRET   │
                    └─────────┘
      M1000         ┌───────┬───────┬───────┬───────┐
I 101  ─┤ ├──┬──────│ DPLSY │ K1000 │ K1000 │  Y2   │
             │      └───────┴───────┴───────┴───────┘
             │      ┌───────┬───────┐
             └──────│  SET  │ M1348 │
                    └───────┴───────┘
                    ┌─────────┐
                    │  IRET   │
                    └─────────┘
                    ┌─────────┐
                    │  END    │
                    └─────────┘
```
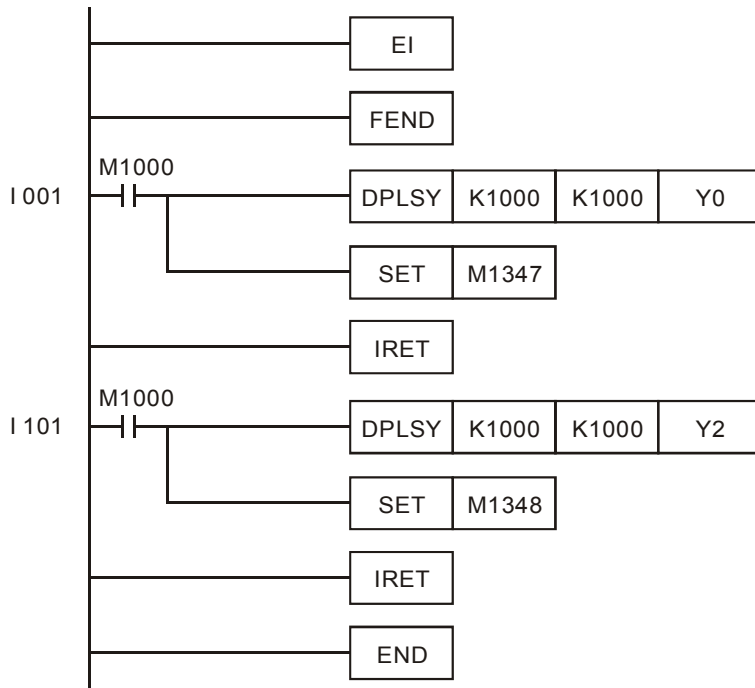
Explanations:

a) Whenever X0 is triggered, Y0 will output 1,000 pulses; whenever X1 is triggered, Y2 will output 1,000 pulses.

b) When X triggers Y pulse output, there should be an interval of at least one scan time between the end of Y pulse output and the next X-triggered output.

Program Example 2:

```
      X1          ╭────────╮
  ─┤ ├──────────(  M1347  )
                  ╰────────╯
      X2          ┌──────┬───────┬───────┬───────┐
  ─┤ ├────────────│ PLSY │ K1000 │ K1000 │  Y0   │
                  └──────┴───────┴───────┴───────┘
                  ┌────────┐
                  │  END   │
                  └────────┘
```
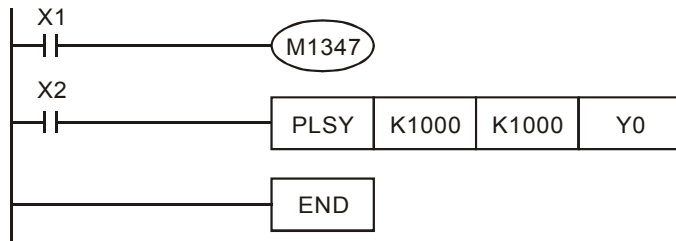
Explanations:

When both X1 and X2 are On, Y0 pulse output will keep operating. However, there will be a short pause (approx. 1 scan time) every 1,000 pulses before the output of the next 1,000 pulses.

API	Mnemonic	Operands			Function	Controllers
58	PWM	S_1 S_2 D			Pulse Width Modulation	ES/EX/SS SA/SX/SC EH/SV

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

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Pulse output width   **S_2**: Pulse output period   **D**: Pulse output device (please use transistor output module)

Explanations:

1. **$S_1 \leq S_2$**.

2. See the specifications of each model for their range of use.

3. In ES/EX/SS series MPU, PWM instruction can only be used once in the program.

4. Flags: See remarks for more details.

5. Range of **S_1**: (t) 0 ~ 32,767ms.

6. Range of **S_2**: (T) 1 ~ 32,767ms (but **$S_1 \leq S_2$**).

7. **D** for all series MPU:

MPU	ES/EX/SS/SA/SX/SC	EH	EH2/SV
Output point	Y1	Y0, Y2	Y0, Y2, Y4, Y6

8. When PWM instruction is used in the program, its outputs cannot be the same as those of API 57 PLSY and API 59 PLSR.

9. PWM instruction designates the pulse output width in **S_1** and pulse output period in **S_2** and outputs from output device **D**.

10. For SA/SX/SC series MPU, When, **$S_1 \leq 0$** or **$S_2 \leq 0$** or **$S_1 > S_2$**, there will be operational errors (M1067 and M1068 will not be On), and there will be no output from the pulse output device. When **$S_1 = S_2$**, the pulse output device will keep being On.

11. For EH/EH2/SV series MPU, When, **$S_1 < 0$** or **$S_2 \leq 0$** or **$S_1 > S_2$**, there will be operational errors (M1067 and M1068 will be On), and there will be no output from the pulse output device. When **$S_1 = 0$**, M1067 and M1068 will not be On and there will be no output from the pulse output device. When **$S_1 = S_2$**, the the pulse output device will keep being On.

12. **S_1** and **S_2** can be changed when PWM instruction is being executed.

13. For SA/EH series MPU, there is no limitation on the times using this instruction in the program. However, for SA/SX/SC/EH series MPU, two instructions are allowed to be executed at the same time; for EH2/SV series MPU, four instructions are allowed to be executed at the same time.

Program Example:

When X0 = On, Y1 will output the pulses as below. When X0 = Off, Y1 output will also be Off.
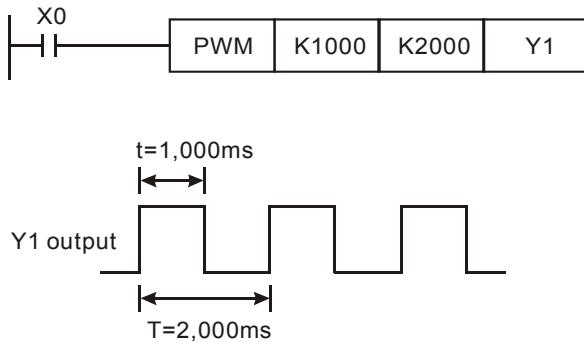
Remarks:

1. Flags for ES/EX/SS/SA/SX/SC series MPU:

   M1070:   Y1 pulse output time unit switch. When Off: 1ms; when On: 100us

2. Flags and special registers for EH/EH2/SV series MPU:

   M1010:   (EH/EH2/SV) When On, CH0, CH1, CH2 and CH3 will output pulses when END instruction is executed. Off when the output starts.

   M1070:   (EH/EH2/SV) The setting of time unit of CH0 has to work with D1371.

   M1071:   (EH/EH2/SV) The setting of time unit of CH1 has to work with D1372.

   M1258:   (EH/EH2/SV) CH0 pulse output signals reverse.

   M1259:   (EH/EH2/SV) CH1 pulse output signals reverse.

   M1334:   (EH/EH2/SV) CH0 pulse output pauses.

   M1335:   (EH/EH2/SV) CH1 pulse output pauses.

   M1336:   (EH/EH2/SV) CH0 pulse output has been sent.

   M1337:   (EH/EH2/SV) CH1 pulse output has been sent.

   M1520:   (EH2/SV) CH2 pulse output pauses.

   M1521:   (EH2/SV) CH3 pulse output pauses.

   M1522:   (EH2/SV) CH2 pulse output has been sent.

   M1523:   (EH2/SV) CH3 pulse output has been sent.

   M1526:   (EH2/SV) CH2 pulse output signals reverse.

   M1527:   (EH2/SV) CH3 pulse output signals reverse.

   M1530:   (EH2/SV) The setting of time unit of CH2 has to work with D1373.

   M1531:   (EH2/SV) The setting of time unit of CH3 has to work with D1374.

   D1336:   (EH/EH2/SV) Low word of the current number of output pulses from CH0.

   D1337:   (EH/EH2/SV) High word of the current number of output pulses from CH0.

   D1338:   (EH/EH2/SV) Low word of the current number of output pulses from CH1.

   D1339:   (EH/EH2/SV) High word of the current number of output pulses from CH1.

   D1371:   (EH/EH2/SV) Time unit of CH0 output pulses when M1070 = On.

   D1372:   (EH/EH2/SV) Time unit of CH1 output pulses when M1071 = On.

D1373:   (EH2/SV) Time unit of CH2 output pulses when M1530 = On.

D1374:   (EH2/SV) Time unit of CH3 output pulses when M1531 = On.

D1375:   (EH2/SV) Low word of the current number of output pulses from CH2.

D1376:   (EH2/SV) High word of the current number of output pulses from CH2.

D1377:   (EH2/SV) Low word of the current number of output pulses from CH3.

D1378:   (EH2/SV) High word of the current number of output pulses from CH3.

3. Time unit settings for EH/EH2/SV series MPU:

You cannot modify M1070 in the program.

D1371, D1372, D1373 and D1374 determine the time unit of the output pulses from CH0, CH1, CH2 and CH3 and the default setting is K1. If your set value is not within the range, the default value will be adopted.

D1371, D1372, D1373, D1374	K0	K1	K2	K3
Time unit	10us	100us	1ms	10ms

4. When there are many high speed pulse output instructions (PLSY, PWM, PLSR) in a program for Y0 output, and provided these instructions are being executed in the same scan period, PLC will set up and output the instructions with the fewest steps.

API	Mnemonic			Operands			Function		Controllers	
59	D	PLSR		S_1 S_2 S_3 D			Pulse Ramp		ES/EX/SS SA/SX/SC EH/SV	

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		PLSR: 9 steps
S_1					*	*	*	*	*	*	*	*	*	*	*		DPLSR: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*		
S_3					*	*	*	*	*	*	*	*	*	*	*		
D		*															

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Maximum speed of pulse output   **S_2**: Total number of output pulses   **S_3**: Acceleration/deceleration time (ms)

D: Pulse output device (please use transistor output module)

Explanations:

1. See the specifications of each model for their range of use.

2. For ES/EX/SS series MPU, PLSR instruction can be used twice in the program but the outputs cannot be overlapped.

3. Flags: See remarks of API 57 PLSY.

4. Range of **S_1**: 10 ~ 32,767Hz (16-bit); 10 ~ 200,000Hz (32-bit). The maximum speed has to be 10's multiple; if not, the 1s digit will be left out. 1/10 of the maximum speed is the variation of one acceleration or deleration. Please be aware if the variation reponds to the acceleration/deceleration demand from the step motor, in case the step motor may crash.

5. Range of **S_2**: 110 ~ 32,767 (16-bit); 110 ~ 2,147,483,647 (32-bit). If **S_2** is less than 110, the pulet output will be abnormal.

6. Range of **S_3**: below 5,000ms. The acceleration time and deceleration time have to be the same.

    a) The acceleration/deceleration time has to be 10 times longer than the maximum scan time (D1012). If not, the slope of accleration and deceleration will be incorrect.

    b) The minimum set value of acceleration/deceleration time can be obtained from the following equation:

    $$S_3 \geqslant \frac{90,000}{S_1}$$

    If the set value is less than the result obtained from the equation, the acceleration/deceleration time will be longer. If the set value is less than 90,000/**S_1**, use the result of 90,000/**S_1** as the set value.

    c) The maximum set value of acceleration/deceleration time can be obtained from the following equation:

    $$S_3 \leqslant \frac{S_2}{S_1} \times 818$$

    d) The speed variation is fixed to 10 steps. If the input acceleration/deceleration time is longer than the maximum set value, the acceleration/deceleration time will follow the maximum set time. If shorter than the minimum set value, the accleration/deceleration time will follow the minimum set time.

7. **D** for all series MPU:

MPU	ES/EX/SS/SA/SX/SC	EH	EH2/SV
Output point	Y0, Y1	Y0, Y2	Y0, Y2, Y4, Y6

8. EH series MPU has two groups pf A-B phase pulse output CH0 (Y0, Y1) and CH1 (Y2, Y3). EH2/SV series MPU has four groups pf A-B phase pulse output CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks of API 57 PLSY for how to set up.

9. PLSR instruction is a pulse output instruction with acclerating and decelerating functions. The pulses accelerate from the static status to target speed and decelerates when the target distance is nearly reached. The pulse output will stop when the target distance is reached.

10. When PLSR instruction is executed, after S_1, S_2 and S_3 are set, the pulses will output from **D**. The output starts at the frequency of increasing S_1/10 at a time. The time forf every frequency is fixed at S_3/9.

11. S_1, S_2 and S_3 can be changed when PLSR instruction is being executed.

12. For ES/EX/SS/SA/SX/SC series MPU, when all the Y0 pulses have been sent, M1029 will be On; when all the Y1 pulses have been sent, M1030 will be On. Next time when PLSR instruction is enabled, M1029 or M1030 will be 0 again and after the pulse output is completed, it will become 1 again.

13. For EH/EH2/SV series MPU, when all the CH0 (Y0, Y1) pulses have been sent, M1029 will be On; when all the CH1 (Y2, Y3) pulses have been sent, M1030 will be On; when CH2 (Y4, Y5) pulses have been sent, M1036 will be On; when CH3 (Y6, Y7) pulses have been sent, M1037 will be On. Next time when PLSR instruction is enabled, M1029, M1030, M1036 or M1037 will be 0 again and after the pulse output is completed, they will become 1 again.

14. During every acceleration section, the number of pulses (frequency × time) may not all be integers. PLC will round up the number to an integer before the output. Therefore, the acceleration time of every section may not be exactly the same. The offset is determined upon the frequency and the decimal after rounding up. In order to ensure the correct number of output pulses, PLC will supplement insufficient pulses in the last section.

15. For SA/EH series MPU, there is no limitation on the times of using this instruction in the program. However, for SA/SX/SC/EH series MPU, two instructions can be exexuted at the same time; for EH2/SV series MPU, four instructions can be executed at the same time.

Program Example:

1. When X0 = On, the pulses will output at the maximum frequency 1,000Hz with the total number D10 at 3,000ms from Y0. The frequency will increase by 1,000/10Hz at a time and every frequency will last for 3,000/9ms.

2. When X10 is Off, the output will be interrupted. When X0 is On again, the counting of pulses will start from 0.

```
   X0
───┤ ├────────[ PLSR │ K1000 │ D10 │ K3000 │ Y0 ]
```

Output: Y0 or Y2
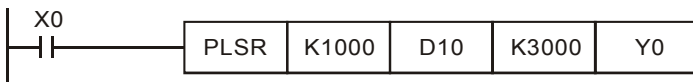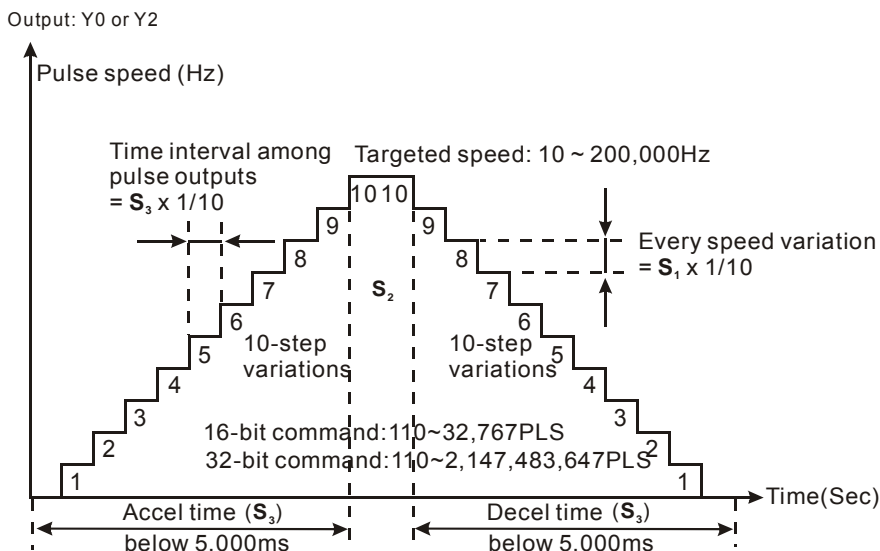
Remarks:

1. The outputs cannot be the same as those of API 57 PLSY and API 58 PWM.

2. When there are many high speed pulse output instructions (PLSY, PWM, PLSR) in a program for Y0 output, and provided these instructions are being executed in the same scan period, PLC will set up and output the instructions with the fewest steps.

3. With M1133 ~ M1135 and D1133, Y0 of SA/SX/SC series MPU can output pulses at up to 50KHz. See 2.11 for more details of special D and special M.

   Range of output frequecies for all series:

MPU	ES/EX/SS	SA/SX/SC	EH	EH2/SV
Range	Y0: 10 ~ 10,000Hz Y1: 10 ~ 10,000Hz	Y0: 10 ~ 30,000Hz Y1: 10 ~ 30,000Hz	Y0: 10 ~ 200,000Hz Y2: 10 ~ 200,000Hz	Y0: 10 ~ 200,000Hz Y2: 10 ~ 200,000Hz Y4: 10 ~ 200,000Hz Y6: 10 ~ 200,000Hz

Functions in EH series MPU:

1. Relevant devices for EH/EH2/SV series MPU:

X0
┤├ PLSR	K1000	D10	K3000	Y0
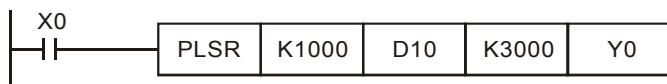
2. The range of pulse speed for this instruction is 10 ~ 200,000Hz. If the set values of maximum speed and acceleration/deceleration time exceed the range, PLC will operate by the default value that is within the range.

Operand		S_1	S_2	S_3	D
Explanation		Max. frequency	Total number of pulses	Accel/Decel time	Output point
Range	16-bit	10 ~ 32,767Hz	110 ~ 32,767	1 ~ 5,000ms	Y0 ~ Y7
	32-bit	10 ~ 200KHz	110 ~ 2,147,483,647		
Definition		K0: No output Kn: Designated frequency	Kn: Designated number	Flag: M1067, M1068	See settings of D1220, D1221
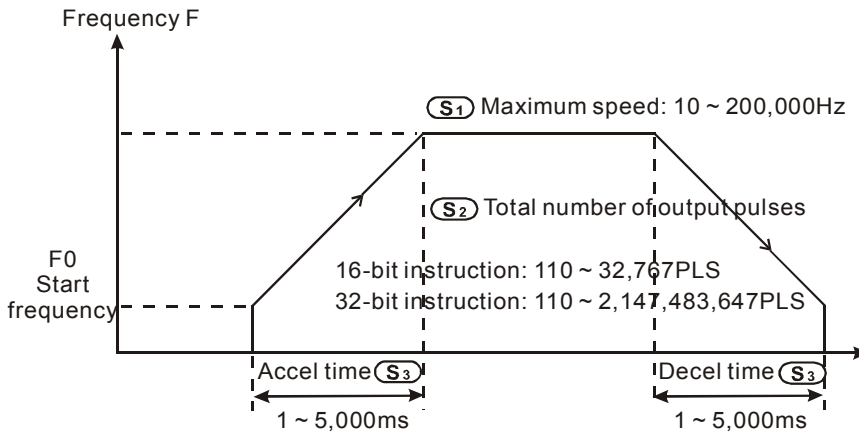
3. The acceleration/deceleration of EH/EH2/SV series MPU is based on the number of pulses. If the output cannot reach the maximum acceleration frequency within the acceleration/deceleration time offered, the instruction will automatically adjust the acceleration/deceleration time and the maximum frequency.

4. The operands have to be set before the execution of the instruction.

5. All acceleration/deceleration instructions are included with the brake function. The brake function will be enabled when PLC is performing acceleration and the switch contact is suddenly Off. The deceleration will operate at the slope of the acceleration.

API	Mnemonic	Operands			Function	Controllers
60	IST	S	D_1	D_2	Initial State	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device in the designated operation mode   **D₁**: The smallest No. of designated step in auto mode

D₂: The biggest No. of designated step in auto mode

Explanations:

1. **S** will occupy 8 consecutive points.

2. Range of **D₁** and **D₂**: for SA/SX/SC/EH/EH2/SV S20 ~ S899; for ES/EX/SS S20 ~ S127; **D₂** > **D₁**.

3. See the specifications of each model for their range of use.

4. ES/SA series MPU does not support E, F index register modification.

5. IST instruction can only be used once in the program.

6. Flags: M1040 ~ M1047. See remarks for more details.

7. IST instruction is a handy instruction specifically for the initial status of step ladder control procedure to accommodate special auxiliary relay.

Program Example 1:

1. Use of IST instruction

```
  M1000
───┤ ├─────┬─────┬─────┬─────┬─────┐
            │ IST │ X10 │ S20 │ S60 │
            └─────┴─────┴─────┴─────┘
```

S   X10: Individual operation   X14: Continuous operation
     X11: Zero return   X15: Zero return enabled switch
     X12: Step operation   X16: Start switch
     X13: One cycle operation   X17: Stop switch

2. When IST instruction is being executed, the following special auxiliary relays will switch automatically.

   M1040: Operation forbidden   S0: Initiates manual operation
   M1041: Operation starts   S1: Initiates zero return
   M1042: Pulse output enabled   S2: Initiates auto operation
   M1047: STL monitor enabled

3. S10 ~ S19 are for zero return and cannot be used as general steps. When S0 ~ S9 are in use, S0 ~ S2 represent manual operation mode, zero return mode and auto operation mode. Therefore, in the program , you have to write the circuit of the three steps in advance.

4. When switched to S1 (zero return) mode, any On in S10 ~ S19 will result in no zero return.

5. When switched to S2 (auto operation) mode, any On of the S in **D₁** ~ **D₂** or M1043 = On will result in no auto operation.

Program Example 2:

1. Robot arm control (by IST instruction):

   a) Motion request: Separate the big ball and small ball and move them to different boxes. Configure the control panel for the control.

   b) Motions of the robot arm: descending, clipping ball, ascending, right shifting, releasing ball, ascending, left shifting.

   c) I/O devices:

2. Operation modes:

   Manual operation: Turn On/Off of the load by a single button.

   Zero return: Press the zero return button to automatically zero-return the machine.

   Auto operation:

   a) Single step operation: Press "auto start" button for every one step forward.

   b) One cycle operation: Press "auto start" button at the zero point. After a cycle of auto operation, the operation will stops at the zero point. Press "auto stop" button in the middle of the operation to stop the operation and press "auto start" to restart the operation. The operation will resume until it meets the zero point.

   c) Continuous operation: Press "auto start" button at the zero point to resume the operation. Press "auto stop" to operate until it meets the zero point.

3. The control panel:

   a) Ball size sensor X0.

   b) Robot arm: left limit X1, big ball right limit X2, small ball right limit X3, upper limit X4, lower limit X5.

   c) Robot arm: ascending Y0, descending Y1, right shifting Y2, left shifting Y3, clipping Y4.

Start Circuit

Manual Operation Mode

Zero Return Mode

SFC:

Ladder Diagram:

Auto Operation Modes

SFC:

```
          ┌─────┐
          │ S2  │
          └──┬──┘
   M1041 ────┤
   M1044 ────┤
          ┌──┴──┐
          │ S20 ├────────( Y1 )
          └──┬──┘
             ├──────────────────────────────────┐
     X5 ─────┤                          X5 ──────┤
     X0 ─────┤                          X0 ──────┤
          ┌──┴──┐   ┌──────┬──────┐        ┌──┴──┐   ┌──────┬──────┐
          │ S30 ├───┤ SET  │  Y4  │        │ S40 ├───┤ SET  │  Y4  │
          └──┬──┘   ├──────┼──────┼──────┐ └──┬──┘   ├──────┼──────┼──────┐
                    │ TMR  │  T0  │ K30  │            │ TMR  │  T1  │ K30  │
                    └──────┴──────┴──────┘            └──────┴──────┴──────┘
     T0 ─────┤        X4                  T1 ──────┤        X4
          ┌──┴──┐   ─│/│─                      ┌──┴──┐   ─│/│─
          │ S31 │────────( Y0 )                │ S41 │────────( Y0 )
          └──┬──┘                              └──┬──┘
     X4 ─────┤        X2                  X4 ─────┤        X3
          ┌──┴──┐   ─│/│─                      ┌──┴──┐   ─│/│─
          │ S32 │────────( Y2 )                │ S42 │────────( Y2 )
          └──┬──┘                              └──┬──┘
     X2 ─────┤                          X3 ──────┤
             ├──────────────────────────────────┘
          ┌──┴──┐        X5
          │ S50 │────── ─│/│───( Y1 )
          └──┬──┘
     X5 ─────┤
          ┌──┴──┐   ┌──────┬──────┐
          │ S60 ├───┤ RST  │  Y4  │
          └──┬──┘   ├──────┼──────┼──────┐
                    │ TMR  │  T2  │ K30  │
                    └──────┴──────┴──────┘
     T2 ─────┤        X4
          ┌──┴──┐   ─│/│─
          │ S70 │────────( Y0 )
          └──┬──┘
     X4 ─────┤        X1
          ┌──┴──┐   ─│/│─
          │ S80 │────────( Y3 )
          └──┬──┘
     X1 ─────┤
             ▼
          ┌─────┐
            S2
```

Ladder Diagram:

Rung	Instruction	Comment
S2 M1041 M1044 ⊣⊢⊣⊢	SET S20	Enter auto operation mode
S20	(Y1)	Robot arm descending
X5 X0 ⊣⊢⊣⊢	SET S30	
X5 X0 ⊣⊢⊣/⊢	SET S40	
S30	SET Y4	Clipping tightly
	TMR T0 K30	
T0 ⊣⊢	SET S31	
S31 X4 ⊣/⊢	(Y0)	Robot arm ascending to upper limit (X4 On)
X4 ⊣⊢	SET S32	
S32 X2 ⊣/⊢	(Y2)	Right shifting
X2 ⊣⊢	SET S50	
S40	SET Y4	Clipping tightly
	TMR T1 K30	
T1 ⊣⊢	SET S41	
S41 X4 ⊣/⊢	(Y0)	Robot arm ascending to upper limit (X4 On)
X4 ⊣⊢	SET S42	
S42 X3 ⊣/⊢	(Y2)	Right shifting
X3 ⊣⊢	SET S50	
S50 X5 ⊣/⊢	(Y1)	Robot arm descending
X5 ⊣⊢	SET S60	
S60	RST Y4	Clipping released
	TMR T2 K30	
T2 ⊣⊢	SET S70	
S70 X4 ⊣/⊢	(Y0)	Robot arm ascending to upper limit (X4 On)
X4 ⊣⊢	SET S80	
S80 X1 ⊣/⊢	(Y3)	Robot arm left shifting to left limit (X1 On)
X1 ⊣⊢	(S2)	
	RET	
	END	

Remarks:

Flag explanations:

M1040:   When On, all step operations are forbidden.

1.   Manual mode: M1040 keeps being On

2.   Zero return/one cycle operation mode: Between the timing of pressing "auto stop" and "auto start" buttons, M1040 will keep being On.

3.   Step mode: M1040 keeps being On until "auto start" button is pressed.

4.   Continuous operation mode: When PLC goes from STOP to RUN, M1040 will keep being On and turn Off when "auto start" button is pressed.

M1041:   Step operation starts. Special M for initial S2 to move to the next step.

1.   Manual/zero return mode: M1041 keeps being Off.

2.   Step/one cycle operation mode: M1041 will only be On when "auto start" button is pressed.

3.   Continuous operation mode: M1041 keeps On when "auto start" button is pressed; Off when "auto stop" button is pressed.

M1042:   Enabling pulse output. Sending pulses once when "auto start" button is pressed.

M1043:   On when zero return is completed.

M1044:   In continuous operation mode, M1044 has to be On to more S2 to the next step.

M1045:   All output resets are forbidden.

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)

1.   When M1045 is Off, and any of the S among $D_1 \sim D_2$ is On, SET Y output and the step in action will be reset to Off.

2.   When M1045 is On, SET Y output will be remained but the step in action will be reset to Off

If the machine executes zero return (at the zero point) and goes from zero return (S1) to manual (S0), no matter M1045 is On or Off, SET Y output will be remained but the step in action will be reset to Off.

M1046:   STL state setting. On when any of the steps is On. When M1047 is forced On, On of any S will result in On of M1046. D1040 ~ D1047 will record the No. of the previous 8 points before On of S.

M1047:   On for enabling STL monitor. When IST instruction starts to be executed, M1047 will be forced On. In every scan time, as long as IST instruction is still On, M1047 will be forced On. M1047 monitors all the S.

D1040 ~
D1047:   On status of step No. 1 ~ 8

API	Mnemonic			Operands				Function	Controllers
61	D	SER	P	S₁ S₂ D n				Search a Data Stack	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SER, SERP: 9 steps
S₁							*	*	*	*	*	*	*			DSER, DSERP: 17 steps
S₂					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*			
n					*	*							*			

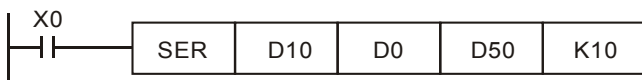|	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Start device for data stack comparison   **S₂**: Data to be compared   **D**: Start device for storing comparison result   **n**: Length of data to be compared

Explanations:

1. When **S₂** are used in device F, only 16-bit instruction is applicable.

2. **D** will occupy 5 consecutive points.

3. Range of **n**: for 16-bit instruction 1 ~ 256; for 32-bit instruction 1 ~ 128.

4. See the specifications of each model for their range of use.

5. The **n** data in the registers starting from **S₁** are compared with **S₂** and the results are stored in the registers starting from **D**.

6. In the 32-bit instruction, **S₁**, **S₂**, **D** and **n** will designate 32-bit registers.

7. For **D**, the 16-bit counters and 32-bit counters in SA/SX/SC series MPU cannot be mixed when being used.

Program Example:

1. When X0 = On, the data stack consist of D10 ~ D19 will be compared against D0 and the result will be stored in D50 ~ D52. If there are equivalent values appearing during the comparison, D50 ~ D52 will all be 0.

2. The data are compared algebraically. (-10 < 2).

3. The No. of the register with the smallest value among the compared data will be recorded in D53; the biggest will be recorded in D54. When there are more than one smallest value or biggest value, device D will record the No. of the register with bigger value.

X0					
⊣├	SER	D10	D0	D50	K10

S₁	Content	Data to be compared	Data No.	Result
D10	88	S₂	0	
D11	100		1	Equal
D12	110		2	
D13	150		3	
D14	100	D0 = K100	4	Equal
D15	300		5	
D16	100		6	Equal
D17	5		7	Smallest
D18	100		8	Equal
D19	500		9	Biggest

D	Content	Description
D50	4	Total number of data with equivalent values
D51	1	No. of the first equivalent value
D52	8	No. of the last equivalent value
D53	7	No. of the smallest value
D54	9	No. of the biggest value

API	Mnemonic		Operands	Function		Controllers
62	D	ABSD	S₁ S₂ D n	Absolute Drum Sequencer		ES/EX/SS **SA/SX/SC EH/SV**

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ABSD: 9 steps
S₁							*	*	*	*	*	*	*			DABSD: 17 steps
S₂											*	*	*			
D		*	*	*												
n					*	*										

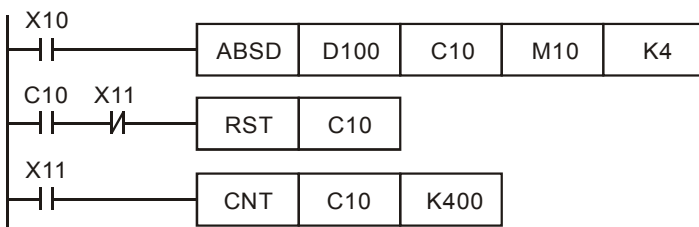| PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Start device in the data table   **S₂**: No. of counter   **D**: Start No. of the devices for the comparison results

n: Number of data for comparison

Explanations:

1. When **S₁** designates KnX, KnY, KnM and KnS, the 16-bit instruction has to designate K4 and 32-bit instruction has to designate K8.

2. Range of **n**: 1 ~ 64

3. See the specifications of each model for their range of use.

4. ABSD instruction is for the absolute control of the multiple output pulses generated by the present value in the counter.

5. **S₂** 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.

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 comprison results will be stored in M10 ~ M13.

3. When X10 = Off, the original On/Off status of M10 ~ M13 will be remained.

```
 X10
──┤├──────┌──────┬──────┬──────┬──────┐
          │ ABSD │ D100 │ C10  │ M10  │ K4 │
          └──────┴──────┴──────┴──────┘
 C10  X11
──┤├──┤/├──┌──────┬──────┐
          │ RST  │ C10  │
          └──────┴──────┘
 X11
──┤├──────┌──────┬──────┬──────┐
          │ CNT  │ C10  │ K400 │
          └──────┴──────┘
```

4. M10~ M13 will be On when the present value in C10 ≦ upper bound value or ≧ lower bound value.

Lower bound value	Upper bound value	Present value in C10	Output
D100 = 40	D101 = 100	40 ≦ C10 ≦ 100	M10 = On
D102 = 120	D103 = 210	120 ≦ C10 ≦ 210	M11 = On

Lower bound value	Upper bound value	Present value in C10	Output
D104 = 140	D105 = 170	140 $\leq$ C10 $\leq$ 170	M12 = On
D106 = 150	D107 = 390	150 $\leq$ C10 $\leq$ 390	M13 = On

5. If the lower bound value > upper bound value, when C10 < upper bound value (60) or > upper bound value (140), M12 will be On.

Lower bound value	Upper bound value	Present value in C10	Output
D100 = 40	D101 = 100	40 $\leq$ C10 $\leq$ 100	M10 = On
D102 = 120	D103 = 210	120 $\leq$ C10 $\leq$ 210	M11 = On
D104 = 140	D105 = 60	60 $\leq$ C10 $\leq$ 140	M12 = On
D106 = 150	D107 = 390	150 $\leq$ C10 $\leq$ 390	M13 = On
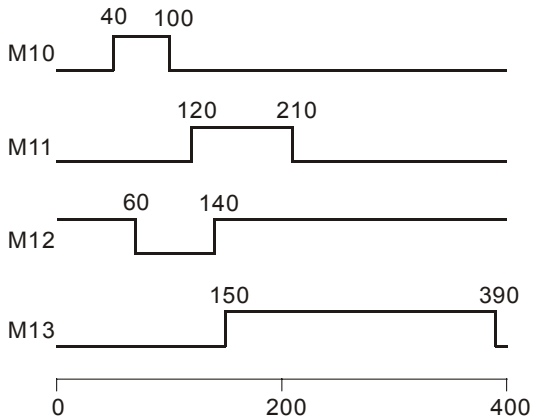
API	Mnemonic	Operands	Function	Controllers
63	INCD	S₁ S₂ D n	Incremental Drum Sequencer	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	INCD: 9 steps
S₁							*	*	*	*	*	*	*			
S₂												*				
D		*	*	*												
n					*	*										

| PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Start device in the data table   **S₂**: No. of counter   **D**: Start No. of the devices for the comparison results

n: Number of data for comparison

Explanations:

1. When **S₁** designates KnX, KnY, KnM and KnS, it has to designate K4.

2. In the 16-bit instruction, **S₂** has to designate C0 ~ C198 and will occupy 2 consecutive No. of counters.

3. Range of **n**: 1 ~ 64

4. See the specifications of each model for their range of use.

5. Flag: M1029 (instruciton execution completed)

6. INCD instruction is for the relative control of the multiple output pulses generated by the present value in the counter.

7. The present value in **S₂** is compared with **S₁**. **S₂** will be reset to 0 whenever a comparison is completed. The current number of data processed in temporarily stored in **S₂** + 1.

8. When n data have been processed, M1029 will be On for one scan period.

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 present value in C10 is compared against the set values in D100 ~ D104. The present value will be reset to 0 whenever a comparison is completed.

3. The current number of data having been processed is temporarily stored in C11.

4. The number of times of reset is temporarily stored in C11.

5. Whenever the content in C11 pluses 1, M10 ~ M14 will also correspondingly change. See the timing diagram below.

6. After the 5 groups of data have been compared, M1029 will be On for one scan period.

7. When X0 goes from On to Off, C10 and C11 will both be reset to 0 and M10 ~ M14 will all be Off. When X0 is On again, the instruction will start its execution again from the beginning.
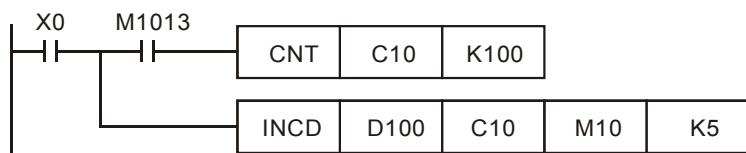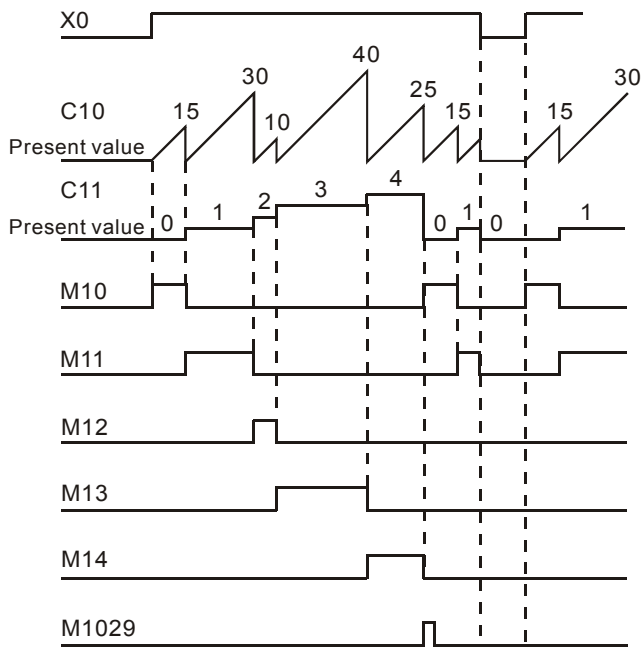
API	Mnemonic	Operands	Function	Controllers
64	TTMR	**D** **n**	Teaching Timer	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TTMR: 5 steps
D													*			
n					*	*										

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device No. for storing the "On" time of button switch   **n**: Multiple setting

Explanations:

1. **D** will occupy 2 consecutive devices.

2. Range of **n**: 0 ~ 2

3. See the specifications of each model for their range of use.

4. For SA series MPU, TTMR instruction can be used 8 times in the program.

5. The "On" time (unit: 100ms) of the external button switch is stored in device No. **D** + 1. The "On" time (unit: second) of the switch is multiplied by **n** and stored in **D**.

6. Multiple setting:

   When n = 0, unit of **D** = second

   When n = 1, unit of **D** = 100ms (D × 10)

   When n = 2, unit of **D** = 10ms (D × 100)

Program Example 1:

1. The "On" (being pressed) time of button switch X0 is stored in D1. The setting of n is stored in D0. Therefore, the button switch will be able to adjust the set value in the timer.

2. When X0 goes Off, the content in D1 will be cleared to 0, but the content in D0 will remain.
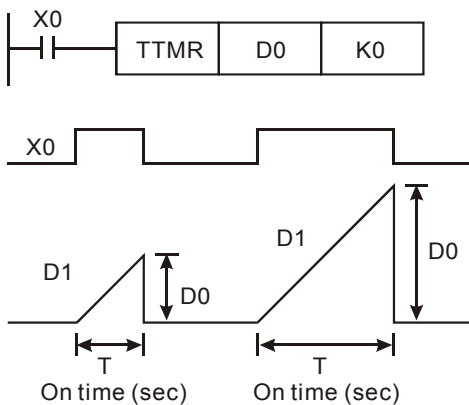
3. Assume the "On" time of X0 is T (sec.), see the relation between D0, D1 and n in the table below.

n	D0	D1 (unit: 100ms)
K0 (unit: s)	1 × T	D1 = D0 × 10
K1 (unit: 100 ms)	10 × T	D1 = D0
K2 (unit: 10 ms)	100 × T	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 timing unit for timer T0 ~ T9 is 0.1 sec. The timing unit for the teaching timer is 1 sec.

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. Store the "On" time (sec.) of X10 in D200.

6. M0 refers to the pulses generated from one scan period after the button switch of the teaching timer X10 is released.

7. Use the set number of the DIP switch as the indirectly designated pointer and send the content in D200 to D100E (D100 ~ D109).

```
 M10
──┤├──      ┌──────┬──────┬──────┐
            │ TMR  │  T0  │ D100 │
            └──────┴──────┴──────┘
 M11
──┤├──      ┌──────┬──────┬──────┐
            │ TMR  │  T1  │ D101 │
            └──────┴──────┴──────┘
                      ⬇

 M19
──┤├──      ┌──────┬──────┬──────┐
            │ TMR  │  T9  │ D109 │
            └──────┴──────┴──────┘
 M1000
──┤├──      ┌──────┬──────┬──────┐
            │ BIN  │ K1X0 │  E   │
            └──────┴──────┴──────┘
 X10
──┤├──      ┌──────┬──────┬──────┐
            │ TTMR │ D200 │  K0  │
            └──────┴──────┴──────┘
 X10
──┤├──      ┌──────┬──────┐
            │ PLF  │  M0  │
            └──────┴──────┘
 M0
──┤├──      ┌──────┬──────┬──────┐
            │ MOV  │ D200 │ D100E│
            └──────┴──────┴──────┘
```
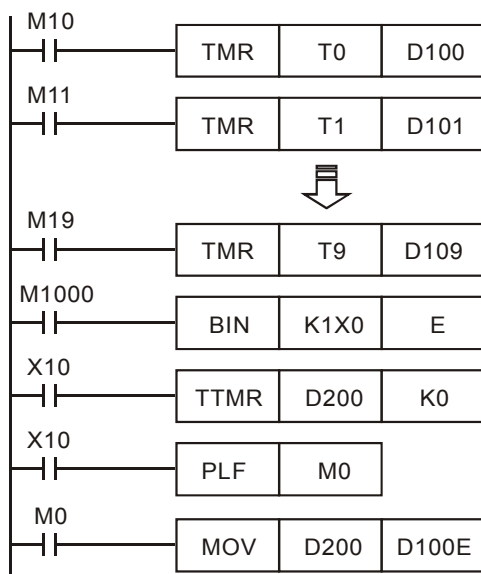
Remarks:

1. For SA series MPU, TTMR instruction can be used 8 times in the program. But in a subroutine or interruption subroutine, the instruction can only be used once.

2. For EH series MPU, there is no limitation on the times using this instruction in the program and 8 instructions can be executed at the same time.

API	Mnemonic	Operands			Function	Controllers
65	STMR	S	m	D	Special Timer	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	STMR: 7 steps
S											*					
m			*	*	*											
D		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: No. of timer   **m**: Set value in timer (unit: 100ms)   **D**: No. of start output device

Explanations:

1. Range of **S**: for SA/SX/SC T0 ~ T191; for EH/EH2/SV T0 ~ T199

2. Range of **m**: 1 ~ 32,767

3. **D** will occupy 4 consecutive devices.

4. See the specifications of each model for their range of use.

5. STMR instruction is used for Off-delay, one shot timer and flashing sequence.

6. The No. of timers designated by STMR instructions can be used only once.

Program Example:

1. When X10 = On, STMR instruction will designate timer T0 and set the set value in T0 as 5 seconds.

2. Y0 is the contact of Off-delay. When X10 goes from Off to On, Y0 will be On. When X10 goes from On to Off, Y0 will be Off after a five seconds of delay.

3. When X10 goes from On to Off, there will be a five seconds of Y1 = On output.

4. When X10 goes from Off to On, there will be a five seconds of Y2 = On output.

5. When X10 goes from Off to On, Y3 will be On after a five seconds of delay. When X10 goes from On to Off, Y3 will be Off after a five seconds of delay.
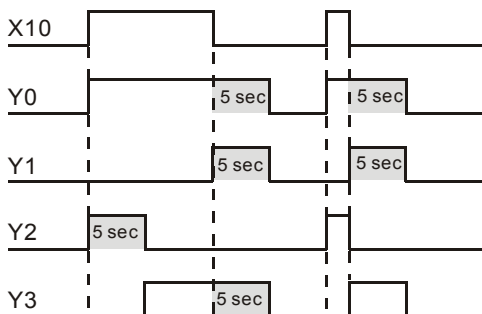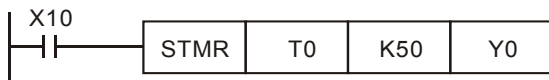
6. Add a b contact of Y3 after X10, and Y1 and Y2 can operate for flashing sequence output. When X10 goes Off, Y0, Y1 and Y3 will be Off and the content in T10 will be reset to 0.
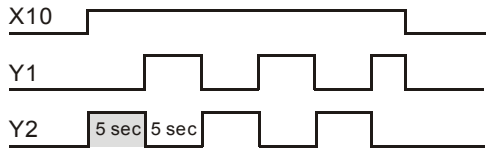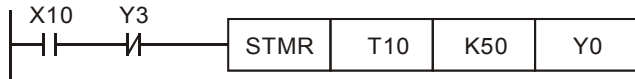
API	Mnemonic		Operands	Function	Controllers
66	ALT	P	**D**	Alternate State	ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Destination device

Explanations:

1. See the specifications of each model for their range of use.

2. When ALT instruction is executed, "On" and "Off" of **D** will switch.

3. This instruction adopts pulse execution instructions (ATLP).

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.
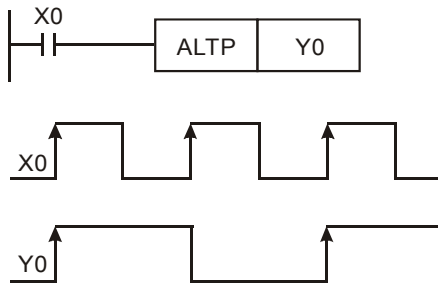
Program Example 2:

Using a single switch to enable and disable control. At the beginning, M0 = Off, so Y0 = On and Y1 = Off. When X10 switches between On/Off for the first time, M0 will be On, so Y1 = On and Y0 = Off. For the second time of On/Off switching, M0 will be Off, so Y0 = On and Y1 = Off.
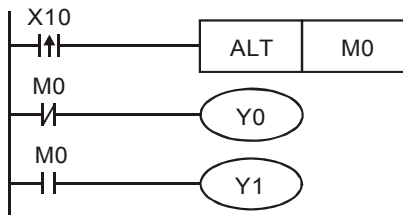
Program Example 3:

Generating flashing. When X10 = On, T0 will generate a pulse every 2 seconds and Y0 output will switch between On and Off following the T0 pulses.

API	Mnemonic	Operands				Function	Controllers
67	RAMP	S_1 S_2 D n				Ramp Variable Value	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices												Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RAMP: 9 steps
S_1													*			
S_2													*			
D													*			
n					*	*										

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Start of ramp signal   **S_2**: End of ramp signal   **D**: Duration of ramp signal   **n**: Scan times

Explanations:

1. Range of **n**: 1 ~ 32,767

2. **D** will occupy 2 consecutive points.

3. See the specifications of each model for their range of use.

4. Flags: M1026 (enabling RAMP; see remarks for more details); M1029 (RAMP execution completed).

5. This instruction is for obtaining slope (the relation between linearity and scan time). Before using this instruction, you have to preset the scan time.

6. The set value of start ramp signal is pre-written in D10 and set value of end ramp signal in D11. When X10 = On, D10 increases towards D11 through n (= 100) scans (the duration is stored in D12). The times of scans are stored in D13.

7. In the program, 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 above program for example, n = J100, the time for D10 to increase to D11 will be 3 seconds (30ms × 100).

8. When X10 goes Off, the instruction will stop its execution. When X10 goes On again, the content in D12 will be reset to 0 for recalculation.

9. When M1026 = Off, M1029 will be On and the content in D12 will be reset to the set value in D10.

Program Example:

When this instruction is used with analog signal outputs, it will be able to buffer START and STOP.

The scan times is stored in D13
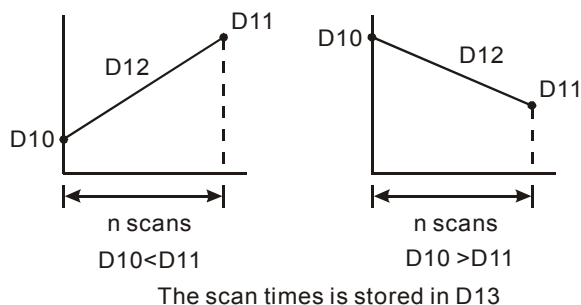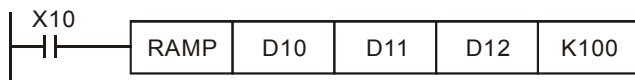
Remarks:

D12 for enabling On/Off of M1026:
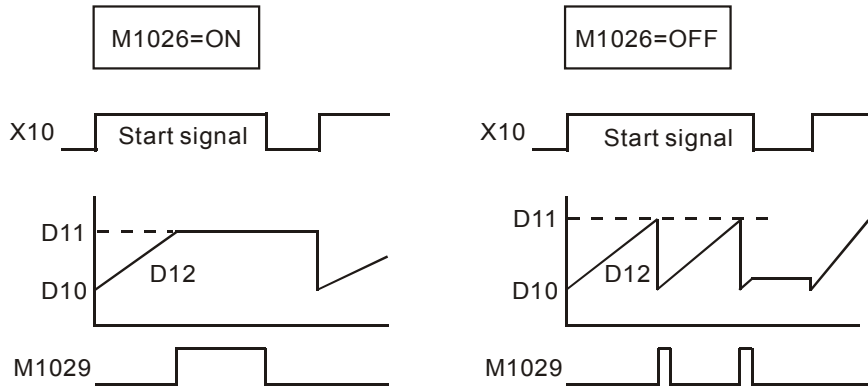
API	Mnemonic	Operands	Function	Controllers
69	SORT	S m₁ m₂ D n	Sort Tabulated Data	ES/EX/SS SA/SX/SC EH/SV

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													*				
m₁					*	*											
m₂					*	*											
D													*				
n					*	*							*				

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for the original data   **m₁**: Groups of data to be sorted   **m₂**: Number of columns of data

D: Start device for the sorted data   **n**: Reference value for data sorting

Explanations:

1. Range of **m₁**: 1 ~ 32.

2. Range of **m₂**: 1 ~ 6

3. Range of **n**: 1 ~ m_2

4. See the specifications of each model for their range of use.

5. Flag: M1029 (SORT execution completed).

6. The sorted result is stored in m1 × m2 registers starting from the device designated in **D**. Therefore, if **S** and **D** designate the same register, the sorted result will be the same as the data designated in **S**.

7. It is better that the start No. designated in **S** is 0.

8. The sorting will be completed after **m₁** times of scans. After the sorting is completed, M1029 will be On.

9. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:

1. When X0 = On, the sorting will start. When the sorting is completed, M1029 will be On. DO NOT change the data to be sorted during the execution of the instruction. If you wish to change the data, please make X0 go from Off to On again.
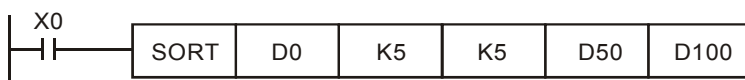
2. Example table of data sorting

Column / Row	Columns of data: m_2				
	Data Column				
	1	2	3	4	5
	Students No.	Physics	English	Math	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

Groups of data: m_1

Sorted data when D100 = K3.

Column / Row	Columns of data: m_2				
	Data Column				
	1	2	3	4	5
	Students No.	Physics	English	Math	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

Groups of data: m_1

Sorted data when D100 = K5.

Column / Row	Columns of data: m_2				
	Data Column				
	1	2	3	4	5
	Students No.	Physics	English	Math	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

Groups of data: m_1

API	Mnemonic		Operands	Function	Controllers
70	D	TKY	Ⓢ Ⓓ₁ Ⓓ₂	Ten Key Input	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		TKY: 7 steps
S	*	*	*	*													DTKY: 13 steps
D₁								*	*	*	*	*	*	*	*		
D₂		*	*	*													

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for key input   **D₁**: Device for storing keyed-in value   **D₂**: Key output signal

Explanations:

1. **S** will occupy 10 consecutive points; **D₂** will occupy 11 consecutive points.

2. See the specifications of each model for their range of use.

3. For SA series MPU, **S** and **D₂** do not support E, F index register modification.

4. This instruction designates 10 external input points (representing decimal numbers 0 ~ 9) starting from **S**. The 10 points are respectively connected to 10 keys. By pressing the keys, you can enter a 4-digit decimal figure 0 ~ 9,999 (16-bit instruction) or a 8-digit figure 0 ~ 99,999,999 (32-bit instruction) and store the figure in **D₁**. **D₂** is used for storing key status.

5. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:

1. Connect the 10 input points starting from X0 to the 10 keys (0 ~ 9). When X20 = On, the instruction will be executed and the keyed-in values will be stored in D0 in bin form. The key status will be stored in M10 ~ M19.
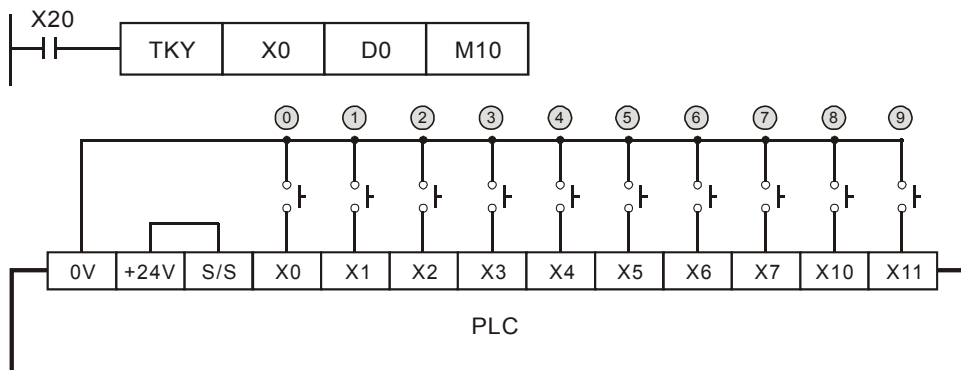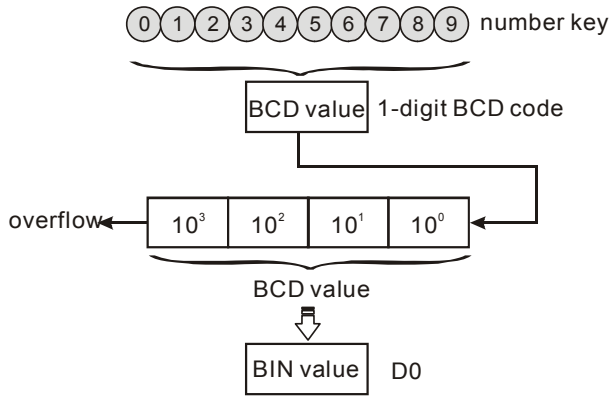
2. As shown in the timing chart below, the 4 points X5, X3, X0, and X1 connected to the keys are entered in order and you can obtain the result 5,301. Store the result in D0. 9,999 is the maximum value allowed to stored in D0. Once the value exceeds 4 digits, the highest digit will overflow.

3. M12 = On when from X2 is pressed to the other key is pressed. Same to other keys.

4. When any of the keys in X0 ~ X11 is pressed, one of M10 ~ 19 will be On correspondingly.

5. M20 = On when any of the keys is pressed.

6. When X20 goes Off, the keyed-in value prior to D0 will remain unchanged, but M10 ~ M20 will all be Off.

API	Mnemonic		Operands	Function	Controllers
71	D	HKY	S D_1 D_2 D_3	Hexadecimal Key Input	ES/EX/SS SA/SX/SC EH/SV

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
S	*															
D₁		*														DHKY: 17 steps
D₂											*	*	*	*	*	
D₃		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

**S**: Start device for key scan input   **D₁**: Start device for key scan output   **D₂**: Device for storing keyed-in value

D₃: Key output signal

Explanations:

1. **S** will occupy 4 consecutive points.

2. **D₁** will occupy 4 consecutive points.

3. **D₃** will occupy 8 consecutive points.

4. See the specifications of each model for their range of use.

5. For SA series MPU, **S**, **D₁** and **D₃** do not support E, F index register modification.

6. Flags: M1029 (On whenever a matrix scan period is completed); M1167 (HKY input modes switch). See remarks for more details.

7. This instruction designates 4 continuous external input points starting from **S** and 4 continuous external input points starting from **D₁** to construct a 16-key keyboard by a matrix scan. The keyed-in value will be stored in **D₂** and **D₃** is used for storing key status. If several keys are pressed at the same time, the first key pressed has the priority.

8. The keyed-in value is termporarily stored in D0. When the 16-bit instruction HKY is in use, 9,999 is the maximum value D0 is able to store. When the value exceeds 4 digits, the highest digit will overflow. When the 32-bit instruction DHKY is in use, 99,999,999 is the maximum value D0 is able to store. When the value exceeds 8 digits, the highest digit will overflow.

9. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:
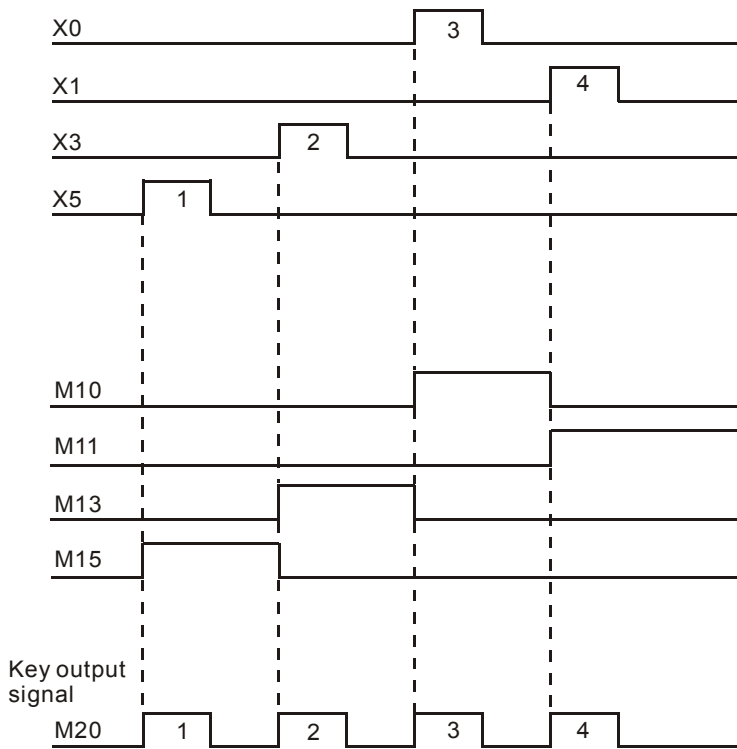
1. Designate 4 input points X10 ~ X13 and the other 4 input points Y10 ~ Y13 to construct a 16-key keyboard. When X4 = On, the instruction will be executed and the keyed-in value will be stored in D0 in bin form. The key status will be stored in M0 ~ M7.

```
    X4
  ──┤├──┤ HKY │ X10 │ Y10 │ D0 │ M0 │
```
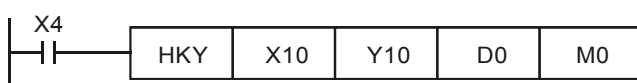
2. Key in numbers:
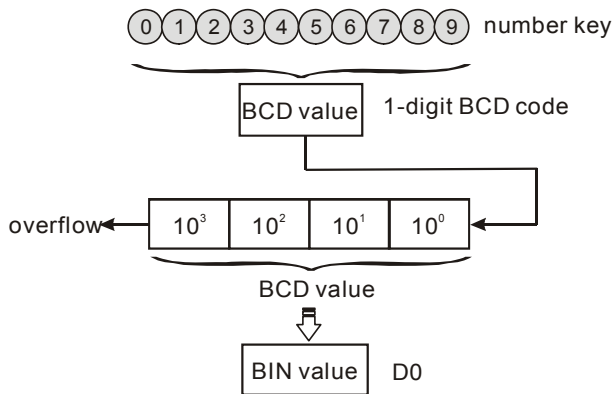
3. Function keys input:

   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) When many keys are pressed at the same time, the
      first key pressed has the priority.
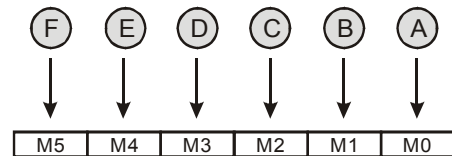
4. Key output signal:

   a) When any of A ~ F is pressed, M6 will be On for once.

   b) When any of 0 ~ 9 is pressed, M7 will be On for once.

5. When X4 goes Off, the keyed-in value prior to D0 will remain unchanged, but M0 ~ M7 will all be Off.

6. External wiring:

PLC (transistor output)
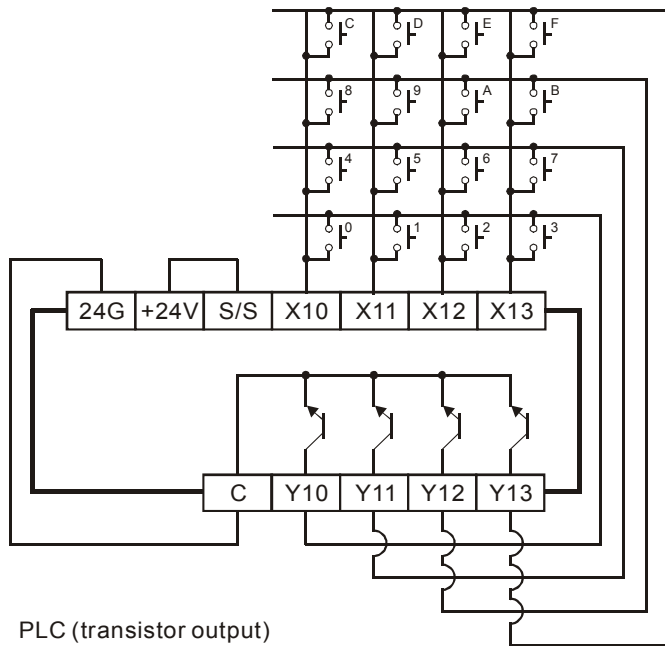
Remarks:

1. When this instruction is being executed, it will require 8 scans to obtain one valid keyed-in value. A scan period that is too long or too short may result in poor keyed-in effect, which can be avoided by the following methods:

    a) If the scan period is too short, I/O may not be able to respond in time, resulting in not being able to read the keyed-in value correctly. In this case, please fix the scan time.

    b) If the scan period is too long, the key may respond slowly. In this case, write this instruction into the time interruption subroutine to fix the time for the execution of this instruction.

2. Functions of M1167:

    a) When M1167 = On, HKY instruction will be able to input the hexadecimal value of 0 ~ F.

    b) When M1167 = Off, HKY instruction will see A ~ F as function keys.

3. Functions of D1037 (only supports EH/EH2/SV series MPU):

    Write D1037 to set the overlapping time for keys (unit: ms). The overlapping time will vary upon different program scan time and the settings in D1037.

API	Mnemonic	Operands				Function	Controllers
72	DSW	S	D_1	D_2	n	Digital Switch	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSW: 9 steps
S	*															
D$_1$		*														
D$_2$											*	*	*			
n					*	*										

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for switch scan input   **D$_1$**: Start device for switch scan output   **D$_2$**: Device for storing the set value of switch   **n**: Groups of switches

Explanations:

1. Range of **n**: 1 ~ 2

2. See the specifications of each model for their range of use.

3. Flag: M1029 (DSW execution completed)

4. This instruction designates 4 or 8 consecutive external input points starting from **S** and 4 consecutive external input points starting from **D$_1$** to scan read 1 or 2 4-digit DIP switches. The set values of DIP switches are stored in **D$_2$**. **n** decides to read 1 or 2 4-digit DIP switches.

5. There is no limitation on the times of using this instruction in the program. However, for SA series MPU, only one instruction can be executed at a time. For EH series MPU, two instructions are allowed to be executed at a time.

Program Example:

1. The first group of DIP switches consist of X20 ~ X23 and Y20 ~ Y23. The second group of switches consist of X24 ~ X27 and Y20 ~ Y23. When X10 = On, the instruction will be executed and the set values of the first group switches will be read and converted into bin values before being stored in D20. The set values of the second group switches will be read, converted into bin values and stored in D21.

```
   X10
  ──┤├──  DSW │ X20 │ Y20 │ D20 │ K2
```
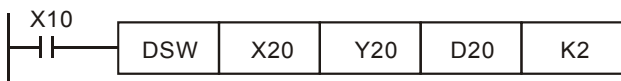
2. When X10 = On, the Y20 ~ Y23 auto scan cycle will be On. Whenever a scan cycle is completed, M1029 will be On for a scan period.

3. Please use transistor output for Y20 ~ Y23. Every pin 1, 2, 4, 8 shall be connected to a diode (0.1A/50V) before connecting to the input terminals on PLC.
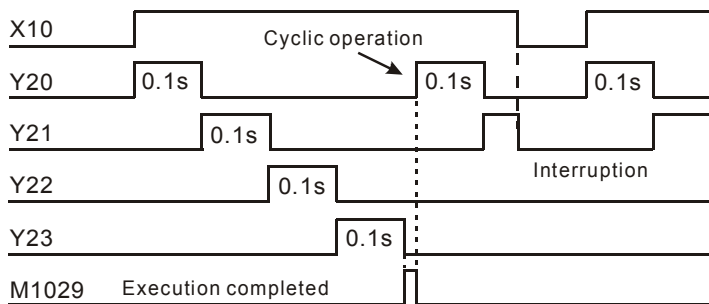
4. Wiring for DIP swich input:
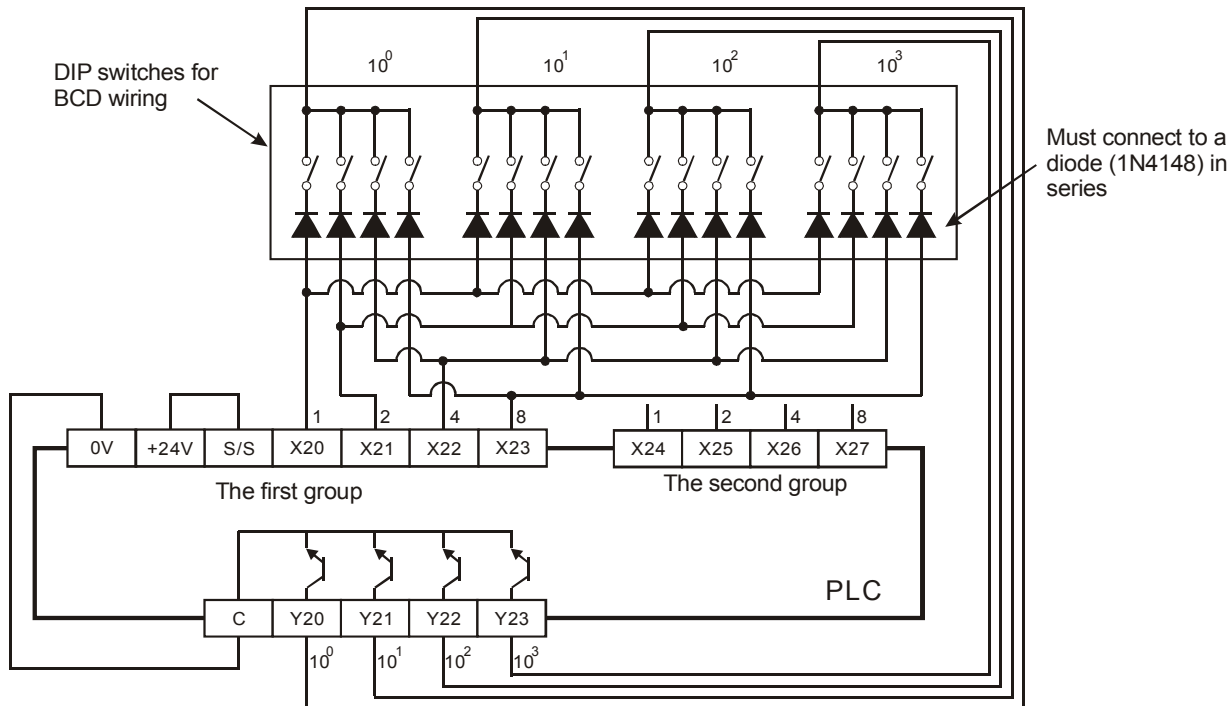
Remarks:

1. When **n** = K1, **D₂** will occupy one register. When **n** = K2, **D₂** will occupy 2 consecutive registers.

2. Follow the methods below for the transistor scan output:

   a) When X10 = On, DSW instruction will be executed. When X10 goes Off, M10 will keep being On until the scan output completes a scan cycle and go Off.

   b) When X10 is used as a button switch, whenever X10 is pressed once, M10 will be reset to Off when the scan output designated by DSW instruction completes a scan cycle. The DIP switch data will be read completely and the scan output will only operate during the time when the button switch is pressed. Therefore, even the scan output is a transistor type, the life span of the transistor can be extended because it does not operate too frequently.

API	Mnemonic		Operands	Function		Controllers
73	SEGD	P	**S** **D**	Seven Segment Decoder		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices										Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SEGD, SEGDP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	
D							*	*	*	*	*	*	*	*	*	

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device to be decoded   **D**: Output device after the decoding

Explanations:

See the specifications of each model for their range of use.

Program Example:

When X10 = On, the contents (0 ~ F in hex) of the lower 4 bits (b0 ~ b3) of D10 will be decoded into a 7-segment display for output. The decoded results will be stored in Y10 ~ Y17. If the content exceeds 4 bits, the lower 4 bits are still used for the decoding.

```
 X10
──┤├──┌──────┬──────┬──────┐
      │ SEGD │ D10  │ K2Y10│
      └──────┴──────┴──────┘
```
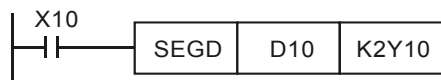
Decoding table of the 7-segment display:

Hex	Bit combi-nation	Composition of the 7-segment display	B0(a)	B1(b)	B2(c)	B3(d)	B4(e)	B5(f)	B6(g)	Data displayed
					Status of each segment					
0	0000		ON	ON	ON	ON	ON	ON	OFF	0
1	0001		OFF	ON	ON	OFF	OFF	OFF	OFF	1
2	0010		ON	ON	OFF	ON	ON	OFF	ON	2
3	0011		ON	ON	ON	ON	OFF	OFF	ON	3
4	0100		OFF	ON	ON	OFF	OFF	ON	ON	4
5	0101		ON	OFF	ON	ON	OFF	ON	ON	5
6	0110		ON	OFF	ON	ON	ON	ON	ON	6
7	0111		ON	ON	ON	OFF	OFF	ON	OFF	7
8	1000		ON	ON	ON	ON	ON	ON	ON	8
9	1001		ON	ON	ON	ON	OFF	ON	ON	9
A	1010		ON	ON	ON	OFF	ON	ON	ON	A
B	1011		OFF	OFF	ON	ON	ON	ON	ON	b
C	1100		ON	OFF	OFF	ON	ON	ON	OFF	C
D	1101		OFF	ON	ON	ON	ON	OFF	ON	d
E	1110		ON	OFF	OFF	ON	ON	ON	ON	E
F	1111		ON	OFF	OFF	OFF	ON	ON	ON	F

API	Mnemonic	Operands	Function	Controllers
74	SEGL	S D n	Seven Segment with Latch	ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device to be displayed in 7-segment display   **D**: Start device for 7-segment display scan output

n: Polarity setting of output signal and scan signal

Explanations:

1. Range of **n**: 0 ~ 7. See remarks for more details.

2. For ES series MPU, the instruction can only be used once in the program. For EH series MPU, the instruction can be used twice in the program. For SA series MPU, there is no limitation on the times of using the instruction, but only one instruction can be executed at a time.

3. For ES/EX/SS/SA/SX/SC series MPU, the last digit of **D** should be 0 and it does not support E, F index register modification.

4. Flag: M1029 (SEGL execution completed)

5. This instruction occupies 8 or 12 continuous external input points starting from **D** for displaying 1 or 2 4-digit 7-segment display data and outputs of scanned signals. Every digit carries a 7-segment display drive (to convert the BCD codes into 7-segment display signal). The drive also carries latch control signals to retain the 7-segment display.

6. **n** decides there be 1 group or 2 groups of 4-digit 7-segment display and designates the polarity for the output.

7. When there is 1 group of 4-digit output, 8 output points will be occupied. When there are 2 groups of 4-digit output, 12 output points will be occupied.

8. When this instruction is being executed, the scan output terminals will circulate the scan in sequence. When the drive contact of the instruction goes from Off to On again, the scan output terminal will restart the scan again.

Program Example:

1. When X10 = On, this instruction starts to be executed, Y10 ~ Y17 construct a 7-segment display scan circuit. The value in D10 will be converted into BCD codes and sent to the first group 7-segment display. The value in D11 will be converted into BCD codes as well and sent to the second group 7-segment display. If the values in D10 and D11 exceed 9,999, operational error will occur.

```
  X10
──┤├──┬──────┬──────┬──────┬──────┐
      │ SEGL │ D10  │ Y10  │  K4  │
      └──────┴──────┴──────┴──────┘
```
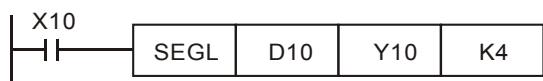
2. When X10 = On, Y14 ~ Y17 will circulate the scan automatically. Every cycle requires 12 scan period. Whenever a cycle is completed, M1029 will be On for a scan period.

3. When there is 1 group of 4-digit 7-segment display, **n** = 0 ~ 3.

   a) Connect the already decoded 7-segment display terminals 1, 2, 4, 8 in parallel an connect them to Y10 ~ Y13 on the PLC. Connect the latch terminals of each digit to Y14 ~ Y17 on the PLC.

   b) When X10 = On, the instruction will be executed and the content in D10 will be sent to the 7-segment displays in sequence by the circulation of Y14 ~ Y17.

4. When there is 2 groups of 4-digit 7-segment display, **n** = 4 ~ 7.

   a) Connect the already decoded 7-segment display terminals 1, 2, 4, 8 in parallel an connect them to Y20 ~ Y23 on the PLC. Connect the latch terminals of each digit to Y14 ~ Y17 on the PLC.

   b) The contents in D10 are sent to the first group 7-segment display. The contents in D11 are sent the the second group 7-segment display. If D10 = K1234 and D11 = K4321, the first group will display 1 2 3 4, and the second group will display 4 3 2 1.

5. Wiring of the 7-segment display scan output:
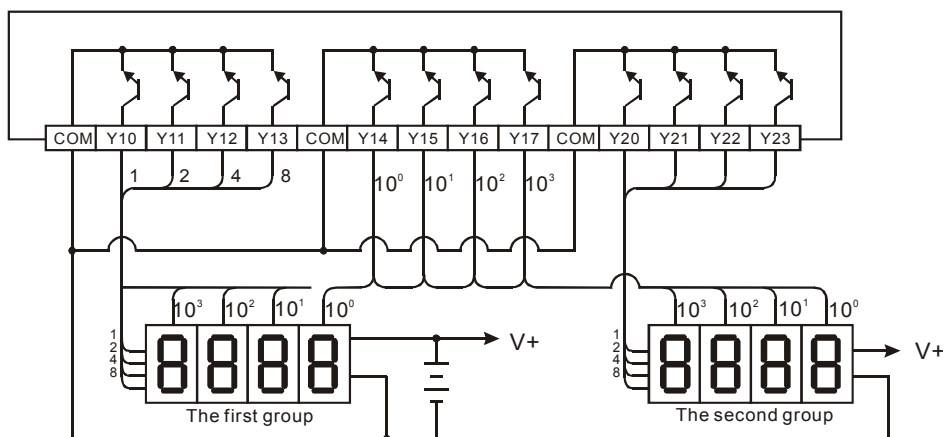
Remarks:

1. ES/EX/SS series MPU (V4.9 and above) supports this instruction but only supports 1 group of 4-digit 7-segment display and 8 points of output. This instruction can only be used once in the program. Range of **n**: 0 ~ 3.

2. **D** of ES/EX/SS series MPU can only designate Y0.

3. When this instruction is executed, the scan time has to be longer than 10ms. If the scan time is shorter than 10ms, please fix the scan time at 10ms.

4. **n** is for setting up the polarity of the transistor output and the number of groups of the 4-digit 7-segment display.

5. 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). Therefore, when output point Y is On, the signal output will be in low voltage.
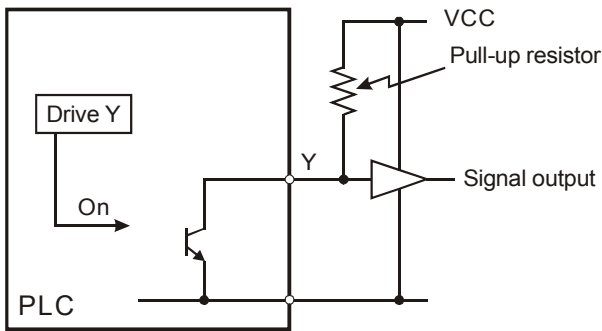
6. Positive logic (negative polarity) output of BCD code

BCD value				Y output (BCDcode)				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

7. Negative logic (positive polarity) output of BCD code

BCD value				Y output (BCDcode)				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

8. Scan latched signal display

Positive logic (negative polarity)		Negative logic (positive polarity)	
Y output (latch)	Output signal	Y output (latch)	Output signal
1	0	0	1

9. Settings of **n**:

Groups of 7-segment display	1 group				2 groups			
Y output of BCD code	$+$		$-$		$+$		$-$	
Scan latched signal display	$+$	$-$	$+$	$-$	$+$	$-$	$+$	$-$
n	0	1	2	3	4	5	6	7

$+$: Positive logic (negative polarity) output $-$: Negative logic (positive polarity) output

10. The polarity of transistor output and the polarity of the 7-segment display input can be the same or different by the setting of **n**.

API	Mnemonic	Operands	Function	Controllers
75	ARWS	S D_1 D_2 n	Arrow Switch	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ARWS: 9 steps
S	*	*	*	*												
D₁											*	*	*	*	*	
D₂		*														
n					*	*										

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for key input   **D₁**: Device to be displayed in 7-segment display   **D₂**: Start device for 7-segment display scan output   **n**: Polarity setting of output signal and scan signal

Explanations:

1. **S** will occupy 4 consecutive points.

2. Range of **n**: 0 ~ 3. See remarks of API 74 SEGL for more details.

3. There no limitation on the times of using this instruction in the program. However, only one instruction is allowed to be executed at a time.

4. **S** and **D₂** of SA series MPU do not support E, F index register modification and can only designate the devices whose last digit is 0 (e.g. Y0, Y10….)

5. See the specifications of each model for their range of use.

6. The output points designated by this instruction shall be transistor output.

7. When using this instruction, please fix the scan time, or place this instruction in the time interruption subroutine (I6□□ ~ I8□□).

Program Example:

1. When this instruction is executed, X20 is defined as down key, X21 is defined as up key, X22 is defined as right key and X23 is defined as left key. The keys are used for setting up and displaying external set values. The set values (range: 0 ~ 9,999) are stored in D20.

2. When X10 = On, digit 10^3 will be the valid digit for setup. If you press the left key at this time, the valid digit will circulate as $10^3 \rightarrow 10^0 \rightarrow 10^1 \rightarrow 10^2 \rightarrow 10^3 \rightarrow 10^0$.

3. If you press the right key at this time, the valid digit will circulate as $10^3 \rightarrow 10^2 \rightarrow 10^1 \rightarrow 10^0 \rightarrow 10^3 \rightarrow 10^2$. During the circulation, the digit indicators connected Y24 ~ Y27 will also be On interchangeably following the circulation.

4. If you press the up key at this time, the valid digit will change as $0 \rightarrow 1 \rightarrow 2 … \rightarrow 8 \rightarrow 9 \rightarrow 0 \rightarrow 1$. If you press the down key, the valid digit will change as $0 \rightarrow 9 \rightarrow 8 … \rightarrow 1 \rightarrow 0 \rightarrow 9$. The changed value will also be displayed in the 7-segment display.

```
   X10
 ──┤├──── ARWS │ X20 │ D20 │ Y20 │ K0
```
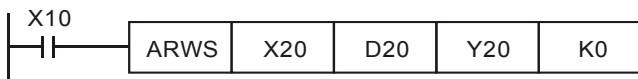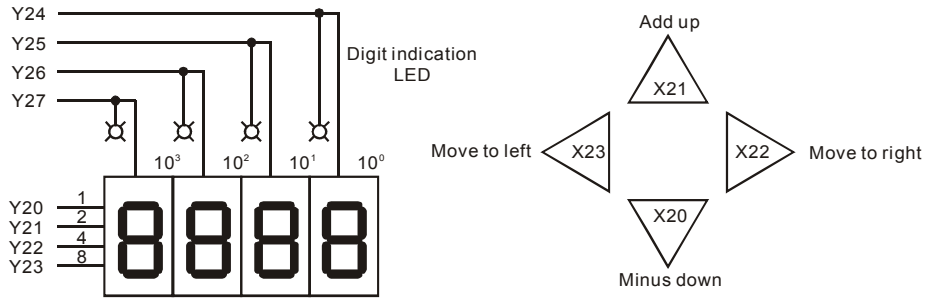
7-segment display for the 4-digit set value

The 4 switches are used for moving the digits and increasing/decreasing set values.

API	Mnemonic	Operands	Function	Controllers
76	ASC	**S** **D**	ASCII Code Conversion	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: English letter to be converted into ASCII code   **D**: Device for storing ASCII code

Explanations:

1. **S**: enter 8 Engligh letters by using WPLSoft on computer or enter ASCII code by HPP.

2. See the specifications of each model for their range of use.

3. Flag: M1161 (8/16 bit mode switch)

4. If the execution of this instruction is connected to a 7-segment display, the error message can be displayed by English letters.

Program Example:

1. When X0 = On, convert A ~ H into ASCII code and stored it in D0 ~ D3.
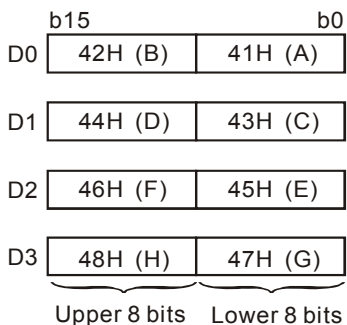
2. When M1161 = On, every ASCII code converted from the letters will occupy the lower 8 bits (b7 ~ b0) of a register. The upper 8 bits are invalid (filled by 0). One register stores a letter.
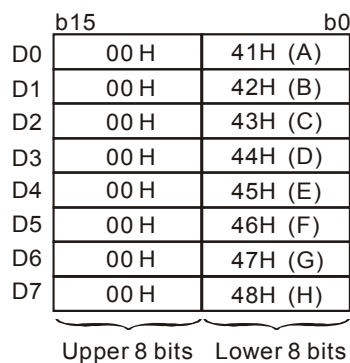
API	Mnemonic	Operands		Function		Controllers
77	PR	S D		Print (ASCII Code Output)		ES/EX/SS SA/SX/SC EH/SV

	Type	Bit Devices				Word Devices										Program Steps
OP		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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Device for storing ASCII code   **D**: External ASCII code output points

Explanations:

1. **S** will occupy 4 consecutive points.

2. **D** will occupy 10 consecutive points.

3. This instruction can only be used twice in the program.

4. See the specifications of each model for their range of use.

5. Flags: M1029 (PR execution completed); M1027 (number of PR outputs)

6. This instruction will output the ASCII codes in the 4 registers starting from **S** from the output devices in the order designated in **D**.

Program Example 1:

1. Use API 76 ASC to convert A ~ H into ASCII codes and store them in D0 ~ D3 and use this instruction to output the codes in sequence.

2. When M1027 = Off and X10 goes On, the instruction will be executed. Designate Y10 (low bits) ~ Y17 (high bits) as the data output points and Y20 for scan signals. Designate Y21 for the monitor signals during the execution. In this mode, you can execute an output for 8 letters in sequence. During the output, if the drive contact goes Off, the data output will stop immediately and all the outputs will go Off.

3. During the execution of the instruction, when X10 goes Off, all the data output will be interrupted. When X10 is On again, the output will be restarted.
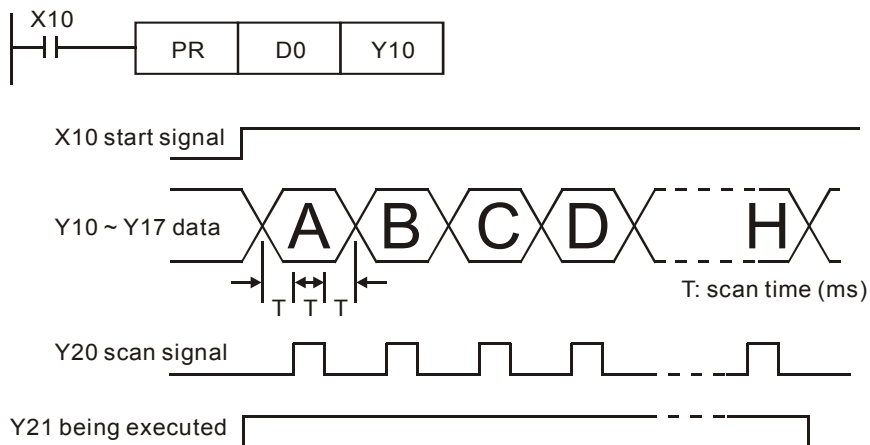
Program Example 2:

1. PR instruction is for outputing a string of 8 bits. When the special auxiliary relay M1027 = Off, PR is able to execute an output of maximum 8 letters in string. When M1027 = On, PR is able to execute an output of 1 ~ 16 letters in string.

2. When M1027 = On and X10 goes from Off to On, the instruction will be executed. Designate Y10 (low bits) ~ Y17 (high bits) as the data output points and Y20 for scan signals. Designate Y21 for the monitor signals during the execution. In this mode, you can execute an output for 16 letters in sequence. During the output, if the drive contact goes Off, the data output will stop after it is completed.

3. When the string encounters 00H (NUL), the string output will finish. The letters coming after it will not be processed.

4. When X10 goes from On to Off, the data output will automatically stop after one cycle. If X10 keeps being On, M1029 will not be enabled.
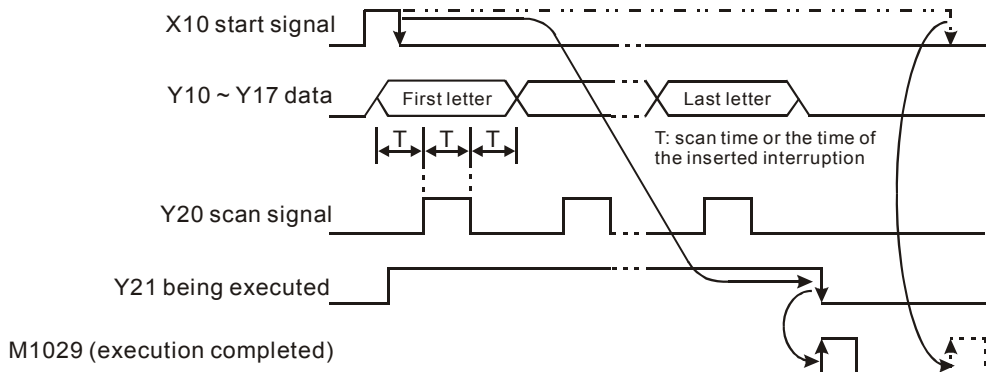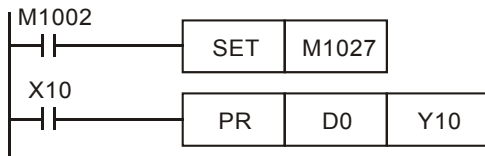
Remarks:

1. Please use transistor output for the output designated by this instruction.

2. When using this instruction, please fix the scan time or place this instruction in a timed interruption subroutine.

API	Mnemonic			Operands				Function	Controllers
78	D	FROM	P	(m₁) (m₂) (D) (n)				Read CR Data in Special Modules	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FROM, FROMP: 9 steps
m₁					*	*							*			
m₂					*	*							*			DFROM, DFROMP: 17 steps
D								*	*	*	*	*	*	*	*	
n					*	*							*			

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

m₁: No. of special module   **m₂**: 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. Range of **m₁** (16-bit and 32-bit): for ES/SA: 0 ~ 7, for EH/EH2 :0 ~ 255, for SV: 0 ~ 107.

2. Range of **m₂** (16-bit and 32-bit): for ES/SA: 0 ~ 48, for EH: 0 ~ 254, for EH2/SV: 0 ~ 499.

3. Range of **n**:

   a) 16-bit: for ES/SA: 1 ~ (49 – **m₂**), for EH: 1 ~ (255 – **m₂**), for EH2/SV: 1 ~ (500 – **m₂**).

   b) 32-bit: for ES/SA: 1 ~ (49 – **m₂**)/2, for EH: 1 ~ (255 – **m₂**)/2, for EH2/SV: 1 ~ (500 – **m₂**)/2.

4. ES series MPU does not support E, F index register modification.

5. **m₁**, **m₂** and **n** of EH series MPU do not support word device D.

6. Flag: M1083 (On when allowing interruptions during FROM/TO instruction).

7. This instruction is for reading the data in the CR in special modules.

8. The 16-bit instruction can designate **D** = K1 ~ K4; the 32-bit instruction can designate **D** = K1 ~ K8.

Program Example:

1. Read CR#29 of special module No.0 into D0 and CR#30 into D1. Only 2 groups of data is read at a time (n = 2).

2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the data read will not be changed.

X0	FROM	K0	K29	D0	K2

API	Mnemonic			Operands				Function		Controllers	
79	D	TO	P	(m₁) (m₂) (S) (n)				Write CR Data into Special Modules		ES/EX/SS SA/SX/SC EH/SV	

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TO, TOP: 9 steps
m₁					*	*							*			DTO, DTOP: 17 steps
m₂					*	*							*			
S					*	*	*	*	*	*	*	*	*	*	*	
n					*	*							*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

m₁: No. of special module   **m₂**: 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. Range of **m₁** (16-bit and 32-bit): for ES/SA: 0 ~ 7, for EH/EH2: 0 ~ 255, for SV: 0 ~ 107.

2. Range of **m₂** (16-bit and 32-bit): for ES/SA: 0 ~ 48, for EH: 0 ~ 254, for EH2/SV: 0 ~ 499.

3. Range of **n**:

    a) 16-bit: for ES/SA: 1 ~ (49 – **m₂**), for EH: 1 ~ (255 – **m₂**), for EH2/SV: 1 ~ (500 – **m₂**).

    b) 32-bit: for ES/SA: 1 ~ (49 – **m₂**)/2, for EH: 1 ~ (255 – **m₂**)/2, for EH2/SV: 1 ~ (500 – **m₂**)/2.

4. ES series MPU does not support E, F index register modification.

5. **m₁**, **m₂** and **n** of EH series MPU do not support word device D.

6. Flag: M1083 (On when allowing interruptions during FROM/TO instruction). See remarks for more details.

7. This instruction is for writing the data into the CR in special modules.

8. The 16-bit instruction can designate **S** = K1 ~ K4; the 32-bit instruction can designate **S** = K1 ~ K8.

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. Only 1 group of data is written in at a time (n = 1).

2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the data written will not be changed.

```
   X0
───┤ ├───┤ DTO │ K0 │ K12 │ D10 │ K1 │
```
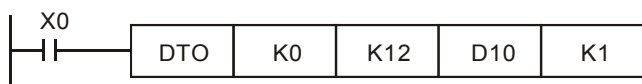
3. Operand rules

    a) **m₁**: The No. of special modules connected to PLC MPU. No. 0 is the module closest to te MPU. Maximum 8 modules are allowed to connected to a PLC MPU and they will not occupy any I/O points.

    b) **m₂**: CR#. CR (control register) is the n 16-bit memories built in the special module, numbered in decimal as #0 ~ #n. All operation status and settings of the special module are contained in the CR.

    c) FROM/TO instruction is for reading/writing 1 CR at a time. DFROM/DTO instruction is for reading/writing 2 CRs at a time.

Higher 16-bit	Lower 16-bit
CR #10	CR #9

← Designated CR number
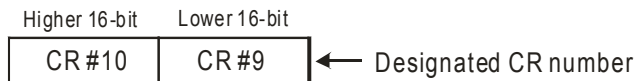
d) Number of groups "n" to be transmitted: n = 2 in 16-bit instructions and n = 1 in 32-bit instructions mean the same.

16-bit instruction when n=6   32-bit instruction when n=3
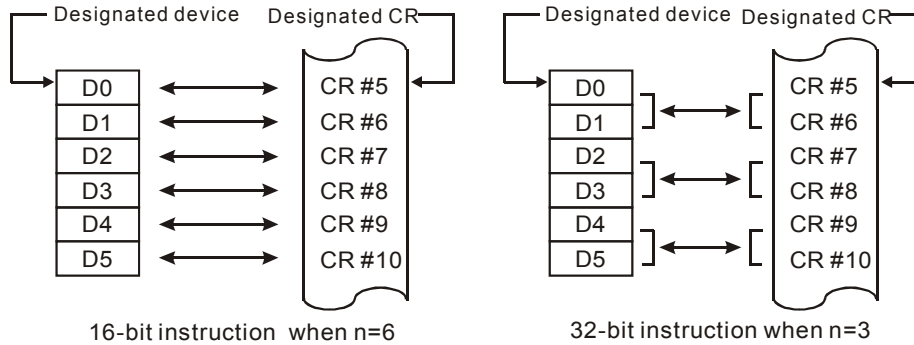
4. ES/EX/SS series MPU does not have M1083. During the execution of FROM/TO instruction, all external or internal interruption subroutines will be forbidden. The interruptions are allowed only after FROM/TO instruction finishes its execution. FROM/TO instruction can also be used in an interruption subroutine.

5. M1083 for switching instruction modes in SA/SX/SC/EH/EH2/SV series MPU:

   a) When M1083 = Off, during the execution of FROM/TO instruction, all external or internal interruption subroutines will be forbidden. The interruptions are allowed only after FROM/TO instruction finishes its execution. FROM/TO instruction can also be used in an interruption subroutine.

   b) When M1083 = On and an interruption signal occurs during the execution of FROM/TO instruction, the interruption will be processed first (with a 100us delay) and the execution of FROM/TO will be stopped. After the interruption subroutine finishes its execution, the program will jump to the next instructio of FROM/TO. FROM/TO cannot be used in an interruption subroutine.

FROM/TO Application Example 1:

Adjust the A/D conversion curve of DVP-04AD. Set the OFFSET value of CH1 as 0V (= $K0_{LSB}$) and GAIN value as 2.5V (= $K2,000_{LSB}$).

1. Write H0 to CR#1 of anlog input module No. 0 and set CH1 as mode 0 (voltage input: -10V ~ +10V).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 ~ CH4.
3. When X0 goes from Off to On, write the OFFSET value $K0_{LSB}$ into CR#18 and the GAIN value $K2,000_{LSB}$ into

CR#24.

FROM/TO Application Example 2:

Adjust the A/D conversion curve of DVP-04AD. Set the OFFSET value of CH2 as 2mA (= K400$_{LSB}$) and GAIN value as 18mA (= K3,600$_{LSB}$).

M1002	TO	K0	K1	H18	K1
	TO	K0	K33	H0	K1
X0	TO	K0	K19	K400	K1
	TO	K0	K25	K3600	K1
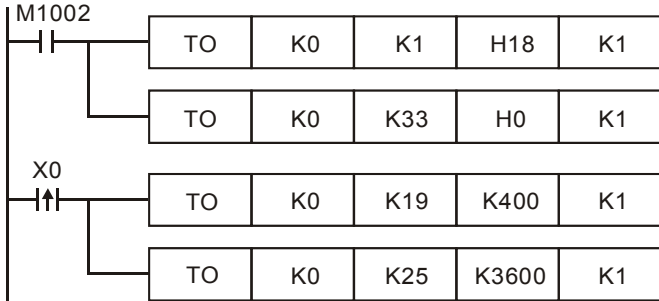
1. Write H18 to CR#1 of anlog input module No. 0 and set CH2 as mode 3 (current input: -20mA ~ +20mA).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 ~ CH4.
3. When X0 goes from Off to On, write the OFFSET value K400 $_{LSB}$ into CR#19 and the GAIN value K3,600 $_{LSB}$ into CR#25.

FROM/TO Application Example 3:

Adjust the D/A conversion curve of DVP-02DA. Set the OFFSET value of CH2 as 0mA (= K0$_{LSB}$) and GAIN value as 10mA (= K1,000$_{LSB}$).

M1002	TO	K1	K1	H18	K1
	TO	K1	K33	H0	K1
X0	TO	K1	K22	K0	K1
	TO	K1	K28	K1000	K1
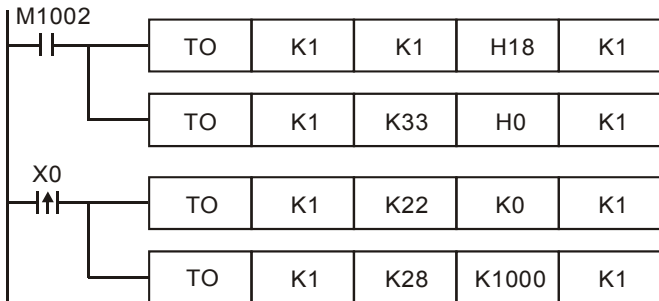
1. Write H18 to CR#1 of anlog output module No. 1 and set CH2 as mode 3 (current output: 0mA ~ +20mA).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 and CH2.
3. When X0 goes from Off to On, write the OFFSET value K0 $_{LSB}$ into CR#22 and the GAIN value K1,000 $_{LSB}$ into CR#28.

FROM/TO Application Example 4:

Adjust the D/A conversion curve of DVP-02DA. Set the OFFSET value of CH2 as 2mA (= K400$_{LSB}$) and GAIN value as 18mA (= K3,600$_{LSB}$).

1. Write H10 to CR#1 of anlog output module No. 1 and set CH2 as mode 2 (current output: +4mA ~ +20mA).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 and CH2.
3. When X0 goes from Off to On, write the OFFSET value K400 $_{LSB}$ into CR#23 and the GAIN value K3,600 $_{LSB}$ into CR#29.

M1002	TO	K1	K1	H10	K1
	TO	K1	K33	H0	K1
X0	TO	K1	K23	K400	K1
	TO	K1	K29	K3600	K1
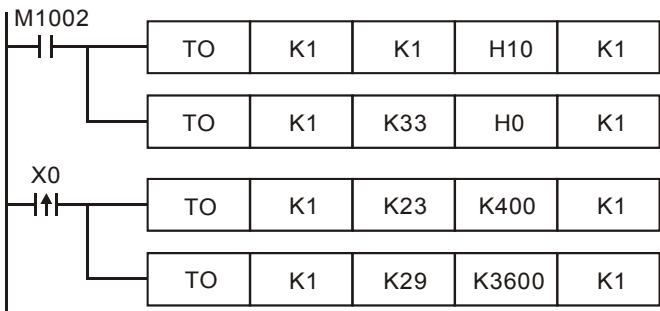
FROM/TO Application Example 5:

When DVP-04AD-S is used with DVP-02DA-S

M1000	FROM	K0	K0	D0	K1
LD= H88 D0	TO	K0	K1	H3030	K1
	TO	K0	K2	K32	K2
	FROM	K0	K6	D20	K4
M1000	FROM	K1	K0	D0	K1
	CMP	H49	D0	M0
M1013	INC	D100
	ADD	D101	K5	D101
LD= K4000 D100	RST	D100
LD= K4000 D101	RST	D101
M1	TO	K1	K1	H10	K1
M1	TO	K1	K10	D100	K2
	END
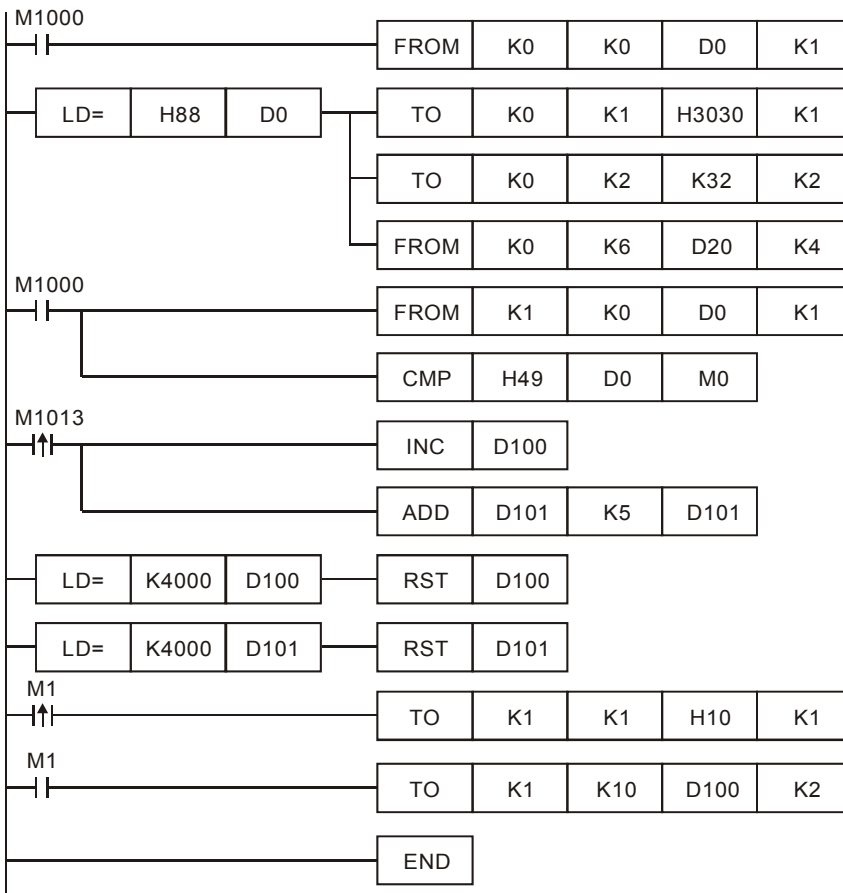
1. Read CR#0 of the extension module No. 0 and see if it is DVP-04AD-S: H88.

2. If D0 = H88, set the input modes: (CH1, CH3) mode 0, (CH2, CH4) mode 3.

3. Set the average times in CH1 and CH2 from CR#2 and CR#3 as K32.

4. Read the average of input signals at CH1 ~ CH4 from CR#6 ~ CR#9 and store the 4 data in D20 ~ D23.

5. Read CR#0 of the extension module No. 1 and see if it is DVP-02DA-S: H49.

6. D100 increases K1 and D101 increases K5 every second.

7. When D100 and D101 reach K4,000, they will be cleared as 0.

8. See if the model is DVP-02DA-S when M1 = On. If so, set up output mode: CH1 in mode 0 and CH2 is mode 2.

9. Write the output settings of D100 and D101 into CR#10 and CR#11. The analog output will change by the changes in D100 and D101.

API	Mnemonic	Operands	Function	Controllers
80	RS	S m D n	Serial Communication Instruction	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		RS: 9 steps
S													*				
m					*	*							*				
D													*				
n					*	*							*				

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for the data to be transmitted   **m**: Length of data to be transmitted   **D**: Start device for receiving data   **n**: Length of data to be received

Explanations:

1. Range of **m**: 0 ~ 256

2. Range of **n**: 0 ~ 256

3. See the specifications of each model for their range of use.

4. ES series MPU does not support E, F index register modification.

5. This instruction is a handy instruction exclusively for MPU to use RS-485 serial communication interface. The user has to pre-store word data in **S** data register, set up data length **m** and the data receiving register **D** and received data length **n**. If E, F index registers are used to modify **S** and **D**, the user cannot change the set values of E and F when the instruction is being executed; otherwise errors may cause in data writing or reading.

6. Designate **m** as K0 if you do not need to send data. Designate **n** as K0 if you do not need to receive data.

7. 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.

8. During the execution of RS instruction, changing the data to be transmitted will be invalid.

9. If the peripheral devices, e.g. AC motor drive, are equipped with RS-485 serial communication and its communication format is open, you can use RS instruction to design the program for the data transmission between PLC and the peripheral device.

10. If the communication format of the peripheral device is Modbus, DVP series PLC offers handy communication instructions API 100 MODRD, API 101 MODWR, and API 150 MODRW, to work with the device. See explanations of the instructions in this application manual.

11. For the special auxiliary relays M1120 ~ M1161 and special data registers D1120 ~ D1131 relevant to RS-485 communication, see remarks for more details.

Program Example 1:

1. Write the data to be transmitted in advance into registers starting from D100 and set M1122 (sending request flag) as On.

2. When X10 = On, RS instruction will be executed and PLC will start to wait for the sending and receiving of data. D100 starts to continuousl send out 10 data and when the sending is over, M1122 will be automatically reset to

Off (DO NOT use the program to execute RST M1122). After 1ms of waiting, PLC will start to receive the 10 data. Store the data in consecutive registers starting from D120.

3. When the receiving of data is completed, M1123 will automatically be On. After the program finishes processing the received data, M1123 has to be reset to Off and the PLC will start to wait for the sending and receiving of data again. DO NOT use the program to continuously execute RST M1123.

```
M1002
──┤├──────┬──────┬─────────────────────┬──────┐
          │ MOV  │ H86   │ D1120 │  Set up communication protocol 9600,7,E,1
          ├──────┼───────┼───────┤
          │ SET  │ M1120 │       │  Retain communication protocol
          ├──────┼───────┼───────┤
          │ MOV  │ K100  │ D1129 │  Set up communication time-out 100ms
Sending request
pulses
──┤↑├─────┬────────────────────────────────────┐
          │ Write in the data to be transmitted in advance
          ├──────┬───────┐
          │ SET  │ M1122 │  Set up sending request
X10
──┤├──────┬──────┬───────┬───────┬───────┐
          │ RS   │ D100  │ K10   │ D120  │ K10
Receiving
completed
──┤├──────┬────────────────────────────┐
          │ Process of received data
M1123     ├──────┬───────┐
          │ RST  │ M1123 │  Receiving of data is completed the flag is reset.
          └──────┴───────┘  The flag is reset.
```

Program Example 2:

Switching between 8-bit mode (M1161 = On) and 16-bit mode (M1161 = Off)

1. 8-bit mode:

   The head code and tail code of the data are set up by M1126 and M1130 together with D1124 ~ D1126. When PLC is executing RS instruction, the head code and tail code set up by the user will be sent out automatically. M1161 = On indicates PLC in 8-bit conversion mode. The 16-bit data will be divided into the higher 8 bits and lower 8 bits. The higher 8 bits are ignored and only the lower 8 bits are valid for data transmission.

```
M1000
──┤├────────( M1161 )
X0
──┤├──────┬──────┬───────┬───────┬───────┐
          │ RS   │ D100  │ K4    │ D120  │ K7
```

Sending data: (PLC -> external equipment)

STX	D100L	D101L	D102L	D103L	ETX1	ETX2

Head code — S source data register, starting from the lower 8 bits of D100 — Tail code 1 / Tail code 2
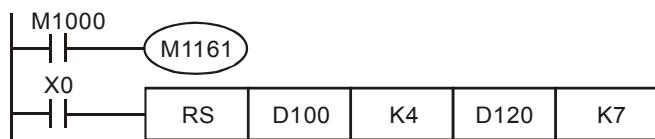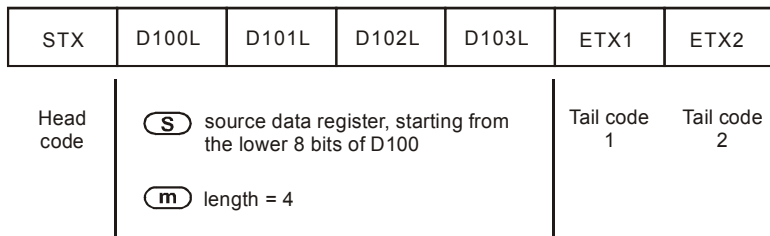
m length = 4

Receiving data: (External equipment -> PLC)

D120L	D121L	D122L	D123L	D124L	D125L	D126L

Head code	Ⓢ received data register, starting from the lower 8 bits of D120	Tail code 1	Tail code 2
	ⓝ length = 7		

When receiving data, PLC will receive the head code and tail code of the data from the external equipment; therefore, the user has to be aware of the setting of data length **n**.

2. 16-bit mode:

    The head code and tail code of the data are set up by M1126 and M1130 together with D1124 ~ D1126. When PLC is executing RS instruction, the head code and tail code set up by the user will be sent out automatically. M1161 = Off indicates PLC in 16-bit conversion mode. The 16-bit data will be divided into the higher 8 bits and lower 8 bits for data transmission.

```
  M1001
  ──┤├──────( M1161 )
   X0
  ──┤├────┌──────┬──────┬──────┬──────┬──────┐
          │  RS  │ D100 │  K4  │ D120 │  K7  │
          └──────┴──────┴──────┴──────┴──────┘
```

Sending data: (PLC -> external equipment)

STX	D100L	D100L	D101L	D101L	ETX1	ETX2

Head code	Ⓢ source data register, starting from the lower 8 bits of D100	Tail code 1	Tail code 2
	ⓜ length = 4		

Receiving data: (External equipment -> PLC)

D120L	D120H	D121L	D121H	D122L	D122H	D123L

Head code	Ⓓ received data register, starting from the lower 8 bits of D120	Tail code 1	Tail code 2
	ⓝ length = 7		

When receiving data, PLC will receive the head code and tail code of the data from the external equipment; therefore, the user has to be aware of the setting of data length **n**.

Program Example 3:

Connect PLC to VFD-B series AC motor drives (AC motor drive in ASCII Mode; PLC in 16-bit mode and M1161 = Off). Write in the 6 data starting from parameter address H2101 in VFD-B in advance as the data to be transmitted.

```
 M1002
 ─┤├──────────┬──────[ MOV  │ H86  │ D1120 ]   Set up communication protocol 9600,7,E,1
               │
               ├──────[ SET  │ M1120 ]            Retain communication protocol
               │
               └──────[ MOV  │ K100 │ D1129 ]   Set up communication time-out 100ms
 Sending request
 pulses
 ─┤↑├─────────┬──────[ Write in data to be transmitted in advance ]
               │
               └──────[ SET  │ M1122 ]            Set up sending request
 X10
 ─┤├─────────────────[ RS  │ D100 │ K17 │ D120 │ K35 ]
 Receiving
 completed
 ─┤├─────────┬──────[ Process of received data ]
               │
 M1123        └──────[ RST  │ M1123 ]            Receiving of data is completed.
                                                  The flag is reset.
```
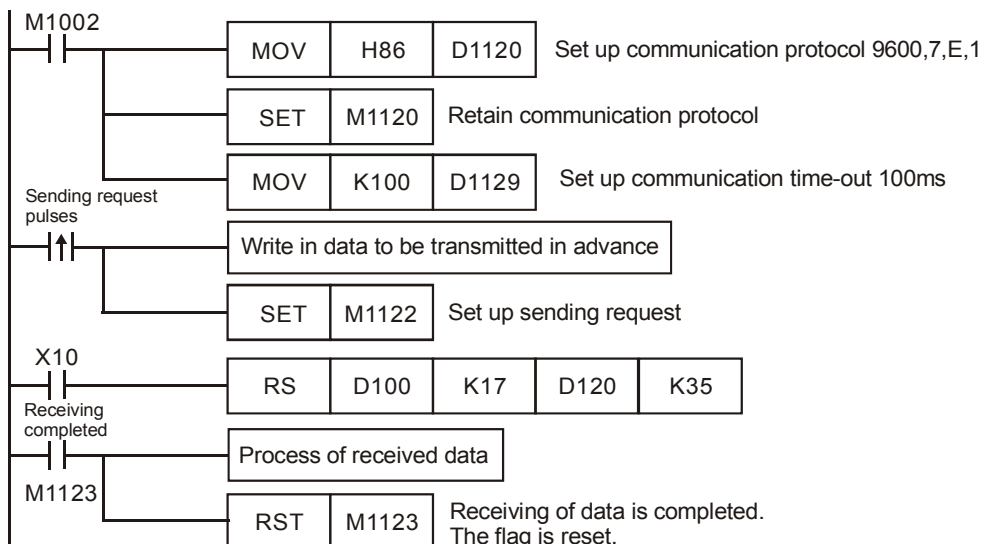
PLC ⇨ VFD-B, PLC sends **": 01 03 2101 0006 D4 CR LF "**

VFD-B ⇨ PLC, PLC receives **": 01 03 0C 0100 1766 0000 0000 0136 0000 3B CR LF "**

Registers for sent data (PLC sends out message)

Register	Data		Explanation	
D100 low	': '	3A H	STX	
D100 high	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D101 low	'1'	31 H	ADR 0	
D101 high	'0'	30 H	CMD 1	Instruction code: CMD (1,0)
D102 low	'3'	33 H	CMD 0	
D102 high	'2'	32 H	Start data address	
D103 low	'1'	31 H		
D103 high	'0'	30 H		
D104 low	'1'	31 H		
D104 high	'0'	30 H	Number of data (counted by words)	
D105 low	'0'	30 H		
D105 high	'0'	30 H		
D106 low	'6'	36 H		
D106 high	'D'	44 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
D107 low	'4'	34 H	LRC CHK 0	
D107 high	CR	D H	END	
D108 low	LF	A H		

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	

Register	Data		Explanation
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	

Program Example 4:

Connect PLC to VFD-B series AC motor drives (AC motor drive in RTU Mode; PLC in 16-bit mode and M1161 = On).

Write in H12 to parameter address H2000 in VFD-B in advance as the data to be transmitted.
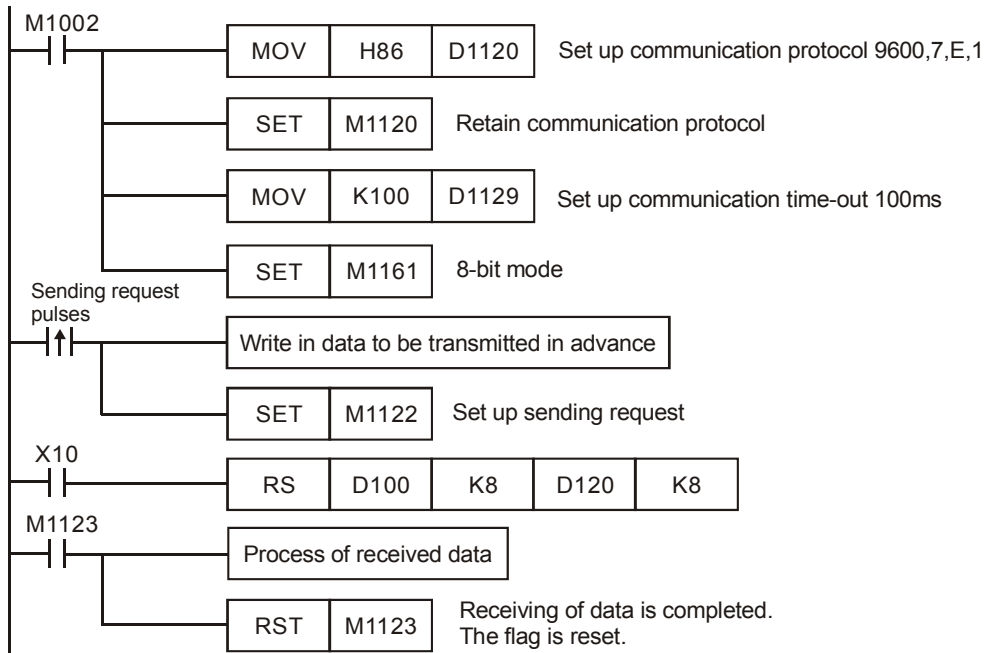
PLC ⇨ VFD-B, PLC sends: **01 06 2000 0012 02 07**

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

Remarks:

1. Flags for the RS-485 communication of RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions

Flag	Function	Action
M1120	For retain the communication setting. After the first program scan is completed, the communication setting will be reset according to the setting in the special data register D1120. When the second program scan starts and RS instruction is being executed, the communication settings will all be reset according to the settings in D1120. If your communication protocol is fixed, you can set M1120 to On and the communication protocol will not be reset whenever RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instruction is executed. In this case, even the settings in D1120 are modified, the communication protocol will not be changed.	Set up and reset by the user.
M1121	Off when the RS-485 communication data is being transmitted.	By the system.
M1122	Sending request. When you need to send out or receive data by RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions, you have to set M1122 to On by a pulse instruction. When these instructions start to execute, PLC will start to send out or receive data. When the data transmission is completed, M1122 will be reset automatically.	Set up by the user; reset automatically by the system.
M1123	Receiving is completed. When the execution of RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions is completed, M1123 will be set to On. You can process the data received when M1123 is On in the program. You have to reset M1123 to Off when the process of received data is completed.	Set up automatically by the system; reset by the user.
M1124	Waiting for receiving. On when PLC is waiting for receiving data.	By the system.
M1125	Receiving status cleared. When M1125 = On, the waiting for receiving status of PLC will be cleared. You have to reset M1125 to Off after the staus is cleared.	Set up and reset by the user.
M1126	User/system defined STX/ETX selection of RS instruction (see the next table for details.)	
M1130	User/system defined STX/ETX selection of RS instruction (see the next table for details.)	
M1127	Data transmission is completed for communication instructions (RS instruction not included)	Set up automatically by the system; reset by the user.
M1128	Data being sent/received indication	By the systme.
M1129	Receiving time-out. If you already set up a communication time-out in D1129 and the data have not been received completey when the time-out set is reached, M1129 will be On. You have to reset M1129 to Off after the problem is solved.	Set up automatically by the system; reset by the user.
M1131	On when the data are converted into hex of MODRD/RDST/MODRW instructions when in ASCII mode; otherwise, M1131 is Off.	By the system
M1140	Data receiving error of MODRD/MODWR/MODRW instructions	
M1141	Parameter error of MODRD/MODWR/MODRW instructions	
M1142	Data receiving error of VFD-A handy commands	
M1143	ASCII/RTU mode selection (used with MODRD/MODWR/MODRW instructions). On = RTU; Off = ASCII	Set up and reset by the user.
M1161	8/16-bit mode selection. On = 8-bit; Off = 16-bit	

2. Special data register for the RS-485 communication of RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF /MODRW instructions

Special D	Function
D1038	For setting up the data responding delay time when a PLC MPU using RS-485 communication is used as a slave. Range: 0 ~ 10,000 (unit: 0.1ms)
D1050 ~ D1055	When MODRD/RDST instruction is executed, PLC will automatically convert the ASCII characters in D1070 ~ D1085 into hex and store the hex value in D1050 ~ D1055.
D1070 ~ D1085	When the RS-485 communication instructions built in PLC are executed, the receiving end will respond with a message and the messages will be stored in D1070 ~ D1085. You can check on the responded data stored in these registers (not applicable for RS instruction).
D1089 ~ D1099	When the RS-485 communication instructions built in PLC are executed, the data sent will be stored in D1089 ~ D1099. You can check on whether the data sent are correct by checking these registers (not applicable for RS instruction).
D1120	RS-485 communication protocol. See the next table for more details.
D1121	The communication address of PLC when it operates as a slave.
D1122	Remaining number of words of the data being sent
D1123	Remaining number of words of the data being received
D1124	Definition of the start word (STX). See the table above for more details.
D1125	Definition of the first end word (ETX1) of RS instruction. See the table above for more details.
D1126	Definition of the second end word (ETX2) of RS instruction. See the table above for more details.
D1129	Abnormal communication time-out (in ms). When D1129 = 0, there will be no time-out occurring. When D1129 > 0 and RS/MODRD/ MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions are being executed, if the first word has not been received within designated time or the time interval between any two words exceeds the value (>0) after PLC enters the receiving mode, PLC will automatically set M1129 to On. You can also use M1129 for handling the communication time-out. Please be noted that you have to reset M1129 after the time-out.
D1130	Error code sent back by Modbus
D1168	For RS instruction, when the received number of words = the low byte of D1168, the interruption I150 will be triggered.
D1169	For RS instruction, when the received data length = the low byte of D1169, the interruption I160 will be triggered. When D1169 = 0, I160 will not be triggered.
D1256 ~ D1295	When the RS-485 communication instruction MODRW built in PLC is executed, the data sent will be stored in D1256 ~ D1295. You can check on whether the data sent are correct by checking these registers.
D1296 ~ D1311	For MODRW instruction, PLC will automatically convert the ASCII characters into hex.

3. How to set up RS-485 communication protocol in D1120

	Content	0	1
b0	Data length	7	8
b1 **b2**	Parity bits	00: None 01: Odd 11: Even	
b3	Stop bits	1 bit	2 bits
b4 **b5** **b6** **b7**	0001 (H1) : 110 0010 (H2) : 150 0011 (H3) : 300 0100 (H4) : 600 0101 (H5) : 1200 0110 (H6) : 2400 0111 (H7) : 4800 1000 (H8) : 9600 1001 (H9) : 19200 1010 (HA) : 38400 1011 (HB) : 57600 (does not support ES/SS V5.8 and below) 1100 (HC) : 115200 (does not support ES/SS V5.8 and below)		
b8	Start word	None	D1124
b9	First end word	None	D1125
b10	Second end word	None	D1126
b15 ~ b11	Not defined		

4. When RS instruction is in use, the frequently used communication format in the peripheral device will define the start word and end word of the control string. Therefore, you can set up the start word and end word in D1124 ~ D1126 or use the start word and end word defined by the PLC. When you use M1126, M1130 and D1124 ~ D1126 to set up the start word and end word, b8 ~ b10 of D1120 have to be set as 1 to make valid the RS-485 communication protocol. See the table below for how to set up.

		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)

5. Example of how to set up the communication format:

Assume there is a communication format: Baud rate 9600 7, N, 2

STX : ":"

ETX1 : "CR"

ETX2 : "LF"

Check the table and obtain the communication format H788 and write it into D1120.

```
      b15                                    b0
D1120 |0|0|0|0|0|1|1|1|1|0|0|0|1|0|0|0|
       |    0     |  7  |  8  |  8  |
       | Don't care |
```
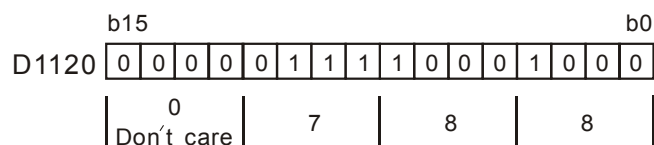
When STX, ETX1 and EXT2 are in use, please be aware of the On and Off of the special auxiliary relays M1126 and M1130.

6. M1143 is for the selection of ASCII mode or RTU mode. On = RTU mode; Off = ASCII mode.

Take the standard Modbus format for example:

In ASCII mode (M1143 = Off)

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:
…….	The n × 8-bit data consists of 2n ASCII codes
DATA 0	
LRC CHK Hi	LRC checksum:
LRC CHK Lo	The 8-bit checksum consists of 2 ASCII 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 hex '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):

   Fixed as ':' (3AH)

Address:

   '0' '0': Broadcasting to all drivers

   '0' '1': To the driver at address 01

   '0' 'F': To the driver at address 15

   '1' '0': To the driver at address 16

     ….and so on, maximum to the driver at address 255 ('F' 'F')

Function code:

   '0' '3': Read contents of many registers

   '0' '6': Write 1 word to register

   '1' '0': Write in contents of many registers

Data characters: The data sent by the user.

LRC checksum:

   LCR checksum is 2's complement of the value added from Address to Data Content.

   For example: 01H + 03H + 21H + 02H + 00H + 02H = 29H. 2's complement of 29H = D7H

End word (END):

Fixed as END Hi = CR (0DH), END Lo = LF (0AH)

For example: Read 2 continuous data stored in the registers of the driver 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 (counted by words)	'0'
	'0'
	'0'
	'2'
LRC checksum	'D'
	'7'
END	CR
	LF

Responding message:

STX	': '
Address	'0'
	'1'
Function code	'0'
	'3'
Number of data (counted by byte)	'0'
	'4'
Content in start address 2102H	'1'
	'7'
	'7'
	'0'
Content of address 2103H	'0'
	'0'
	'0'
	'0'
LRC check	'7'
	'1'
END	CR
	LF

In RTU mode (M1143 = On)

START	See the following explanation
Address	Communication address: In 8-bit binary
Function	Function code: In 8-bit binary
DATA (n-1)	Data:
…….	$n \times$ 8-bit data
DATA 0	
CRC CHK Low	CRC checksum:
CRC CHK High	16-bit CRC consists of 2 8-bit binary
END	See the following explanation

START:

For ES/EX/SS/SA/SX series MPU, no input signal can be ≥ 10ms.

See the table below for EH/EH2/SV series MPU:

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:

00H: Broadcasting to all drivers

01H: To the driver at address 01

0FH: To the driver at address 15

10H: To the driver at address 16…. And so on, maximum to the driver at address 254 (FE H)

Function code:

03H: Read contents of many registers

06H: Write 1 word to register

10H: Write in contents of many registers

Data characters: The data sent by the user.

CRC checksum: Starting from Address and ending at Data Content.

Step 1: Make the 16-bit register (CRC register) = FFFFH

Step 2: Exclusive OR the first 8-bit message and the low 16-bit CRC register. Store the result in the CRC register.

Step 3: Right shift CRC register for a bit and fill "0" into the high bit.

Step 4: Check the value shifted to the right. If it is 0, fill in the new value obtained in step 3 and store the value in CRC register; otherwise, Exclusive OR A001H and CRC register and store the result in the CRC register.

Step 5: Repeat step 3 – 4 and finish operations of all the 8 bits.

Step 6: Repeat step 2 – 5 for obtaining the next 8-bit message until the operation of all the messages are completed. The final value obtained in the CRC register is the CRC checksum. The CRC checksum has to be placed interchangeably in the checksum of the message.

END:

For ES/EX/SS V5.8 (and below) and SA/SX V1.1 (and below) series MPU, keep no input signal be ≥ 10ms.

See the table below for EH/EH2/SV series MPU:

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

For example: Read 2 continuous data stored in the registers of the driver at address 01H (see the table below).

The start register is at address 2102H.

Inquiry message:

Address	01 H
Function	03 H
Start data address	21 H
	02 H
Number of data (counted by words)	00 H
	02 H
CRC CHK Low	6F H
CRC CHK High	F7 H

Responding message:

Address	01 H
Function	03 H
Number of data (counted by byte)	04 H
Content in data address 8102H	17 H
	70 H
Content in data address 8103H	00 H
	00 H
CRC CHK Low	FE H
CRC CHK High	5C H

7. Timing diagram of RS-485 communication flag:
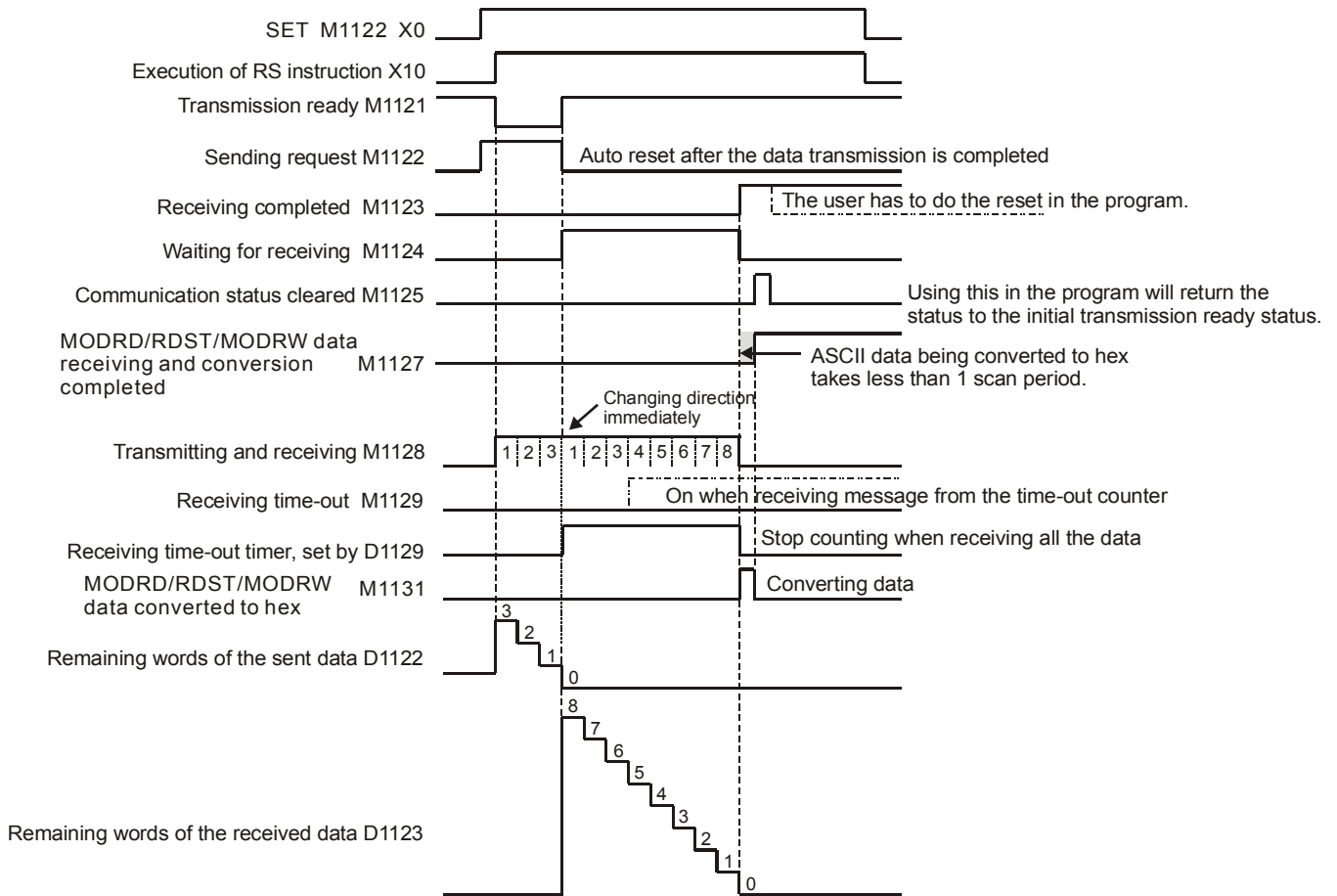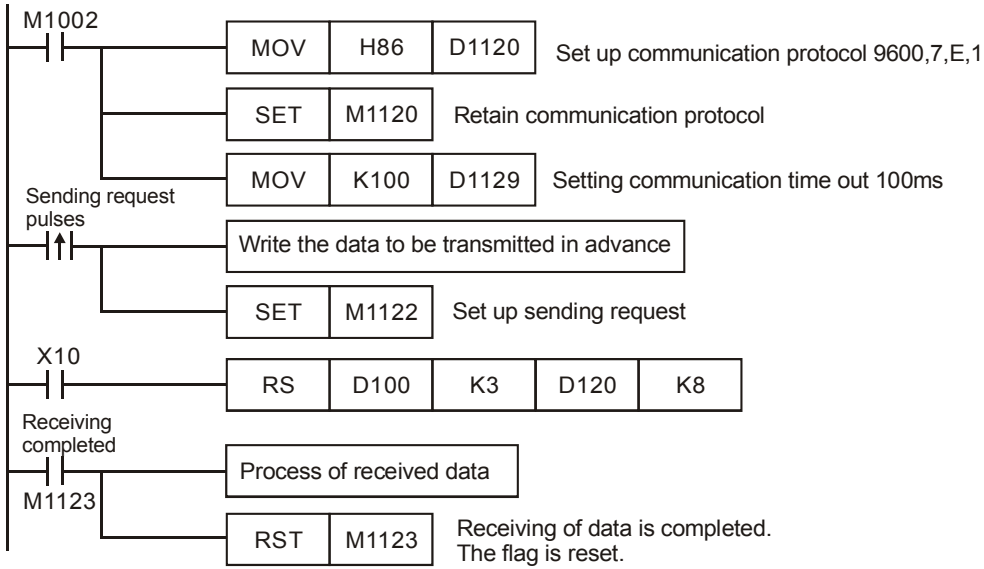
API	Mnemonic		Operands	Function	Controllers
81	D PRUN P		**S** **D**	Parallel Run	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PRUN, PRUNP: 5 steps
S							*		*							DPRUN, DPRUNP: 9 steps
D								*	*							

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device   **D**: Destination device

Explanations:

1. The most right digit of X, Y and M of KnX, KnY and KnM has to be 0.

2. When **S** designates KnX, **D** has to designate KnM; when **S** designates KnM, **D** has to designate KnY.

3. See the specifications of each model for their range of use.

4. This instruction sends the content in **S** to **D** in the form of octal system.

Program Example 1:

When X3 = On, the content in K4X10 will be sent to K4M10 in octal form.
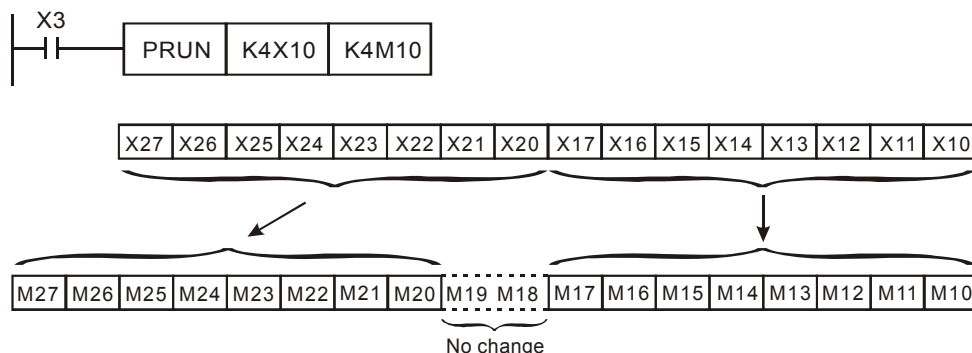
Program Example 2:

When X2 = On, the content in K4M10 will be sent to K4Y10 in octal form.

API	Mnemonic		Operands	Function		Controllers
82	ASCI	P	S D n	Converts Hex to ASCII		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ASCI, ASCIP: 7 steps
S					*	*	*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*										

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for source data   **D**: Start device for storing the converted result   **n**: Number of bits to be converted

Explanations:

1. Range of **n**: 1 ~ 256

2. See the specifications of each model for their range of use.

3. Flag: M1161 (8/16 bit mode switch)

4. 16-bit conversion mode: When M1161 = Off, the instruction converts every bit of the hex data in **S** into ASCII codes and send them to the 8 high bits and 8 low bits of **D**. **n** = the converted number of bits.

5. 8-bit conversion mode: When M1161 = On, the instruction converts every bit of the hex data in **S** into ASCII codes and send them to the 8 low bits of **D**. **n** = the number of converted bits. (All 8 high bits of **D** = 0)

Program Example 1:

1. M1161 = Off: The 16-bit conversion mode

2. When X0 = On, convert the 4 hex values in D10 into ASCII codes and send the result to registers starting from D20.
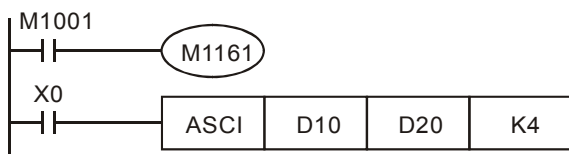
3. Assume

(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** = 6, the bit structure will be as:

```
b15      D10 = H 0123                    b0
0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 1
|    0    |    1    |    2    |    3    |

b15      D11 = H 4567                    b0
0 1 0 0 0 1 0 1 0 1 1 0 0 1 1 1
|    4    |    5    |    6    |    7    |

              Converted to

b15          D20                         b0
0 0 1 1 0 1 1 1 0 0 1 1 0 1 1 0
|    ˙7˙ → H 37    |    ˙6˙ → H 36    |

b15          D21                         b0
0 0 1 1 0 0 0 1 0 0 1 1 0 0 0 0
|    ˙1˙ → H 31    |    ˙0˙ → H 30    |

b15          D22                         b0
0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0
|    ˙3˙ → H 33    |    ˙2˙ → H 32    |
```

6. When **n** = 1 ~ 16:

D \ n	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 Llow 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								
D27 High byte								

D \ n	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 Llow 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: The 8-bit conversion mode

2. When X0 = On, convert the 4 hex values in D10 into ASCII codes and send the result to registers starting from D20.
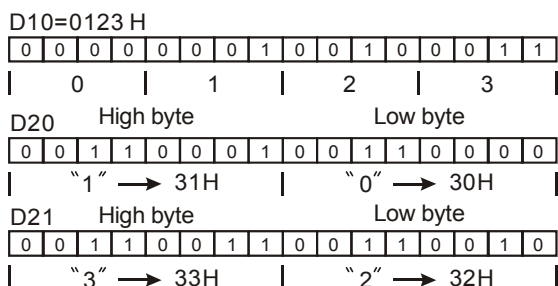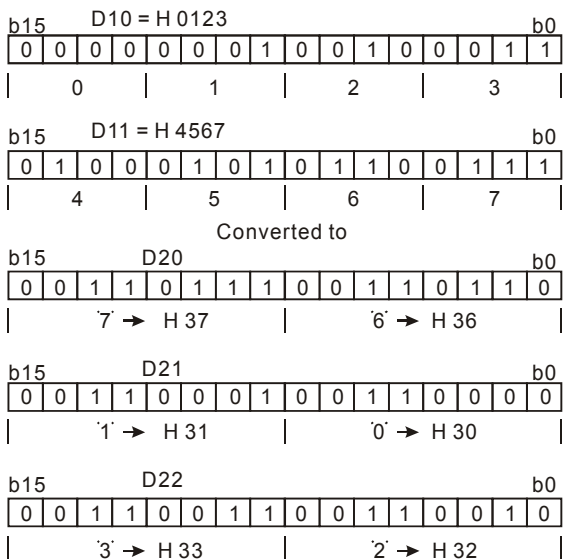
```
  M1000
  ─┤├──────────( M1161 )

  X0
  ─┤├──────┌──────┬──────┬──────┬──────┐
           │ ASCI │ D10  │ D20  │  K4  │
           └──────┴──────┴──────┴──────┘
```

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** = 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 D20=2 is 32H

0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0

| 3 | 2 |

ASCII code of D21=3 is 33H

0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

| 3 | 3 |

5. When **n** = 4, the bit structure will be as:

b15　　D10 = H 0123　　b0

0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1

| 0 | 1 | 2 | 3 |

Converted to

b15　　D20　　b0

0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

'0' ➔ H 30

b15　　D21　　b0

0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1

'1' ➔ H 31

b15　　D22　　b0

0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0

'2' ➔ H 32

b15　　D23　　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			
D30					change			
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				"3"	"2"	"1"
D34		change					"3"	"2"
D35								"3"

API	Mnemonic		Operands	Function		Controllers
83	HEX	P	S D n	Converts ASCII to Hex		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	HEX, HEXP: 7 steps
S					*	*	*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*										

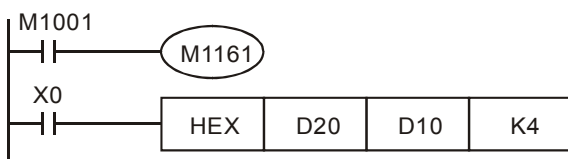|	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for source data **D**: Start device for storing the converted result **n**: Number of bits to be converted

Explanations:

1. Range of **n**: 1 ~ 256

2. See the specifications of each model for their range of use.

3. Flag: M1161 (8/16 bit mode switch)

4. 16-bit conversion mode: When M1161 = Off, the instruction is in 16-bit conversion mode. ASCII codes of the 8 high bits and 8 low bits of the hex data in **S** are converted into hex value and sent to **D** (every 4 bits as a group). **n** = the number of bits converted into ASCII codes.

5. 8-bit conversion mode: When M1161 = On, the instruction is in 8-bit conversion mode. Every bit of the hex data in **S** are converted into ASCII codes and sent to the 8 low bits of **D**. **n** = the number of converted bits. (All 8 high bits of **D** = 0)

Program Example 1:

1. M1161 = Off: The 16-bit conversion mode

2. When X0 = On, convert the ASCII codes stored in the registers starting from D20 into hex value and send the result (every 4 bits as a group) to registers starting from D10. **n** = 4.

```
  M1001
──┤├──────────────( M1161 )
   X0
──┤├────┬─────┬──────┬──────┬──────┐
        │ HEX │ D20  │ D10  │  K4  │
        └─────┴──────┴──────┴──────┘
```

3. Assume

S	ASCII code	Converted to hex	S	ASCII code	Converted to hex
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"
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 |
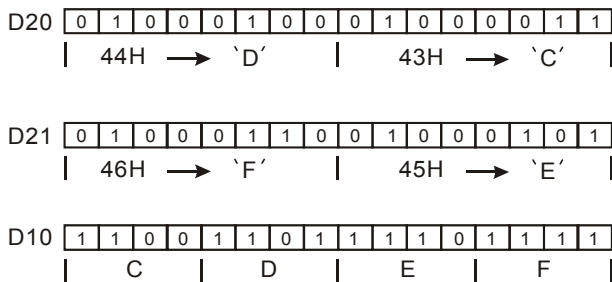
      C   D   E   F

5. When **n** = 1 ~ 16:

n \ D	D13	D12	D11	D10
1				***C H
2				**CD H
3				*CDE H
4				CDEF H
5	The undesignated parts in the registers in use are all 0.		***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: The 8-bit converstion mode

```
  M1000
  ──┤├──────────( M1161 )

  X0
  ──┤├──────┌──────┬─────┬─────┬─────┐
            │ HEX  │ D20 │ D10 │ K4  │
            └──────┴─────┴─────┴─────┘
```
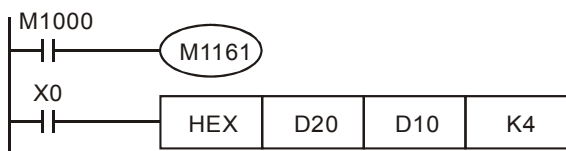
2. Assume

S	ASCII code	Converted to hex	S	ASCII code	Converted to hex
D20	H 43	"C"	D28	H 34	"4"
D21	H 44	"D"	D29	H 35	"5"
D22	H 45	"E"	D30	H 36	"6"
D23	H 46	"F"	D31	H 37	"7"
D24	H 38	"8"	D32	H 30	"0"
D25	H 39	"9"	D33	H 31	"1"
D26	H 41	"A"	D34	H 32	"2"
D27	H 42	"B"	D35	H 33	"3"

3. When **n** = 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 ~ 16:
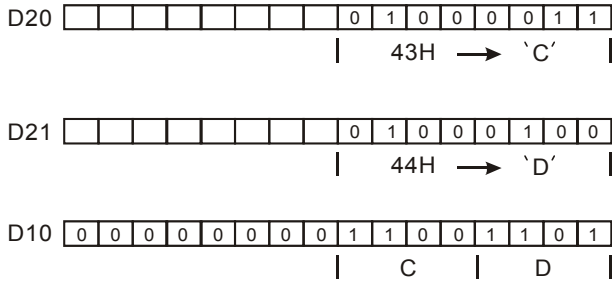
n \ D	D13	D12	D11	D10
1				***C H
2				**CD H
3				*CDE H
4				CDEF H
5	The used registers which are not specified are all 0		***C H	DEF8 H
6			**CD H	EF89 H
7			*CDE H	F89A H
8			CDEF H	89AB H
9		***C H	DEF8 H	9AB4 H
10		**CD H	EF89 H	AB45 H
11		*CDE H	F89A H	B456 H
12		CDEF H	89AB H	4567 H
13	***C 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
84	CCD P	S D n	Check Code	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start device for source data   **D**: Device for storing the sum check result   **n**: Number of data

Explanations:

1. Range of **n**: 1 ~ 256

2. See the specifications of each model for their range of use.

3. Flag: M1161 (8/16 bit mode switch)

4. The sum check is used for ensuring the correctness of the data transmission.

5. 16-bit conversion mode: When M1161 = Off, the instruction is in 16-bit conversion mode. The instruction sums up **n** data (8 bits as a unit) from the start register designated in **S** and stores the results in the registers designated in **D**. The parity bits are stored in **D** + 1.

6. 8-bit conversion mode: When M1161 = On, the instruction is in 8-bit conversion mode. The instruction sums up **n** data (8 bits as a unit; only 8 low bits are valid) from the start register designated in **S** and stores the results in the registers designated in **D**. The parity bits are stored in **D** + 1.

Program Example 1:

1. M1161 = Off: The 16-bit conversion mode

2. When X0 = On, the instruction will sum up 6 data stored in the register designated in D0 (8 bits as a unit; **n** = 6 indicates D0 ~ D2 are designated) and store the result in the register designated in D100. The parity bits are stored in D101.

```
  M1000
───┤/├──────────( M1161 )
  X0
───┤ ├──┬─────┬──────┬──────┬
        │ CCD │  D0  │ D100 │  K6 │
        └─────┴──────┴──────┴──────┘
```
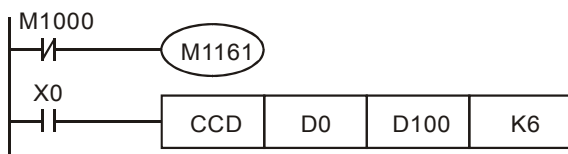
(S)	Content of data
D0 low byte	K100 = 0 1 1 0 0 1 0 0
D0 high byte	K111 = 0 1 1 0 1 1 1①
D1 low byte	K120 = 0 1 1 1 1 0 0 0
D1 high byte	K202 = 1 1 0 0 1 0 1 0
D2 low byte	K123 = 0 1 1 1 1 0 1①
D2 high byte	K211 = 1 1 0 1 0 0 1①
D100	K867
D101	0 0 0 1 0 0 0①

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

Program Example 2:

1. M1161 = On: The 8-bit conversion mode

2. When X0 = On, the instruction will sum up 6 data stored in the register designated in D0 (8 bits as a unit; **n** = 6 indicates D0 ~ D5 are designated) and store the result in the register designated in D100. The parity bits are stored in D101.

```
  M1000
───┤├──────────( M1161 )

   X0
───┤├────┬──────┬──────┬──────┬
         │ CCD  │  D0  │ D100 │  K6  │
         └──────┴──────┴──────┴
```

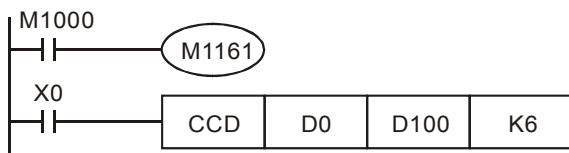| (S)	Content of data
D0 low byte	K100 = 0 1 1 0 0 1 0 0
D1 low byte	K111 = 0 1 1 0 1 1 1①
D2 low byte	K120 = 0 1 1 1 1 0 0 0
D3 low byte	K202 = 1 1 0 0 1 0 1 0
D4 low byte	K123 = 0 1 1 1 1 0 1①
D5 low byte	K211 = 1 1 0 1 0 0 1①
D100	K867
D101	0 0 0 1 0 0 0①

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
85	VRRD	P	**S** **D**	Volume Read		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: No. of VR   **D**: Device for storing the volume of VR

Explanations:

1. Range of **S**: 0 ~ 7; without function card: 0 ~ 1.

2. See the specifications of each model for their range of use.

3. Flags: M1178, M1179. See remarks for more details.

4. VRRD instruction is used for reading 2 points (No.0, No.1) of PLC or the VR rotary switch volume change in the 6 points of the function cards (No.2 ~ No.7) and converting the data into values 0 ~ 255 (stored in **D**).

5. If you are to set up the timer by the VR volume, simply rotate the VR to modify the set time in the timer. If you are to acquire a value larger than 255, multiply **D** by a constant.

Program Example 1:

1. When X0 = On, VR0 volume changed will be converted into an 8-bit BIN value (0 ~ 255) and stored in D0.

2. When X1 = On, the imer T0 will start to time with the content in D0 as the set value in the timer.

```
X0
├─┤├──┤ VRRD │ K0 │ D0 │

X1
├─┤├──┤ TMR  │ T0 │ D0 │
```
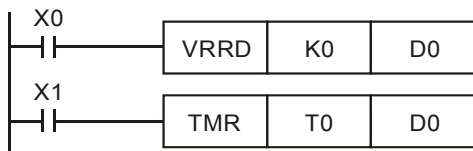
Program Example 2:

1. Read the VR volume in order: The VR0 ~ VR7 rotary switches on the PLC correspond to **S** = K0 ~ K7 of VRRD instruction. E index register modification is used in the example below, K0E = K0 ~ K7.

2. The timer converts the scale 0 ~ 10 on the rotary switch into 0 ~ 255. The timing unit of T0 ~ T7 is 0.1 second; therefore, the set time in the timer will be 0 ~ 25.5 seconds.

```
M1000
├─┤├──┤ RST  │ E    │

      ┤ FOR  │ K8   │
M1000
├─┤├──┤ VRRD │ K 0E │ D100E │

      ┤ INC  │ E    │

      ┤ NEXT │
```

```
  X10
  ─┤├──────────────┌─────┬─────┬──────┐
                   │ TMR │ T0  │ D100 │
                   └─────┴─────┴──────┘
  T0
  ─┤├──────────────( Y000 )
  
  X17      ⇩
  ─┤├──────────────┌─────┬─────┬──────┐
                   │ TMR │ T7  │ D107 │
                   └─────┴─────┴──────┘
  T7
  ─┤├──────────────( Y007 )

                   ┌─────────┐
                   │   END   │
                   └─────────┘
```
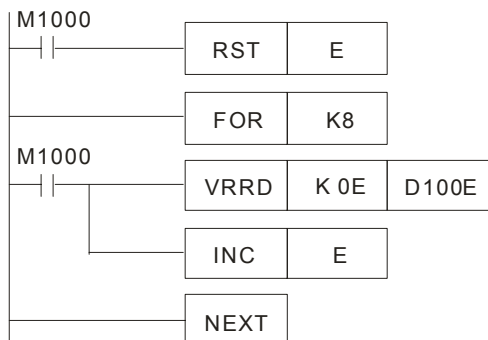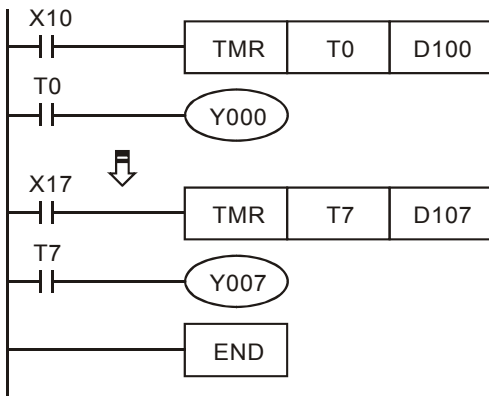
3. Operation of FOR ~ NEXT instruction:

    a) In the area between FOR ~ NEXT instruction, FOR designating K8 indicates the loop between FOR ~ NEXT will be executed repeatedly for 8 times before the next instruction is executed.

    b) Between FOR ~ NEXT (INC E), E will be 0, 1, 2, …7 plusing 1. Therefore, the 8 VR rotary switch volumes will be VR0→D100, VR1→D101, VR2→D102…VR7→D107 and be read to designated registers in order.

Remarks:

1. VR refers to **V**ariable **R**esister.

2. The 2 points of VR rotary switch built in SA/SC/EH/EH2/SV series MPU can be used together with special D and special M.

Device	Function
M1178	Enabling VR0
M1179	Enabling VR1
D1178	VR0 value
D1179	VR1 value

3. If there is no VR extension card inserted in the PLC, setting up the No. of rotary switches as K2 ~ K7 in VRRD and VRSC instruction in the program will result in errors in grammar check.

API	Mnemonic		Operands		Function		Controllers
86	VRSC	P	**S** **D**		Volume Scale		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: No. of VR   **D**: Device for storing the scale of VR

Explanations:

1. Range of **S**: 0 ~ 7; without function card: 0 ~ 1

2. See the specifications of each model for their range of use.

3. VRSC instruction is used for reading 2 points (No.0, No.1) of PLC or the VR rotary switch scale (0 ~ 10) in the 6 points of the function cards (No.2 ~ No.7) and storing the data in **D**. If the position of the VR falls in the middle of two scales, VRSC will round up the value into an integer of 0 ~ 10.
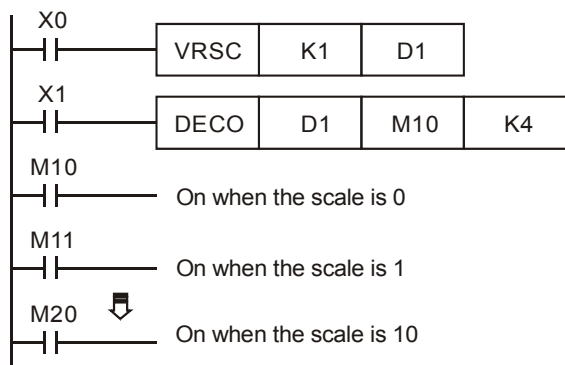
Program Example 1:

When X0 = On, the scale of VR0 (0 ~ 10) will be stored in D10.

```
   X0
───┤ ├──────┤ VRSC │ K0 │ D10 │
```

Program Example 2:

1. When the VR is used as DIP switch, they will correspond to scale 0 ~ 10 and only one of M10 ~ M20 will be On. Use API 41 DECO instruction to decode the scales into M10 ~ M25.

2. When X0 = On, store the scale (0 ~ 10) of VR1 into D1.

3. When X1 = On, use API 41 DECO to decode the scales into M10 ~ M25.

```
   X0
───┤ ├──────┤ VRSC │ K1  │ D1  │
   X1
───┤ ├──────┤ DECO │ D1  │ M10 │ K4 │
   M10
───┤ ├────────── On when the scale is 0
   M11
───┤ ├────────── On when the scale is 1
   M20
───┤ ├────────── On when the scale is 10
```

Remarks:

If the MPU is not inserted with a VR extension card, and the No. of the rotary switches inVRRD or VRSC instruction in the program are set as K2 ~ K7, errors will occur in the execution of grammar check.

API		Mnemonic		Operands	Function			Controllers
87	D	ABS	P	ⓓ	Absolute Value			ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices									Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ABS, ABSP: 3 steps
D								*	*	*	*	*	*	*	*	DABS, DABSP: 5 steps

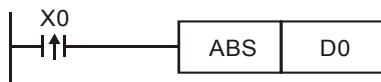|	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device of the absolute value

Explanations:

1. See the specifications of each model for their range of use.

2. This instruction obtains the absolute value of the content in the designated in **D**.

3. This instruction adopts pulse execution instructions (ABSP, DABSP).

Program Example:

When X0 = Off→On, obtain the absolute value of the content in D0.

```
   X0
  ──┤↑├──────────┌──────┬──────┐
                 │ ABS  │  D0  │
                 └──────┴──────┘
```

API	Mnemonic		Operands		Function		Controllers	
88	D	PID	S_1 S_2 S_3 D		PID Control Loop		ES/EX/SS SA/SX/SC EH/SV	

Type	Bit Devices				Word Devices												Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		PID : 9 steps
S_1													*				DPID: 17 steps
S_2													*				
S_3													*				
D													*				

| PULSE									16-bit									32-bit							
ES	EX	SS	SA	SX	SC	EH	SV		ES	EX	SS	SA	SX	SC	EH	SV		ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Set value (SV)   **S₂**: Present value (PV)   **S₃**: Parameter   **D**: Output value (MV)

Explanations:

1. In 16-bit instruction, **S₃** will occupy 20 consecutive devices; in 32-bit instruction, **S₃** will occupy 21 consecutive devices.

2. See the specifications of each model for their range of use.

3. See the Remarks below for the times of using PID instruction allowed in the program.

4. This instruction is specifically for PID control. PID operation will be executed by the scan only when the sampling time is reached. PID refers to "proportion, integration and differential". PID control is widely applied to many machines, pneumatic and electronic equipments.

5. For the 16-bit instruction, the parameters are **S₃** ~ **S₃**+19; for the 32-bit instruction, the parameters are **S₃** ~ **S₃**+20. After all the parameters are set up, PID instruction will start to be executed and the results will be stored in **D**. **D** has to be the data register area without latched function. (If you wish to designate a latched data register area, place the data register in the latched area at the beginning of the program and clear it as 0.)

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 goes Off, the instruction will not be executed and the data prior to the instruction will stay intact.

```
   X0
───┤├───┤ PID │ D0 │ D1 │ D100 │ D150 │
```
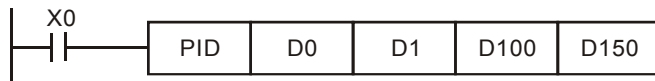
Remarks:

1. ES/EX/SS series MPU V5.7 (and above) supports PID instruction. Other versions do not support the instruction.

2. There is no limitation on the times of using this instruction. However, the register No. designated in **S₃** cannot be repeated.

3. For the 16-bit instruction, **S₃** will occupy 20 registers. In the program example above, the area designated in **S₃** is D100 ~ D119. Before the execution of PID instruction, you have to transmit the setting value to the designated register area by MOV instruction, If the designated registers are latched, use MOVP instruction to transmit all setting value at a time.

4. Settings of **S₃** in the 16-bit instruction

Device No.	Function	Setup Range	Explanation
S_3:	Sampling time (T_S) (unit: 10ms)	1 ~ 2,000 (unit: 10ms)	If T_S is less than 1 program scan time, PID instruction will be executed for 1 program scan time. If T_S= 0, PID instruction will not be enabled. The minimum T_S has to be longer than the program scan time.
S_3 +1:	Proportional gain (K_P)	0 ~ 30,000 (%)	The magnified error proportional value between SV – PV.
S_3 +2:	Integral gain (K_I)	0 ~ 30,000 (%)	The magnified proportional value of every sampling time unit × the accumulated value of the error.
S_3 +3:	Differential gain (K_D)	-30,000 ~ 30,000 (%)	The magnified proportional value of the varied error in every samping timme unit.
S_3 +4:	Control direction (DIR)	0: automatic control 1: forward control (E = SV - PV) 2: inverse 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 be filled in with the appropriate parameter K_P, K_I and K_D (not avaliable in the 32-bit instruction). 4: Exclusively for the adjusted temperature control (not avaliable in the 32-bit instruction).	
S_3 +5:	The range that error value (E) doesn't work	0 ~ 32,767	E = the error of SV – PV. When S_3 +5 = K0, the function will not be enabled, e.g. when S_3 +5 is set as 5, MV of E between -5 and 5 will be 0.
S_3 +6:	Upper bound of output value (MV)	-32,768 ~ 32,767	Ex: if S_3 +6 is set as 1,000, the output will be 1,000 when MV is bigger than 1,000. S_3 +6 has to be bigger or equal S_3 +7; otherwise the upper bound and lower bound 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, the output will be -1,000 when MV 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 output will be 1,000 when the integral value is bigger than 1,000 and the integration will stop. S_3 +8 has to be bigger or equal S_3 +9; otherwier the upper bound and lower bound will switch.
S_3 +9:	Lower bound of integral value	-32,768 ~ 32,767	Ex: if S_3 +9 is set as -1,000, the output will be -1,000 when the integral value is smaller than -1,000 and the integration will stop.
S_3 +10,11:	Accumulated integral value	32-bit floating point	The accumulated integral value is only for reference. You can still clear or modify it (in 32-bit floating point) according to your need.
S_3 +12:	The previous PV	-	The previous PV is only for reference. You can still modify it according to your need.
S_3 +13: ～ S_3 +19:	For system use only.		

5. When parameter setting exceeds its range, the upper bound and lower bound will become the setting value. However, if the motion direction (DIR) exceeds the range, it will be set to 0.

6. PID instruction can be used in interruption subroutines, step points and CJ instruction.

7. 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 the interruption subroutine of the timer.

8. PV of PID instruction has to be stable before the execution of PID instruction. If you are to extract the input value

of DVP04AD/04DA/06XA/04PT/04TC for PID operation, please be aware of the A/D conversion time of these modules.

9. For the 32-bit instruction, If S_3 designates the parameter setting area of PID instruction as D100 ~ D120, S_3 occupies 21 registers. Before the execution of PID instruction, you have to use MOV instrction first to send the setting value to the register area for setup. If the designated registers are latched one, use MOVP instruction to send all the setting value at a time.

10. Settings of S_3 in the 32-bit instruction

Device No.	Function	Setup range	Explanation
S_3 :	Sampling time (T_S) (unit: 10ms)	1 ~ 2,000 (unit: 10ms)	If T_S is less than 1 program scan time, PID instruction will be executed for 1 program scan time. If T_S= 0, PID instruction will not be enabled. The minimum T_S has to be longer than the program scan time.
S_3 +1:	Proportional gain (K_P)	0 ~ 30,000 (%)	The magnified error proportional value between SV – PV.
S_3 +2:	Integral gain (K_I)	0 ~ 30,000 (%)	The magnified proportional value of every sampling time unit × the accumulated value of the error.
S_3 +3:	Differential gain (K_D)	-30,000 ~ 30,000 (%)	The magnified proportional value of the varied error in every samping timme unit.
S_3 +4:	Control direction (DIR)	0: automatic control 1: forward control (E = SV – PV) 2: inverse control (E = PV – SV)	
S_3 +5, 6:	The range that 32-bit error value (E) doesn't work	0 ~ 2,147,483,647	E = the error of SV – PV. When S_3 +5,6 = K0, the function will not be enabled, e.g. when S_3 +5,6 is set as 5, MV of E between -5 and 5 will be 0.
S_3 +7, 8:	Upper bound of 32-bit output value (MV)	-2,147,483,648 ~ 2,147,483,647	Ex: if S_3 +7,8 is set as 1,000, the output will be 1,000 when MV is bigger than 1,000. S_3 +7,8 has to be bigger or equal S_3 +9,10; otherwise the upper bound and lower bound will switch.
S_3 +9, 10:	Lower bound of 32-bit output value (MV)	-2,147,483,648 ~ 2,147,483,647	Ex: if S_3 +9,10 is set as -1,000, the output will be -1,000 when MV is smaller than -1,000.
S_3 +11, 12:	Upper bound of 32-bit integral value	-2,147,483,648 ~ 2,147,483,647	Ex: if S_3 +11,12 is set as 1,000, the output will be 1,000 when the integral value is bigger than 1,000 and the integration will stop. S_3 +11,12 has to be bigger or equal S_3 +13,14; otherwier the upper bound and lower bound will switch.
S_3 +13, 14:	Lower bound of 32-bit integral value	-2,147,483,648 ~ 2,147,483,647	Ex: if S_3 +13,14 is set as -1,000, the output will be -1,000 when the integral value is smaller than -1,000 and the integration will stop.
S_3 +15, 16:	32-bit accumulated integral value	32-bit floating point	The accumulated integral value is only for reference. You can still clear or modify it (in 32-bit floating point) according to your need.
S_3 +17, 18:	32-bit previous PV	-	The previous PV is only for reference. You can still modify it according to your need.
S_3 +19: ≀ S_3 +20:	For system use only.		

11. 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. The PID operation is conducted according to the speed and the differential PV.

2. The PID operation has three control directions: automatic, foreward and inverse. Forward or inverse are designated in **S₃** +4. Other relevant settings of PID operation are set by the registers designated in **S₃** ~ **S₃** + 5.

3. Basic PID equation:

$$MV = K_P * E(t) + K_I * E(t)\frac{1}{S} + K_D * PV(t)S$$

Control direction	PID equation
Forward, automatic	E(t) = SV – PV
Inverse	E(t) = PV – SV

$PV(t)S$ is the differential value of $PV(t)$; $E(t)\frac{1}{S}$ is the integral value of $E(t)$. When E(t) is less than 0

as the control direction is selected as forward or inverse, E(t) will be regarded as "0".

The equation above illustrates that this instruction is different from a general PID instruction by the variable use of the differential value. To avoid the flaw that the transient differential value is too big when a general PID instruction is executed for the first time, our PID instruction monitors the differentiation status of the PV. When the variation of PV is too big, this instruction will reduce the output of MV.

4. Symbol explanation:

MV : Output value

K_P : Proprotional gain

$E(t)$: Error value

PV : Present measured value

SV : Target value

K_D : Differential gain

$PV(t)S$: Differential value of PV(t)

K_I : Integral gain

$E(t)\frac{1}{S}$: Integral value of E(t)

5. Temperature Control Equation:

When **S₃** +4 is K3 and K4, the equation used in diagram 2 (see below) will be changed as:

$$MV = \frac{1}{K_P}\left[E(t) + \frac{1}{K_I}\left(E(t)\frac{1}{S}\right) + K_D * PV(t)S\right]$$

In which the error value is fixed as E(t) = SV – PV

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 you have no idea how to adjust the parameters, you can select K3 (auto-tuning) and after all the parameters are adjusted (the control direction will be automatically set as K4), you can modify your parameters to better ones according to the result of the control.
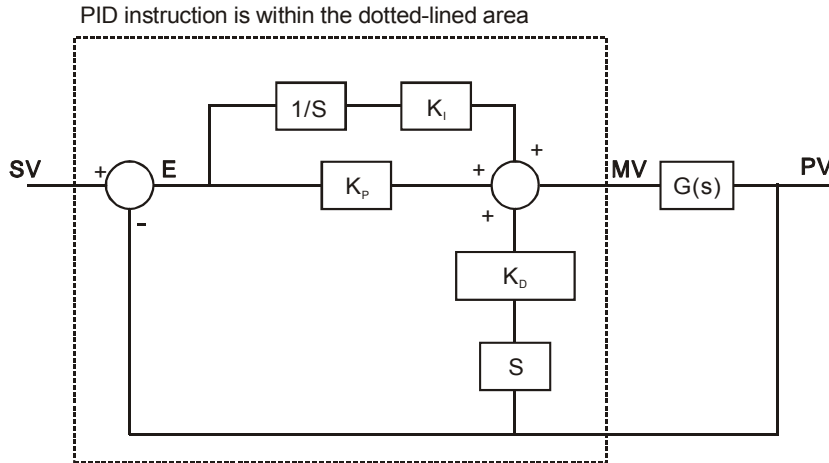
6. Control diagrams:

PID instruction is within the dotted-lined area

Diagram 1:S_3 + 4 = K0 ~ K2

In Diagram 1, S is differentiation, referring to "PV – previous PV / sampling time". 1 / S is integration, referring to "(previous integral value + error value) × sampling time". G(S) refers to the device being controlled.

PID operation is within the dotted-lined area

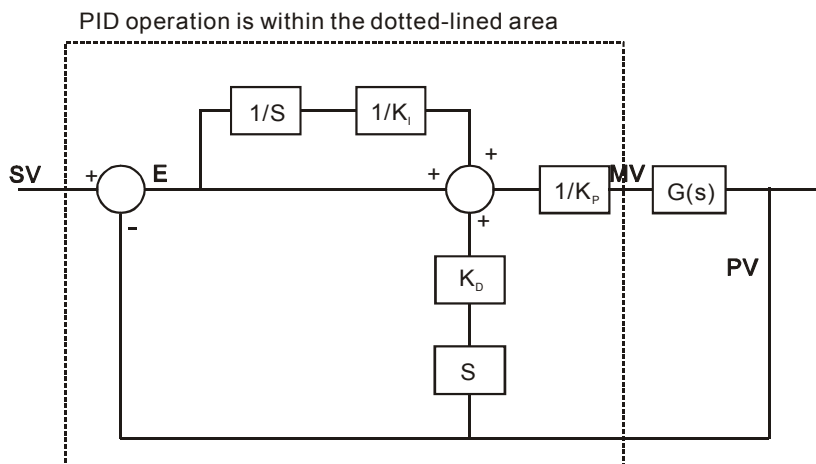Diagram 2: S_3 + 4 = K3 ~ K4

In Diagram 2, $1/K_I$ and $1/K_P$ refer to "divided by K_I" and "divided by K_P". Due to that this is exclusively for temperature control, you have to use PID instruction together with GPWM instruction. See Application 3 for more details.

7. Notes:

a) S_3 + 6 ~ S_3 + 13 are only available in SA/SX/SC/EH/EH2/SV series, and ES/EX/SS (v5.7 and above) series MPU.

b) PID instruction can only be used once in ES/EX/SS (v5.6 and below) series MPU. There is no limitation on the times of using PID instruction in ES/EX/SS (v5.7 and above) series and SA/SX/SC/EH/EH2/SV series MPU.

c) S_3 + 3 of ES/EX/SS (v5.7 and below), SA/SX/SC (v1.1 and below) and EH (v1.0 and below) series MPU can only be the value within 0 ~ 30,000.

d) There are a lot of circumstances where PID instruction can be applied; therefore, please choose the control functions appropriately. For example, when you select parameter auto-tuning for the temperature (S_3 + 4 = K3), you cannot use it in a motor control environment in case improper control may occur.

e) When you adjust the three main parameters, K_P, K_I and K_D (S_3 + 4 = K0 ~ K2), you have to adjust K_P first (according to your experiences) and set K_I and K_D as 0. When you can roughly handle the control, you then adjust K_I (increasingly) and K_D (increasingly) (see example 4 below for how to adjust). K_P = 100 refers to 100%, i.e. the gain of the error is 1. K_P < 100% will decrease the error and K_P > 100% will increase the error.

f) When you select the parameter exclusively for temperature control (S_3 + 4 = K3, K4), it is suggested that you store the parameter in D register in the latched area in case the automatically adjusted parameter will disappear after the power is cut off. There is no guarantee that the adjusted parameter is suitable for every control. Therefore, you can modify the adjusted parameter according to your actual need, but it is suggested that you modify only K_I or K_D.

g) PID instruction can to work with many parameters; therefore please do not randomly modify the parameters in case the control cannot be executed normally.

Example 1: Diagram of using PID instruction in position control (S_3 + 4 = 0)
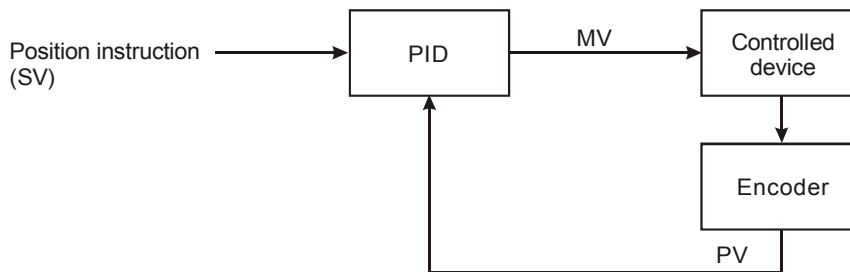
Example 2: Diagram of using PID instruction with AC motor drive on the control (S_3 + 4 = 0)
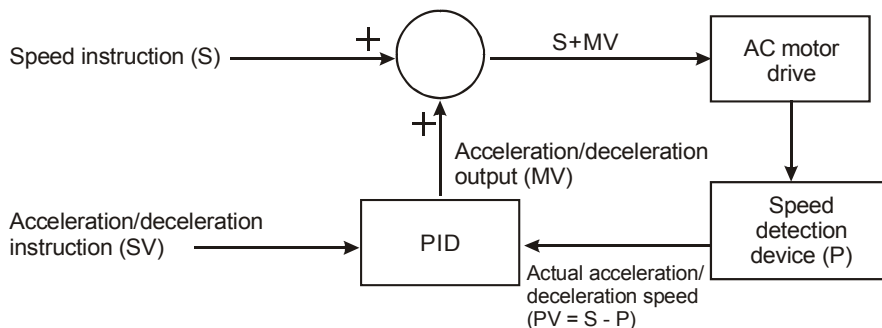
Example 3: Diagram of using PID instruction in temperature control (**S$_3$** + 4 = 1)
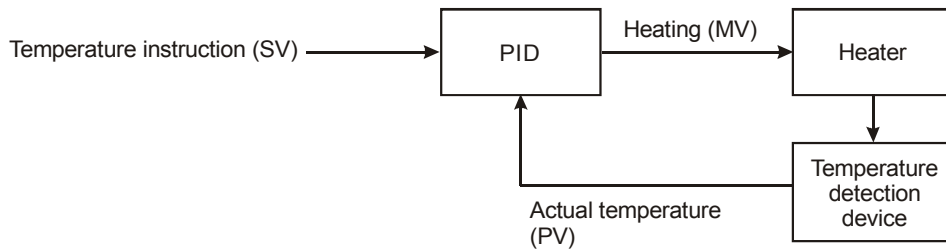
Example 4: How to adjust PID parameters

Assume that the transfer function of the controlled device G(S) in a control system is a first-order function

$G(s) = \dfrac{b}{s+a}$ (most models of motors are first-order function), SV = 1, and sampling time (T$_S$) = 10ms, we

suggest you to follow the steps below for adjusting the parameters.

Step 1: Set K$_I$ and K$_D$ as 0 and K$_P$ as 5, 10, 20 and 40. Record the SV and PV respectively and the results are as the figure below.
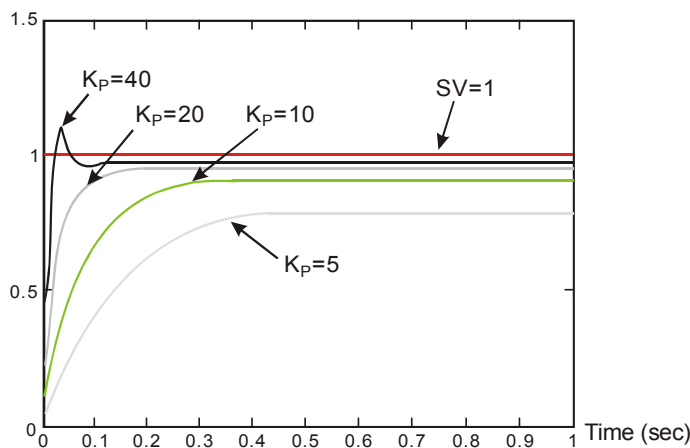
Step 2: From the figure, we can see that when K$_P$ = 40, there will be over-reaction, so we will not select it. When K$_P$ = 20, the PV reaction curve will be close to SV and there will not be over-reaction, but due to its fast start-up with big transient MV, we will consider to put it aside. When K$_P$ = 10, the PV reaction curve will get close to SV value more smoothly, so we will use it. Finally when K$_P$ = 5, we will not consider it due to the slow reaction.

Step 3: Select K$_P$ = 10 and adjust K$_I$ from small to big (e.g. 1, 2, 4 to 8). K$_I$ should not be bigger than K$_P$. Adjust K$_D$ from small to big (e.g. 0.01, 0.05, 0.1 and 0.2). K$_D$ should not exceed 10% of K$_P$. Finally we obtain the figure of PV and SV below.

Graph showing PV=SV, with curve labeled $K_P=10, K_I=8, K_D=0.2$. Time (sec) axis from 0 to 1.
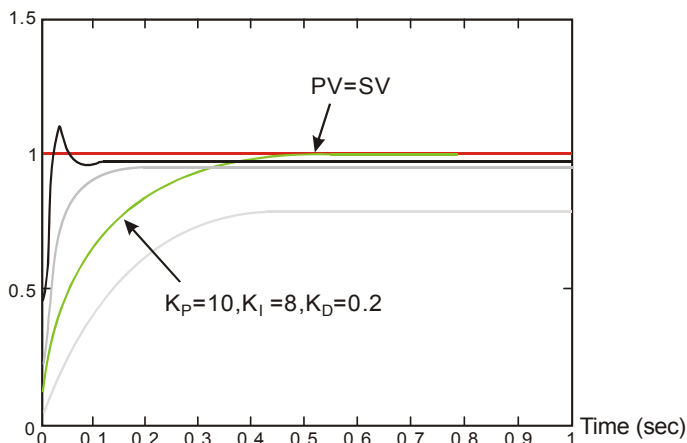
Note: This example is only for your reference. Please adjust your parameters to proper ones according to your actual condition of the control system.

Application Examples:

Application 1 Using PID instruction in the pressure control system (use the diagram of Example 1).

Purpose: Enabling the control system to reach the target pressure.

Explanation: 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.
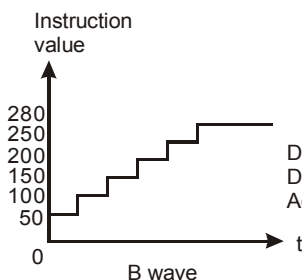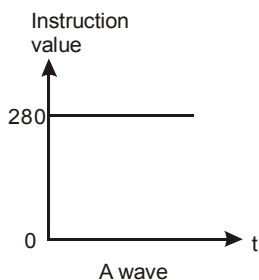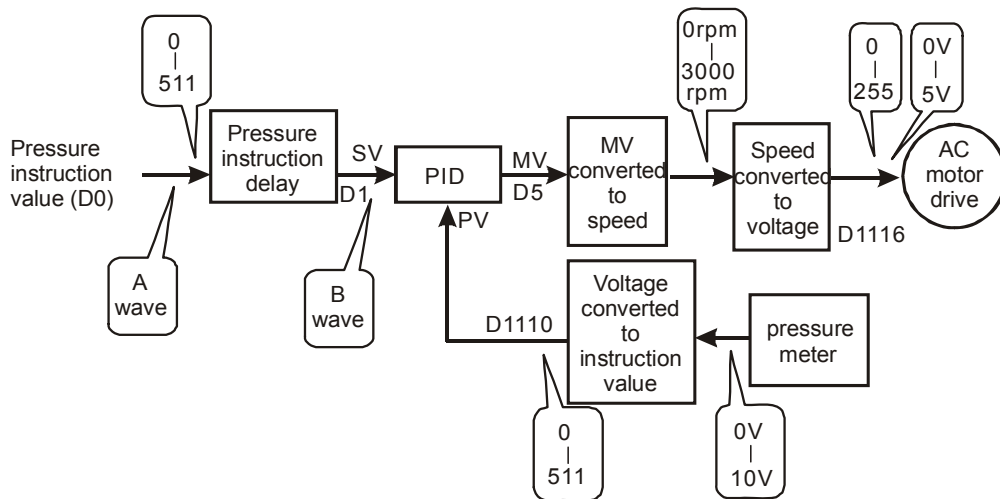
D2: Instruction interval value
D3: Instruction interval time
Adjusted by the user according to the actual condition

A wave

B wave

The example program of the instruction delay:
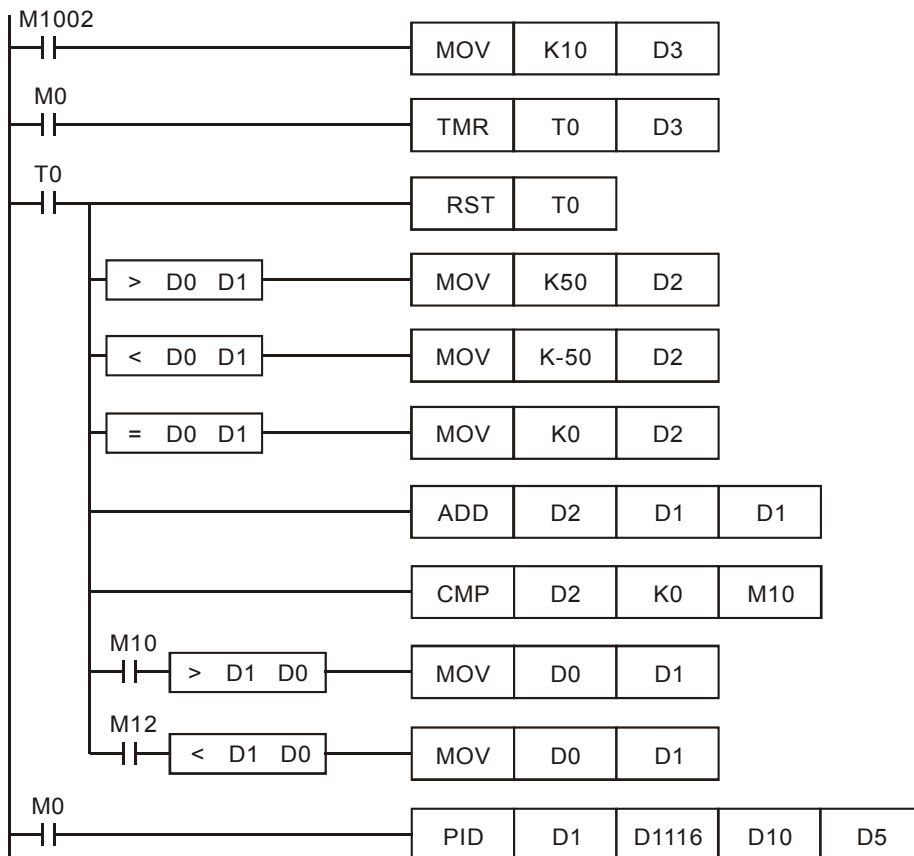
Application 2 Speed control system and pressure control system work individually (use diagram of Example 2).

Purpose: After the speed control operates in open loop for a period of time, adding into it the pressure control system (PID instruction) for close loop control.

Explanation: Since the speed and pressure control systems are not interrelated, we have to structure a open loop for speed control first following by a close loop pressure control. If you fear that the control instruction of the pressure control system changes too fast, you can consider to add the instruction delay illustrated in Application 1 into the control. See the control diagram below.
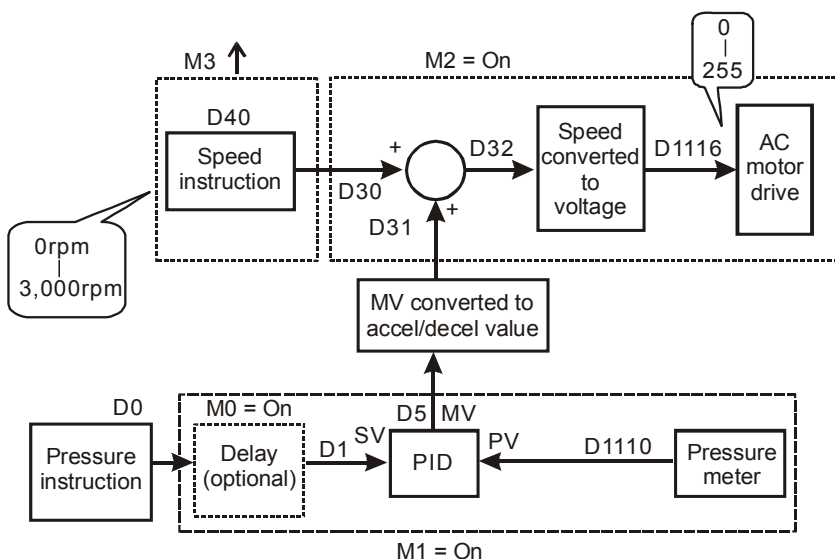
Part of the example program:

```
  M1002
  ┤├─────────────────────────── MOV  K1000  D40

  ┌M0─┐
  ┤ / ├─────────────────────────  MOV  D0  D1
  └───┘

  M1
  ┤/├───────────────────────────  MOV  K0  D5

  M3
  ┤↑↓├──────────────────────────  MOV  D40  D30

  M2
  ┤├──┬────────────────────────   ADD  D30  D31  D32
      │
      ├──[ >  D32  K3000 ]──────   MOV  K3000  D32
      │
      ├──[ <  D32  K0 ]─────────   MOV  K0  D32
      │
      ├─────────────────────────   DIV  D32  K11  D32
      │
      ├──[ >  D32  K255 ]───────   MOV  K255  D32
      │
      └─────────────────────────   MOV  D32  D1116

  M1
  ┤├───────────────────────────   PID  D1  D1110  D10  D5
```
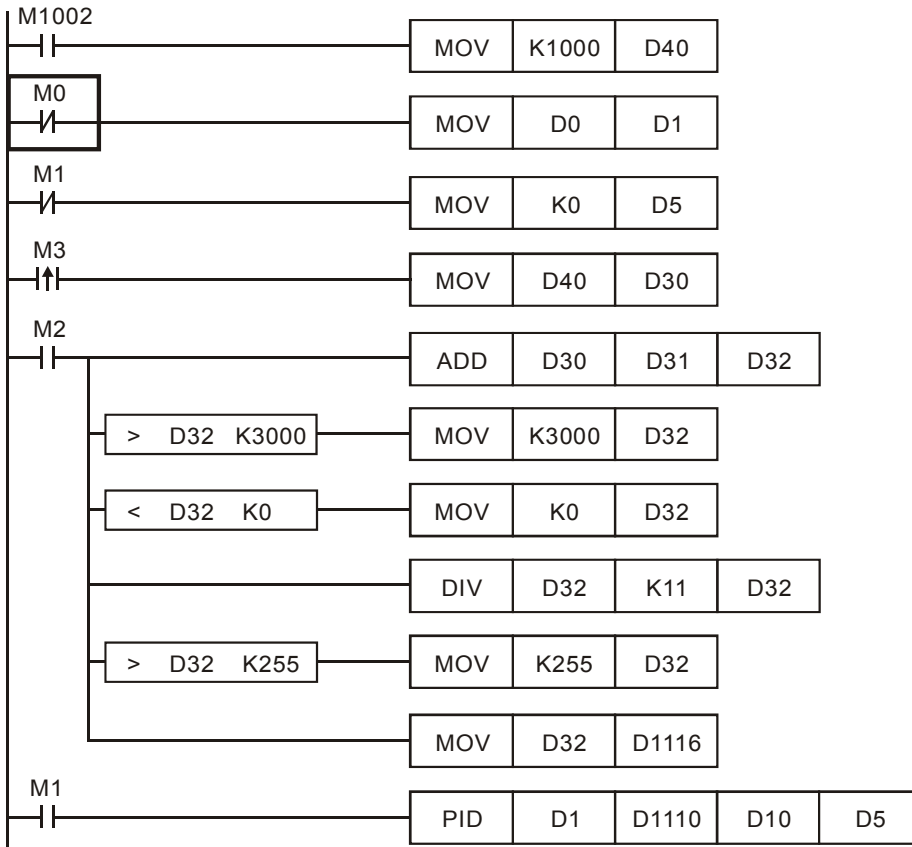
Application 3 Using auto-tuning on the parameter for the temperature control.

Purpose: Using auto-tuning to calculate the most suitable parameters for PID temperature control.

Explanation: You may not be familiar with the temperature environment for the first time, so you can use auto-tuning ($S_3 + 4$ = K3) for an initial adjustment. After this, PID instruction will become exclusively for temperature control ($S_3 + 4$ = K4). In this example, the control environment is an oven. See the example program below.

```
  M1002
  ┤├──┬────────────────────────   MOV  K4000  D20
      │
      ├─────────────────────────   MOV  K400  D200
      │
      ├─────────────────────────   MOV  K800  D10
      │
      └─────────────────────────   TO  K0  K2  K2  K1

  M1013
  ┤↑├──────────────────────────   FROM  K0  K6  D11  K1

  M0
  ┤├──┬────────────────────────   MOV  K3  D204
      │
      └─────────────────────────   RST  M0
```
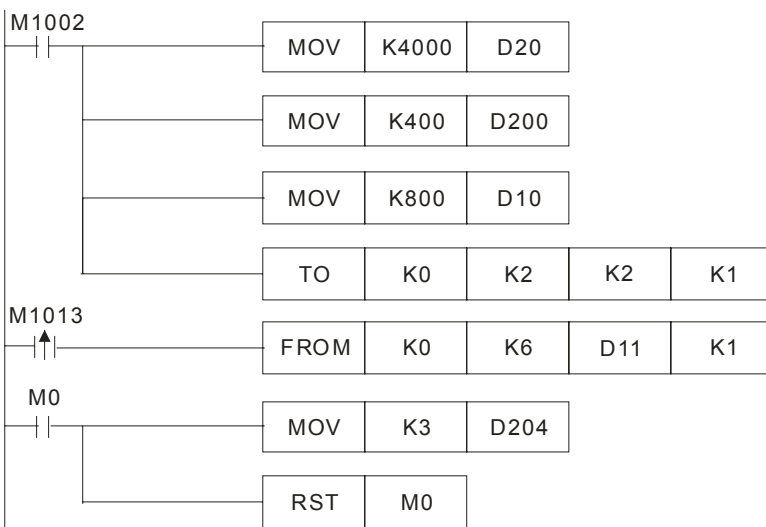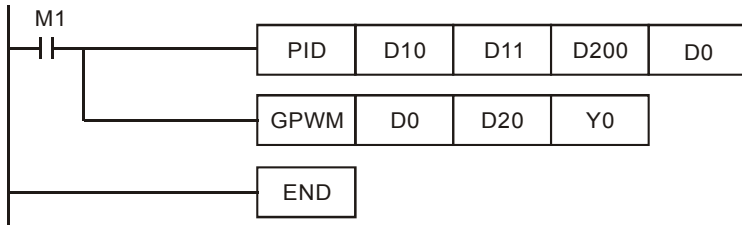
The experiment result of auto-tuning:
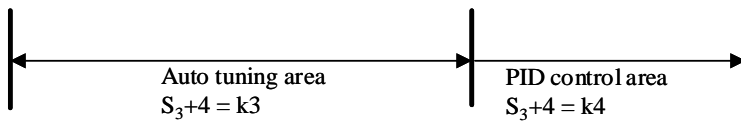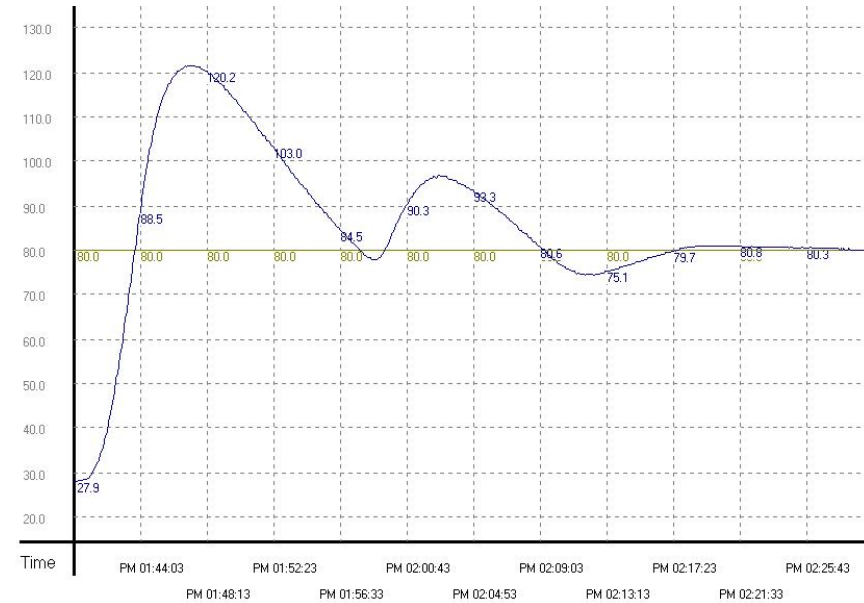
The experiment result of using the adjusted parameter exclusively for temperature control after auto-tuning:
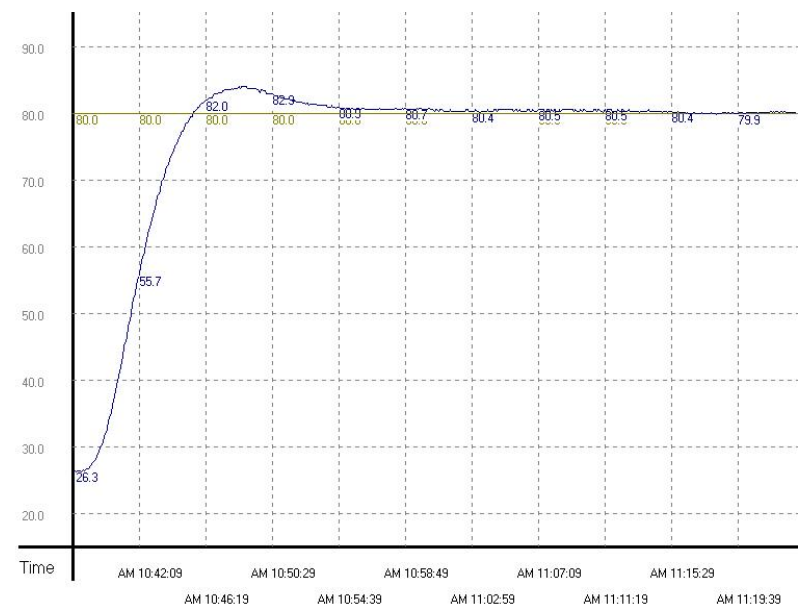
From the figure above, we can see that the temperature control after auto-tuning is working fine and we use only

approximately 20 minutes for the control. Next, we modify the target temperature from 80°C to 100°C and obtain the result below.
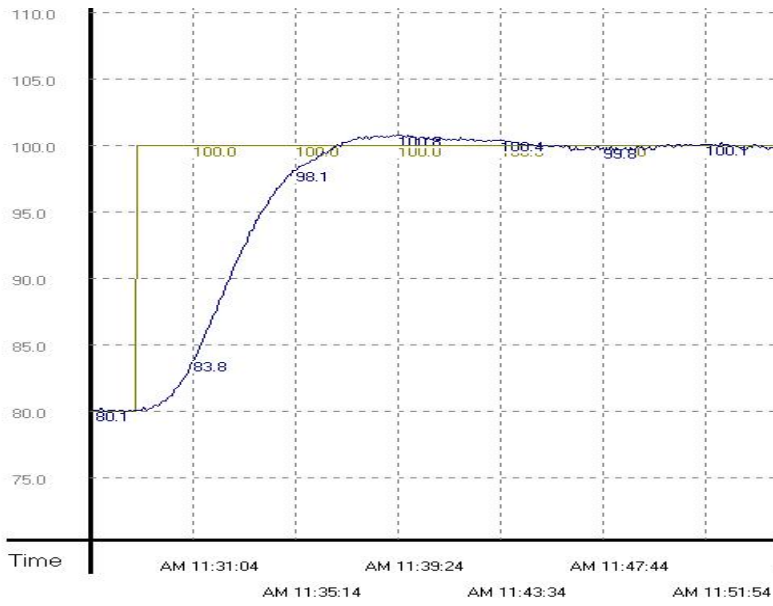
From the result above, we can see that when the parameter is 100°C, we can still control the temperature without spending too much time.

API	Mnemonic	Operands			Function			Controllers	
100	MODRD	S_1 S_2 n			Read Modbus Data			ES/EX/SS	SA/SX/SC	EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Address of communication device   **S_2**: Address of data to be read   **n**: Length of read data

Explanations:

1. Range of **S_1**: K0 ~ K254

2. Range of **n**: K1 ≤ n ≤ K6

3. See the specifications of each model for their range of use.

4. ES/EX/SS series MPU does not support E, F index register modification.

5. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143

6. MODRD is a drive instruction exclusively for peripheral communication equipment in MODBUS ASCII mode /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 controlling communication (read data) of Delta drives.

7. If the address of **S_2** is illegal to the designed communication device, the device will respond with an error, PLC will records the error code in D1130 and M1141 will be On.

8. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1085. After receiving the feedback data is completed, PLC will auto-check if all data are correct. If there is an error, M1140 will be On.

9. In ASCII mode, due to that the feedback data are all in ASCII, PLC will convert the feedback data into numerals and store them in D1050 ~ D1055. D1050 ~ D1055 will be invalid in RTU mode.

10. After M1140 or M1141 turn On, the program will send a correct datum to the peripheral equipment. If the feedback datum is correct, M1140 and M1141 will be reset.

Program Example 1:

Communication between PLC and VFD-S series AC motor drives (ASCII Mode, M1143 = Off)

```
  M1002
  ─┤├──────────┬──────┬──────┬──────┐
                │ MOV  │ H87  │ D1120 │  Set up communication protocol 9600, 8, E, 1
                ├──────┼──────┴──────┘
                │ SET  │ M1120 │  Retain communication protocol
                ├──────┼──────┬──────┐
                │ MOV  │ K100 │ D1129 │  Set up communication time-out: 100ms
  X1            └──────┴──────┴──────┘
  ─┤↑├─────────┬──────┬──────┐
                │ SET  │ M1122 │  Set up sending request
  X0            └──────┴──────┘
  ─┤├──────────┬──────┬──────┬──────┬──────┐
                │MODRD │  K1  │ H2101│  K6  │  Set up communication instruction device address 01
                └──────┴──────┴──────┴──────┘  data address H2101 data length 6 words
  ─┤├──────────┬─────────────────────────┐    The received data are stored in D1070 ~ D1085 in ASCII format.
                │ Process of receiving data │    PLC will automatically convert the data into numerals and store
  M1127         └─────────────────────────┘      them in D1050 ~ D1055.
  ─┤├──────────┬──────┬──────┐           Sending/receiving of data is completed. The flag is reset.
  receiving     │ RST  │ M1127 │
  completed     └──────┴──────┘
```
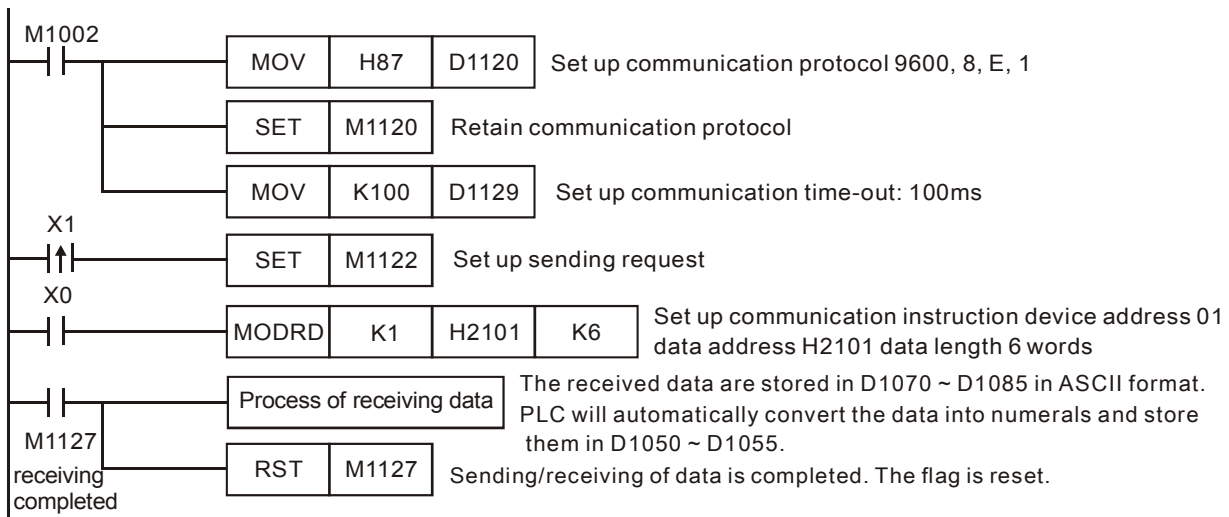
PLC ⇨ VFD-S, PLC sends: "**01 03 2101 0006 D4**"

VFD-S ⇨ PLC , PLC receives: "**01 03 0C 0100 1766 0000 0000 0136 0000 3B**"

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1089 low	'0'	30 H	ADR 1	Address of AC motor
D1089 high	'1'	31 H	ADR 0	drive: ADR (1,0)
D1090 low	'0'	30 H	CMD 1	Instruction code: CMD
D1090 high	'3'	33 H	CMD 0	(1,0)
D1091 low	'2'	32 H		
D1091 high	'1'	31 H	Starting data address	
D1092 low	'0'	30 H		
D1092 high	'1'	31 H		
D1093 low	'0'	30 H		
D1093 high	'0'	30 H	Number of data (counted by words)	
D1094 low	'0'	30 H		
D1094 high	'6'	36 H		
D1095 low	'D'	44 H	LRC CHK 1	Checksum: LRC CHK
D1095 high	'4'	34 H	LRC CHK 0	(0,1)

Registers for received data (responding messages)

Register	DATA		Explanation	
D1070 low	'0'	30 H	ADR 1	
D1070 high	'1'	31 H	ADR 0	
D1071 low	'0'	30 H	CMD 1	
D1071 high	'3'	33 H	CMD 0	
D1072 low	'0'	30 H	Number of data (counted by byte)	
D1072 high	'C'	43 H		
D1073 low	'0'	30 H		PLC automatically convert
D1073 high	'1'	31 H	Content of	ASCII codes to numerals
D1074 low	'0'	30 H	address 2101 H	and store the numeral in
D1074 high	'0'	30 H		D1050 = 0100 H
D1075 low	'1'	31 H		PLC automatically convert
D1075 high	'7'	37 H	Content of	ASCII codes to numerals
D1076 low	'6'	36 H	address 2102 H	and store the numeral in
D1076 high	'6'	36 H		D1051 = 1766 H
D1077 low	'0'	30 H		PLC automatically convert
D1077 high	'0'	30 H	Content of	ASCII codes to numerals
D1078 low	'0'	30 H	address 2103 H	and store the numeral in
D1078 high	'0'	30 H		D1052 = 0000 H

Register	DATA		Explanation
D1079 low	'0'	30 H	Content of address 2104 H — PLC automatically convert ASCII codes to numerals and store the numeral in D1053 = 0000 H
D1079 high	'0'	30 H	
D1080 low	'0'	30 H	
D1080 high	'0'	30 H	
D1081 low	'0'	30 H	Content of address 2105 H — PLC automatically convert ASCII codes to numerals and store the numeral in D1054 = 0136 H
D1081 high	'1'	31 H	
D1082 low	'3'	33 H	
D1082 high	'6'	36 H	
D1083 low	'0'	30 H	Content of address 2106 H — PLC automatically convert ASCII codes to numerals and store the numeral in D1055 = 0000 H
D1083 high	'0'	30 H	
D1084 low	'0'	30 H	
D1084 high	'0'	30 H	
D1085 low	'3'	33 H	LRC CHK 1
D1085 high	'B'	42 H	LRC CHK 0

Program Example 2:

Communication between PLC and VFD-S series AC motor drives (RTU Mode, M1143 = On)
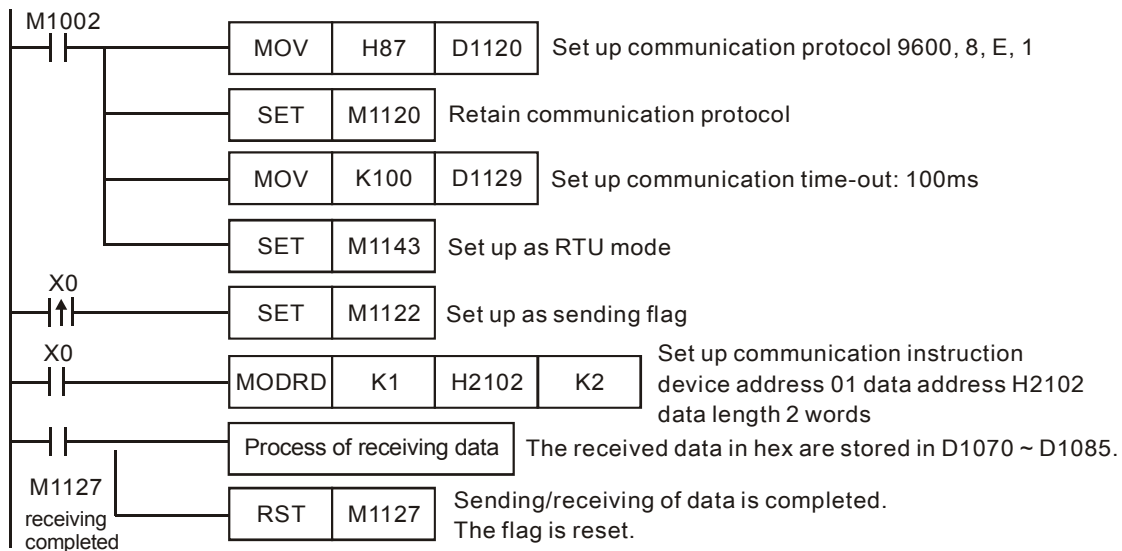
PLC ⇨ VFD-S, PLC sends: **01 03 2102 0002 6F F7**

VFD-S ⇨ PLC, PLC receives: **01 03 04 1770 0000 FE 5C**

Registers for sent data (sending messages)

Register	DATA	Explanation
D1089 low	01 H	Address
D1090 low	03 H	Function
D1091 low	21 H	Starting data address
D1092 low	02 H	
D1093 low	00 H	Number of data (counted by words)
D1094 low	02 H	
D1095 low	6F H	CRC CHK Low
D1096 low	F7 H	CRC CHK High

Registers for received data (responding messages)

Register	DATA	Explanation
D1070 low	01 H	Address
D1071 low	03 H	Function
D1072 low	04 H	Number of data (counted by bytes)
D1073 low	17 H	Content of address 2102 H
D1074 low	70 H	
D1075 low	00 H	Content of address 2103 H
D1076 low	00 H	
D1077 low	FE H	CRC CHK Low
D1078 low	5C H	CRC CHK High

Program Example 3:

1. In the communication between PLC and VFD-S series AC motor drive (ASCII Mode, M1143 = Off), retry when communication time-out, data receiving error and sending address error occur.

2. When X0=On, PLC will read the data in VFFD-S data adress H2100 of device 01 and stores the data in ASCII format in D1070 ~ D1085. PLC will automatically convert the data into numerals and stores them in D1050 ~ D1055.

3. M1129 will be On when communication time-out occurs. The program will trigger M1129 and send request to M1122 for reading the data again.

4. M1140 will be On when data receiving error occurs. The program will trigger M1140 and send request to M1122 for reading the data again.

5. M1141 will be On when sending address error occurs. The program will trigger M1141 and send request to M1122 for reading the data again.

            DVP-PLC Application Manual

Remarks:

1. The activation criteria placed before the three instructions, API 100 MODRD, API 105 RDST, and API 150 MODRW (Function Code H03), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF); otherwise, the data stores in the receiving registers will be incorrect.

2. 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.

API	Mnemonic	Operands			Function		Controllers
101	MODWR	S_1	S_2	n	Write Modbus Data		ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Address of communication device   **S_2**: Address of data to be read   **n**: Data to be written

Explanations:

1. Range of **S_1**: K0 ~ K254

2. See the specifications of each model for their range of use.

3. ES/EX/SS series MPU does not support E, F index register modification.

4. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143

5. MODWR is a drive instruction exclusively for peripheral communication equipment in MODBUS ASCII mode/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 controlling communication (write data) of Delta drives.

6. If the address of **S_2** is illegal to the designed communication device, the device will respond with an error, PLC will records the error code in D1130 and M1140 will be On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. For error codes, see the user manual of VFD-S.

7. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1076. After receiving the feedback data is completed, PLC will auto-check if all data are correct. If there is an error, M1140 will be On.

8. After M1140 or M1141 turn On, the program will send a correct datum to the peripheral equipment. If the feedback datum is correct, M1140 and M1141 will be reset.

Program Example 1:

Communication between PLC and VFD-S series AC motor drives (ASCII Mode, M1143 = Off)
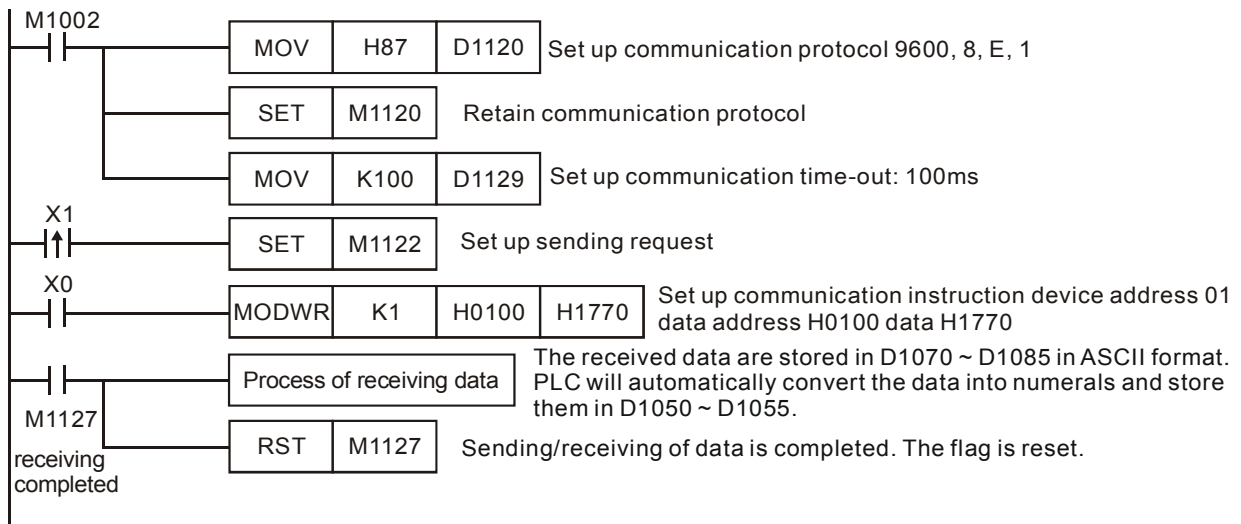
PLC ⇨ VFD-B, PLC sends: " **01 06 0100 1770 71** "

VFD-B ⇨ PLC, PLC receives: " **01 06 0100 1770 71** "

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1089 low	'0'	30 H	ADR 1	Address of AC motor drive:
D1089 high	'1'	31 H	ADR 0	ADR (1,0)
D1090 low	'0'	30 H	CMD 1	Instruction code: 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	Error checksum: LRC CHK
D1095 high	'1'	31 H	LRC CHK 0	(0,1)

PLC receiving data register (response messages)

Register	DATA		Explanation
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-S series AC motor drives (RTU Mode, M1143 = On)

```
        M1002
        ─┤├──────┬──────┌─────────┬──────┬──────┐
                 │      │   MOV   │  H87 │ D1120│  Set up communication protocol 9600, 8, E, 1
                 │      └─────────┴──────┴──────┘
                 │      ┌─────────┬──────────────┐
                 │      │   SET   │    M1120     │  Retain communication protocol
                 │      └─────────┴──────────────┘
                 │      ┌─────────┬──────┬──────┐
                 │      │   MOV   │ K100 │ D1129│  Set up communication time-out: 100ms
                 │      └─────────┴──────┴──────┘
                 │      ┌─────────┬──────────────┐
                 │      │   SET   │    M1143     │  Set up as RTU mode
                 │      └─────────┴──────────────┘
         X0      │      ┌─────────┬──────────────┐
        ─┤↑├─────┤      │   SET   │    M1122     │  Set up as sending flag
                 │      └─────────┴──────────────┘
         X0      │      ┌─────────┬──────┬──────┬──────┐
        ─┤├──────┤      │ MODWR   │  K1  │H2000 │ H12  │  Set up communication instruction device address 01
                 │      └─────────┴──────┴──────┴──────┘  data address H2000 Write in data H12
                 │      ┌──────────────────────────┐
        ─┤├──────┤      │ Process of receiving data │  The received data in hex are stored in D1070 ~ D1085.
        M1127    │      └──────────────────────────┘
        receiving│      ┌─────────┬──────────────┐
        completed└──────│   RST   │    M1127     │  Sending/receiving of data is completed.
                        └─────────┴──────────────┘  The flag is reset.
```
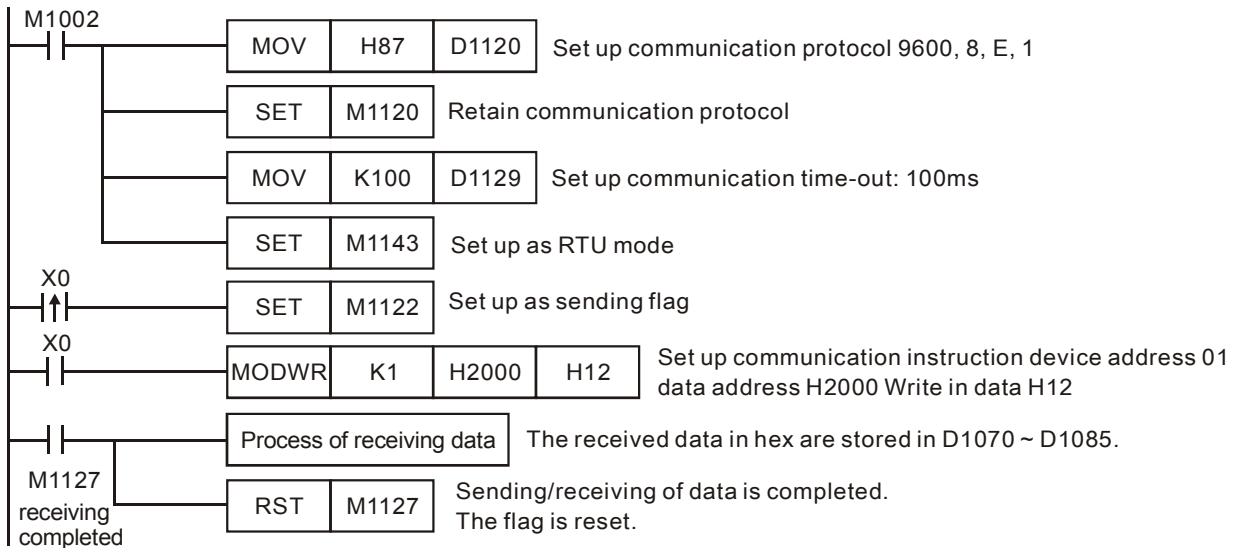
PLC ⇨ VFD-S, PLC sends: **01 06 2000 0012 02 07**

VFD-S ⇨ PLC, PLC receives: **01 06 2000 0012 02 07**
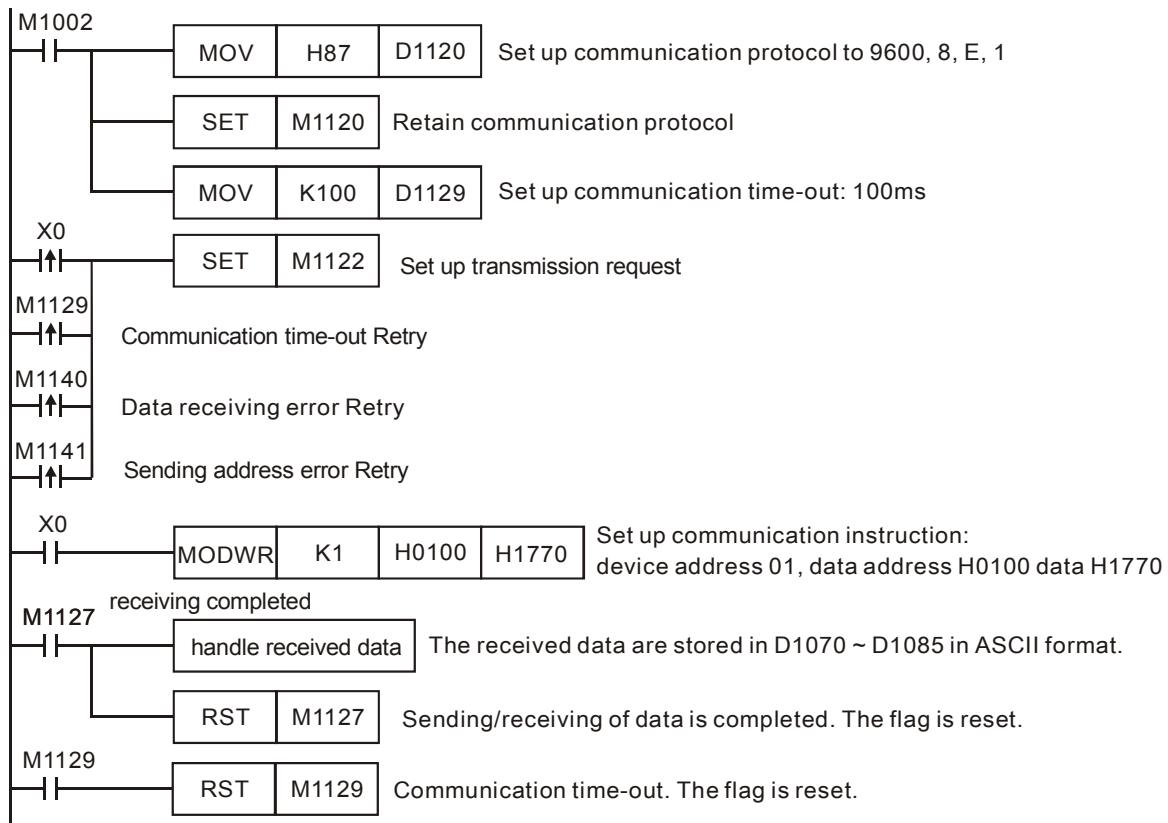
Registers for sent data (sending messages)

Register	DATA	Explanation
D1089 low	01 H	Address
D1090 low	06 H	Function
D1091 low	20 H	Data address
D1092 low	00 H	
D1093 low	00 H	Data contents
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	Explanation
D1070 low	01 H	Address
D1071 low	06 H	Function
D1072 low	20 H	Data address
D1073 low	00 H	
D1074 low	00 H	Data contents
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-S series AC motor drive (ASCII Mode, M1143 = Off), retry when communication time-out, data receiving error and sending address error occur.

2. When X0 = On, PLC will write H1770(K6000) into VFD-S data adress H0100 of device 01.

3. M1129 will be On when communication time-out occurs. The program will trigger M1129 and send request to M1122 for writing the data again.

4. M1140 will be On when data receiving error occurs. The program will trigger M1140 and send request to M1122 for writing the data again.

5. M1141 will be On when sending address error occurs. The program will trigger M1141 and send request to M1122 for writing the data again.

```
M1002
─┤├──────────┌──────┬──────┬──────┐
             │ MOV  │ H87  │ D1120│   Set up communication protocol to 9600, 8, E, 1
             └──────┴──────┴──────┘
             ┌──────┬──────┐
             │ SET  │ M1120│   Retain communication protocol
             └──────┴──────┘
             ┌──────┬──────┬──────┐
             │ MOV  │ K100 │ D1129│   Set up communication time-out: 100ms
             └──────┴──────┴──────┘
X0
─┤↑├──────────┌──────┬──────┐
             │ SET  │ M1122│   Set up transmission request
             └──────┴──────┘
M1129
─┤↑├──   Communication time-out Retry

M1140
─┤↑├──   Data receiving error Retry

M1141
─┤↑├──   Sending address error Retry

X0
─┤├──────────┌───────┬────┬───────┬───────┐   Set up communication instruction:
             │ MODWR │ K1 │ H0100 │ H1770 │   device address 01, data address H0100 data H1770
             └───────┴────┴───────┴───────┘
M1127       receiving completed
─┤├──────────┌────────────────────┐
             │ handle received data│   The received data are stored in D1070 ~ D1085 in ASCII format.
             └────────────────────┘
             ┌──────┬──────┐
             │ RST  │ M1127│   Sending/receiving of data is completed. The flag is reset.
             └──────┴──────┘
M1129
─┤├──────────┌──────┬──────┐
             │ RST  │ M1129│   Communication time-out. The flag is reset.
             └──────┴──────┘
```

Program Example 4:

1. In the communication between PLC and VFD-S series AC motor drive (ASCII Mode, M1143 = Off), retry when communication time-out, data receiving error and sending address error occur. Times of retry = D0 (default = 3). When communication Retry is successful, the user can return to controlling by triggering criteria.

2. When X0 = On, PLC will write H1770(K6000) into VFD-S data adress H0100 of device 01.

3. M1129 will be On when communication time-out occurs. The program will trigger M1129 and send request to M1122 for writing the data again. Times of Retry = D0 (default = 3)

4. M1140 will be On when data receiving error occurs. The program will trigger M1140 and send request to M1122 for writing the data again. Times of Retry = D0 (default = 3)

5. M1141 will be On when sending address error occurs. The program will trigger M1141 and send request to M1122 for writing the data again. Times of Retry = D0 (default = 3)
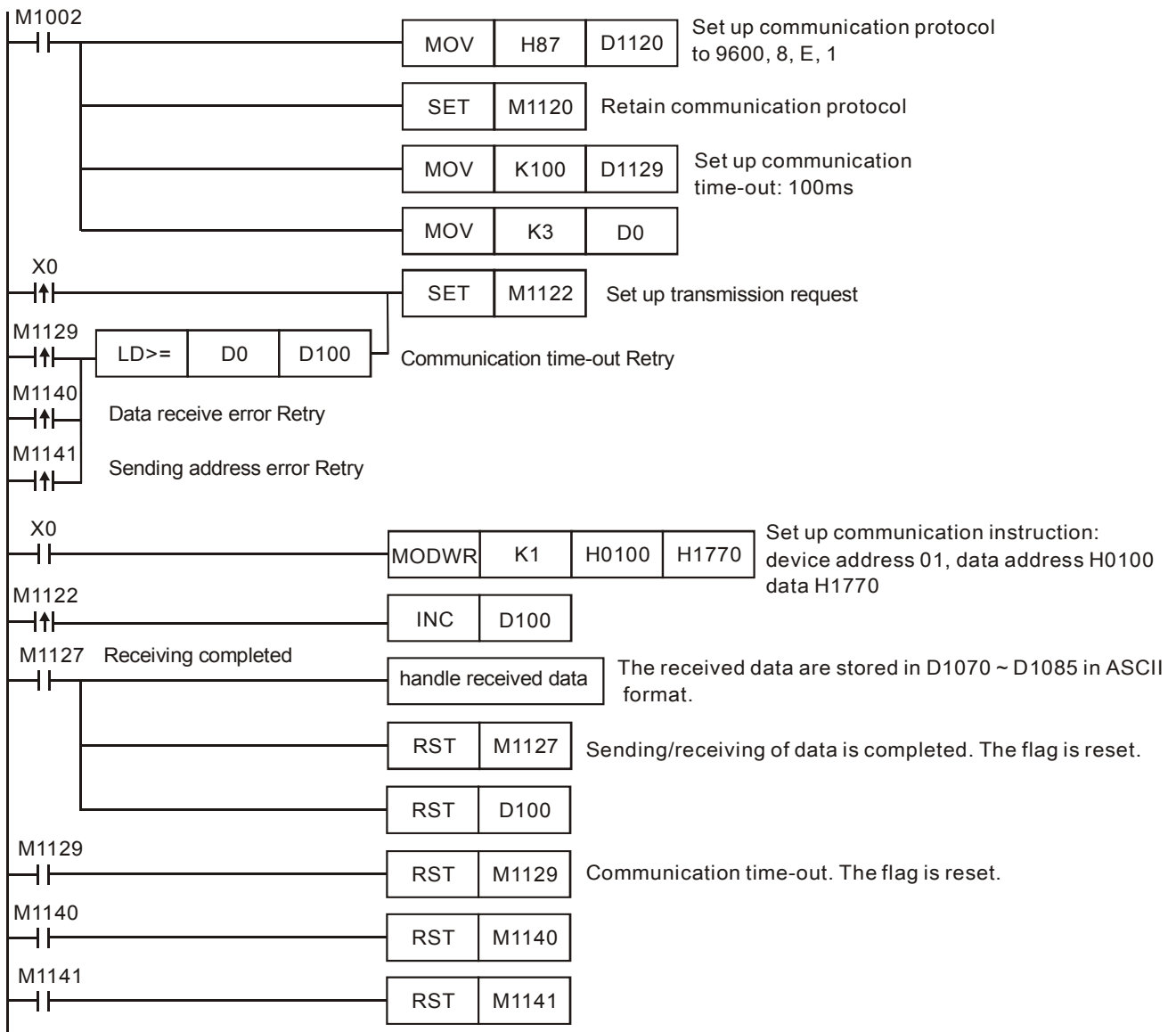
Remarks:

1. For the registers for flag settings, see explanations in API 80 RS.

2. The activation criteria placed before the two instructions, API 101 MODWR and API 150 MODRW (Function Code H06, H10), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF) and have to enable sending request M1122 first.

3. 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.

API	Mnemonic	Operands			Function	Controllers
102	FWD	**S₁**	**S₂**	**n**	Forward Running of VFD-A	ES/EX/SS/SA/SX/SC EH/SV

Type OP	Bit Devices					Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FWD: 7 steps
S₁					*	*							*			
S₂					*	*							*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

API	Mnemonic	Operands			Function	Controllers
103	REV	**S₁**	**S₂**	**n**	Reverse Running of VFD-A	ES/EX/SS/SA/SX/SC EH/SV

Type OP	Bit Devices					Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	REV: 7 steps
S₁					*	*							*			
S₂					*	*							*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

API	Mnemonic	Operands			Function	Controllers
104	STOP	**S₁**	**S₂**	**n**	Stop VFD-A	ES/EX/SS/SA/SX/SC EH/SV

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
S₁					*	*							*			
S₂					*	*							*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Address of communication device S_2: Rotation frequency of AC motor drive n: Target to be instructed
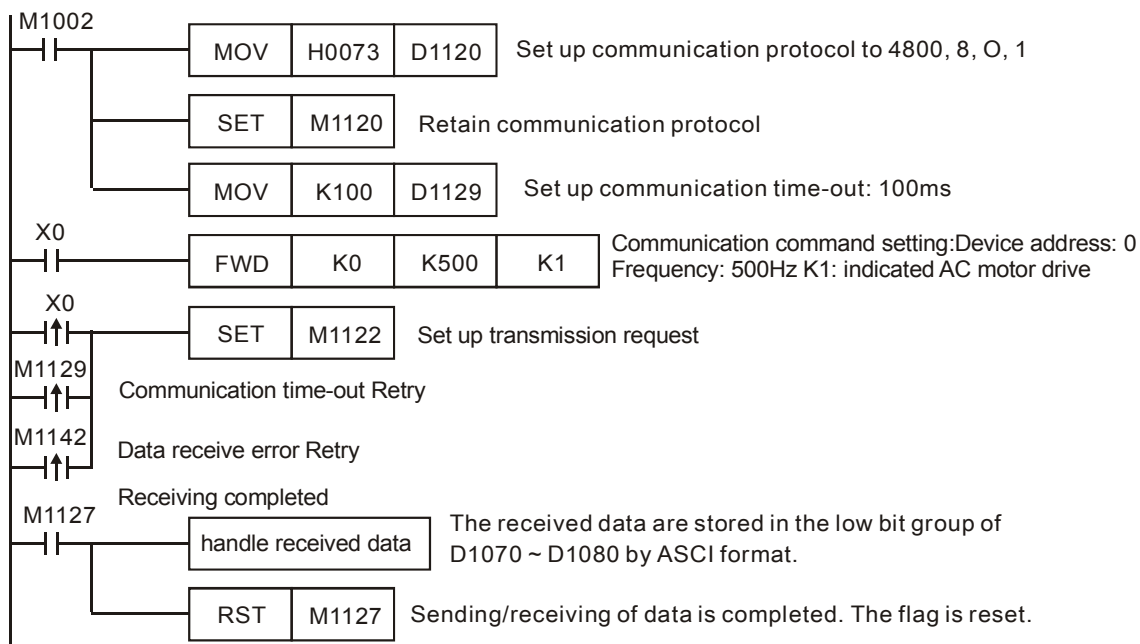
Explanations:

1. Range of S_1: K0 ~ K31

2. Range of n: K1 or K2

3. See the specifications of each model for their range of use.

4. ES series MPU does not support E, F index register modification.

5. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143

6. FWD/REV/STOP are handy instructions exclusively for Delta VFD-A/H series AC motor drive to perform forward running/reverse running/stop. Be sure to set up communication time-out (D1129) when executing this instruction.

7. S_2 = operation frequency of AC motor drive. Set frequency in A-series AC motor drive: K0 ~ K4,000   (0.0Hz ~

400.0Hz). Set frequency in H-series: K0 ~ K1,500 (0Hz ~ 1,500Hz).

8. **n** = instructed target. **n**=1: AC motor drive at designated address. **n**=2: all connected AC motor drives.

9. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1080. After receiving the feedback data is completed, PLC will auto-check if all data are correct. If there is an error, M1142 will be On. When **n** = 2, PLC will not receive any data.

Program Example :

Communication between PLC and VFD-A series AC drives, retry for communication time-out and received data error.

```
M1002
--| |----+----[ MOV | H0073 | D1120 ]   Set up communication protocol to 4800, 8, O, 1
         |
         +----[ SET | M1120 ]            Retain communication protocol
         |
         +----[ MOV | K100 | D1129 ]     Set up communication time-out: 100ms
X0
--| |---------[ FWD | K0 | K500 | K1 ]   Communication command setting:Device address: 0
                                         Frequency: 500Hz K1: indicated AC motor drive
X0
--|↑|----+----[ SET | M1122 ]            Set up transmission request
M1129    |
--|↑|----+   Communication time-out Retry
M1142    |
--|↑|----+   Data receive error Retry
M1127    |   Receiving completed
--| |----+----[ handle received data ]   The received data are stored in the low bit group of
         |                                D1070 ~ D1080 by ASCI format.
         +----[ RST | M1127 ]            Sending/receiving of data is completed. The flag is reset.
```

PLC ⇨ VFD-A, PLC sends: "**C ♥ ☺ 0001 0500** "

VFD-A ⇨ PLC, PLC sends: "**C ♥ ♠ 0001 0500** "

Registers for sent data (sending messages)

Register	DATA		Explanation
D1089 low	'C'	43 H	Start word of instruction
D1090 low	'♥'	03 H	Checksum
D1091 low	'☺'	01 H	Instructed target
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	Running instruction
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	Start word of instruction
D1071 low	'♥'	03 H	Checksum
D1072 low	'♠'	06 H	Reply authorization (correct: 06H, incorrect: 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	Running instruction
D1078 low	'5'	35 H	
D1079 low	'0'	30 H	
D1080 low	'0'	30 H	

Remarks:

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.

API	Mnemonic	Operands	Function	Controllers
105	RDST	Ⓢ ⓝ	Read VFD-A Status	ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Address of communicatino device   **n**: Target to be instructed

Explanations:

1. Range of **S**: K0 ~ K31

2. Range of **n**: K0 ~ K3

3. See the specifications of each model for their range of use.

4. ES series MPU does not support E, F index register modification.

5. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143

6. **n**: Instructed target (to be read) in AC motor drive
   - n=0, frequency
   - n=1, output frequency
   - n=2, output current
   - n=3, running instruction

7. Data sent back (feedback) from AC motor drive (11 bytes, see VFD-A user manual) are stored in the low bytes of D1070 ~ D1080.

   "Q, S, B, Uu, Nn, ABCD"

Feedback	Explanation	Data storage
Q	Start word: 'Q' (51H).	D1070 low
S	Checksum code: 03H.	D0171 low
B	Instruction authorization. correct: 06H, incorrect: 07H.	D1072 low
U	Communication address (address: 00~31). "Uu" = ("00" ~ "31") indicated	D1073 low
U	in ASCII format.	D1074 low
N	Instructed target (00 ~ 03)."Nn" = ("00 ~ 03") indicated in ASCII format.	D1075 low
N		D1076 low
A		D1077 low
B	Instructed data. The content of "ABCD" differs upon the instructed targets	D1078 low
C	(00 ~ 03). 00 ~ 03 indicate frequency, current and running mode	D1079 low
D	respectively. Please refer to the explanations below for details.	D1080 low
	Nn = "00" Frequency instruction = ABC.D (Hz) Nn = "01" Output instruction = ABC.D (Hz) Nn = "02" Output current = ABC.D (A) PLC will automatically convert the ASCII characters of "ABCD" into numerals and store the numeral in D1050. For example, assume "ABCD" = "0600", PLC will convert ABCD into K0600 (0258 H) and store it in the special register D1050.	

	Nn = "03"	Running instruction					
	'A' =	'0'	Stop,		'5'	JOG (forward)	
		'1'	Forward running		'6'	JOG (reverse)	
		'2'	Stop,		'7'	JOG (reverse)	
		'3'	Reverse running		'8'	Abnormal	
		'4'	JOG (forward),				
		ES series PLCs will convert the ASCII characters of "A" into a numeral and store the numeral in D1051. For example, assume "A" = "3", PLC will convert A into K3 and store it in the special register D1051. SA/EH series PLCs will store the numeral in low bytes of D1051.					
	'B' =	b7	b6	b5	b4	Source of running instruction	
		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 signal frequency instruction	
		1	0	1	0	RS-485 communication interface	
		1	0	1	1	Up/Down control	
		b3	= 0	No DC braking stop	1	DC braking stop	
		b2	= 0	No DC braking startup	1	DC braking startup	
		b1	= 0	Forward running	1	Reverse running	
		b0	= 0	Stop	1	Running	
		ES series PLCs will store "B" in special auxiliary relay M1168 (b0) ~ M1175 (b7). SA/EH series PLCs will store "B" (in hex) in the high bytes of special register D1051.					
	"CD" =	"00"	No abnormal record		"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"		
		ES/SA/EH series PLCs will convert the ASCII characters of "CD" into a numerals and store the numeral in D1052. For example, assume "CD" = "16", PLC will convert CD into K16 and store it in the special register D1052.					

Remarks:

1. The activation criteria placed before the three instructions, API 100 MODRD, API 105 RDST and API 150 MODRW (Function Code 03), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF); otherwise, the data stores in the receiving registers will be incorrect.

2. For the registers for flag settings, see explanations in API 80 RS.

3. 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.

API	Mnemonic	Operands	Function	Controllers
106	RSTEF	S n	Reset Abnormal VFD-A	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices												Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		RSTEF: 5 steps
S					*	*							*				
n					*	*							*				

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Address of communication device   **n**: Target to be instructed

Explanations:

1. Range of **S**: K0 ~ K31

2. Range of **n**: K1 or K2

3. See the specifications of each model for their range of use.

4. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143

5. RSTEF is a handy communication instruction exclusively for Delta VFD-A series AC motor drives and is used for reset when the AC motor drive operates abnormally.

6. **n**: instructed target. **n**=1: AC motor drive at assigned address. **n**=2: all connected AC motor drives.

7. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1089. If **n** = 2, there will be no feedback data.

Remarks:

1. The activation criteria placed before the three instructions, API 100 MODRD, API 105 RDST and API 150 MODRW (Function Code 03), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF); otherwise, the data stores in the receiving registers will be incorrect.

2. For the registers for flag settings, see explanations in API 80 RS.

3. 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.

API	Mnemonic	Operands	Function	Controllers
107	LRC P	**S** **n** **D**	Checksum LRC Mode	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Start operation device for ASCII mode checksum   **n**: Number of calculated bits   **D**: Start device for storing the operation result   LRC checksum: See remarks.

Explanations:

1. Range of **n**: K1 ~ K256

2. See the specifications of each model for their range of use.

3. Flag: M1161 (switching between 8/16 bit modes)

4. **n** has to be even. If **n** does not fall within its range, an operation error will occur, the instruction will not be executed, M1067, M1068 = On and D1067 will record the error code H'0E1A.

5. In 16-bit conversion mode: When M1161 = Off, **S** divides its hex data area into higher 8 bits and lower 8 bits and performs LRC checksum operation on each bit. The data will be sent to the higher 8 bits and lower 8 bits in **D**. **n** = the number of calculated bits.

6. In 8-bit conversion mode: When M1161 = On, **S** divides its hex data area into higher 8 bits (invalid data) and lower 8 bits and performs LRC checksum operation on each bit. The data will be sent to the lower 8 bits in **D** and occupy 2 registers. **n** = the number of calculated bits. (All higher bits in **D** are "0".)

Program Example:

When PLC communicates with VFD-S series AC motor drives (In ASCII mode, M1143 = Off), (In 8-bit mode, M1161 = On), the sent data write in advance the 6 data read starting from H2101 of VFD-S.

```
 M1002
──┤├──┬──┤ MOV │ H86  │ D1120 │ Set up communication protocol to 9600, 7, E, 1
      │
      ├──┤ SET │ M1120 │ Retain communication protocol
      │
      └──┤ MOV │ K100 │ D1129 │ Set up communication time-out: 100ms

 sending request pulse
──┤↑├──┬──┤ Write in sent data in advance
 pulse │
       └──┤ SET │ M1122 │ Set up transmission request

 X10
──┤├──┬──┤ RS │ D100 │ K17 │ D120 │ K35 │
 receiving completed

 M1123
──┤├──┬──┤ Process of receiving data
      │
      └──┤ RST │ M1123 │ Sending/receiving of data is completed.
                         The flag is reset.
```

PLC ⇨ VFD-S, PLC sends: **": 01 03 2101 0006 D4 CR LF "**

Registers for sent data (sending messages)

Register	DATA		Explanation	
D100 low	': '	3A H	STX	
D101 low	'0'	30 H	ADR 1	Address of AC motor
D102 low	'1'	31 H	ADR 0	drive: ADR (1,0)
D103 low	'0'	30 H	CMD 1	Instruction code: CMD
D104 low	'3'	33 H	CMD 0	(1,0)
D105 low	'2'	32 H	Starting data address	
D106 low	'1'	31 H		
D107 high	'0'	30 H		
D108 low	'1'	31 H		
D109 low	'0'	30 H	Number of data (counted by words)	
D110 low	'0'	30 H		
D111 low	'0'	30 H		
D112 low	'6'	36 H		
D113 low	'D'	44 H	LRC CHK 1	Error checksum: LRC
D114 low	'4'	34 H	LRC CHK 0	CHK (0,1)
D115 low	CR	A H	END	
D116 low	LF	D H		

The error checksum LRC CHK (0,1) can be calculated by LRC instruction (in 8-bit mode, M1161 = On).

LRC checksum: 01 H + 03 H + 21 H + 01 H + 00 H + 06 H = 2C H. Obtain 2's complement, D4H, and store 'D'(44H) in the lower 8 bits of D113 and '4'(34H) in the lower 8 bits of D114.

Remarks:

1. The format of ASCII mode with a communication datum

STX	': '	Start word = ': ' (3AH)
Address Hi	' 0 '	Communication:
Address Lo	' 1 '	8-bit address consists of 2 ASCII codes
Function Hi	' 0 '	Function code:
Function Lo	' 3 '	8-bit function consists of 2 ASCII codes
DATA (n-1)	' 2 '	Data content:
…….	' 1 '	n × 8-bit data consists of 2n ASCII
DATA 0	' 0 '	codes
	' 2 '	
	' 0 '	
	' 0 '	
	' 0 '	
	' 2 '	
LRC CHK Hi	' D '	LRC checksum:
LRC CHK Lo	' 7 '	8-bit checksum consists of 2 ASCII codes
END Hi	CR	End word:
END Lo	LF	END Hi = CR (0DH), END Lo = LF(0AH)

2. LRC checksum: 2's complement of the summed up value of communication address and data. For example, 01 H + 03 H + 21 H + 02 H + 00 H + 02 H = 29 H. Obtain 2's complement = D7H.

API	Mnemonic		Operands			Function		Controllers		
108	CRC	P	**S** **n** **D**			Checksum CRC Mode		ES/EX/SS SA/SX/SC EH/SV		

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							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start operation device for RTU mode checksum   **n**: Number of calculated bits   **D**: Start device for storing the operation result   CRC checksum: See remarks.

Explanations:

1.   Range of **n**: K1 ~ K256

2.   Flags: M1161 (switching between 8/16-bit modes)

3.   If **n** does not fall within its range, an operation error will occur, the instruction will not be executed, M1067, M1068 = On and D1067 will record the error code H'0E1A.

4.   In 16-bit conversion mode: When M1161 = Off, **S** divides its hex data area into higher 8 bits and lower 8 bits and performs CRC checksum operation on each bit. The data will be sent to the higher 8 bits and lower 8 bits in **D**. **n** = the number of calculated bits.

5.   In 8-bit conversion mode: When M1161 = On, **S** divides its hex data area into higher 8 bits (invalid data) and lower 8 bits and performs CRC checksum operation on each bit. The data will be sent to the lower 8 bits in **D** and occupy 2 registers. **n** = the number of calculated bits. (All higher 8 bits in **D** are "0".)

Program Example:

When PLC communicates with VFD-S series AC motor drives (In RTU mode, M1143 = On), (In 16-bit mode, M1161 = On), the sent data write in advance H12 into H2000 of VFD-S.
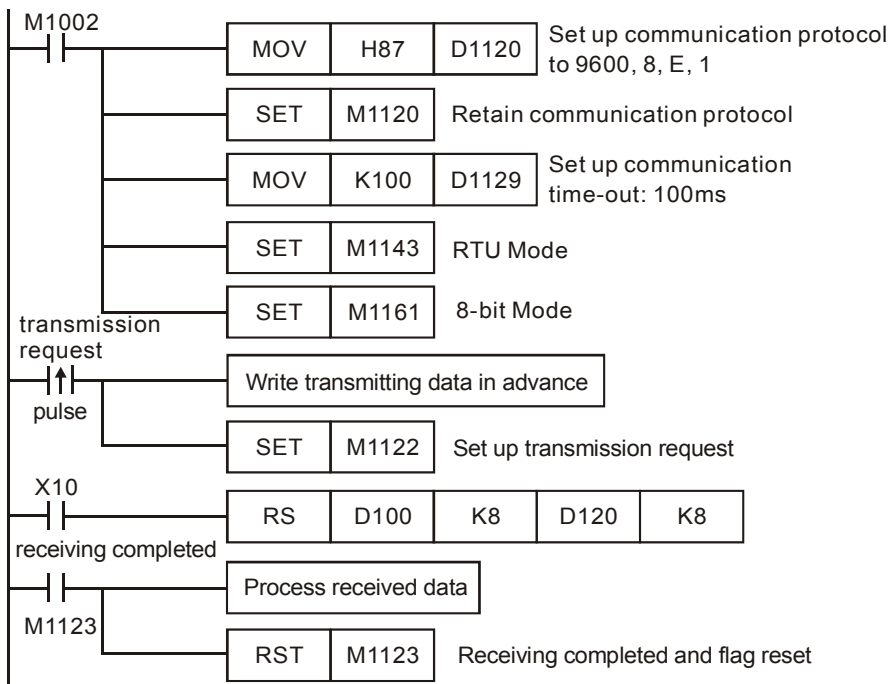
PLC ⇨ VFD-S, PLC sends: **01 06 2000 0012 02 07**

Registers for sent data (sending 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 0
D107 low	07 H	CRC CHK 1

The error checksum CRC CHK (0,1) can be calculated by CRC instruction (in 8-bit mode, M1161 = On).

CRC checksum: 02 H is stored in the lower 8 bits of D106 and 07 H in the lower 8 bits of D107,

Remarks:

1. The format of RTU mode with a communication datum

	START	Time interval
	Address	Communication address: 8-bit binary
	Function	Function code: 8-bit binary
	DATA (n-1)	Data content:
	…….	n × 8-bit data
	DATA 0	
	CRC CHK Low	CRC checksum:
	CRC CHK High	16-bit CRC checksum consists of 2 8-bit binaries
	END	Time interval

2. CRC checksum starts from Address and ends at Data content.

The operation of CRC checksum:

Step 1: Make the 16-bit register (CRC register) = FFFFH

Step 2: Exclusive OR the first 8-bit byte message instruction and the low-bit 16-bit CRC register. Store the result in CRC register.

Step 3: Shift the CRC register one bit to the right and fill 0 in the higher bit.

Step 4: Check the value that shifts to the right. If it is 0, store the new value from Step 3 into the CRC register, otherwise, Exclusive OR A001H and the CRC register, and store the result in the CRC register.

Step 5: Repeat Step 3 ~ 4 and finish calculating the 8 bits.

Step 6: Repeat Steps 2 ~ 5 for obtaining the next 8-bit message instruction until all the message instructions are calculated. In the end, the obtained CRC register value is the CRC checksum. Be aware that CRC checksum should be placed in the checksum of the message instruction.

API	Mnemonic		Operands	Function		Controllers
109	SWRD	P	**D**	Read Digital Switch		ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: Device for storing the read value

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1104 ~ M1111 (status of digital switch)

3. This instruction stores the value read from digital switch function card into **D**.

4. The read value is stored in the low byte in D. Every switch has a corresponding bit.

5. When there is no digital function card inserted, the error message C400 (hex) will appear in grammar check.

Program Example:

1. There are l 8 DIP switches on the digital switch function card. After the switches are read by SWRD instruction, the status of each switch will correspond to M0 ~ M7.

```
M1000
 ┤├────────────[ SWRD │ K2M0 ]

M0
 ┤├────────────( Y0 )

M1
 ┤├────────────[ MOV │ K2M0 │ D0 ]

M2
 ┤├────────────[ CNT │ C0 │ K10 ]

M3
 ┤├────────────[ RST │ C0 ]

M4
 ┤├────────────[ TMR │ T0 │ K100 ]
```
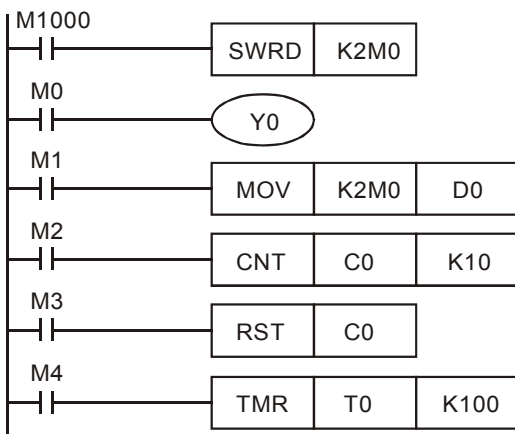
2. The status of M0 ~ M7 can be executed by each contact instruction.

3. The execution of END instruction indicates that the process of input is completed. REF (I/O refresh) instruction will be invalid.

4. When SWRD instruction uses the data in digital switch function card, it can read minimum 4 bits (K1Y*, K1M* or K1S*).

Remarks:

When digital switch function card is inserted, the status of the 8 DIP switches will correspond to M1104 ~ M1111.

API	Mnemonic			Operands			Function	Controllers
110	D	ECMP	P	S_1 S_2 D			Floating Point Compare	ES/EX/SS SA/SX/SC EH/SV

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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Binary floating point comparison value 1   **S₂**: Binary floating point comparison value 2   **D**: Comparison result

Explanations:

1. **D** occupies 3 consecutive devices.

2. See the specifications of each model for their range of use.

3. The binary floating point values **S₁** and **S₂** are compared with each other. The comparison result (>, =, <) is stored in **D**.

4. If **S₁** or **S₂** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the comparison.

Program Example:

1. Designated device M10 and M10 ~ M12 are automatically occupied.

2. When X0 = On. DECMP instruction will be executed and one of M10 ~ M12 will be On. When X0 = Off, DECMP instruction will not be executed and M10 ~ M12 will remain their status before X0 = Off.

3. To obtain results ≧, ≦, ≠, serial-parallel M10 ~ M12.

4. Use RST or ZRST instruction to clear the result.

```
  X0
──┤├──────┌─────────┬──────┬──────┬──────┐
          │ DECMP   │  D0  │ D100 │ M10  │
          └─────────┴──────┴──────┴──────┘
   M10
  ──┤├────────  On when (D1,D0)>(D101,D100)
   M11
  ──┤├────────  On when (D1,D0)=(D101,D100)
   M12
  ──┤├────────  On when (D1,D0)<(D101,D100)
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands				Function		Controllers	
111	D	EZCP	P	S₁ S₂	S	D		Floating Point Zone Compare		ES/EX/SS	SA/SX/SC	EH/SV

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
S₁					*	*							*			
S₂					*	*							*			
S					*	*							*			
D		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Lower bound of binary floating point   **S₂**: Upper bound of binary floating point   **S**: Binary floating point comparison result   **D**: Comparison result

Explanations:

1. **D** occupied 3 consecutive devices.

2. **S₁** ≤ **S₂**. See the specifications of each model for their range of use.

3. **S** is compared with **S₁** and **S₂** and the result (>, =, <) is stored in **D**.

4. If **S₁** or **S₂** is andesignated constant K or H, the instruction will convert the constant into a binary floating point value before the comparison.

5. When **S₁** > **S₂**, **S₁** will be used as upper/lower bound for the comparison.

Program Example:

1. Designated device M0 and M0 ~ M2 are automatically occupied.

2. When X0 = On. DEZCP instruction will be executed and one of M0 ~ M2 will be On. When X0 = Off, EZCP instruction will not be executed and M0 ~ M2 will remain their status before X0 = Off.
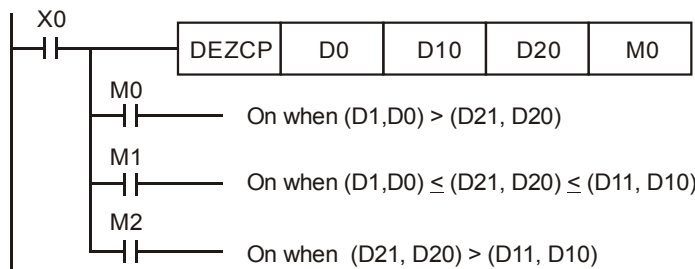
3. Use RST or ZRST instruction to clear the result.

```
   X0
───┤├────┬──── DEZCP │ D0 │ D10 │ D20 │ M0
         │
   M0    │
───┤├────┴──── On when (D1,D0) > (D21, D20)
   M1
───┤├───────── On when (D1,D0) ≤ (D21, D20) ≤ (D11, D10)
   M2
───┤├───────── On when (D21, D20) > (D11, D10)
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic		Operands	Function	Controllers
112	D MOVR P		**S** **D**	Move Floating Point Data	ES/EX/SS SA/SX/SC EH/SV

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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S：Source floating point data   **D**：Destination device

Explanations:

1. **S** can only be in floating point (FX.XX).

2. See the specifications of each model for their range of use.

3. This instruction is able to enter floating point values directly in **S**.

4. When the instruction is executed, the content in **S** is moved directly into **D**. When the instruction is not executed, the content in **D** will not be modified.

Program Example:

1. User DMOVR instruction to move 32-bit floating point data.

2. When X0 = Off, the content in (D11、D10) remains unchanged. When X0 = On, the present value F1.20000004768372 will be moved to data registers (D11, D10).

```
   X0
───┤ ├──────────┤ DMOVR │ F1.20000004768372 │ D10 │
```
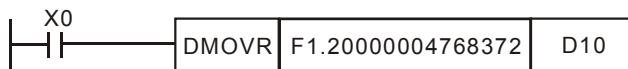
Remarks:

This instruction only supports ES V6.1, SA/SX_V1.1, SV_V1.2, EH_V1.2, EH2/SV_V1.0 and above versions.

API		Mnemonic		Operands	Function		Controllers
116	D	RAD	P	⒮ ⒟	Angle → Radian		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source (angle)   **D**: Result (radian)

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. Radian = degree × (π/180)

4. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

5. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

6. If the result = 0, the zero flag M1020 = On.

Program Example:

When X0 = On, designate the degree of binary floating point (D1, D0). Convert the angle into radian and store the result in binary floating point in (D11, D10).
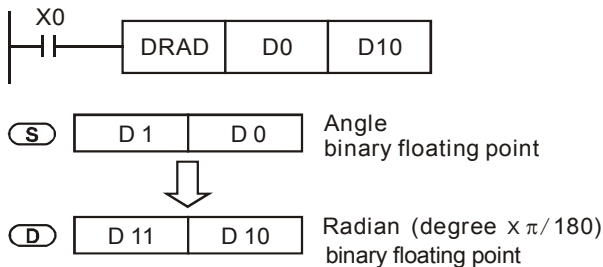
```
  X0
──┤ ├──┤ DRAD │ D0 │ D10 ├
```

⒮ | D 1 | D 0 | Angle
binary floating point

⬇

⒟ | D 11 | D 10 | Radian (degree × π/180)
binary floating point

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands		Function			Controllers		
117	D	DEG	P	Ⓢ Ⓓ		Radian → Angle			ES/EX/SS SA/SX/SC EH/SV		

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source (radian)   **D**: Result (angle)

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. Degree = radian × (180/π)

4. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

5. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

6. If the result = 0, the zero flag M1020 = On.

Program Example:

When X0 = On, designate the angle of binary floating point (D1, D0). Convert the radian into angle and store the result in binary floating point in (D11, D10).
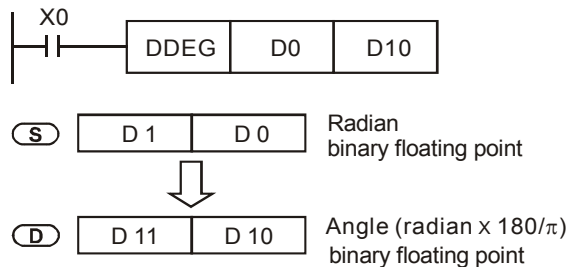
```
 X0
─┤├─────┤ DDEG │ D0 │ D10 │
```

Ⓢ | D 1 | D 0 | Radian
              binary floating point

            ⬇

Ⓓ | D 11 | D 10 | Angle (radian x 180/π)
              binary floating point

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands		Function			Controllers
118	D	EBCD	P	S D		Float to Scientific Conversion			ES/EX/SS SA/SX/SC EH/SV

Type		Bit Devices						Word Devices									Program Steps
OP		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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source   **D**: Result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. This instruction converts binary floating point value in the register designated by **S** into decimal floating point value and stores it in the register designated by **D**.

4. PLC conducts floating point operation in binary format. DEBCD instruction is exclusively for converting floating points from binary to decimal.

5. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

6. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

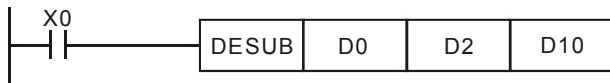7. If the result = 0, the zero flag M1020 = On.
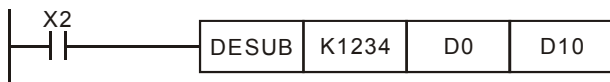
Program Example:

When X0 = On, the binary floating points in D1 and D0 will be converted into decimal floating points and stored in D3 and D2.
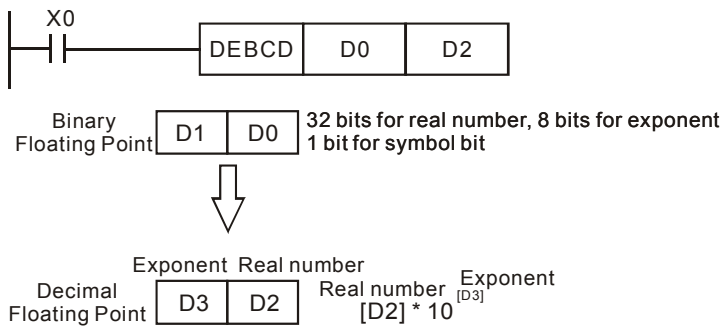
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic		Operands		Function	Controllers
119	D EBIN P		$\boxed{S}$ $\boxed{D}$		Scientific to Float Conversion	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source   **D**: Result

Explanations:

1. See the specifications of each model for their range of use.

2. Flag: M1020 (zero flag)

3. This instruction converts decimal floating point value in the register designated by **S** into binary floating point value and stores it in the register designated by **D**.

4. DEBIN instruction is exclusively for converting floating points from decimal to binary.

5. Range of decimal floating point real numbers: -9.999 ~ +9,999. Range of exponants: -41 ~ +35. Range of PLC decimal floating points: $\pm 1{,}175 \times 10^{-41} \sim \pm 3{,}402 \times 10^{+35}$.

6. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X1 = On, the decimal floating points in D1 and D0 will be converted into binary floating points and stored in D3 and D2.
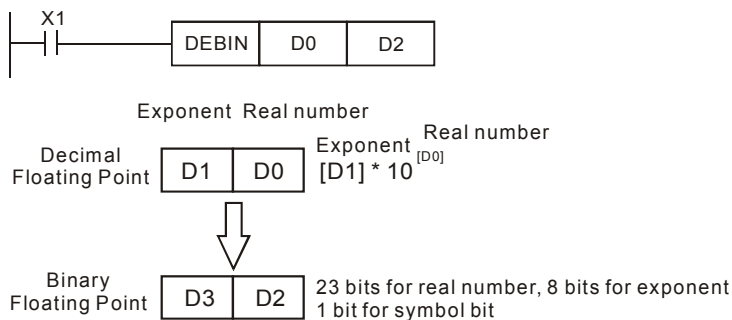
Program Example 2:

1. Use FLT instruction (API 149) to convert BIN integer into binary floating point before performing floating point operation. The value to be converted must be BIN integer and use DEBIN instruction to convert the floating point into a binary one.

2. When X0 = On, move K3,140 to D0 and K-3 to D1 to generate decimal floating point ($3.14 = 3140 \times 10^{-3}$).
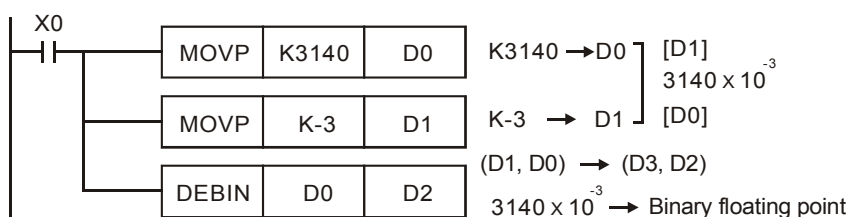
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands			Function	Controllers
120	D	EADD	P	S_1	S_2	D	Floating Point Addition	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices												Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEADD, DEADDP: 13 steps
S_1					*	*							*			
S_2					*	*							*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

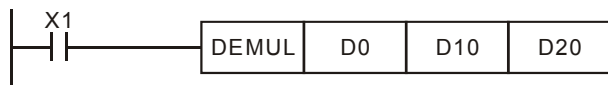**S₁**: Summand   **S₂**: Addend   **D**: Sum

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. $S_1 + S_2 = D$. The floating point value in the register designated by S_1 and S_2 are added up and the result is stored in the register designated by **D**. The addition is conducted in binary floating point system.

4. If **S₁** or **S₂** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.

5. **S₁** and **S₂** can designate the same register. In this case, if the "continuous execution" instruction is in use, during the period when the criteria contact in On, the register will be added once in every scan by pulse execution instruction DEADDP.

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X0 = On, binary floating point (D1, D0) $+$ binary floating point (D3, D2) and the result is stored in (D11, D10).

X0			
├─┤├─	DEADD	D0	D2	D10
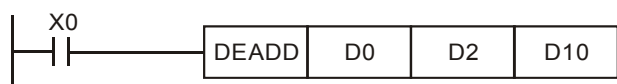
Program Example 2:

When X2 = On, binary floating point (D11, D10) + K1234 (automatically converted into binary floating point) and the result is stored in (D21, D20).

X2			
├─┤├─	DEADD	D10	K1234	D20
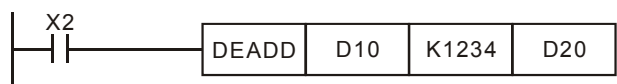
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands			Function		Controllers
121	D	ESUB	P	S_1 S_2 D			Floating Point Subtraction		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DESUB, DESUBP: 13 steps
S_1					*	*							*			
S_2					*	*							*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Minuend   **S₂**: Subtrahend   **D**: Remainder

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. $S_1 - S_2 = D$. The floating point value in the register designated by **S₂** is subtracted from the floating point value in the register assigned by **S₁** and the result is stored in the register designated by **D**. The subtraction is conducted in binary floating point system.

4. If **S₁** or **S₂** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.

5. **S₁** and **S₂** can designate the same register. In this case, if the "continuous execution" instruction is in use, during the period when the criteria contact in On, the register will be subtracted once in every scan by pulse execution instruction DESUBP.

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X0 = On, binary floating point (D1, D0) $-$ binary floating point (D3, D2) and the result is stored in (D11, D10).

```
 X0
─┤ ├──────[ DESUB │ D0 │ D2 │ D10 ]
```

Program Example 2:

When X2 = On, K1234 (automatically converted into binary floating point) $-$ binary floating point (D1, D0) and the result is stored in (D11, D10).

```
 X2
─┤ ├──────[ DESUB │ K1234 │ D0 │ D10 ]
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands			Function		Controllers
122	D	EMUL	P	S_1 S_2 D			Floating Point Multiplication		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices												Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		DEMUL, DEMULP: 13 steps
S_1					*	*							*				
S_2					*	*							*				
D													*				

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. **$S_1 \times S_2 = D$**. The floating point value in the register assigned by **S_1** is multiplied with the floating point value in the register designated by **S_2** and the result is stored in the register designated by **D**. The multiplication is conducted in binary floating point system.

4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.

5. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" instruction is in use, during the period when the criteria contact in On, the register will be multiplied once in every scan by pulse execution instruction DEMULP.

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X1 = On, binary floating point (D1, D0) × binary floating point (D11, D10) and the result is stored in (D21, D20).

X1	DEMUL	D0	D10	D20

Program Example 2:

When X2 = On, K1234 (automatically converted into binary floating point) × binary floating point (D1, D0) and the result is stored in (D11, D10).

X2	DEMUL	K1234	D0	D10
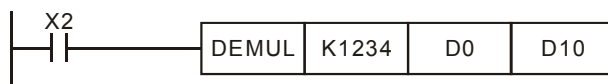
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands			Function		Controllers
123	D	EDIV	P	S_1 S_2 D			Floating Point Division		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEDIV, DEDIVP: 13 steps
S_1					*	*							*			
S_2					*	*							*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

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

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. **$S_1 \div S_2 = D$**. The floating point value in the register designated by **S_1** is divided by the floating point value in the register assigned by **S_2** and the result is stored in the register designated by **D**. The division is conducted in binary floating point system.

4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.

5. If **S_2** = 0, operation error will occur, the instruction will not be executed, M1067, M1068 = On and D1067 will recorded the error code H'0E19.

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X1 = On, binary floating point (D1, D0) ÷ binary floating point (D11, D10) and the quotient is stored in (D21, D20).

```
  X1
──┤├──────┬─────────┬──────┬──────┬──────┐
          │  DEDIV  │  D0  │ D10  │ D20  │
          └─────────┴──────┴──────┴──────┘
```

Program Example 2:

When X2 = On, binary floating point (D1, D0) ÷ K1234 (automatically converted into binary floating point) and the result is stored in (D11, D10).

```
  X2
──┤↓├─────┬─────────┬──────┬───────┬──────┐
          │  DEDIV  │  D0  │ K1234 │ D10  │
          └─────────┴──────┴───────┴──────┘
```
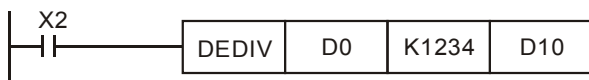
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands		Function	Controllers
124	D	EXP	P	**S** **D**		Exponent of Binary Floating Point	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices						Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DEXP, DEXPP: 9 steps
S					*	*							*			
D													*			

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Device for operation source   **D**: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. e = 2.71828 as the base and **S** as exponent for EXP operation: $EXP^{(D+1, D)} = [S+1, S]$

4. Both positive and negative values are valid for **S**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S** should be converted into a floating point value.

5. The content in **D** = e^S; e = 2.71828, ^{S} = designated source data

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) into binary floating point and store it in register (D11, D10).

2. When M1= On, use (D11, D10) as the exponent for EXP operation and store the binary floating point result in register (D21, D20).

3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × 10$^{[D31]}$) and store it in register (D31, D30).

```
M0
├┤├──┬──── RST    M1081

     └──── DFLT   D0     D10

M1
├┤├─────── DEXP   D10    D20

M2
├┤├─────── DEBCD  D20    D30
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands		Function	Controllers
125	D	LN	P	Ⓢ	Ⓓ	Natural Logarithm of Binary Floating Point	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Device for operation source   **D**: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. This instruction performs natural logarithm "LN" operation by **S**: LN〔**S** + 1, **S**〕=〔**D** + 1, **D**〕

4. Only positive values are valid for **S**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S** should be converted into a floating point value.

5. e^D = **S** .The content in **D** = ln**S**; **S** = designated source data.

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) into binary floating point and store it in register (D11, D10).

2. When M1= On, use register (D11, D10) as the real number for LN operation and store the binary floating point result in register (D21, D20).

3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × 10[D31]) and store it in register (D31, D30).

```
M0
─┤├─┬──────[ RST  | M1081 ]

     └──────[ DFLT | D0  | D10 ]

M1
─┤├────────[ DLN  | D10 | D20 ]

M2
─┤├────────[ DEBCD| D20 | D30 ]
```
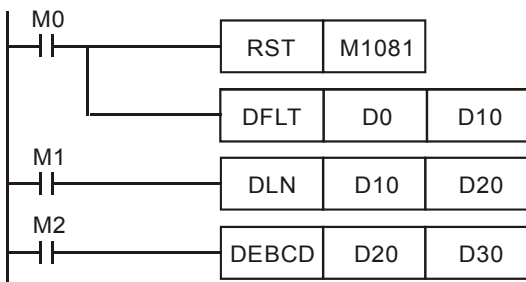
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands			Function		Controllers
126	D	LOG	P	S_1 S_2 D			Logarithm of Binary Floating Point		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		DLOG, DLOGP: 13 steps
S_1					*	*							*				
S_2					*	*							*				
D													*				

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Device for base S_2: Device for operation source   D: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. This instruction performs "log" operation of the content in S_1 and S_2 and stores the result in **D**.

4. Only positives are valid for the content in S_1 and S_2. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, S_1 and S_2 should be converted into floating point values.
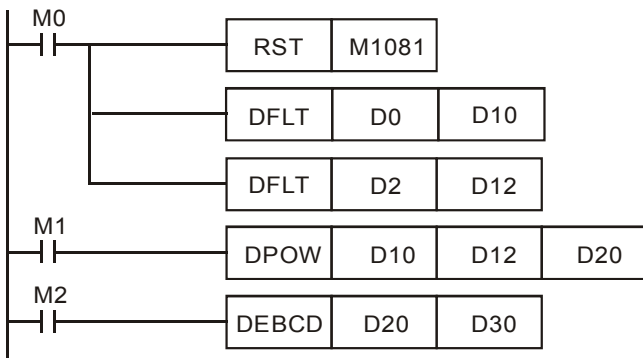
5. $S_1^D = S_2$, **D** = ? → $Log_{S_1}^{S_2}$ = **D**

   Example: Assume S_1 = 5, S_2 = 125, **D** = log_5^{125} = ?

   $S_1^D = S_2$ → 5^D = 125 → **D** = log_5^{125} = 3

6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

8. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) and (D3, D2) into binary floating points and store them in the 32-bit registers (D11, D10) and (D13, D12).

2. When M1= On, perform log operation on the binary flotiing points in 32-bit registers (D11, D10) and (D13, D12) and store the result in the 32-bit register (D21, D20).

3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × 10$^{[D31]}$) and store it in register (D31, D30).

```
 M0
──┤├──────┬───┤ RST  │ M1081 │
          │
          ├───┤ DFLT │  D0   │ D10  │
          │
          └───┤ DFLT │  D2   │ D12  │
 M1
──┤├──────────┤ DLOG │ D10   │ D12  │ D20 │
 M2
──┤├──────────┤ DEBCD│ D20   │ D30  │
```
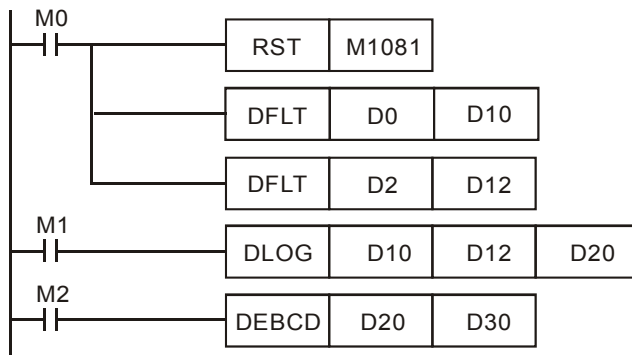
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic		Operands		Function	Controllers
127	D ESQR P		Ⓢ Ⓓ		Floating Point Square Root	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DESQR, DESQRP: 9 steps
S					*	*							*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device   **D**: Operation result

Explanations:

1. Range of **S**: ≥ 0

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag); M1067 (operation error)

4. This instruction performs a square root operation on the content in the register designated by **S** and stores the result in the register designated by **D**. The square root operation is performed in floating point system.

5. If **S** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.

6. If the result of the operation = 0, the zero flag M1020 = On.

7. **S** can only be a positive value. Performing any square root operation on a negative value will result in an "operation error" and this instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code H'0E1B.

Program Example 1:

When M0 = On, calculate the square root of the binary floating point (D1, D0) and store the result in register (D11, D10).

```
    X0
 ─┤ ├──────┌───────┬─────┬─────┐
            │ DESQR │ D0  │ D10 │
            └───────┴─────┴─────┘
```

$\sqrt{(D1, D0)}$ → (D11, D10)
binary floating point   binary floating point
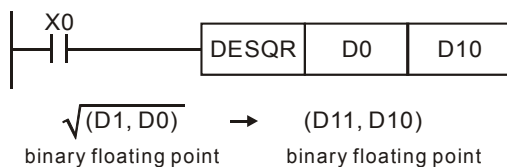
Program Example 2:

When M2 = On, calculate the square root of K1,234 (automatically converted into binary floating point) and store the result in register (D11, D10).

```
    X2
 ─┤ ├──────┌───────┬───────┬─────┐
            │ DESQR │ K1234 │ D10 │
            └───────┴───────┴─────┘
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands			Function	Controllers
128	D	POW	P	S_1 S_2 D			Floating Point Power Operation	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		DPOW, DPOWP: 13 steps
S_1					*	*							*				
S_2					*	*							*				
D													*				

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Device for base.　　**S_2**: Device for exponent.　　**D**: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.

2. This instruction performs power multiplication of binary floating point **S_1** and **S_2** and stores the result in **D**.

   D = POW〔**S_1** + 1, **S_1**〕^〔**S_2** + 1, **S_2**〕

3. Only positives are valid for the content in **S_1**. Both positives and negatives are valid for the content in **S_2**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S_1** and **S_2** should be converted into floating point values.

   Example: When S_1^{S2} = **D**, **D** = ?

   　　　Assume **S_1** = 5, **S_2** = 3, **D** = 5^3 =125

4. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

5. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

6. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) and (D3, D2) into binary floating points and store them in the 32-bit registers (D11, D10) and (D13, D12).

2. When M1= On, perform POW operation on the binary floting points in 32-bit registers (D11, D10) and (D13, D12) and store the result in the 32-bit register (D21, D20).

3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × $10^{[D31]}$) and store it in register (D31, D30).

```
 M0
─┤├─┬── [ RST   | M1081 ]
     │
     ├── [ DFLT  | D0  | D10 ]
     │
     └── [ DFLT  | D2  | D12 ]
 M1
─┤├──── [ DPOW  | D10 | D12 | D20 ]
 M2
─┤├──── [ DEBCD | D20 | D30 ]
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands	Function		Controllers
129	D	INT	P	Ⓢ Ⓓ	Float to Integer		ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device   **D**: Converted result

Explanations:

1. **S** occupies 2 consecutive devices. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. The binary floating point value of the register designated by **S** is converted to BIN integer and stored in the register designated by **D**. The decimal of BIN integer is left out.

4. This instruction is the inverse operation of API 49 FLT instruction.

5. If the converstion result = 0, the zero flag M1020 = On
   If there is any decimal left out, the borrow flag M1021 = On.
   If the result exceeds the range listed below, the 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 (D1, D0) will be converted into BIN integer and the result will be stored in (D10). The decimal of BIN integer will be left out.

2. When X1 = On, the binary floating point (D21, D20) will be converted into BIN integer and the result will be stored in (D31, D30). The decimal of BIN integer will be left out.
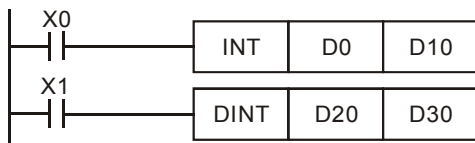
```
 X0
─┤ ├──────┤ INT  │ D0  │ D10 │

 X1
─┤ ├──────┤ DINT │ D20 │ D30 │
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands		Function		Controllers	
130	D	SIN	P	**S**	**D**	Sine		ES/EX/SS	SA/SX/SC	EH/SV

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							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value   **D**: SIN result

Explanations:

1. 0° ≤ **S** < 360°. See the specifications of each model for their range of use.

2. Flags: M1018 (angle or radian); M1020 (zero flag)

3. **S** can be an angle or radian, decided by M1018.

4. When M1018 = Off, the program will be in radian mode and the RAD value = angle × π /180

5. When M1018 = On, the program will be in angle mode and the range of angle should be "0° ≤ angle < 360°"

6. If the result = On, M1020 = On.

7. The SIN value obtained by **S** is calculated and stored in the register designated by **D**. The figure below offers the relation between radian and the result.
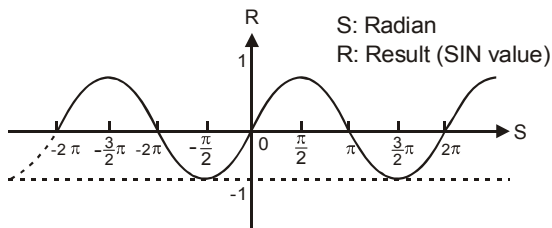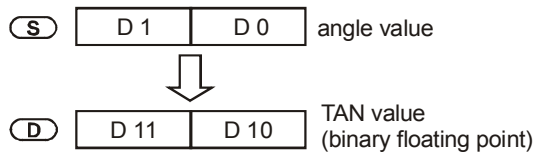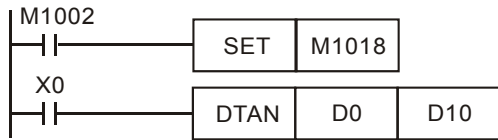

Program Example 1:

When M1018 = Off, the program is in radian mode. When X0 = On, use the RAD value of binary floating point (D1, D0) and obtain its SIN value. The binary floating point result will be stored in (D11, D10).

Program Example 2:

When M1018 = Off, the program is in radian mode. Input terminals X0 and X1 select the angle. The angles are converted into RAD value for calculating the SIN value.

```
      X0
      ┤├        ┌──────┬────────┬────────┐
                │ MOVP │  K30   │  D10   │    (K30 ─→ D10) ┐
                └──────┴────────┴────────┘                 │
      X1                                                   │
      ┤├        ┌──────┬────────┬────────┐                 │
                │ MOVP │  K60   │  D10   │    (K60 ─→ D10) ┘
                └──────┴────────┴────────┘
    M1000
      ┤├───┬──  ┌──────┬────────┬────────┐
           │    │ FLT  │  D10   │  D14   │    (D10 ─→ D15, D14)  binary floating point
           │    └──────┴────────┴────────┘
           │    ┌───────┬──────────┬─────────────┬──────┐
           ├──  │ DEDIV │K31415926 │ K1800000000 │ D20  │   (π/180) ─→ (D21, D20)
           │    └───────┴──────────┴─────────────┴──────┘     binary        binary
           │                                                 floating point floating point
           │    ┌───────┬────────┬────────┬────────┐
           ├──  │ DEMUL │  D14   │  D20   │  D40   │   (D15, D14) angle  x π/180 ─→
           │    └───────┴────────┴────────┴────────┘   (D41, D40) RAD binary floating point
           │    ┌──────┬────────┬────────┐
           └──  │ DSIN │  D40   │  D50   │    (D41, D40) RAD ─→  (D51, D50) SIN
                └──────┴────────┴────────┘                      binary floating point
```
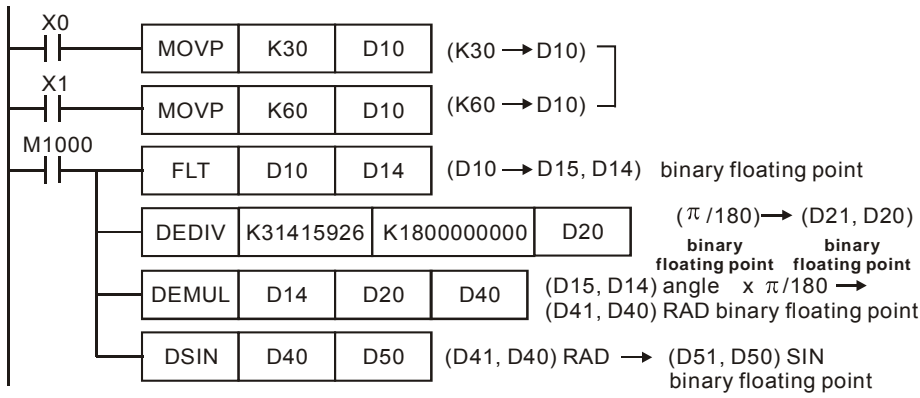
Program Example 3:

When M1018 = On, the program is in angle mode. When X0 = On, use the angle of (D1, D0) to obtain SIN value and store the binary floating point result in (D11, D10). (0° ≤ angle < 360°)
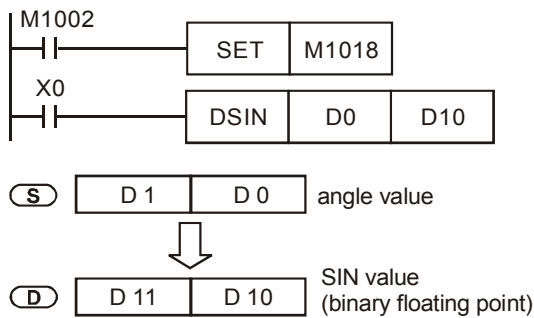
```
    M1002
      ┤├        ┌──────┬────────┐
                │ SET  │ M1018  │
                └──────┴────────┘
      X0
      ┤├        ┌──────┬────────┬────────┐
                │ DSIN │   D0   │  D10   │
                └──────┴────────┴────────┘
```

```
  (S)  │  D 1  │  D 0  │   angle value

              ⇩

  (D)  │ D 11  │ D 10  │   SIN value
                           (binary floating point)
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands		Function	Controllers
131	D	COS	P	Ⓢ Ⓓ		Cosine	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F			DCOS, DCOSP: 9 steps
S					*	*							*					
D													*					

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source value   **D**: COS result

Explanations:

1. 0° ≤ **S** < 360°. See the specifications of each model for their range of use.

2. Flags: M1018 (angle or radian); M1020 (zero flag)

3. **S** can be an angle or radian, decided by M1018.

4. When M1018 = Off, the program will be in radian mode and the RAD value = angle × π /180

5. When M1018 = On, the program will be in angle mode and the range of angle should be "0° ≤ angle < 360°"

6. If the result = On, M1020 = On.

7. The COS value obtained by **S** is calculated and stored in the register designated by **D**. The figure below offers the relation between radian and the result.
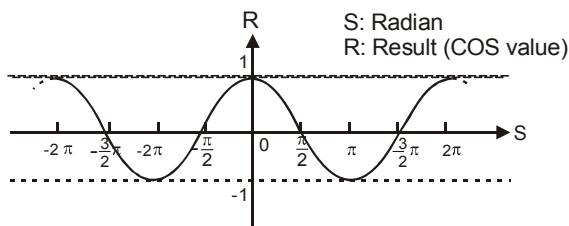
8. Switch between radian and angle by M1018: When M1018 = Off, **S** will be a RAD value; when M1018 = On, **S** will be an angle (0° ~ 360°).

Program Example 1:

When M1018 = Off, the program is in radian mode. When X0 = On, use the RAD value of binary floating point (D1, D0) and obtain its COS value. The binary floating point result will be stored in (D11, D10).
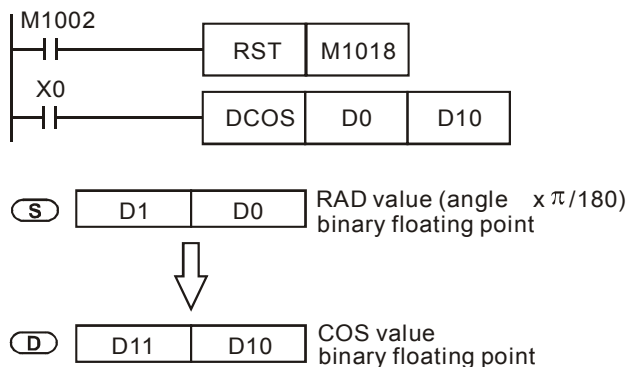
Program Example 2:

When M1018 = On, the program is in angle mode. When X0 = On, use the angle of (D1, D0) to obtain COS value and

store the binary floating point result in (D11, D10). (0° ≤ angle < 360°)
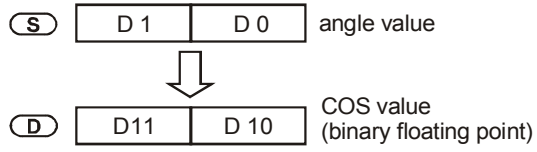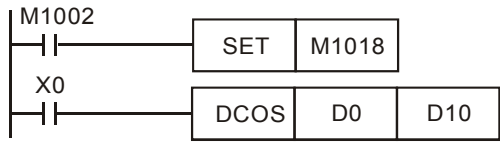
```
  M1002
───┤ ├──────────────┤ SET  │ M1018 │
   X0
───┤ ├──────────────┤ DCOS │  D0  │ D10 │
```

```
 (S)  │  D 1  │  D 0  │  angle value
                ⇩
 (D)  │  D11  │  D 10 │  COS value
                         (binary floating point)
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic		Operands		Function		Controllers
132	D	TAN	P	S D	Tangent		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices						Word Devices										Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DTAN, DTANP: 9 steps
S					*	*							*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source value   **D**: TAN result

Explanations:

1. $0° \leq$ **S** $< 360°$. See the specifications of each model for their range of use.

2. Flags: M1018 (angle or radian); M1020 (zero flag)

3. **S** can be an angle or radian, decided by M1018.

4. When M1018 = Off, the program will be in radian mode and the RAD value = angle × π /180

5. When M1018 = On, the program will be in angle mode and the range of angle should be "$0° \leq$ angle $< 360°$"

6. If the result = On, M1020 = On.

7. The TAN value obtained by **S** is calculated and stored in the register designated by **D**. The figure below offers the relation between radian and the result.
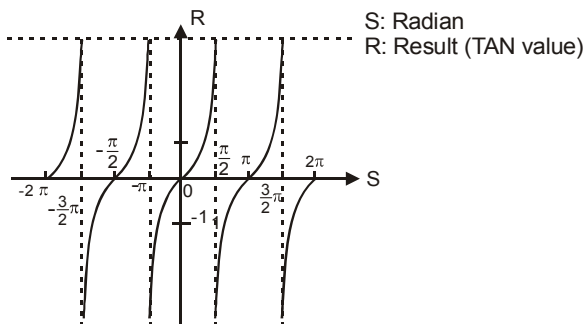
Program Example 1:

When M1018 = Off, the program is in radian mode. When X0 = On, use the RAD value of binary floating point (D1, D0) and obtain its TAN value. The binary floating point result will be stored in (D11, D10).
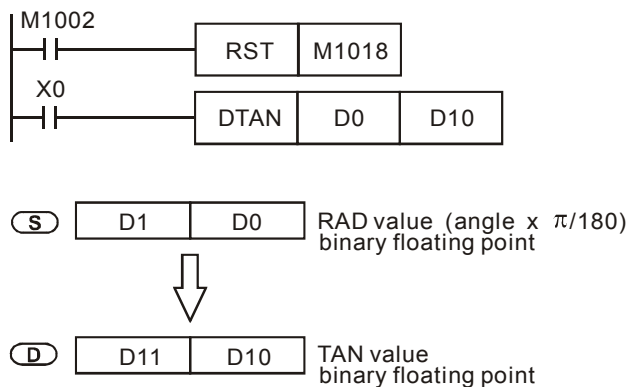
Program Example 2:

When M1018 = On, the program is in angle mode. When X0 = On, use the angle of (D1, D0) to obtain TAN value and

store the binary floating point result in (D11, D10). (0° ≤ angle < 360°)

```
      M1002
      ─┤├──────────────┤ SET │ M1018 │
       X0
      ─┤├──────────────┤ DTAN │ D0 │ D10 │
```

```
 ⓢ │ D 1  │ D 0  │  angle value

              ⬇

 ⓓ │ D 11 │ D 10 │  TAN value
                     (binary floating point)
```

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands		Function		Controllers
133	D	ASIN	P	S D		Arc Sine		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices						Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F			DASIN, DASINP: 9 steps
S					*	*							*					
D													*					

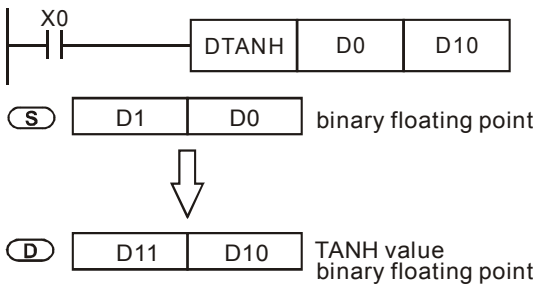| PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source value (binary floating point)   **D**: ASIN result

Explanations:

1. See the specifications of each model for their range of use.

2. Flag: M1020 (zero flag)

3. ASIN value=sin^{-1}. The figure below offers the relation between the entered sin value and the result.

S: Entered data (sine value)
R: ASIN value (radian)
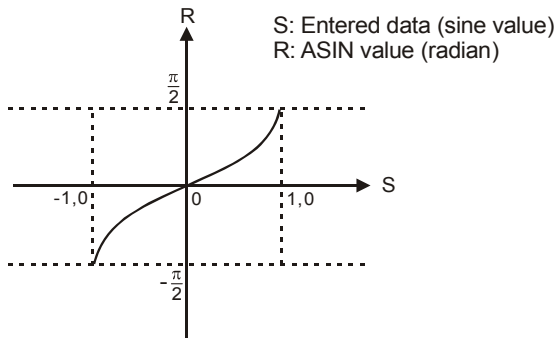
4. The decimal floating point of the SIN value designated by **S** should be within -1.0 ~ +1.0. If the value falls without the range, M1067 and M1068 will be On without performing any action.

5. If the result = 0, M1020 = On.

Program Example:

When X0 = On, obtain the ASIN value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).
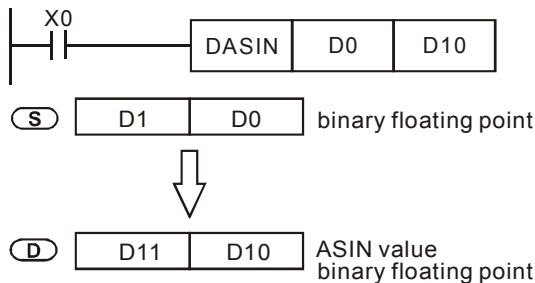

```
    X0
    ┤├────────[ DASIN │ D0 │ D10 ]
```

| S | D1 | D0 | binary floating point |

| D | D11 | D10 | ASIN value binary floating point |

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands		Function			Controllers
134	D	ACOS	P	**S** **D**		Arc Cosine			ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DACOS, DACOSP: 9 steps
S					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value (binary floating point)   **D**: ACOS result

Explanations:

1. See the specifications of each model for their range of use.

2. Flag: M1020 (zero flag)

3. ACOS value=$\cos^{-1}$. The figure below offers the relation between the entered cos value and the result.

R
S: Entered data (cosine value)
R: ACOS value (radian)
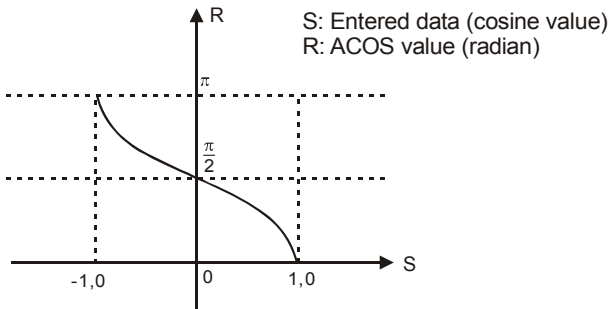
4. The decimal floating point of the COS value designated by **S** should be within -1.0 ~ +1.0. If the value falls without the range, M1067 and M1068 will be On without performing any action.

5. If the result = 0, M1020 = On.

Program Example:

When X0 = On, obtain the ACOS value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).
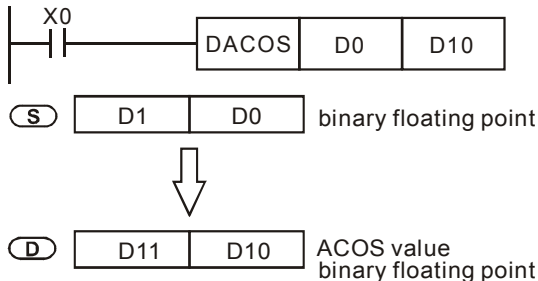
| X0 | DACOS | D0 | D10 |

S | D1 | D0 | binary floating point

D | D11 | D10 | ACOS value
binary floating point

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands		Function	Controllers
135	D	ATAN	P	Ⓢ Ⓓ		Arc Tangent	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source value (binary floating point)　　**D**: ATAN value

Explanations:

1. See the specifications of each model for their range of use.

2. Flag: M1020 (zero flag)

3. ATAN value=$\tan^{-1}$. The figure below offers the relation between the entered tan value and the result.

S: Entered data (tangent value)
R: ATAN value (radian)
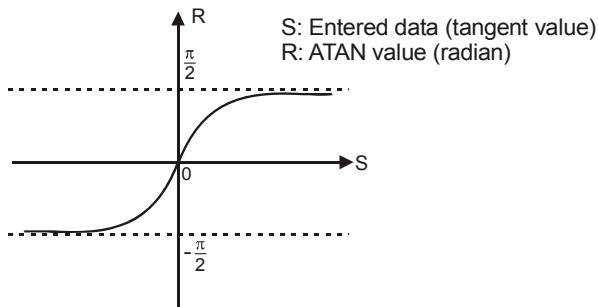
4. If the result =0, M1020 = On.

Program Example:

When X0 = On, obtain the ATAN value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).
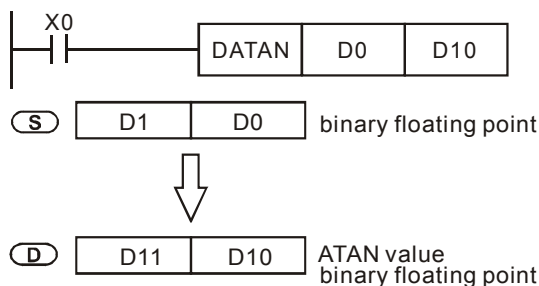
Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function		Controllers
136	D	SINH	P	Ⓢ Ⓓ	Hyperbolic Sine		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSINH, DSINHP: 9 steps
S					*	*							*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source value (binary floating point)   **D**: SINH value

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. SINH value=$(e^s - e^{-s})/2$. The result is stored in **D**.

Program Example:

1. When X0 = On, obtain the SINH value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).

```
     X0
   ──┤├──────┌────────┬──────┬──────┐
             │ DSINH  │  D0  │ D10  │
             └────────┴──────┴──────┘
```

Ⓢ | D1 | D0 | binary floating point

⬇

Ⓓ | D11 | D10 | SINH value
binary floating point

2. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

3. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

4. If the result = 0, the zero flag M1020 = On.

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API		Mnemonic		Operands		Function	Controllers
137	D	COSH	P	Ⓢ Ⓓ		Hyperbolic Cosine	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCOSH, DCOSHP: 9 steps
S					*	*							*			
D													*			

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value (binary floating point)   **D**: COSH value

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. COSH value=$(e^s+e^{-s})/2$. The result is stored in **D**.

Program Example:

1. When X0 = On, obtain the COSH value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).

5. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

6. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

7. If the result = 0, the zero flag M1020 = On.

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic		Operands	Function		Controllers
138	D TANH	P	(S) (D)	Hyperbolic Tangent		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DTANH, DTANHP: 9 steps
S					*	*							*			
D													*			

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value (binary floating point)   **D**: TANH result

Explanations:

1. See the specifications of each model for their range of use.

2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)

3. TANH value=$(e^s - e^{-s})/(e^s + e^{-s})$. The result is stored in **D**.

Program Example:

1. When X0 = On, obtain the TANH value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).

    ```
      X0
    ├──┤ ├──────┤ DTANH │  D0  │  D10 │
    ```

    (S) | D1 | D0 | binary floating point

    ⇩

    (D) | D11 | D10 | TANH value
                       binary floating point

2. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.

3. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.

4. If the result = 0, the zero flag M1020 = On.

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic	Operands	Function	Controllers
143	DELAY	Ⓢ	Delay Instruction	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: delay time (unit: 0.1ms)

Explanations:

1. Range of **S**: K1 ~ K1,000. See the specifications of each model for their range of use.
2. After DELAY instruction is executed, the program after DELAY in every scan period will execute delay outputs according to the delay time designated by the user.

Program Example:

When X0 goes from Off to On and generates an external interruption, the interruption subroutine will execute DELAY for 2ms before executing the next step, X1 = On and Y0 = On.
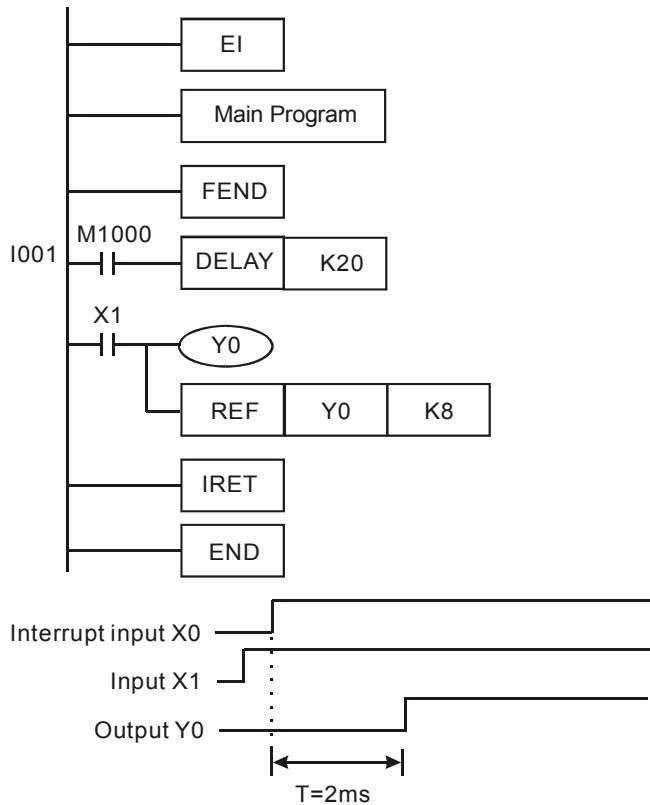
Remarks:

1. User can define the delay time based on their needs.
2. The delay time may increase due do the influences from communication, high-speed counters and high-speed pulse output instructions.
3. The delay time of designated external output (transistor or relay) will increase due to the delay on the transistor

or relay itself. See 2.3 for more information.

API	Mnemonic	Operands	Function	Controllers
144	GPWM	**S₁** **S₂** **D**	General PWM Output	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices													Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F			GPWM: 7 steps
S₁													*					
S₂													*					
D		*	*	*														

PULSE									16-bit									32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Width of output pulse   **S₂**: Pulse output cycle   **D**: Pulse output device

Explanations:

1. **S₂** occupies 3 consecutive devices.

2. **S₁** ≤ **S₂**. See the specifications of each model for their range of use.

3. Range of **S₁**: t = 0 ~ 32,767ms.

4. Range of **S₂**: t = 1 ~ 32,767ms.

5. **S₂** +1 and **S₂** +2 are parameters for the system. Do not occupy them.

6. Pulse output devices **D**: Y, M, S.

7. When being executed, GPWM instruction designates **S₁** and **S₂** and that pulses output will be from device **D**.

8. When **S₁** ≤ 0, there will be no pulse output. When **S₁** ≥ **S₂**, the pulse output device will keep being On.

9. **S₁** and **S₂** can be modified when GPWM instruction is being executed.

Program Example:

When X0 = On, D0 = K1,000, D2 = K2,000, and Y10 will output the pulse illustrated below. When X0 = Off, Y10 output will be Off.
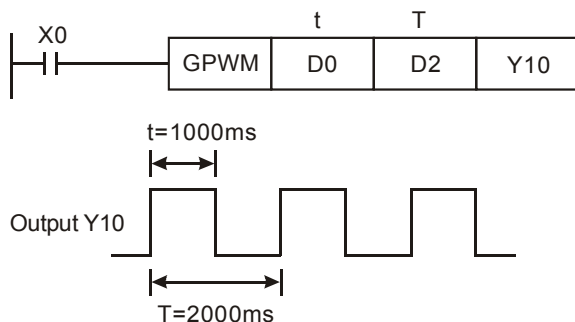
Explanations:

1. This instruction counts by the scan cycle; therefore the maximum offset will be one PLC scan cycle. **S₁**, **S₂** and (**S₂** - **S₁**) should > PLC scan cycle; otherwise, errors will occur during GPWM outputs.

2. Please note that placing this instruction in a subroutine or interruption will cause inaccurate GPWM outputs.

API	Mnemonic	Operands				Function	Controllers
145	FTC	**S₁**	**S₂**	**S₃**	**D**	Fuzzy Temperature Control	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	FTC: 9 steps
S₁					*	*							*			
S₂					*	*							*			
S₃													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Set value (SV)   **S₂**: Present value (PV)   **S₃**: 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. Range of S_2 : 1 ~ 5000 (shown as 0.1°C ~ 500°C). Unit: 0.1°. If (S_3 +1) is set as bit0 = 0, the range will be 0.1°C ~ 500°C. Therefore, when the user obtain an A/D value from the temperature sensor, the value has to be converted into a value between 1 ~ 5,000 by four arithmetic operation instructions.

3. If S_3 < K1, the instruction will not be executed. If S_3 > K200, S3 will adopt K200. S_3 will occupy 7 consecutive devices.

4. See the specifications of each model for their range of use.

5. 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.

6. **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.

7. 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..

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.

```
 X0
─┤ ├─────[ FTC │ D0 │ D1 │ D100 │ D150 ]
```
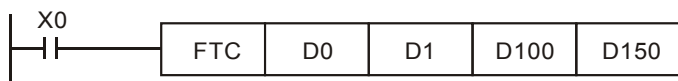
Remarks:

1. Setting of **S₃**:

Device No.	Function	Range	Explanation
S₃	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.
S₃ +1	b0: temperature unit b1: filter function b2 ~ b5: heating environnment b6 ~ b15: reserved	b0 =0 means ^{o}C b0 =1 means ^{o}F	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
S₃ +2 ∤ **S₃** +6	Parameters for system use only. Do not use them.		

2. Control Diagram:
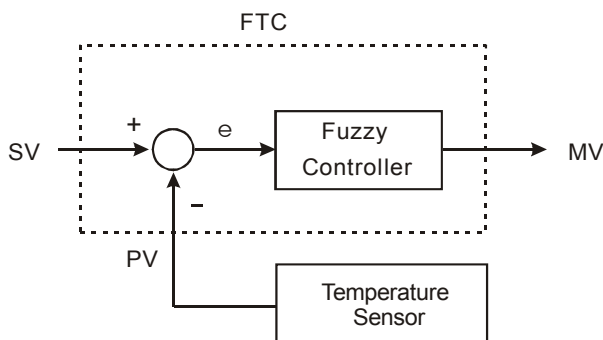
3. Notes and suggestion:

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 **S₃**+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.

4. **Example 1:** control diagram

FTC

SV
D10

+

e

Fuzzy
Controller

MV
D22

GPWM Program

Y0

−

PV D11

PT Module

Temperature
Sensor
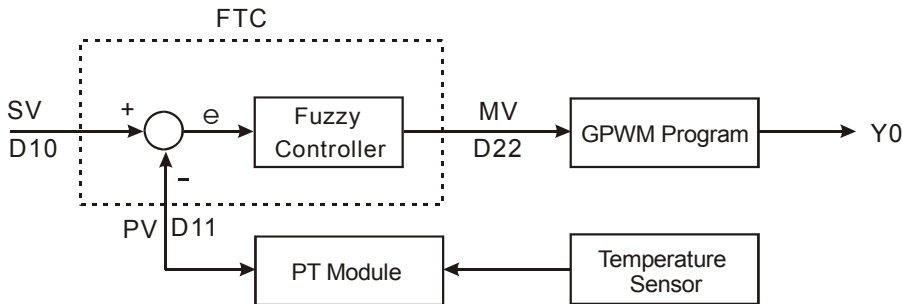
Output D22 (MV) of FTC instruction is the input D22 of GPWM instruction, as the duty cycle of ajustable pulses. D30 is the fixed cycle time of pulses. See below for the timing diagram of Y0 output.

D22

Y0

D30
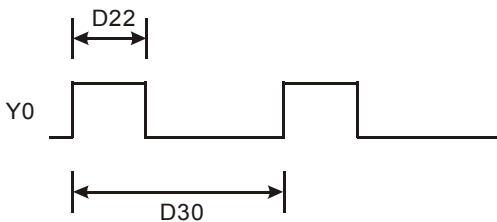
Assume parameter settings: D10 = K1,500 (target temperature), D12 = K60 (sampling time: 6 secs.), D13 = K8 (bit3=1), D30 = K6,000 (=D12*100)

The example control program is indicated as:

M1002

MOV	K1500	D10	
TO	K0	K2	K2	K1
FROM	K0	K6	D11	K1
MOV	K60	D12	
MOV	K8	D13	
MOV	K6000	D30	
SET	M1		

M1

| FTC | D10 | D11 | D12 | D22 |
| GPWM | D22 | D30 | Y0 | |

M1013

| FROM | K0 | K6 | D11 | K1 |
| END | | | | |
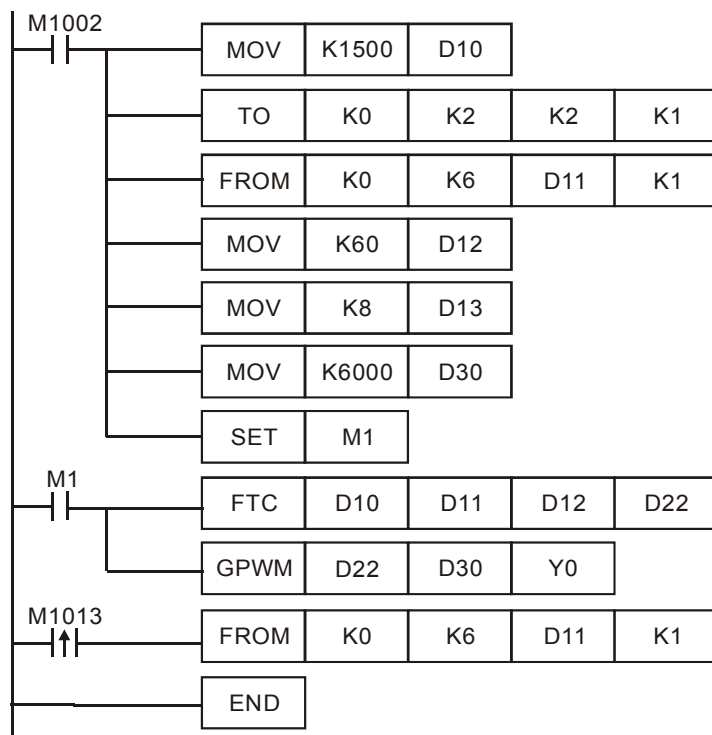
Experiment in an oven which can be heated up to 250°C. See below for the records of target and present temperatures. As shown in the diagram below, we can see that after 48 minutes, the temperature is able to reach the target temperature with ±1°C inaccuracy and exceed approx. 10°C of the target temperature.
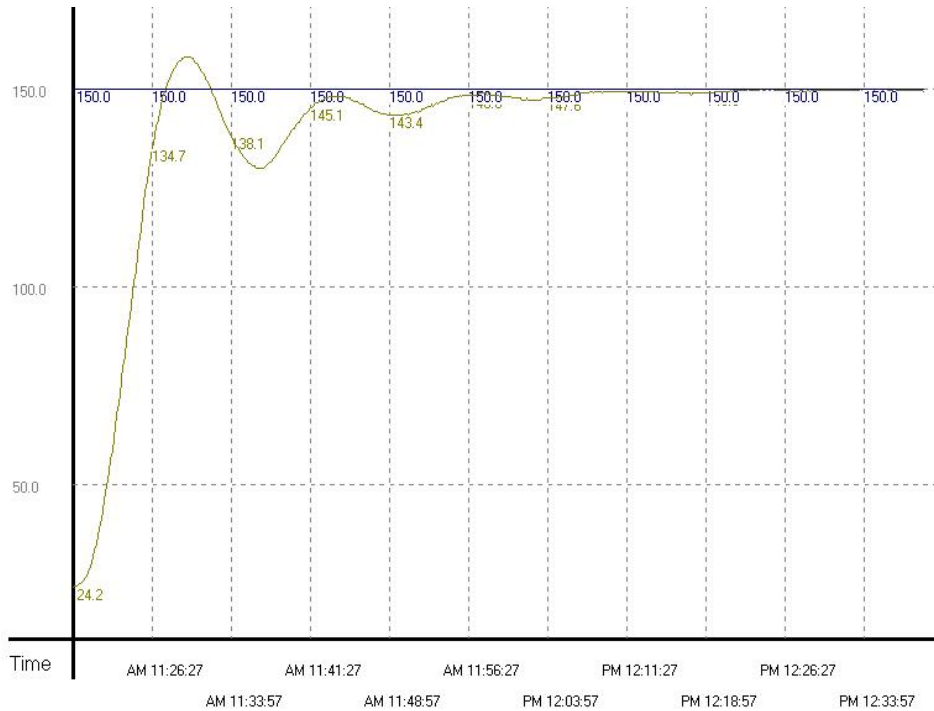
Example 2: Due to that the temperature once exceeds the target temperature, we modify the heating environment into "fast heating environment" (D13 = K16). The results are shown in the diagram below.

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 ±1°C inaccuracy. It seems that we have chosen the right environment, but the sampling time is too long, resulting in the extension of heating time.
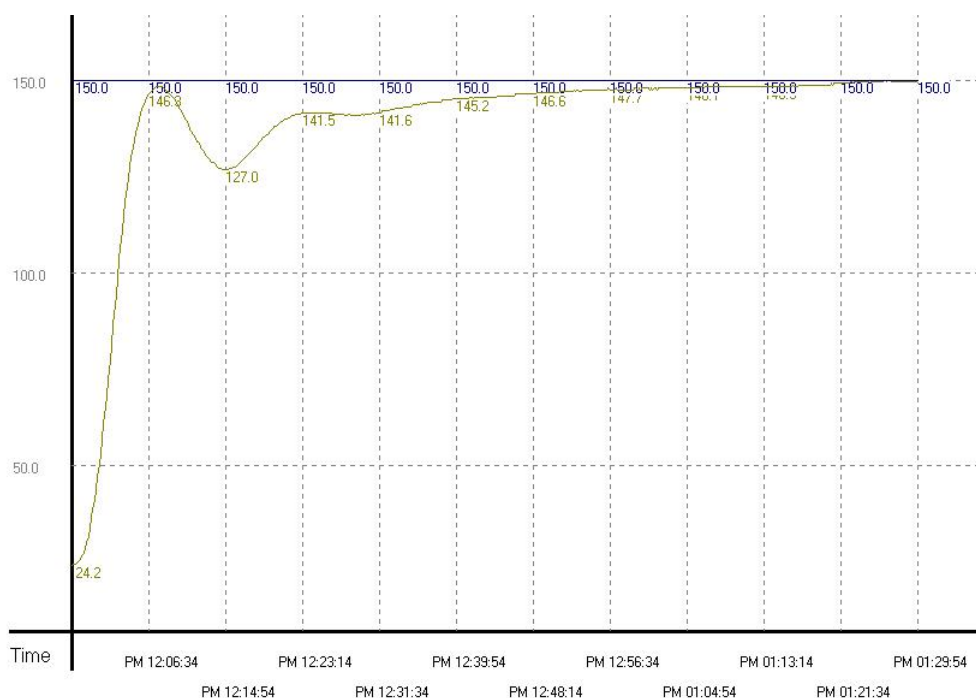
Example 3: To speed up the speed to reach the target temperature, we correct the sampling time as 4 seconds (D12 = K40, D30 = K4,000). The results are shown in the diagram below.

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.
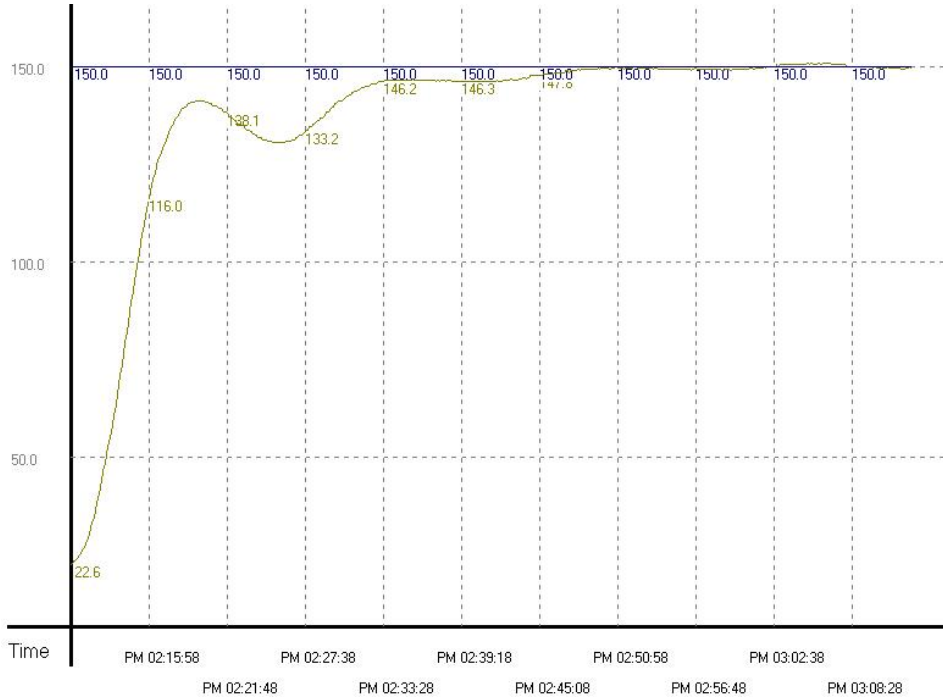
Example 4: To see if we can reach the target temperature faster, we modify the sampling time frim example 3 into 2 seconds (D12 = K20, D30 = K2,000). The results are shown in the diagram below.

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.
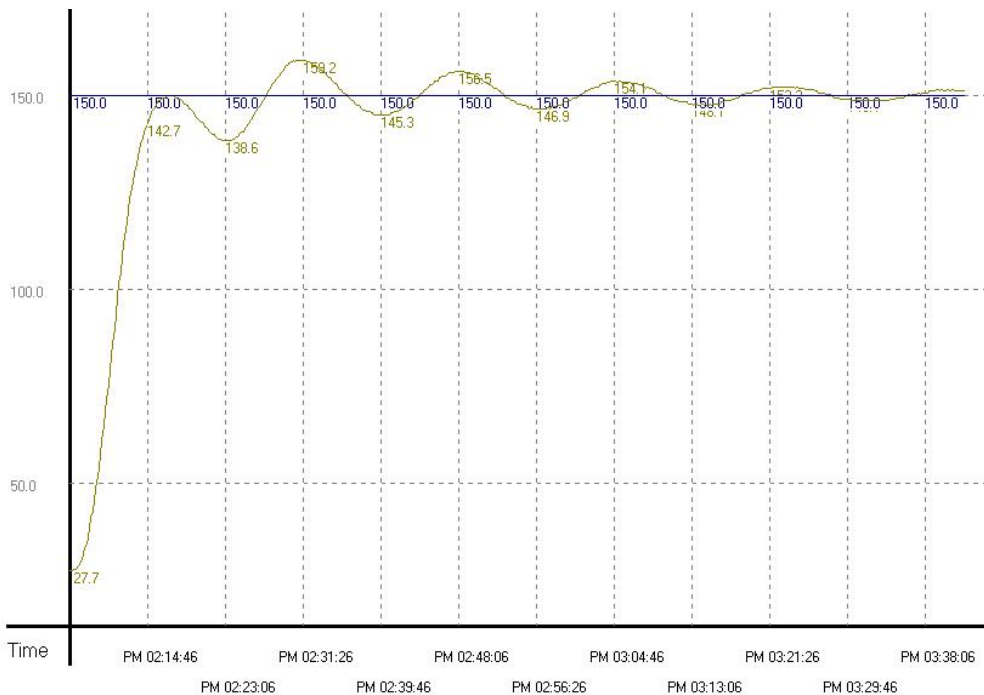
API	Mnemonic	Operands	Function	Controllers
146	CVM	S_1 S_2 D	Valve Control	ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Target time of valve (absolute position)   **S₂**: Time from fully-closed to fully-open of valve (destination)   **D**: Output device

Explanations:

1. **S₁** occupies 3 consecutive registers when in use. **S₁** + 0 are for the user to store the designated value; **S₁** + 1 (the current position of the valve) and **S₁** + 2 are for storing the parameters recorded in the instruction and please DO NOT use and alter these two registers.

2. **D** occupies 2 consecutive output devices when in use. **D** + 0 is the "open" contact and **D** + 1 is the "close" contact.

3. This instruction only supports EH2/SV and does not support EH.

4. The unit of time: 0.1 second. When the scan time of the program exceeds 0.1 second, DO NOT use this instruction to adjust the position of the valve.

5. Frequency of the output device: 10Hz.

6. When the time of **S₁** + 0 $>$ the fully-opened time set in **S₂**, **D** + 0 will keep being On and **D** + 1 being Off. When the time of **S₁** + 0 $<$ 0, **D** + 0 will keep being Off and **D** + 1 being On.

7. When the instruction is enabled, the instruction will start to control the valve from "0" time position. Therefore, if the user cannot be sure whether the valve is at "0" before executing the instruction, please designate **S₁** + 0 as less than 0 and execute the instruction for **S₂** (time) before sending in the correct target control time.

Program Example 1:

1. The control valve
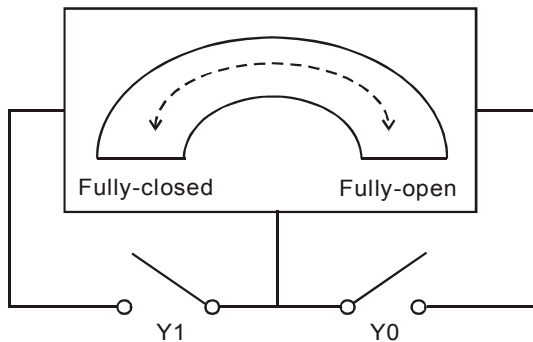
2. Definitions of the control valve:

   a) When Y0 and Y1 = Off: No valve action

b) When Y0 = On and Y1 = Off: Valve "open"

c) When Y0 = Off and Y1 = On: Valve "closed"

d) When Y0 and Y1 = On: The action is prohibited.

3. Timing diagram and program of the control:
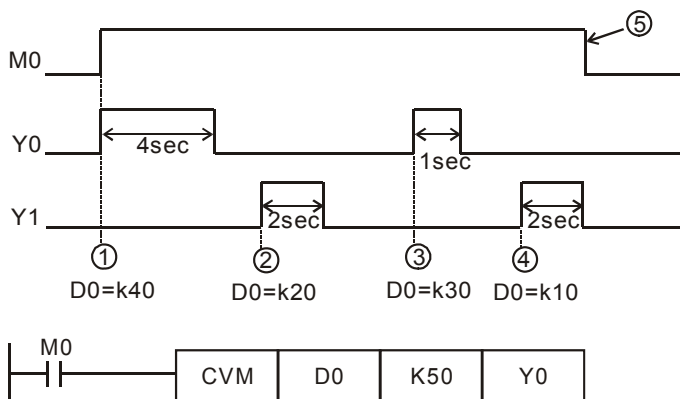
4. Control phases:

1) Phase ①: When M0 = On, D0 = K40 refers to the valve shall be open (Y0 = On, Y1 = Off) till the position of 4 seconds.

2) Phase ②: Change the position of the valve and D0 = K20. Due to that the previous position was at 4 seconds, the valve shall be closed (Y0 = Off, Y1 = On) for 2 seconds, moving the valve to the position of 2 seconds.

3) Phase ③: Change the position of the valve and D0 = K30. Due to that the previous position was at 2 seconds, the valve shall be open (Y0 = On, Y1 = Off) for 1 second, moving the valve to the position of 3 seconds.

4) Phase ④: Change the position of the valve and D0 = K10. Due to that the previous position was at 2 seconds, the valve shall be closed (Y0 = Off, Y1 = On) for 2 seconds, moving the valve to the position of 1 second.

5) Phase ⑤: Switch off X0 and no actions at the valve (Y0 = Off, Y1 = Off).

Program Example 2:

1. Timing diagram and program of the control:

M0		TMR	T0	K50	
	T0	MOV	k-1	D0	
		CVM	D0	K50	Y0
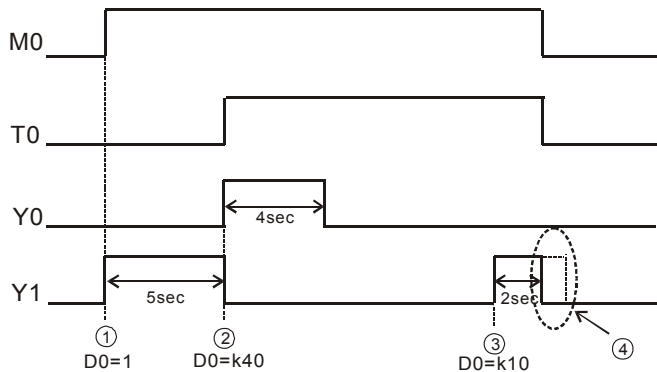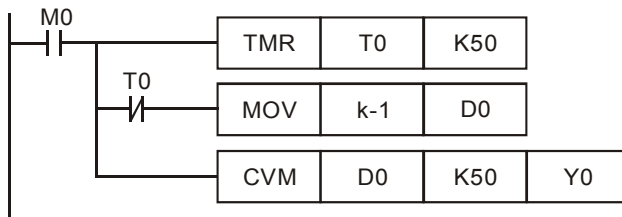
2. Control phases:

    1) Phase ①: When M0 = On, due to that we are not sure about there the valve is, set D0 = K-1 to deliberately close the valve (Y0 = Off, Y1 = On) for 5 seconds and make sure the valve is at the position of 0 second before moving on to the next step.

    2) Phase ②: When T0 = On, allow D0 = K40 to start is action. Open the valve (Y0 = On, Y1 = Off) for 4 seconds, moving the valve to the position of 4 seconds.

    3) Phase ③: Change the position of the valve and D0 = K10. Due to that the previous position was at 4 seconds, the valve shall be closed (Y0 = Off, Y1 = On) for 3 seconds, moving the valve to the position of 1 second.

    4) Phase ④: Switch off M0 and the valve will no longer move (Y0 = Off, Y1 = Off).

API		Mnemonic		Operands	Function		Controllers
147	D	SWAP	P	Ⓢ	Byte Swap		ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Device for swapping 8 high/low byte.

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. As 16-bit instruction: the contents in the 8 high bytes and 8 low bytes are swapped.

4. As 32-bit instruction: the 8 high bytes and 8 low bytes in the two registers swap with each other respectively.

5. This instruction adopts pulse execution instructions (SWAPP, DSWAPP).

Program Example 1:

When X0 = On, the high 8 bytes and low 8 bytes in D0 will swap with each other.
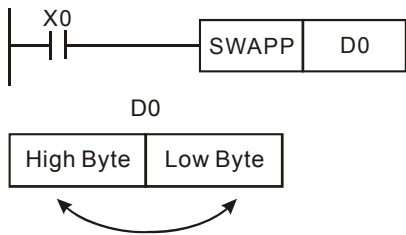
Program Example 2:

When X0 = On, the high 8 bytes and low 8 bytes in D11 will swap with each other and the high 8 bytes and low 8 bytes in D10 will swap with each other.
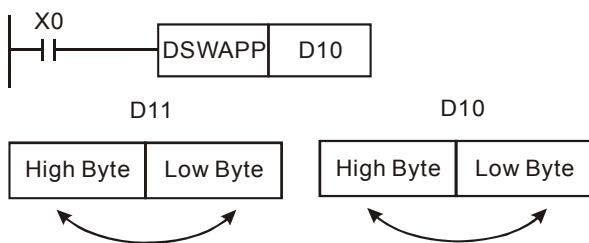
API		Mnemonic		Operands			Function		Controllers
148	D	MEMR	P	(m) (D) (n)			Read File Register		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MEMR, MEMRP: 7 steps
m					*	*							*			DMEMR, DMEMRP: 13 steps
D													*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

m: Address in the file register to be read   **D**: Device for storing the read data (starting from the designated D)

n: Number of data read at a time

Explanations:

1. Range of **m**: K0 ~ K1,599 (SA/SX/SC); K0 ~ K9,999 (EH/EH2/SV)

2. Range of **D**: D2000 ~ D4999 (SA/SX/SC); D2000 ~ D9999 (EH/EH2/SV)

3. Range of **n**: For 16-bit instruction K1 ~ K1,600 (SA/SX/SC), K1 ~ K8,000 (EH/EH2/SV); For 32-bit instruction K1 ~ K800 (SA/SX/SC), K1 ~ K4,000 (EH/EH2/SV)

4. See the specifications of each model for their range of use.

5. Flag: M1101. See explanations below.

6. SA/SX/SC/EH/EH2/SV uses this instruction to read the data in file registers and store them into data registers.

7. SA/SX/SC offers 1,600 16-bit file registers.

8. **m** and **n** of SA/SX/SC do not suppot E and F index register modification.

9. EH/EH2/SV offers 10,000 16-bit file registers.

10. If **m**, **D** and **n** fall without their range, operation error will occur. M1067, M1068 = On and D1067 will record the error code H'0E1A.

Program Example 1:

1. The 16-bit instruction MEMR reads 100 data at address 10 in the file register and store the read data in register D starting from D2000.

2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the previously read data will remain unchanged.

```
  X0
──┤├──┤ MEMR │ K10 │ D2000 │ K100 │
```
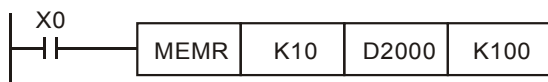
Program Example 2:

1. The 32-bit instruction DMEMR reads 100 data at address 20 in the file register and store the read data in register D starting from D3000.

2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the previously read data will remain unchanged.

```
  X0
──┤├──┤ DMEMR │ K20 │ D3000 │ K100 │
```
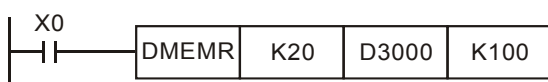
API	Mnemonic		Operands			Function		Controllers
149	D	MEMW	P	S	m	n	Write File Register	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MEMW, MEMWP: 7 steps
S													*			DMEMW, DMEMWP: 13 steps
m					*	*							*			
n					*	*							*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Device for storing the written data (starting from the designated D)   **m**: Address in the file register to be written

n: Number of data to be written at a time

Explanations:

1. Range of **S**: D2000 ~ D4999 (SA/SX/SC); D2000 ~ D9999 (EH/EH2/SV)

2. Range of **m**: K0 ~ K1,599 (SA/SX/SC); K0 ~ K9,999 (EH/EH2/SV)

3. Range of **n**: For 16-bit instruction K1 ~ K1,600 (SA/SX/SC), K1 ~ K8,000 (EH/EH2/SV); For 32-bit instruction K1 ~ K800 (SA/SX/SC), K1 ~ K4,000 (EH/EH2/SV)

4. See the specifications of each model for their range of use.

5. Flag: M1101. See explanations below.

6. SA/SX/SC/EH/EH2/SV uses this instruction to read the data in data registers and write them into file registers.

7. SA/SX/SC offers 1,600 16-bit file registers.

8. **m** and **n** of SA/SX/SC do not suppot E and F index register modification.

9. EH/EH2/SV offers 10,000 16-bit file registers.

10. If **S**, **m** and **n** fall without their range, operation error will occur. M1067, M1068 = On and D1067 will record the error code H'0E1A.

Program Example:

1. When X0 = On, the 32-bit instruction DMEMW writes 100 32-bit data starting from D2001 and D2000 into address 0 ~ 199 in the file register.

2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the previously data written in will remain unchanged.

```
  X0
──┤├───┬──────┬───────┬──────┬──────┐
       │ DMEMW│ D2000 │  K0  │ K100 │
       └──────┴───────┴──────┴──────┘
```
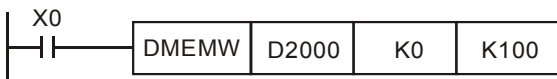
File Register:

1. EH/EH2/SV: When the PLC is powered, it will decide whether to automatically send the data in the file register to the designated data register by M1101 (whether to enable the function of file register), D1101 (start address in file register K0 ~ K9,999), D1102 (number of data to be read in file register K1 ~ k8,000), and D1103 (device for storing read data, starting from designated D, K2,000 ~ K9,999).

2. In EH/EH2/SV, the reading of data from file register to data register D will not be executed if D1101 < 0, D1101

> K9,999, D1103 < K2,000 or D1103 > K9,999.

3. SA/SX/SC: When the PLC is powered, it will decide whether to automatically send the data in the file register to the designated data register by M1101 (whether to enable the function of file register), D1101 (start address in file register K0 ~ K1,599), D1102 (number of data to be read in file register K1 ~ k1,600), and D1103 (device for storing read data, starting from designated D, K2,000 ~ K4,999).

4. In SA/SX/SC, the reading of data from file register to data register D will not be executed if D1101 < 0, D1101 > K1,599, D1103 < K2,000 or D1103 > K4,999.

5. When the reading of data from file register to data register D starts, PLC will stop the reading if the address of file register or data register exceed their range.

6. In PLC program, only API 148 MEMR and API 149 MEMW can be used to read or write the file register. See 2.8.3 for more information on file registers.

7. File registers do not have actual addresses in it. Reading and writing of file registers can only be done through API 148 MEMR, API 149 MEMW or peripheral devices HPP and WPLSoft.

8. If the address in the file register to be read exceeds its range, the read value will be 0.

9. Special relays of file register and other relevant special registers:

Flag	Function
M1101	Whether to enable the function of file register; latched; default = off

Special D	Function
D1101	Start address in file register. SA/SX/SC: K0 ~ K1,599; EH/EH2/SV: K0 ~ K9,999; latched; default = 0
D1102	Number of data to be read in file register. SA/SX/SC: K1 ~ K1,600; EH/EH2/SV: K1 ~ K8,000; latched; default = 0
D1103	Device for storing read data, starting from designated D. SA/SX/SC: K2,000 ~ K4,999; EH/EH2/SV: K2,000 ~ K9,999; latched; default = 2,000

MEMO

API	Mnemonic	Operands					Function	Controllers
150	MODRW	(S₁)	(S₂)	(S₃)	(S)	(n)	Read/Write MODBUS Data	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MODRW: 11 steps
S₁					*	*							*			
S₂					*	*							*			
S₃					*	*							*			
S													*			
n					*	*							*			

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Address of communication device   **S₂**: Function code   **S₃**: Device address of data to be read/written

S: Register for storing read/written data (source or destination)   **n**: Length of read/written data

Explanations:

1. The content of **S₂** shall only be: K3(H3), K6(H6), K16(H10).

2. ES/EX/SS V.4.9 (and above) support the continuous execution instruction (MODRW). Other versions do not support this instruction.

3. ES/EX/SS series MPU does not support E, F index register modification.

4. Flags: M1120 ~ M1131, M1140 ~ M1143. See remarks for more details.

5. Range of **S₁**: K0 ~ K254

6. **S₂**: Funcation code. For example, H03 is for AC motor drive or DVP-PLC to read many data; H06 is for AC motor drive or DVP-PLC to write a single data; H10 is for AC motor drive or DVP-PLC to write many data. Only these function codes are available currently; other function codes are still not executable. See program examples for more information.

7. **S₃**: Device address of data to be read/written. The device address inside the communication device. If the address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.

8. **S**: Register for storing read/written data. The user sets up a register and stores the data to be written in the register in advance. The register can be register for storing the read data.

9. **n**: Length of read/written data. For ES/SA, when M1143 = Off (in ASCII mode), the range of length: K1 ~ K8 (word); when M1143 = On (in RTU mode), the range of length: K1 ~ K16 (word). For EH/EH2/SV, the range of length: K1 ~ K16 (word).

10. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.
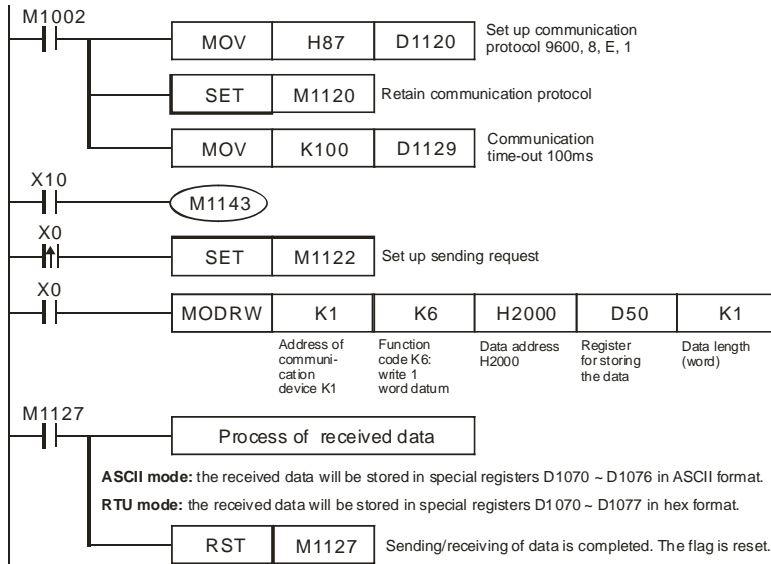
Program Example 1:

1. Function code K3(H3): For reading many data in register

   When PLC is connected to VFD-S AC motor drive: M1143 = Off, in ASCII mode

   When PLC is connected to VFD-S AC motor drive: M1143 = On, in RTU mode

2. When in ASCII mode, the received data will be stored in the designated registers starting from D0 in ASCII format and PLC will automatically convert the data into hex value and store them in special registers D1296 ~ D1311. When the conversion into hex value starts, M1131 will be On and turn Off when the conversion is completed.

3. If necessary, the user can move the hex values stored in D1296 ~ D1131 to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1296 ~ D1311.

4. When in RTU mode, the received data will be stored in the designated registers starting from D0 in hex format.

5. When In ASCII mode or RTU mode, PLC will store the data to be sent in D1256 ~ D1295. If necessary, the user can move the data to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1256 ~ D1295.

6. The data sent back from AC motor drive are stored in the registers designated by the user. After the transmission is completed, PLC will auto-check if the received data are incorrect. M1140 will be On if there is an error.

7. If the device address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.

8. After M1140 = On or M1141 = On, PLC will send another correct datum to AC motor drive. If the data sent back from AC motor drive is correct, M1140 and M1141 will be reset.
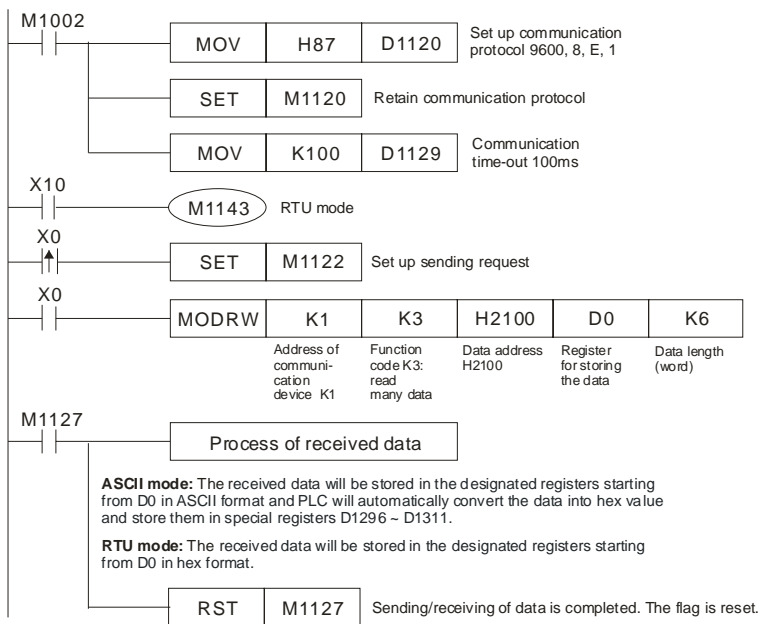
9. ASCII Mode: When PLC is connected to VFD-S AC motor drive.

    PLC ⇨ VFD-S, PLC sends: **"01 03 2100 0006 D5"**

    VFD-S ⇨ PLC, PLC receives: **"01 03 0C 0100 1766 0000 0000 0136 0000 3B"**

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1256 Low	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1256 High	'1'	31 H	ADR 0	
D1257 Low	'0'	30 H	CMD 1	Instruction code: CMD (1,0)
D1257 High	'3'	33 H	CMD 0	
D1258 Low	'2'	32 H	Starting Data Address	
D1258 High	'1'	31 H		
D1259 Low	'0'	30 H		
D1259 High	'0'	30 H		
D1260 Low	'0'	30 H	Number of Data (counted by words)	
D1260 High	'0'	30 H		
D1261 Low	'0'	30 H		
D1261 High	'6'	36 H		
D1262 Low	'D'	44 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
D1262 High	'5'	35 H	LRC CHK 0	

Registers for received data D0 (responding messages)

Register	DATA		Explanation	
D0 Low	'0'	30 H	ADR 1	
D0 High	'1'	31 H	ADR 0	
D1 Low	'0'	30 H	CMD 1	
D1 High	'3'	33 H	CMD 0	
D2 Low	'0'	30 H	Number of Data (counted by byte)	
D2 High	'C'	43 H		
D3 Low	'0'	30 H	Content of address 2100H	PLC automatically convert ASCII codes to numerals and store the numeral in D1296 = H0100
D3 High	'1'	31 H		
D4 Low	'0'	30 H		
D4 High	'0'	30 H		
D5 Low	'1'	31 H	Content of address 2101H	PLC automatically convert ASCII codes to numerals and store the numeral in D1297 = H1766
D5 High	'7'	37 H		
D6 Low	'6'	36 H		
D6 High	'6'	36 H		
D7 Low	'0'	30 H	Content of address 2102H	PLC automatically convert ASCII codes to numerals and store the numeral in D1298 = H0000
D7 High	'0'	30 H		
D8 Low	'0'	30 H		
D8 High	'0'	30 H		
D9 Low	'0'	30 H	Content of address 2103H	PLC automatically convert ASCII codes to numerals and store the numeral in D1299 = H0000
D9 High	'0'	30 H		
D10 Low	'0'	30 H		
D10 High	'0'	30 H		
D11 Low	'0'	30 H	Content of address 2104H	PLC automatically convert ASCII codes to numerals and store the numeral in D1300 = H0136
D11 High	'1'	31 H		
D12 Low	'3'	33 H		
D12 High	'6'	36 H		
D13 Low	'0'	30 H	Content of address 2105H	PLC automatically convert ASCII codes to numerals and store the numeral in D1301 = H0000
D13 High	'0'	30 H		
D14 Low	'0'	30 H		
D14 High	'0'	30 H		
D15 Low	'3'	33 H	LRC CHK 1	
D15 High	'B'	42 H	LRC CHK 0	

10. RTU Mode: When PLC is connected to VFD-S AC motor drive

    PLC ⇨ VFD-S, PLC sends: **"01 03 2100 0006 CF F4"**

    VFD-S ⇨ PLC, PLC receives: **"01 03 0C 0000 0503 0BB8 0BB8 0000 012D 8E C5"**

Registers for sent data (sending messages)

Register	DATA	Explanation	
D1256 Low	01 H	Address	
D1257 Low	03 H	Function	
D1258 Low	21 H	Starting Data Address	
D1259 Low	00 H		
D1260 Low	00 H	Number of Data (counted by words)	
D1261 Low	06 H		
D1262 Low	CF H	CRC CHK Low	
D1263 Low	F4 H	CRC CHK High	

Registers for received data D0 (responding messages)

Register	DATA	Explanation	
D0 Low	01 H	Address	
D1 Low	03 H	Function	
D2 Low	0C H	Number of Data (byte)	
D3 Low	00 H	Content of address 2100H	PLC automatically convert ASCII codes to numerals and store the numeral in D1296 = H0000
D4 Low	00 H		
D5 Low	05 H	Content of address 2101H	PLC automatically convert ASCII codes to numerals and store the numeral in D1297 = H0503
D6 Low	03 H		
D7 Low	0B H	Content of address 2102H	PLC automatically convert ASCII codes to numerals and store the numeral in D1298 = H0BB8
D8 Low	B8 H		
D9 Low	0B H	Content of address 2103H	PLC automatically convert ASCII codes to numerals and store the numeral in D1299 = H0BB8
D10 Low	B8 H		
D11 Low	00 H	Content of address 2104H	PLC automatically convert ASCII codes to numerals and store the numeral in D1300 = H0000
D12 Low	00 H		
D13 Low	01 H	Content of address 2105H	PLC automatically convert ASCII codes to numerals and store the numeral in D1301 = H012D
D14 Low	2D H		
D15 Low	8E H	CRC CHK Low	
D16 Low	C5 H	CRC CHK High	

Program Example 2:

1. Function code K6(H6): For writing a word data to register

    When PLC is connected to VFD-S AC motor drive: M1143 = Off, in ASCII mode

    When PLC is connected to VFD-S AC motor drive: M1143 = On, in RTU mode

2. When in ASCII mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1076.

3. When in RTU mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1077.

4. When In ASCII mode or RTU mode, PLC will store the data to be sent in D1256 ~ D1295. If necessary, the user can move the data to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1256 ~ D1295.

5. After receiving the data sent back from AC motor drive is completed, PLC will auto-check if the received data are incorrect. M1140 will be On if there is an error.

6. If the device address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.

7. After M1140 = On or M1141 = On, PLC will send another correct datum to AC motor drive. If the data sent back from AC motor drive is correct, M1140 and M1141 will be reset.

8. ASCII Mode: When PLC is connected to VFD-S AC motor drive.

   PLC ⇨ VFD-S, PLC sends: **"01 06 0100 1770 71"**

   VFD-S ⇨ PLC, PLC receives: **"01 06 0100 1770 71"**

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1256 Low	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1256 High	'1'	31 H	ADR 0	
D1257 Low	'0'	30 H	CMD 1	Instruction code: CMD (1,0)
D1257 High	'6'	36 H	CMD 0	
D1258 Low	'0'	30 H		
D1258 High	'1'	31 H	Data Address	
D1259 Low	'0'	30 H		
D1259 High	'0'	30 H		
D1260 Low	'1'	31 H		
D1260 High	'7'	37 H	Data content	The content of register D50 (H1770 = K6,000)
D1261 Low	'7'	37 H		
D1261 High	'0'	30 H		
D1262 Low	'7'	37 H	LRC CHK 1	LRC CHK (0,1) is error check
D1262 High	'1'	31 H	LRC CHK 0	

Registers for received data (responding messages)

Register	DATA		Explanation
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

9. RTU Mode: When PLC is connected to VFD-S AC motor drive

   PLC ⇨ VFD-S, PLC sends: **"01 06 2000 0012 02 07"**

   VFD-S ⇨ PLC, PLC receives: **"01 06 2000 0012 02 07"**

Registers for sent data (sending message)

Register	DATA	Explanation	
D1256 Low	01 H	Address	
D1257 Low	06 H	Function	
D1258 Low	20 H	Data Address	
D1259 Low	00 H		
D1260 Low	00 H	Data content	The content of register D50 (H12)
D1261 Low	12 H		
D1262 Low	02 H	CRC CHK Low	
D1263 Low	07 H	CRC CHK High	

Registers for received data (responding message)

Register	DATA	Explanation
D1070 Low	01 H	Address
D1071 Low	06 H	Function
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. Function code K16(H10): For writing many word data into register.

   When PLC is connected to VFD-S AC motor drive: M1143 = Off, in ASCII mode

   When PLC is connected to VFD-S AC motor drive: M1143 = On, in RTU mode

2. When in ASCII mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1076.

3. When in RTU mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1077.

4. When In ASCII mode or RTU mode, PLC will store the data to be sent in D1256 ~ D1295. If necessary, the user can move the data to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1256 ~ D1295.

5. After receiving the data sent back from AC motor drive is completed, PLC will auto-check if the received data are incorrect. M1140 will be On if there is an error.

6. If the device address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.

7. After M1140 = On or M1141 = On, PLC will send another correct datum to AC motor drive. If the data sent back from AC motor drive is correct, M1140 and M1141 will be reset.

8. ASCII Mode: When PLC is connected to VFD-S AC motor drive.

    PLC ⇨ VFD-S, PLC sends: "**01 10 2000 0002 04 0012 1770 30**"

    VFD-S ⇨ PLC, PLC receives: "**01 10 2000 0002 CD**"

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1256 Low	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1256 High	'1'	31 H	ADR 0	
D1257 Low	'1'	31 H	CMD 1	Instruction code: CMD (1,0)
D1257 High	'0'	30 H	CMD 0	
D1258 Low	'2'	32 H	Data Address	
D1258 High	'0'	30 H		
D1259 Low	'0'	30 H		
D1259 High	'0'	30 H		
D1260 Low	'0'	30 H	Number of Registers	
D1260 High	'0'	30 H		
D1261 Low	'0'	30 H		
D1261 High	'2'	32 H		

Register	DATA		Explanation	
D1262 Low	'0'	30 H	Byte Count	
D1262 High	'4'	34 H		
D1263 Low	'0'	30 H	Data contents 1	The content of register D50 (H12)
D1263 High	'0'	30 H		
D1264 Low	'1'	31 H		
D1264 High	'2'	32 H		
D1265 Low	'1'	31 H	Data contents 2	The content of register D51 (H1770 = K6,000)
D1265 High	'7'	37 H		
D1266 Low	'7'	37 H		
D1266 High	'0'	30 H		
D1267 Low	'3'	33 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
D1267 High	'0'	30 H	LRC CHK 0	

Registers for received data (responding messages)

Register	DATA		Explanation
D1070 Low	'0'	30 H	ADR 1
D1070 High	'1'	31 H	ADR 0
D1071 Low	'1'	31 H	CMD 1
D1071 High	'0'	30 H	CMD 0
D1072 Low	'2'	32 H	Data Address
D1072 High	'0'	30 H	
D1073 Low	'0'	30 H	
D1073 High	'0'	30 H	
D1074 Low	'0'	30 H	Number of Registers
D1074 High	'0'	30 H	
D1075 Low	'0'	30 H	
D1075 High	'2'	32 H	
D1076 Low	'C'	43 H	LRC CHK 1
D1076 High	'D'	44 H	LRC CHK 0

9. RTU Mode: When PLC is connected to VFD-S AC motor drives

PLC ⇨ VFD-S, PLC sends: "**01 10 2000 0002 04 0012 1770 C4 7F**"

VFD-S ⇨ PLC, PLC receives: "**01 10 2000 0002 4A 08**"

Registers for send data (sending messages)

Register	DATA	Explanation	
D1256 Low	01 H	Address	
D1257 Low	10 H	Function	
D1258 Low	20 H	Data Address	
D1259 Low	00 H		
D1260 Low	00 H	Number of Registers	
D1261 Low	02 H		
D1262 Low	04 H	Byte Count	
D1263 Low	00 H	Data content 1	The content of register D50 (H12)
D1264 Low	12 H		
D1265 Low	17 H	Data content 2	The content of register D51 (H1770 = K6,000)
D1266 Low	70 H		
D1267 Low	C4 H	CRC CHK Low	
D1268 Low	7F H	CRC CHK High	

Registers for received data (responding messages)

Register	DATA	Explanation
D1070 Low	01 H	Address
D1071 Low	10 H	Function
D1072 Low	20 H	Data Address
D1073 Low	00 H	
D1074 Low	00 H	Number of Registers
D1075 Low	02 H	
D1076 Low	4A H	CRC CHK Low
D1077 Low	08 H	CRC CHK High

Remarks:

1. The activation condition placed before MODRD, RDST and MODRW instructions cannot use rising-edge or falling-edge contacts; otherwise the data stored in the registers for received data will encounter errors.

2. Flags and special registers for MODRW instruction in RS-485 communication. (For details, see API 80 RS).

Flags	Function
M1120	For retaining communication setups. After the setup is made, changes in D1120 will be invalid.
M1121	When Off, RS-485 is sending data.
M1122	Sending request
M1123	Receiving is completed
M1124	Waiting for receiving data
M1125	Disable receiving status
M1126	Selecting STX/ETX system
M1127	Sending/receiving data through MODRD / RDST / MODRW instructions is completed.
M1128	Sending data…/receiving data…
M1129	Receiving data time-out
M1130	User/system defined STX/ETX
M1131	On when MODRD / MODWR / MODRW is converting data to hex
M1140	MODRD / MODWR / MODRW data receiving error
M1141	MODRD / MODWR / MODRW parameter error
M1142	VFD-A handy instruction data receiving error
M1143	ASCII/RTU mode selection (used with MODRD/MODWR/MODRW) (Off = ASCII mode; On = RTU mode)
D1070 ~ D1085	When the built-in RS-485 communication instruction is executed and sends out data, the receiving end will respond with a message and the message will be stored in D1070 ~ D1085. The user can check the registers for the messages.
D1120	RS-485 communication protocol
D1121	PLC communication address (saving PLC communication address; latched)
D1122	Remaining words of the sent data
D1123	Remaining words of the received data
D1124	Start text definition (STX)
D1125	Definition of end text 1 (ETX1)
D1126	Definition of end text 2 (ETX2)
D1129	Abnormal communication time-out. Unit: ms
D1130	Records of error codes sent back from MODBUS
D1256 ~ D1295	When the built-in RS-485 communication instruction MODRW is executed, the sent out data will be stored in D1256 ~ D1295. The user can check whether the instruction is correct by the contents in the registers.
D1296 ~ D1311	PLC will automatically convert the ASCII data stored in the register designated by the user into hex format.

API	Mnemonic	Operands	Function	Controllers
151	PWD	**S** **D**	Detection of Input Pulse Width	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices						Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PWD: 5 steps
S	*															
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device   **D**: Destination device for storing the detected result

Explanations:

1. Range of **S**: X10 ~ X17

2. Range of **D**: D0 ~ D999, occupying 2 consecutive devices. Can only be used once in the program.

3. PWD instruction is for detecting the time span of output signals from X10 ~ X17; the valid frequency range is 1 ~1KHz. When M1169 = On, the instruction will detect the time span of the continuous rising edge and falling edge of the input signals (time unit: 100us). When M1169 = On, the instruction will detect the time span of 2 continuous rising edges of the input signals (time unit: 1us). It cannot designate the same X10 ~ X17 as does DCNT and ZRN instructions.

4. **D** occupies two continuous devices. The longest detectable time is 21,474.83647 seconds, about 357.9139 minutes or 5.9652 hours.

5. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:

When X0 = On, record the time span of X10 = On and store it in D1 and D0.

```
  X0
──┤├──┌──────┬──────┬──────┐
      │ PWD  │ X10  │  D0  │
      └──────┴──────┴──────┘
```
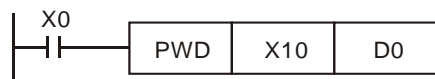
API	Mnemonic	Operands	Function	Controllers
152	RTMU	**D** **n**	Start of the Measurement of Execution Time of I Interruption	ES/EX/SS SA/SX/SC **EH/SV**

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	RTMU: 5 steps
D					*	*							*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC **EH SV**	ES EX SS SA SX SC EH SV

Operands:

D: Device for storing the measuring time (unit: 1us)   **n**: Measurement time base. Parameter range: K10 ~ K500 (time unit: 1us)

Explanations:

1. Range of **D**: K0 ~ K9

2. Range of **n**: K10 ~ K500

3. The designated special D registers (D1156 ~ D1165) can measure up to 10 interruption subroutines. For example, when **D** = K5, the designated D register will be D1161.

4. When RTMU is executed, if the **D** and **n** entered by the user are legal, interruption of the timer will be enabled and the counting starts and the special D designated by **D** is cleared as 0. When RTMD is executed, interruption of the timer is disabled and the calculated time will be assigned to special D designated by RTMD.

5. With API 153 RTMD, RTMU can measure the execution time of "I" interruption service subroutine, which can be reference for dealing with the high-speed response when the user is at the initial stage of developing the program.

API	Mnemonic	Operands	Function	Controllers
153	RTMD	**D**	End of the Measurement of the Execution Time of I Interruption	ES/EX/SS SA/SX/SC **EH/SV**

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

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC **EH SV**	ES EX SS SA SX SC EH SV

Operands:

D: Device to store the measuring time (unit: 1us).

Explanations:

1. Range of **D**: K0 ~ K9. The No. of **D** has to be the same as that designated by **D** in API 152 RTMU; otherwise the result of the measurement may be unexpectable.

Program Example:

When X0 goes from Off to On, the program will enter I001 interruption subroutine. RTMU will activate an 8-bit timer (unit: 10us) and RTMD (when D = K0) will shut down the timer and store the time in the timer in special D registers (D1156 ~ D1165, designated by K0 ~ K9).
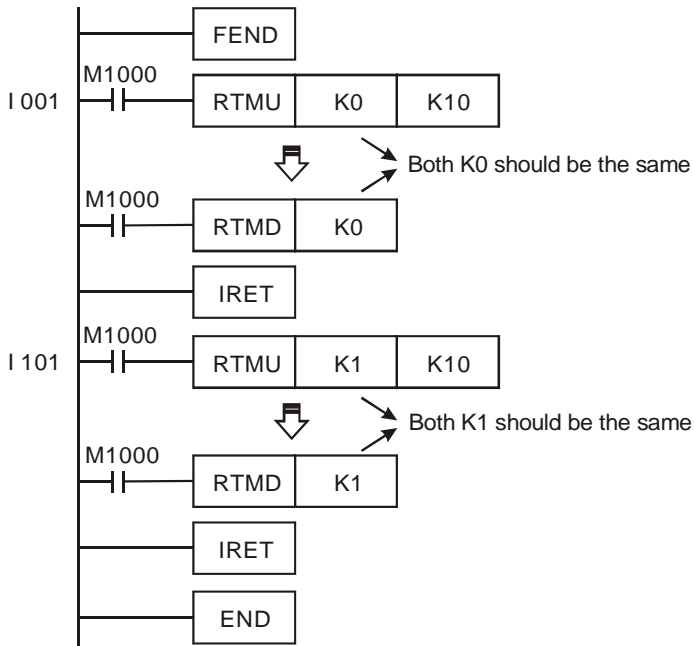
Remarks:

1. We suggest you remove this instruction after you finish developing your PLC program.
2. Due to the lower priority of the interruption enabled by RTMU, when RTMU is enabled, other high-speed pulse input counting or high-speed pulse output may result in failure to trigger the timer.
3. If you activate RTMU but do not activate RTMD before the end of the interruption, the interruption will not be shut down.
4. RTMU instruction activates 1 timer interruption in PLC. Therefore, if many RTMU or RTMD are executed at the same time, confusion in the timer may occur. Please be aware of the situation.

API	Mnemonic		Operands			Function			Controllers
154	RAND	P	S_1 S_2 D			Random Number			ES/EX/SS SA/SX/SC EH/SV

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

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Lower bound of the random number   **S₂**: Upper bound of the random number   **D**: The random number produced

Explanations:

1. $S_1 \leq S_2$; K0 $\leq$ S_1 , S_2 $\leq$ K32,767

2. See the specifications of each model for their range of use.

3. 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.

```
 X10
──┤├──┤ RAND │ D0 │ D10 │ D20 │
```
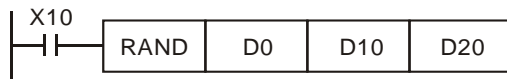
API	Mnemonic		Operands			Function		Controllers
155	D	ABSR	S	D_1	D_2	Read the Absolute Position from a Servo Motor		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DABSR: 13 steps
S	*	*	*	*												
D_1		*	*	*												
D_2								*	*	*	*	*	*	*		

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Input signal from Servo (occupies 3 consecutive devices)   **D₁**: Control signal for controlling Servo (occupies 3 consecutive devices)   **D₂**: Absolute position data (32-bit) read from Servo

Explanations:

1. Operand S and D_1 of SA series MPU do not support E, F index register modification.

2. See the specifications of each model for their range of use.

3. This instruction can only be used once in the program.

4. Flag: see remarks for more details.

5. This instruction reads the absolute position (ABS) of MITSUBISHI MR-J2 servo drive (with absolute position check function).

6. **S** will occupy 3 consecutive devices, **S**, **S** +1, and **S** +2. **S** and **S** +1 are connected to the absolute position (bit 0, bit 1) on the servo for data transmitting. **S** +2 is connected to Servo for transmitting data ready flag. See the wiring example below for more details.

7. **D₁** will occupy 3 consecutive devices, **D₁**, **D₁** + 1, **D₁** + 2. **D₁** is connected to SERVO On (SON) of Servo. **D₁**+1 is connected to ABS transmisstion mode of Servo and **D₁**+2 is connected to ABS request signal. See the wiring example below for more details.
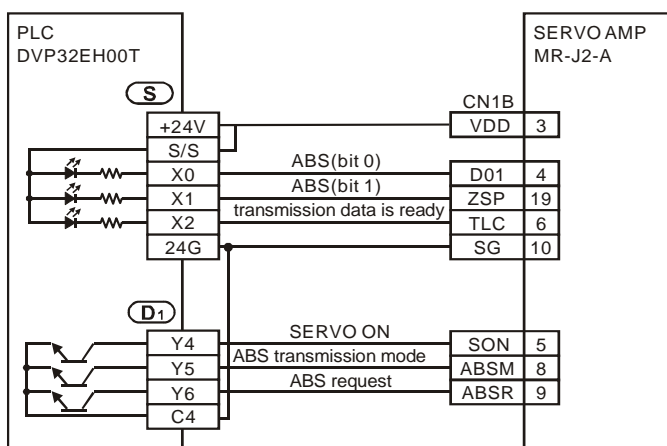
8. **D₂** will occupy 2 consecutive devices **D₂** and **D₂** + 1. **D₂** is the lower 16 bits and **D₂** + 1 is the higher 16 bits. The absolute position data should be written into the present value registers (D1337, D1336) of CH0 pulse (Y0, Y1) or the present value registers (D1339, D1338) of CH1 pulse (Y2, Y3) in EH series MPU; therefore, we suggest you designate the two corresponding registers. If you designate other devices as the registers, you still have to

transmit the data to D1337 and D1336 of CH0 or D1339 and D1338 of CH1. In addition, the absolute position data should be written into the present value registers (D1348, D1349) of CH0 pulse (Y10) or the present value registers (D1350, D1351) of CH1 pulse (Y11) in SC series MPU; therefore, we suggest you designate the two corresponding registers. If you designate other devices as the registers, you still have to transmit the data to D1348 and D1349 of CH0 or D1350 and D1351 of CH1.

9. When DABSR instruction starts to read, after finishing reading the absolute position of SERVO, flag M1029 will be On. The user has to reset the flag.

10. 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.

Program Example:

1. When X7 = On, the 32-bit absolute position data read from Servo will be stored in the present value registers (D1337, D1336) of CH0 pulse in EH MPU. At the same time, the timer T10 is enabled and starts to count for 5 seconds. If the reading of the absolute position is not completed after 5 seconds, M10 will be On, indicating that the reading of absolute position encounters abnormality.

2. When enabling the connection to the system, please synchronize the power input of DVP-PLC EH/EH2/SV and SERVO AMP or activate the power of SERVO AMP earlier than DVP-PLC.
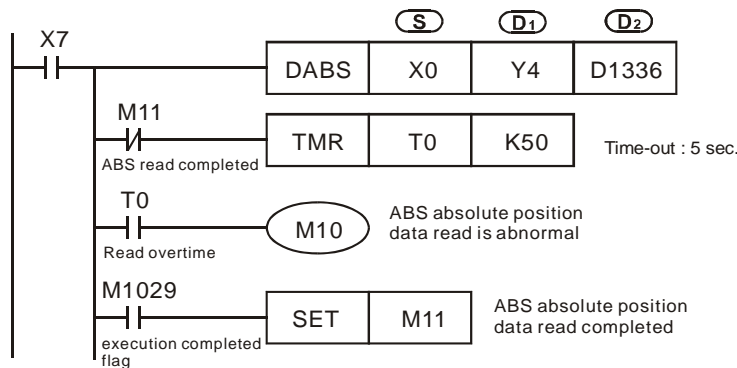
Remarks:

1. If the instruction is interrupted when PLC is still reading the absolute position of SERVO, an ALARM message (ALE5) will occur in SERVO.

2. Timing chart of DABSR instruction reading absolute position:

   a) When DABSR instruction starts to execute, it will drive SERBVO On (SON) and ABS transmittion mode for output.

   b) By "transmission is ready" and "ABS request" signals, you can confirm the transmission and reciept of both sides as well as processing the transmission of the 32-bit present position data plus the 6-bit check data.

   c) The data are transmitted by ABS (bit0, bit1).

SERVO ON — SON

ABS data transmission mode — ABSM

Transmission data is ready — TLC — AMP output

ABS request — ABSR — Controller output

ABS(bit 1) — ZSP — AMP output

ABS(bit 0) — D01 — AMP output

32-bit present value position data +(plus) 6-bit check data
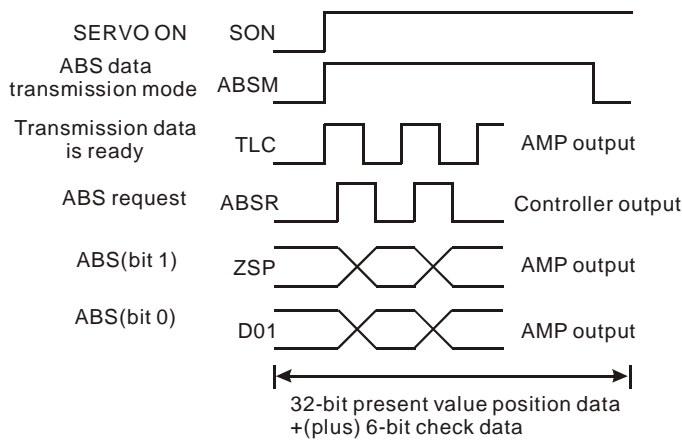
3. This instruction is applicable to the Servo motor equipped with absolute positioning function, e.g. Mitsubishi MR-J2-A Servo drive.

4. Select one of the following methods for the initial reading of present absolute position.

    a) Complete zero point return by using reset signal function to execute API 156 ZRN instruction.

    b) After using JOG or manual operation to adjust the zero point position, input a reset signal in SERVO AMP. See the figure of external switch below for whether to use DVP-PLC for output. For the wiring of DVP-PLC and Mitsubishi MR-H2-□A, see remarks of API 159 DRVA instruction.

Ex: Mitsubishi MR-J2-□A
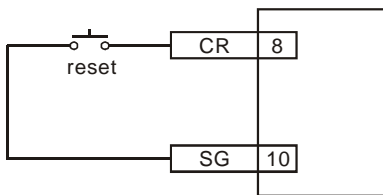
5. Flags explanation:

    M1010： (For EH/EH2/SV series MPU) When M1010 is On, CH0 (Y0, Y1) and CH1 (Y2, Y3) will output pulses while END instruction is being executed. When the output starts, M1010 will automatically turn Off.

    M1029： (For EH/EH2/SV series MPU) When the first group CH0 (Y0, Y1) pulse output or the execution of other relevant instructions are completed, M1029 will turn On.

    M1030： (For EH/EH2/SV series MPU) When the second group CH1 (Y2, Y3) pulse output is completed, M1030 will turn on.

    M1102： (For SC series MPU) When the first group CH0 (Y10) pulse output is completed, M1102 will turn On.

    M1103： (For SC series MPU) When the second group CH1 (Y11) pulse output is completed, M1103 will turn On.

    M1258： (For EH/EH2/SV series MPU) When M1258 is On, CH0 (Y0, Y1) will output reverse pulses.

    M1259： (For EH/EH2/SV series MPU) When M1259 is On, CH1 (Y2, Y3) will output reverse pulses.

    M1305： (For EH/EH2/SV series MPU) PLSV, DPLSV, DRVI, DDRVI, DRVA, DDRVA instructions for CH0 (Y1, Y2) reverse running.

M1306： (For EH/EH2/SV series MPU) PLSV, DPLSV, DRVI, DDRVI, DRVA, DDRVA instructions for CH1 (Y2, Y3) reverse running.

M1334： (For EH series MPU) When M1334 = On, CH0 (Y0, Y1) pulse output will pause.

(For EH2/SV series MPU) When M1334 = On, CH0 (Y0, Y1) pulse output will stop.

(For SC series MPU) When M1334 = On, the DDRVI and DDRVA execution criteria will stop and CH0 (Y10) pulse output will stop immediately without deceleration.

M1335： (For EH series MPU) When M1335 = On, CH1 (Y2, Y3) pulse output will pause.

(For EH2/SV series MPU) When M1335 = On, CH1 (Y2, Y3) pulse output will stop.

(For SC series MPU) When M1335 = On, DDRVI and DDRVA execution criteria will stop and CH1 (Y11) pulse output will stop immediately without deceleration.

M1520： (For EH2/SV series MPU) When M1520 = On, CH2 (Y4, Y5) pulse output will stop.

M1521： (For EH2/SV series MPU) When M1521 = On, CH3 (Y6, Y7) pulse output will stop.

M1336： (For EH/EH2/SV series MPU) CH0 (Y0, Y1) pulse output indication flag

M1337： (For EH/EH2/SV series MPU) CH1 (Y2, Y3) pulse output indication flag

M1346： (For EH/EH2/SV series MPU) ZRN instruction for "enabling CLEAR output signal" flag

6. Special registers:

D1337, D1336： 1. (For EH/EH2/SV series MPU) Registers for the first group (Y0, Y1) output pulse present value of position control instructions (API 156 ZRN, API 157 PLSV, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1337 is for high word; D1336 is for low word.

2. (For EH/EH2/SV series MPU) Registers for storing the current number of output pulses of the first group (Y0, Y1) output of pulse output instructions (API 57 PLSY, API 59 PLSR). D1337 is for high word; D1336 is for low word.

D1338, D1339： 1. (For EH/EH2/SV series MPU) Registers for the second group (Y2, Y3) output pulse present value of position control instructions (API 156 ZRN, API 157 PLSV, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1339 is for high word; D1338 is for low word.

2. (For EH/EH2/SV series MPU) Registers for storing the current number of output pulses of the second group (Y2, Y3) output of pulse output instructions (API 57 PLSY, API 59 PLSR). D1339 is for high word; D1338 is for low word.

D1340 (D1352)： For setting up the frequencies of the first acceleration segment and the last deceleration segment when the position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA) are executing CH0 (CH1) outputs.

<u>Range of setting:</u>

For EH/EH2/SV series MPU, the speed has to be higher than 10Hz. Frequency lower than 10Hz or higher than maximum output frequency will be output by 10Hz. The default setting in EH/EH2/SV series MPU is 200Hz. For SC series MPU, the speed has to be 100 ~ 100KHz. Frequency lower than 100Hz will be output by 100Hz and frequency higher than

100KHz will be output by 100KHz. The default setting in SC series MPU is 100Hz.

Note: During the control of the stepping motor, please consider the resonance and the limitation on the start frequency when you set up the speed.

D1341, D1342： (For EH/EH2/SV series MPU) For setting up the maximum speed when the position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA) are being executed. D1342 is for high word; D1341 is for low word.

Range of setting: 200KHz fixed.

D1343 (D1353)： For setting up the time of the first acceleration segment and the last deceleration segment when the position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA) are executing CH0 (CH1) outputs.

Range of setting:

For EH/EH2/SV series MPU, the acceleration/deceleration time has to be longer than 10ms. The time shorter than 10ms or longer than 10,000ms will be output by 10ms. The default setting in EH/EH2/SV series MPU is 100ms. For SC series MPU, the time has to be 50 ~ 20,000ms. The time shorter than 50ms will be regarded as 50ms.

Note: During the control of the stepping motor, please consider the resonance and the limitation on the start frequency when you set up the speed.

D1348, D1349： (For SC series MPU) Registers for the first group (Y0, Y1) output pulse present value of position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1349 is for high word; D1348 is for low word.

D1350, D1351： (For SC series MPU) Registers for the second group (Y11) output pulse present value of position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1351 is for high word; D1350 is for low word.

API	Mnemonic		Operands	Function	Controllers
156	D ZRN		S_1 S_2 S_3 D	Zero Return	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	ZRN: 9 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DZRN: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
S_3	*	*	*	*												
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Zero return speed   **S_2**: Creep speed   **S_3**: Near p oint signal (DOG)   **D**: Pulse output device (please use transistor output module)

Explanations:

1.   When **S_1** and **S_2** are used in device F, only 16-bit instruction is applicable.

2.   **S_1** and **S_2** of SC series MPU only support device K, H and D. **S_3** of SC series MPU only supports device X10 and X11.

3.   Flag: see remarks of API 155 ABSR and API 158 DDRVI for more details.

4.   **S_1** is the starting speed of zero return operation. For EH/EH2/SV series MPU, the 16-bit instruction can designate the range of the speed, which is 10 ~ 32,767Hz and the range designated by the 32-bit instruction is 10 ~ 200,000Hz. If the designated speed is slower than 10Hz, the zero return will operate at 10Hz and when the designated speed is faster than 200KHz, the zero return will operate at 200KHz. For SC series MPU, the 32-bit instruction can designate the range of speed , which is 100 ~ 100,000Hz. If the designated speed is slower than 100Hz, the zero return will operate at 100Hz, and when the designated speed is faster than 100KHz, the zero return will operate at 100KHz.

5.   **S_2** is the designated low speed after the near point signal (DOG) is On. EH/EH2/SV series MPU can designate the range of **S_2**, which is 10 ~ 32,767Hz and SC series MPU can designate the range 100 ~ 100,000Hz.

6.   **S_3** is the designated near point signal (DOG) input (input from A contact). In EH/EH2/SV series MPU, if devices other than the external output device (X10 ~ X17), e.g. X, Y, M, S are designated, they will be affected by the scan period, resulting in dispersion of the zero point. In addition, please note that the MPU cannot designate the same input points X10 ~ X17 as those designated by DCNT and PWD instructions. SC series MPU can only designate X10 and X11 and cannot designate the same input points as those designated by DCNT instruction.

7.   EH series MPU has two groups of A/B phase pulse output, CH0 (Y0, Y1) and CH1 (Y2, Y3); EH2/SV series MPU has four groups of A/B phase pulse output, CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks for the setup methods.

8.   Zero return output device in different models

Model	SC MPU	EH MPU	EH2/SV MPU
Zero return output	Y10, Y11	Y0, Y2	Y0, Y2, Y4, Y6

9.   When executing API 158 DRVI (releative positioning) or API 159 DRVA (absolute positioning), PLC will automatically store the increasing or decreasing forward/reverse pulses in the present value registers. For

EH/EH2/SV series MPU, Y0: D1337, D1336; Y2: D1339, D1338, Y4: D1376, D1375; Y6: D1378, D1377. For SC series MPU, Y10: D1348, D1349; Y11: D1350, D1351. In this way, you can keep track of the position of the machine at any time. However, due to that the data will be lost when the power of the PLC is switched off, you have to enter the zero point position of the machine when executing zero return for the first time.

Program Example:

When M10= On, Y0 output pulses start to operate zero return at the frequency of 20KHz. When the zero return meets DOG X15 = On, Y0 output pulses will start to operate by creep speed 1KHz until X15 is Off.

```
  M10
───┤├───┤ ZRN │ K20000 │ K1000 │ X15 │ Y0 │
```
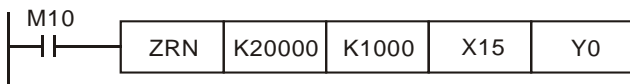
Remarks:

1. Timing chart of the reset signal output for EH/EH2/SV series MPU. (SC series MPU does not support this function.)

   a) When the reset signal flag M1346 = On, after zero return is completed, the PLC can send the reset signal to the servo drive and the signal will last for approximately 20ms. After 20ms, the reset signal will return to Off again.

   b) Output devices for reset signals of EH series MPU:

      CH0 (Y0, Y1) reset output device (Y4)

      CH1 (Y2, Y3) reset output device (Y5)

   c) Output devices for reset signals of EH2/SV series MPU:

      CH0 (Y0, Y1) reset output device (Y10)

      CH1 (Y2, Y3) reset output device (Y11)

      CH2 (Y4, Y5) reset output device (Y12)

      CH3 (Y6, Y7) reset output device (Y13)
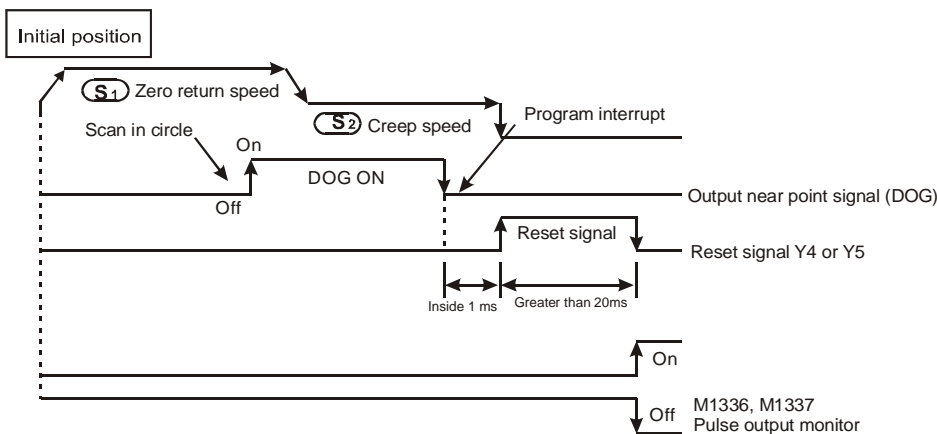
Note: The designated devices, X, Y, M, and S, other than the external input devices X10 ~ X17 will be affected by the scan period, 2 times of the scan period at worst.

2. The zero return operation:

   a) When ZRN instruction is executed, the frequency of the first acceleration segment of CH0 (CH1) in EH/EH2/SV series MPU is set by D1340 (D1352). In SC series MPU, CH0 (CH1) will set creep speed as the

start frequency. The acceleration time of EH/EH2/SV/SC series MPU is set by D1343 (D1353). S1 will start to move when the acceleration reaches the zero return speed.

b) When the DOG signal goes from Off to On, the zero return speed will decelerate to **S₂** by the time set in D1343 (D1353).

c) When the DOG signal goes from On to Off and the pulse output stops, 0 will be written in the present value registers (D1337, D1336) of CH0 pulses, D1339 and D1338 of CH1 pulses, D1375 and D1376 of CH2 pulses, and D1377 and D1378 of CH3 pulses in EH/EH2/SV series MPU; 0 will also be written in D1348 and D1349 of Y10 (CH0) pulses or D1350 and D1351 of Y11 (CH1) pulses in SC series MPU.

d) When the DOG signal goes from On to Off and the reset signal flag M1346 = On, Y4 (CH0) or Y5 (CH1) in EH series MPU will output a reset signal; Y10 (CH0), Y11 (CH1) , Y12 (CH2) and Y13 (CH3) in EH2/SV series MPU will output a reset signal.

e) For EH/EH2/SV series MPU, when the pulse output is completed and M1029, M1030, M1036 and M1037 are enabled, indication flag M1336 sent by CH0 pulses, M1337 by CH1, M1522 by CH2 and M1523 by CH3 will be Off. For SC series MPU, when the pulse output is completed, M1102 and M1103 will be enabled.

f) Due to that ZRN (DZRN) instruction cannot locate the position of DOG, the zero return can only be done towards a single direction. In the zero return operation of EH/EH2/SV series MPU, D1337 and D1336 (present value registers) of CH0 pulses or D1339 and D1338 of CH1 pulses are decreasing. In the zero return operation of SC series MPU, D1348 and D1349 of CH0 pulses or D1350 and D1351 of CH1 pulses are also decreasing.
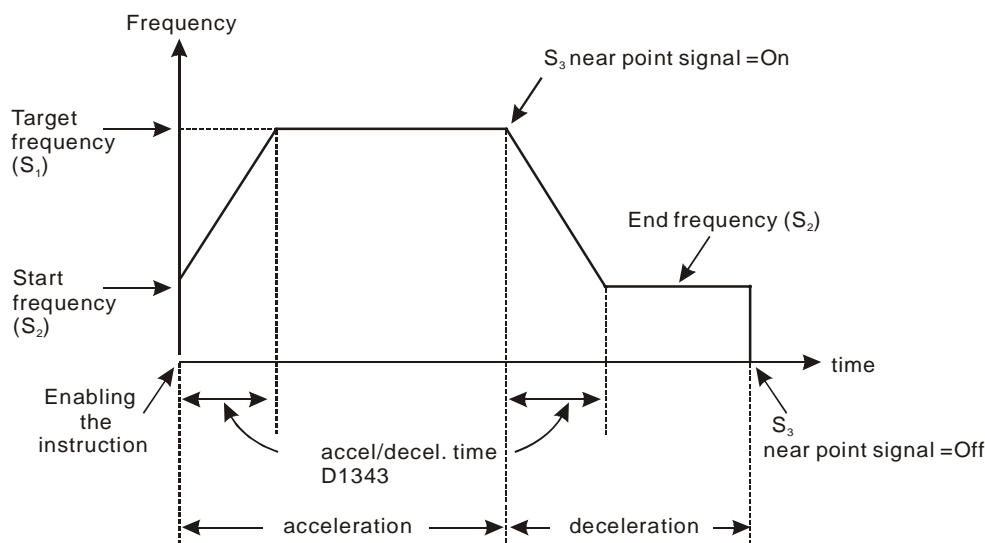
g) ZRN (DZRN) instruction is applicable to servo motor with absolute positioning function, e.g. Mitsubishi MR-J2-A servo drive. Even when the power is switched off, the current position can still be recorded. In addition, the current position of servo drive can be read by API 155 DABSR of EH/EH2/SV/SV series MPU; therefore only one zero return operation is required and no zero return has to be done after the power is switched off.

h) When the drive contact of ZRN instruction is On, CH0 (CH1) will read the acceleration/deceleration time set in D1343 (D1353) and accelerate to the zero return speed, waiting for the DOG and decelerate to creep speed. When the DOG is Off, the pulse output will stop immediately.

i) For SC series MPU, many ZRN instructions can be compiled in the program but only one instruction can be executed when the PLC program is being executed. For example, provided there is already an instruction enabling Y10 output, other instructions enabling also Y10 output will not be executed. The principle of the instruction execution is "first come, first executed".

j) For SC series MPU, when you designate Y10 as the output device, you can choose either X10 or X11 for DOG input in the "acceleration to deceleration" segment. In other words, when designating Y11 as the output device, you can also choose either X10 or X11 for DOG input.

k) For SC series MPU, due to that this instruction does not compare between the number of output pulses, the DOG input (from Off to On) will therefore become the trigger of acceleration converting to deceleration. The "On" time of DOG has to be longer than 10us; otherwise the signal may be regarded as useless interference.

l) For SC series MPU, when the execution of the instrucion enters the deceleration segment and the output frequency reaches creep speed (end frequency), the output will stop when DOG goes from On to Off.

m) For SC series MPU, the current accumulated number of pulses of Y10 is stored in D1348 and D1349 and that of Y11 is stored in D1350 and D1351. Then the program operates from STOP to RUN or from RUN to STOP, the contents will not be cleared to 0.

n) For SC series MPU, M1102 = On indicates the end of Y10 pulse output; M1103 = On indicates the end of Y11 pulse output.

o) For SC series MPU, after the instruction is executed, all parameters cannot be modified unless the execution of the instruction stops.

p) For SC series MPU, when the execution of the stops, all outputs will stop immediately no matter what type of the output it is.

API	Mnemonic		Operands			Function		Controllers
157	D	PLSV	**S** **D₁** **D₂**			Adjustable Speed Pulse Output		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	PLSV: 7 steps
S					*	*	*	*	*	*	*	*	*	*	*	DPLSV: 13 steps
D₁		*														
D₂		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Pulse output frequency   **D₁**: Pulse output device (please use transistor output module)   **D₂**: Output device for the signal of rotation direction

Explanations:

1. See remarks for the setting range of **S**, **D₁** and **D₂**.

2. Flag: see remarks of API 155 ABSR and API 158 DDRVI for more details.

3. **S** is the designated pulse output frequency. The 16-bit instruction can designate its range 0 ~ +32,767Hz, 0 ~ -32,768Hz. The ranges designated by 32-bit instruction are 0 ~ +200,000Hz and 0 ~ -200,000Hz. "+/-" signs indicate forward/backward directions. During the pulse output, the frequency can be changed, but not the frequencies of different directions.

4. **D₁** is the pulse output device. EH series MPU can designate Y0 and Y2 and EH2/SV series MPU can designate Y0, Y2, Y4 and Y6.

5. The operation of **D₂** corresponds to the "+" or "-" of **S**. When **S** is "+", **D₂** will be On; when **S** is "-", **D₂** will be Off.

6. PLSV instruction does not have settings for acceleration and deceleration. Please use API 67 RAMP for the acceleration and deceleration of pulse output frequency.

7. During the pulse output executed by PLSV instruction, the drive contact turning Off will result in the immediate stop of the output without going through a deceleration.

8. When the absolute value of the input frequency during the execution of DPLSV is bigger than 200KHz, the output will operate at 200KHz.

9. For EH/EH2/SV series MPU, D1222, D1223, D1383 and D1384 are the time differences sent between the direction setup signal and pulse output points of CH0, CH1, CH2 and CH3.

10. For EH/EH2/SV series MPU, M1305, M1306, M1532 and M1533 are the flags of the direction signals of CH0, CH1, CH2 and CH3. When S is "+", the output will operate towards a forward direction and the flag will go Off. When S is "-", the output will operate towards a backward direction and the flag will go On.

Program Example:

When M10 = On, Y0 will output pulses at 20KHz. Y5 = On indicates forward pulses.

```
  M10
──┤├──────┌──────┬────────┬─────┬─────┐
          │ PLSV │ K20000 │ Y0  │ Y5  │
          └──────┴────────┴─────┴─────┘
```
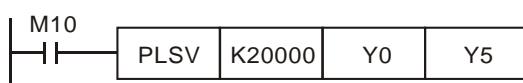
API	Mnemonic		Operands				Function	Controllers
158	D	DRVI	S₁	S₂	D₁	D₂	Drive to Increment	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DRVI: 9 steps
S₁					*	*	*	*	*	*	*	*	*	*	*	DDRVI: 17 steps
S₂					*	*	*	*	*	*	*	*	*	*	*	
D₁		*														
D₂		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Number of output pulses (relative designation)　　**S₂**: Pulse output frequency　　**D₁**: Pulse output device (please use transistor output module)　　**D₂**: Output device for the signal of rotation direction

Explanations:

1. See remarks for the setting range of **S₁**, **S₂**, **D₁** and **D₂**.

2. **S₁** and **S₂** of SC series MPU only support device K, H and D.

3. Flag: see remarks for more details.

4. **S₁** is the number of output pulses (relative designation). For EH/EH2/SV series MPU, the 16-bit instruction can designate the range -32,768 ~ +32,767. The range designated by 32-bit instruction is -2,147,483,648 ~ +2,147,483,647. For SC series MPU, the 32-bit instruction can designate the range -2,147,483,648 ~ +2,147,483,647. "+/-" signs indicate forward/backward directions.

5. **S₂** is the designated pulse output frequency. For EH/EH2/SV series MPU, the 16-bit instruction can designate its range 10 ~ 32,767Hz. The range designated by 32-bit instruction is 10 ~ 200,000Hz. For SC series MPU, the 32-bit instruction can designate the range 100 ~ 100,000Hz.

6. EH series MPU has two groups of A/B phase pulse output, CH0 (Y0, Y1) and CH1 (Y2, Y3). EH2/SV series MPU has four groups of A/B phase pulse output, CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks for the setup methods.

7. Pulse output device **D₁** in different models

Model	SC MPU	EH MPU	EH2/SV MPU
Pulse output end	Y10, Y11	Y0, Y2	Y0, Y2, Y4, Y6

8. The operation of **D₂** corresponds to the "+" or "-" of **S₁**. When **S₁** is "+", **D₂** will be On; when **S₁** is "-", **D₂** will be Off. **D₂** will not be Off immediately after the pulse output is over; it will be Off only when the drive contact of the instruction turns Off.

9. For EH/EH2/SV series MPU, **S₁** is

   - The 32-bit data stored in the present value registers D1337 (high word) and D1336 (low word) of CH0 (Y0, Y1).

   - The 32-bit data stored in the present value registers D1339 (high word) and D1338 (low word) of CH1 (Y2, Y3).

   - The 32-bit data stored in the present value registers D1376 (high word) and D1375 (low word) of CH2 (Y4, Y5).

   - The 32-bit data stored in the present value registers D1378 (high word) and D1377 (low word) of CH3 (Y5,

Y6).

- When in backward direction, the content in the present value register will decrease.

10. For SC series MPU, **S₁** is the 32-bit data stored in the present value registers D1348 (low word) and D1349 (high word) of CH0 (Y10) or the 32-bit data stored in the present value registers D1350 (low word) and D1351 (high word) of CH1 (Y11). When in backward direction, the content in the present value register will decrease. When the program goes from STOP to RUN or from RUN to STOP, the content in the present value register will remain unchanged.

11. When DRVI instruction is executing pulse output, you cannot change the content of all operands. The changes will be valid next time when DRVI instruction is enabled.

12. For EH/EH2/SV series MPU, when the drive contact of DRVI instruction is Off, even the indication flag M1336 sent by CH0 pulses, M1337 sent by CH1 pulses, M1522 sent by CH2 pulses and M1523 sent by CH3 pulses are "On", DRVI instruction will not be driven again.

13. When the absolute value of the input frequency of DDRVI insturction in EH/EH2/SV series MPU is larger than 200KHz, the output will be operated at 200KHz. When the absolute value of the input frequency is smaller than 10Hz, the output will be operated at 10Hz.

14. D1343 (D1353) is for setting up the time of the first acceleartion segment and last deceleration segment of CH0 (CH1). The acceleration and deceleration time of EH/EH2/SV series MPU shall not be shorter than 10ms. The output will be operated for 10ms if the time is shorter than 10ms or longer than 10,000ms (default setting = 100ms). The time range for SC series MPU is 50 ~ 20,000ms. The output will be operated for 20,000ms or 50ms if the time set is longer than 20,000ms or shorter than 50ms.

15. D1340 (D1352) is for setting up the start/end frequency of Y10 (Y11). If **S₂** is less than or equals start/end frequency, the pulse output frequency will be executed by the start/end frequency.

16. For EH/EH2/SV series MPU, M1305 (M1306) is the direction signal of CH0 (CH1). When **S₁** is a positive number, the output will be operated in a forward direction and M1305 (M1306) will be Off. When **S₁** is a negative number, the output will be operated in a backward direction and M1305 (M1306) will be On.

Program Example:

When M10= On, Y0 will output 20,000 pulses (relative designation) at 2KHz. Y5 = On indicates the pulses are executed in forward direction.

M10					
⊣⊢	DRVI	K20000	K2000	Y0	Y5
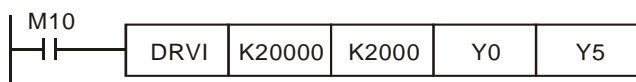
Remarks:

1. Explanations on EH/EH2/SV series MPU:

   a) Relative position control: Designating the traveling distance starting from the current position by "+/ -" signs; also known as a relative driving method.
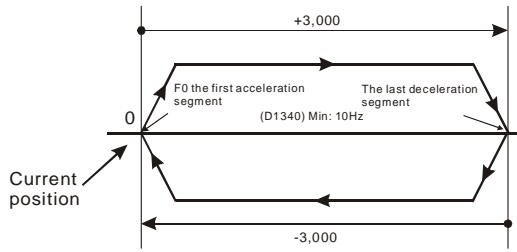
b) Settings of relative positioning and the acceleration/deceleration speed:
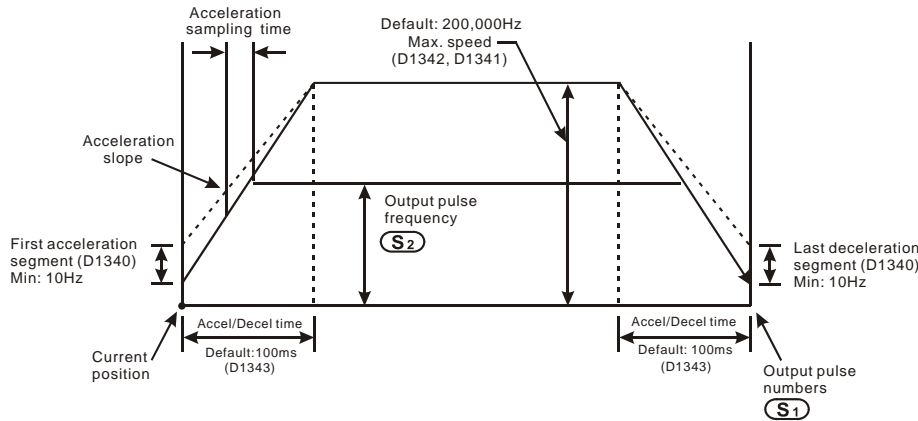
2. Explanations on SC series MPU:

a) Relative position control: Designating the traveling distance starting from the current position by "+/ -" signs; also known as a relative driving method.
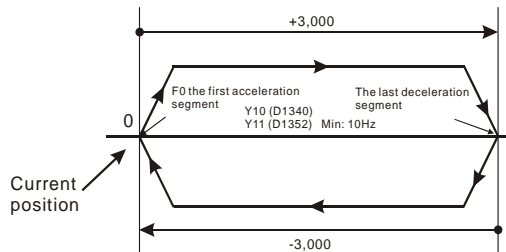
b) Settings of relative positioning and the acceleration/deceleration speed: D1343 (D1353) is for settings of the time of the first acceleration segment and last deceleration segment of Y10 (Y11). D1340 (D1352) is for settings of start/end frequency of Y10 (Y11).

c) Many DRVI instructions can be compiled synchronously in the program, but only one instruction can be activated whenever the PLC executes the program. For example, if Y10 output has already been activated by an instruction,   other instructions that are also used to activate Y10 output will not be excecuted. Therefore, the principle of the instruction activation sequence is "first activated, first executed".

d) When Y10 is activated by DDRVI instruction, the output function of Y10 will be disabled until DDRVI is OFF. The same rule applies to Y11.

e) Once the instruction is activated, all other parameters cannot be modified until the instruction is disabled.

f) When the instruction is disabled but the output has not yet completed:

M1334 = On indicates that Y10 will stop output immediately.

M1334 = Off indicates that Y10 will decelerate according to the deceleration time till it reaches end frequency and stop the pulse output.

M1335 corresponds to Y11 output and applies the same rule.

3. Flags for SC series MPU:

M1102： M1102 = On after Y10 pulse output is completed.

M1103： M1103 = On after Y11 pulse outout is completed.

M1334： Y10 pulse output stops immediately without deceleration when the pulse output instruction is disabled.

M1335： Y11 pulse output stops immediately without deceleration when the pulse output instruction is disabled.

4. Special registers for SC series MPU:

D1348： Low word of the current number of Y10 output pulses.

D1349： High word of the current number of Y10 output pulses.

D1350： Low word of the current number of Y11 output pulses.

D1351： High word of the current number of Y11 output pulses..

D1340： Settings of the first start frequency and the last end frequency of Y10 output pulses.

D1352： Settings of the first start frequency and the last end frequency of Y11 output pulses.

D1343： Settings of the acceleration/deceleration time of Y10 output pulses.

D1353： Settings of the acceleration/deceleration time of Y11 output pulses.

5. Flags for EH/EH2/SV series MPU:

M1010： For EH/EH2/SV, when M1010 = On, CH0, CH1, CH2 and CH3 will output pulses when END instruction is being executed. M1010 will be Off automatically when the output starts.

M1029： For EH/EH2/SV, M1029 = On after CH0 pulse output is completed.

M1030： For EH/EH2/SV, M1030 = On after CH1 pulse output is completed.

M1036： For EH2/SV, M1036 = On after CH2 pulse output is completed.

M1037： For EH2/SV, M1037 = On after CH3 pulse output is completed.

M1305： For EH/EH2/SV, direction signal of CH0.

M1306： For EH/EH2/SV, direction signal of CH1.

M1334： For EH, CH0 pulse output pauses.

For EH2/SV, CH0 pulse output stops.

M1335： For EH, CH1 pulse output pauses.

For EH2/SV, CH1 pulse output stops.

M1336： For EH/EH2/SV, "CH0 sends out pulses" indication.

M1337： For EH/EH2/SV, "CH1 sends out pulses" indication.

M1520： For EH2/SV, CH2 pulse output stops.

M1521： For EH2/SV, CH3 pulse output stops.

M1522： For EH2/SV, "CH2 sends out pulses" indication.

M1523： For EH2/SV, "CH3 sends out pulses" indication.

M1534： For EH2/SV, designated deceleration time of CH0 (should be used with D1348).

M1535： For EH2/SV, designated deceleration time of CH1 (should be used with D1349).

M1536： For EH2/SV, designated deceleration time of CH2 (should be used with D1350).

M1537： For EH2/SV, designated deceleration time of CH3 (should be used with D1351).

M1532： For EH2/SV, direction signal of CH2.

M1533： For EH2/SV, direction signal of CH3.

6. Special registers for EH/EH2/SV series MPU:

D1220： For EH/EH2/SV, phase setting of CH0 (Y0, Y1): D1220 determines the phase by the last two

bits; other bits are invalid.

1. K0: Y0 output
2. K1: Y0, Y1 AB-phase output; A ahead of B.
3. K2: Y0, Y1 AB-phase output; B ahead of A.
4. K3: Y1 output

D1221： For EH/EH2/SV, phase setting of CH1 (Y2, Y3): D1221 determines the phase by the last two

bits; other bits are invalid.

1. K0: Y2 output
2. K1: Y2, Y3 AB-phase output; A ahead of B.
3. K2: Y2, Y3 AB-phase output; B ahead of A.
4. K3: Y3 output

D1222： For EH/EH2/SV, the time difference between the direction signal and pulse output sent by

CH0.

D1223： For EH/EH2/SV, the time difference between the direction signal and pulse output sent by

CH1.

D1229： For EH2/SV, phase setting of CH2 (Y4, Y5): D1229 determines the phase by the last two bits;

other bits are invalid.

1. K0: Y4 output
2. K1: Y4, Y5 AB-phase output; A ahead of B.
3. K2: Y4, Y5 AB-phase output; B ahead of A.
4. K3: Y5 output

D1230： For EH2/SV, phase setting of CH3 (Y6, Y7): D1230 determines the phase by the last two bits;

other bits are invalid.

1. K0: Y6 output
2. K1: Y6, Y7 AB-phase output; A ahead of B.
3. K2: Y6, Y7 AB-phase output; B ahead of A.
4. K3: Y7 output

D1336： For EH/EH2/SV, low word of the current number of output pulses from CH0.

D1337： For EH/EH2/SV, high word of the current number of output pulses from CH0.

D1338： For EH/EH2/SV, low word of the current number of output pulses from CH1.

D1339： For EH/EH2/SV, high word of the current number of output pulses from CH1.

D1340： For EH/EH2/SV, settings of the first start frequency and the last end frequency of CH0.

D1343： For EH/EH2/SV, settings of acceleration/deceleration time for CH0 pulse output.

D1352： For EH/EH2/SV, settings of the first start frequency and the last end frequency of CH1.

D1353： For EH/EH2/SV, settings of acceleration/deceleration time for CH1 pulse output.

D1375： For EH2/SV, low word of the current number of output pulses from CH2.

D1376： For EH2/SV, high word of the current number of output pulses from CH2.

D1377： For EH2/SV, low word of the current number of output pulses from CH3.

D1378： For EH2/SV, high word of the current number of output pulses from CH3.

D1379： For EH2/SV, settings of the first start frequency and the last end frequency of CH2.

D1380： For EH2/SV, settings of the first start frequency and the last end frequency of CH3.

D1348： For EH2/SV, deceleration time for CH0 pulse output when M1534 = On.

D1349： For EH2/SV, deceleration time for CH1 pulse output when M1535 = On.

D1350： For EH2/SV, deceleration time for CH2 pulse output when M1536 = On.

D1351： For EH2/SV, deceleration time for CH3 pulse output when M1537 = On.

D1381： For EH2/SV, settings of acceleration/deceleration time for CH2 pulse output.

D1382： For EH2/SV, settings of acceleration/deceleration time for CH3 pulse output.

D1383： For EH2/SV, the time difference between the direction signal and pulse output sent by CH2.

D1384： For EH2/SV, the time difference between the direction signal and pulse output sent by CH3.

API	Mnemonic		Operands				Function		Controllers	
159	D	DRVA	S_1 S_2 D_1 D_2				Drive to Absolute		ES/EX/SS **SA/SX/SC EH/SV**	

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DRVA: 9 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DDRVA: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D_1		*														
D_2		*	*	*												

	PULSE	16-bit	32-bit
	ES|EX|SS|SA|SX|SC|EH|SV	ES|EX|SS|SA|SX|SC|**EH**|**SV**	ES|EX|SS|SA|SX|**SC**|**EH**|**SV**

Operands:

S_1: Number of output pulses (absolute designation)   **S_2**: Pulse output frequency   **D_1**: Pulse output device (please use transistor output module)   **D_2**: Output device for the signal of rotation direction

Explanations:

1. See remarks for the setting range of **S_1**, **S_2**, **D_1** and **D_2**.

2. **S_1** and **S_2** of SC series MPU only support device K, H and D.

3. Flag: see remarks of API 158 DRVI for more details.

4. **S_1** is the number of output pulses (absolute designation). For EH/EH2/SV series MPU, the 16-bit instruction can designate the range -32,768 ~ +32,767. The range designated by 32-bit instruction is -2,147,483,648 ~ +2,147,483,647. For SC series MPU, the 32-bit instruction can designate the range -2,147,483,648 ~ +2,147,483,647. "+/-" signs indicate forward/backward directions.

5. **S_2** is the designated pulse output frequency. For EH/EH2/SV series MPU, the 16-bit instruction can designate its range 10 ~ 32,767Hz. The range designated by 32-bit instruction is 10 ~ 200,000Hz. For SC series MPU, the 32-bit instruction can designate the range 100 ~ 100,000Hz.

6. EH series MPU has two groups of A/B phase pulse output, CH0 (Y0, Y1) and CH1 (Y2, Y3). EH2/SV series MPU has four groups of A/B phase pulse output, CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks for the setup methods.

7. Pulse output device **D_1** in different models

Model	SC MPU	EH MPU	EH2/SV MPU
Pulse output end	Y10, Y11	Y0, Y2	Y0, Y2, Y4, Y6

8. When **S_1** is larger than the current relative position, **D_2** will be Off; when **S_1** is smaller than the current relative position, **D_2** will be On. **D_2** will not be Off immediately after the pulse output is over; it will be Off only when the drive contact of the instruction turns Off.

9. For EH/EH2/SV series MPU, **S_1** is

    -   The 32-bit data stored in the present value registers D1337 (high word) and D1336 (low word) of CH0 (Y0, Y1).

    -   The 32-bit data stored in the present value registers D1339 (high word) and D1338 (low word) of CH1 (Y2, Y3).

    -   The 32-bit data stored in the present value registers D1376 (high word) and D1375 (low word) of CH2 (Y4, Y5).

    -   The 32-bit data stored in the present value registers D1378 (high word) and D1377 (low word) of CH3 (Y5,

Y6).

When in backward direction, the content in the present value register will decrease.

10. For SC series MPU, S_1 is the 32-bit data stored in the present value registers D1348 (low word) and D1349 (high word) of CH0 (Y10) or the 32-bit data stored in the present value registers D1350 (low word) and D1351 (high word) of CH1 (Y11). When in backward direction, the content in the present value register will decrease. When the program goes from STOP to RUN or from RUN to STOP, the content in the present value register will remain unchanged.

11. For EH/EH2/SV series MPU, when DRVA instruction is executing pulse output, you cannot change the content of all operands. The changes will be valid next time when DRVA instruction is enabled.

12. For EH/EH2/SV series MPU, when the drive contact of DRVA instruction is Off, the pulse output will decelerate to stop and M1029 and M1030 will be enabled. For SC series MPU, the pulse output will decelerate to stop and M1102 and M1103 will be enabled.

13. For EH/EH2/SV series MPU, when the drive contact of DRVA instruction is Off, even the indication flag M1336 sent by CH0 pulses or M1337 sent by CH1 pulses are "On", DRVA instruction will not be driven again.

14. When the absolute value of the input frequency of DRVA and DDRVA insturctions in EH/EH2/SV series MPU is larger than 200KHz, the output will be operated at 200KHz. When the absolute value of the input frequency is smaller than 10Hz, the output will be operated at 10Hz.

15. D1343 (D1353) is for setting up the time of the first acceleartion segment and last deceleration segment of CH0 (CH1). The acceleration and deceleration time of EH/EH2/SV series MPU shall not be shorter than 10ms. The output will be operated for 10ms if the time is shorter than 10ms or for 100ms (default) if the time is longer than 10,000ms. The time range for SC series MPU is 50 ~ 20,000ms. The output will be operated for 20,000ms or 50ms if the time set is longer than 20,000ms or shorter than 50ms.

16. For EH/EH2/SV series MPU, M1305 (M1306) is the direction signal of CH0 (CH1). When S_1 is a positive number, the output will be operated in a forward direction and M1305 (M1306) will be Off. When S_1 is a negative number, the output will be operated in a backward direction and M1305 (M1306) will be On.

17. D1340 (D1352) is for setting up the start/end frequency of Y10 (Y11). If S_2 is less than or equals start/end frequency, the pulse output frequency will be executed by the start/end frequency.

Program Example:

When M10= On, Y0 will output 20,000 pulses (absolute designation) at 2KHz. Y5 = On indicates the pulses are executed in forward direction.

M10					
┤├	DRVA	K20000	K2000	Y0	Y5
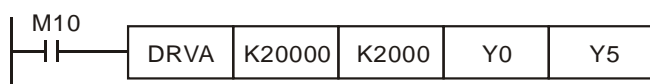
Remarks:

1. Explanations on EH/EH2/SV series MPU:

   a) Absolute position control: Designating the traveling distance starting from the zero point (0); also known as a absolute driving method.
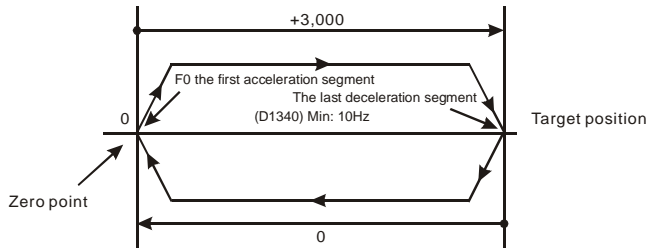
b) Settings of absolute positioning and the acceleration/deceleration speed:
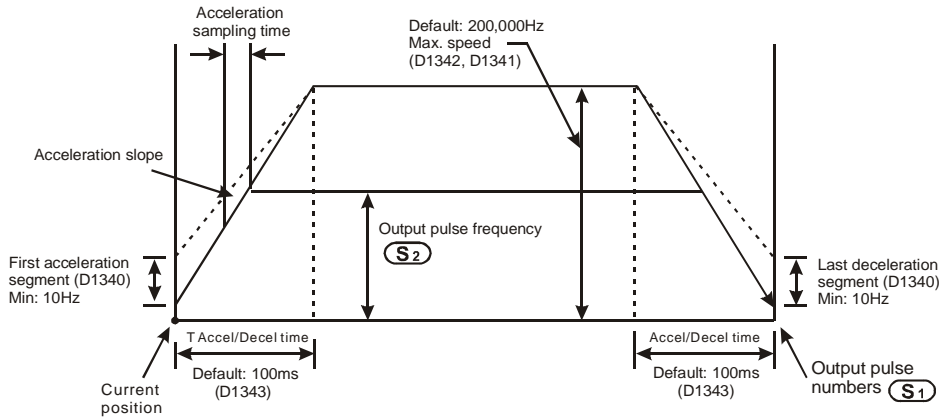
2. Explanations on SC series MPU:

a) Absolute position control: Designating the traveling distance starting from the zero point (0); also known as a absolute driving method.
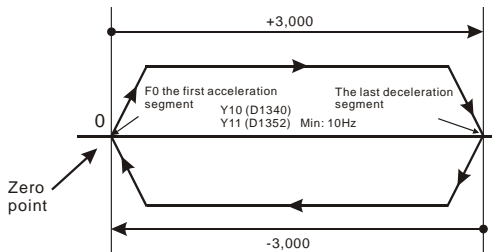
b) Settings of absolute positioning and the acceleration/deceleration speed: D1343 (D1353) is for settings of the time of the first acceleration segment and last deceleration segment of Y10 (Y11). D1340 (D1352) is for settings of start/end frequency of Y10 (Y11).
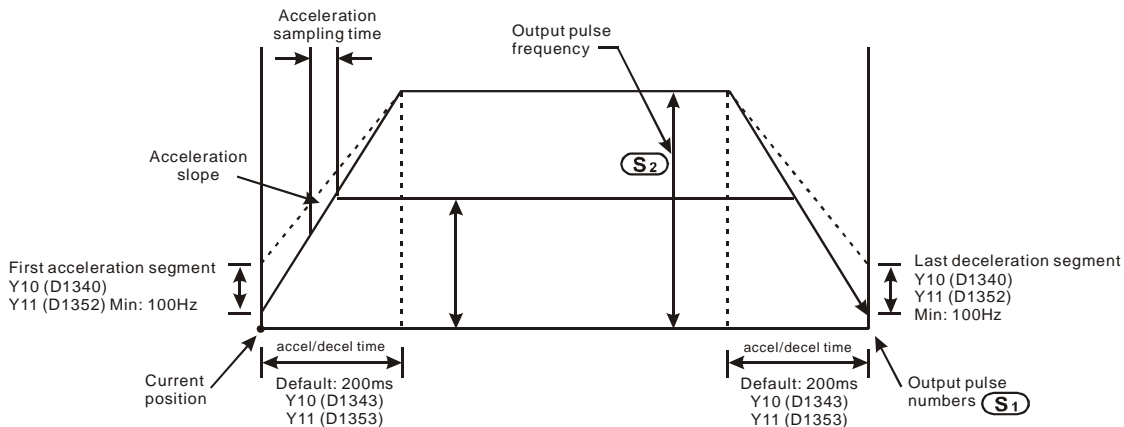
c) Many DRVA instructions can be compiled synchronously in the program, but only one instruction can be activated whenever the PLC executes the program. For example, if Y10 output has already been activated by an instruction, other instructions that are also used to activate Y10 output will not be excecuted. Therefore, the principle of the instruction activation sequence is "first activated, first executed".

d) When Y10 is activated by DDRVA instruction, the output function of Y10 will be disabled until DDRVAis OFF. The same rule applies to Y11.

e) Once the instruction is activated, all other parameters cannot be modified until the instruction is disabled.

f) When the instruction is disabled but the output has not yet completed:

M1334 = On indicates that Y10 will stop output immediately.

M1334 = Off indicates that Y10 will decelerate according to the deceleration time till it reaches end frequency and stop the pulse output.

M1335 corresponds to Y11 output and applies the same rule.

3. See remarks of DDRVI instruction for more details on the flags.

4. Wiring of DVP-EH series and Delta ASDA servo drive:
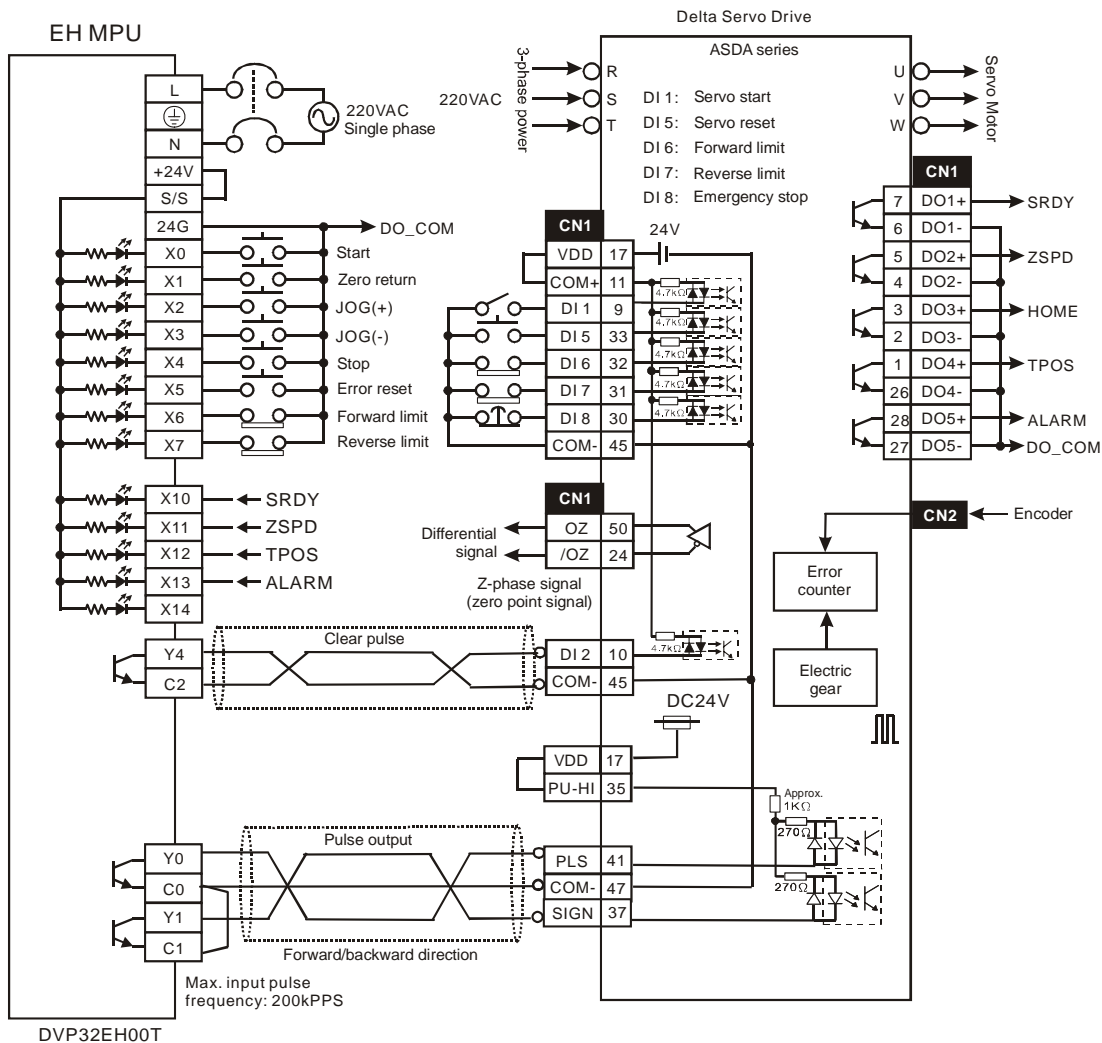
DVP32EH00T

Note:

(a) The parameter setting of Delta ASDA servo drive:

P1-01: position mode

P1-00: pulse input type as Pulse+DIR.

(b)   The forward/reverse limit switch should be connected to SERVO AMP.

(c)   The "clear pulse" signal will clear the current number of pulses left inside the servo.

5.   Wiring of DVP-SC series and Delta ASDA servo drive:
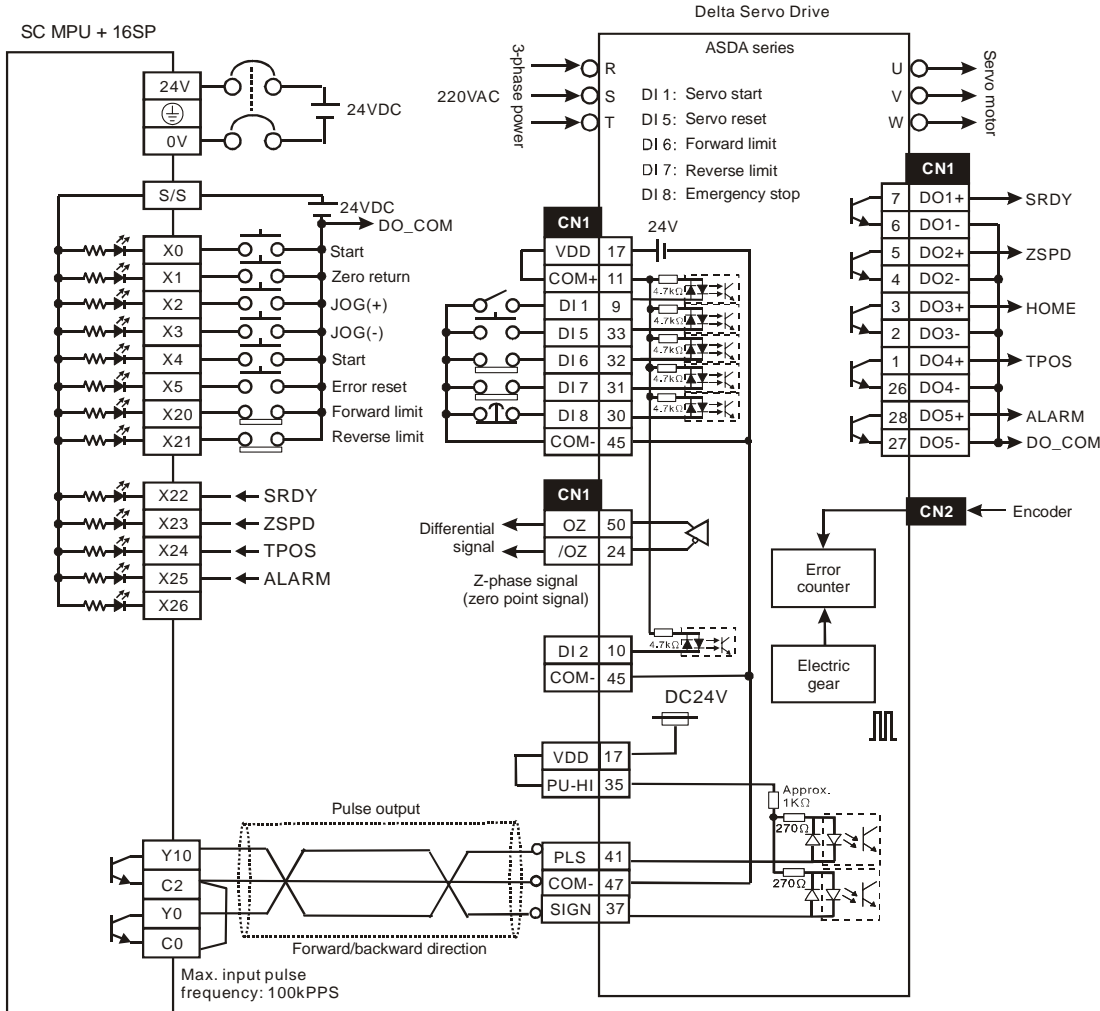
DVP12SC11T+DVP16SP11T

Note:

(a)   The parameter setting of Delta ASDA servo drive:

P1-01: position mode

P1-00: pulse input type as Pulse+DIR.

(b)   The forward/reverse limit switch should be connected to SERVO AMP.

6. Wiring of DVP-EH series PLC and a Mitsubishi MR-J2-☐A Servo drive:
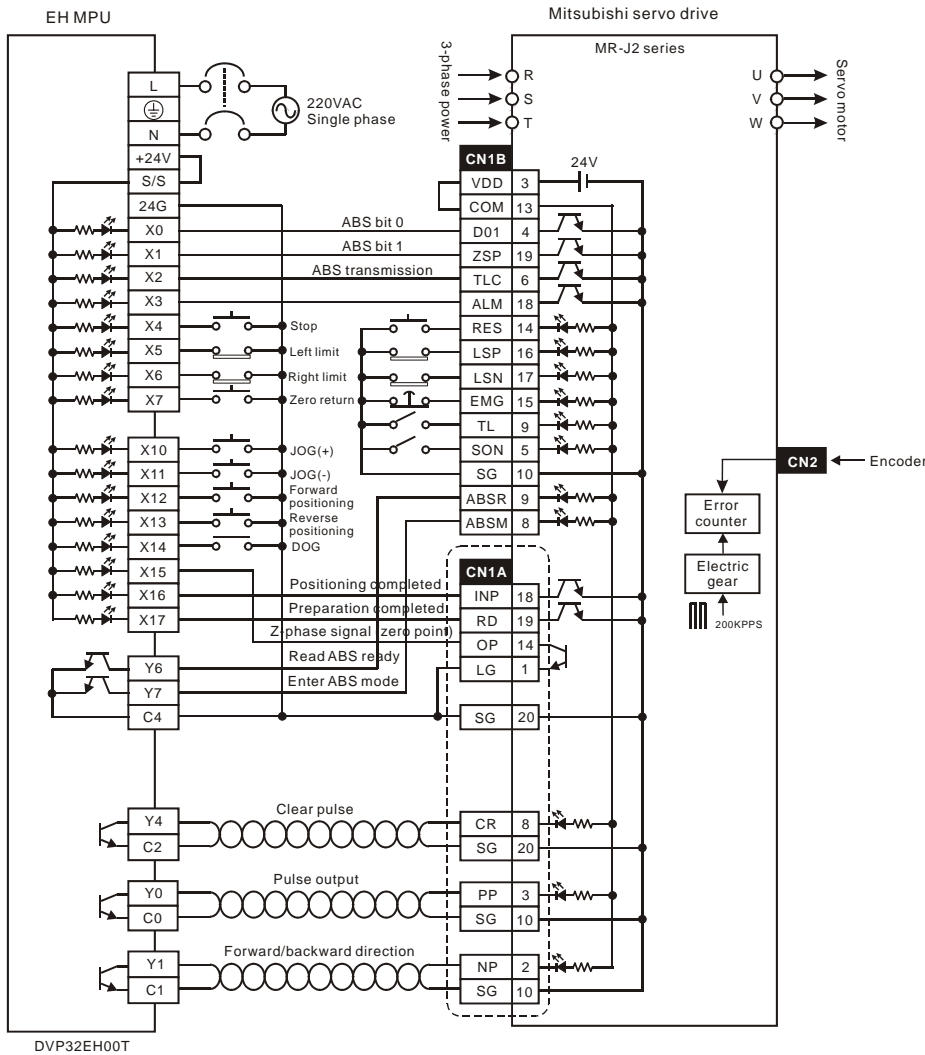
Note:

(a) When detecting an absolute position by using DABSR instruction, the parameter setting of a Mitsubishi MR-J2-☐A servo drive that connects to Delta EH series PLC:

P0: position mode.

P1: using absolute value.

P21: pulse input type as Pulse+DIR.

(b) The forward/reverse limit switch should be connected to SERVO AMP.

(c) When using OP (Z-phase signal) in servo and given that the Z-phase signal is a high-frequency one when the motor is running at high speed, the valid detection can only be possible when the signal is within the range detectable by PLC. When using OP (Z phase signal) of the servo, if Z phase signal is a high frequency signal during high-speed motor operation, the high frequency signal shall be within the available range that can be detected by PLC.

7. Cautions when designing a position control program:

a) There is no limitation on the times of using the position control instructions, API 156 ZRN, API 157 PLSV, API 158 DRVI, and API 159 DRVA. However, the user still have to note that:

i. Do not drive the position control instructions which use the same output CH0(Y0, Y1) or CH1(Y2, Y3) simultaneously. Otherwise, they will be treated as repeated outputs and cannot function normally.

ii. It is recommended that you use the step ladder instruction (STL) to design the position control program (see the example below).

b) How to use the position control instructions (API 156 ABSR, API 157 PLSV, API 158 DRVI, and API 159 DRVA) and pulse output instructions (API 57 PLSY, API 58 PWM and API 59 PLSR) at the same time.

The position control instruction and pulse output instruction share the 32 bits of the present value register (D1337 high word; D1336 low word) of CH0 (Y0, Y1) or the present value register of CH1 (Y2, Y3), which will make the operation complicated. Therefore, it is recommanded that you replace the pulse output instruction with position control instruction.

c) Explanations on the (Y0, Y1) pulses from CH0 and (Y2, Y3) pulses from CH1.

Voltage range: DC5V ~ DC24V

Current range: 10mA ~ 100mA

Output pulse frequency: Y0, Y2 at 200KHz; Y1, Y3 at 10KHz.

8. Settings of pulse output signals in the operation of position control for EH/EH2/SV series MPU:

a) Pulse + DIR (recommended)
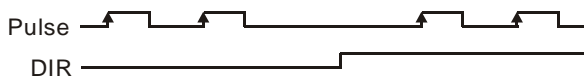
b) CW/CCW (limited frequency at 10KHz)
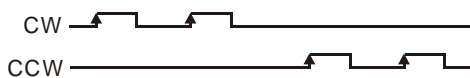
c) A/B-phase output (limited frequency at 10KHz)
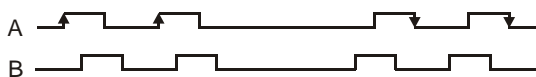
9. Follow the above output settings of PLC for the pulse input parameters of SERVO AMP or stepping motor.

10. For EH/EH2/SV series MPU, when Y0 output adopts many high-speed pulse output instructions (PLSY, PWM, PLSR) and position control instructions (ZRN, PLSV, DRVI, DRVA) in a program and these instructions are executed synchronously in the same scan period, PLC will execute the instruction with the fewest step numbers.
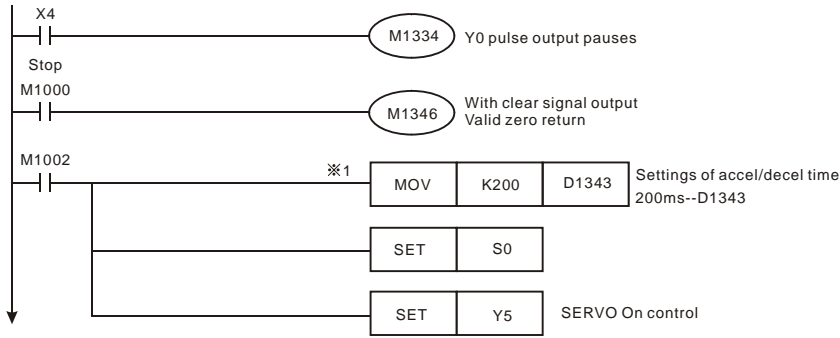
Programming example for forward/reverse operation:

For the wiring, see the wiring drawing of DVP-EH series and Mitsubishi MR-J2-□A servo drive
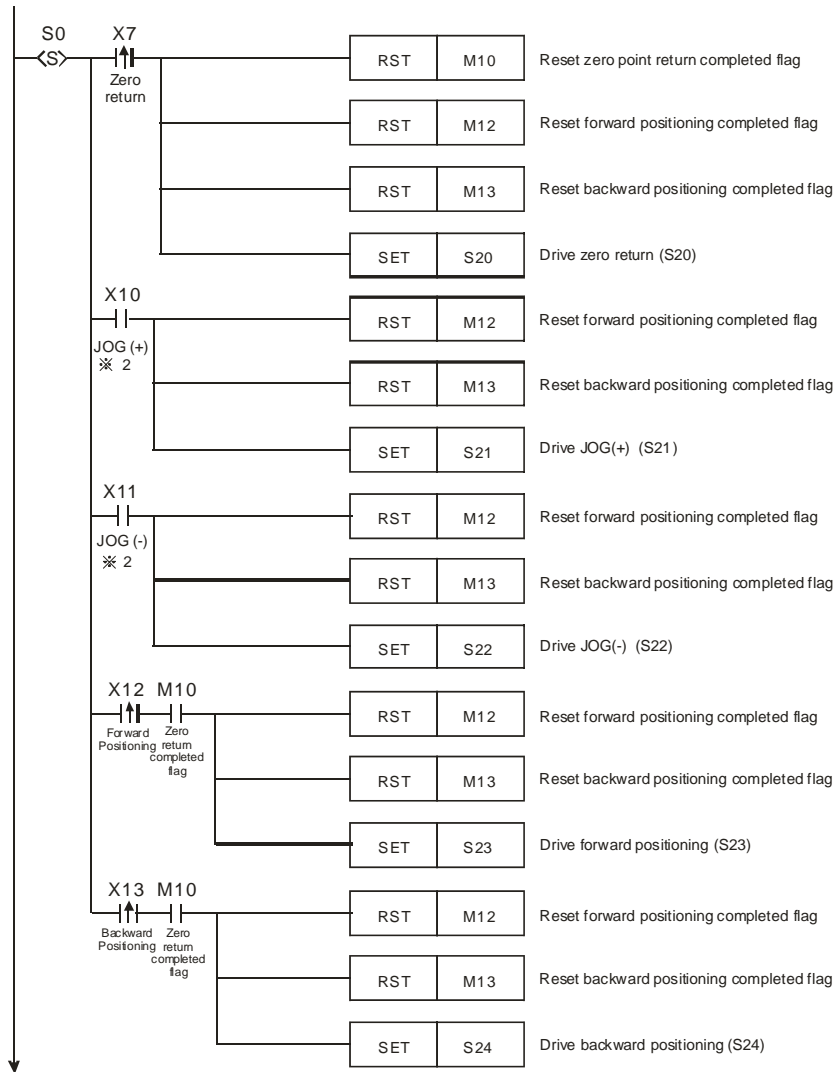
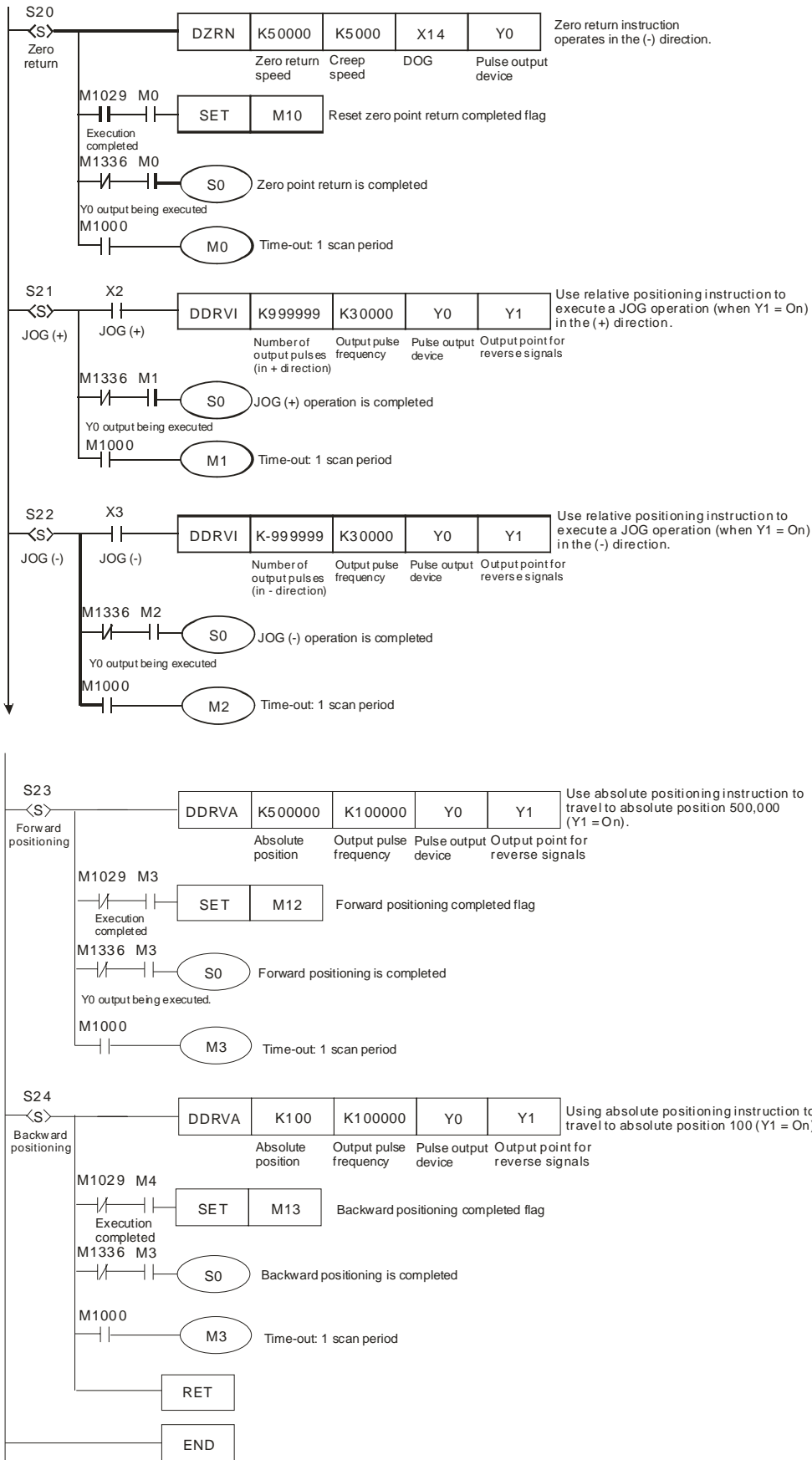One operation mode performs positioning by absolute position:
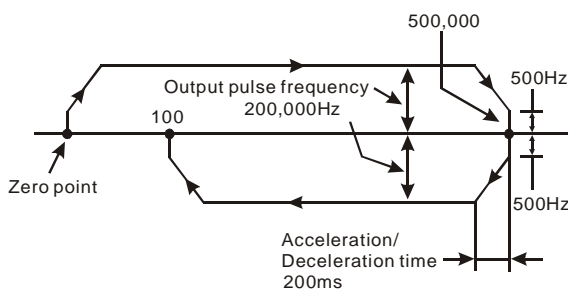
11. Programming example of using step ladder instruction (STL):

※1. If the accel./decel. time (D1343) of CH1 can be default setting, (100ms) this program step can be ignored.

※2. The max. traveling distance of a JOG operation equals to the max. number of output pulses (-2,147,483,648 ~ +2,147,483,647) of API 158 DDRVI instruction. Please re-execute JOG of the traveling distance exceeds the range.

```
S20                DZRN  K50000  K5000   X14    Y0      Zero return instruction
<S>                                                      operates in the (-) direction.
Zero
return                     Zero return  Creep   DOG   Pulse output
                           speed        speed          device

       M1029 M0    SET    M10    Reset zero point return completed flag
       ┤├──┤├

       Execution
       completed
       M1336 M0    S0     Zero point return is completed
       ┤├──┤├

       Y0 output being executed
       M1000       M0     Time-out: 1 scan period
       ┤├


S21       X2       DDRVI  K999999  K30000  Y0    Y1      Use relative positioning instruction to
<S>       ┤├                                             execute a JOG operation (when Y1 = On)
JOG (+)  JOG (+)            Number of  Output pulse Pulse output Output point for   in the (+) direction.
                           output pulses frequency  device    reverse signals
                           (in + direction)

       M1336 M1    S0     JOG (+) operation is completed
       ┤├──┤├

       Y0 output being executed
       M1000       M1     Time-out: 1 scan period
       ┤├


S22       X3       DDRVI  K-999999  K30000  Y0    Y1     Use relative positioning instruction to
<S>       ┤├                                             execute a JOG operation (when Y1 = On)
JOG (-)  JOG (-)            Number of  Output pulse Pulse output Output point for   in the (-) direction.
                           output pulses frequency  device    reverse signals
                           (in - direction)

       M1336 M2    S0     JOG (-) operation is completed
       ┤/├──┤├

       Y0 output being executed
       M1000       M2     Time-out: 1 scan period
       ┤├


S23                DDRVA  K500000  K100000  Y0    Y1     Use absolute positioning instruction to
<S>                                                      travel to absolute position 500,000
Forward                    Absolute   Output pulse Pulse output Output point for   (Y1 = On).
positioning                position   frequency   device    reverse signals

       M1029 M3    SET    M12    Forward positioning completed flag
       ┤/├──┤├

       Execution
       completed
       M1336 M3    S0     Forward positioning is completed
       ┤/├──┤├

       Y0 output being executed.
       M1000       M3     Time-out: 1 scan period
       ┤├


S24                DDRVA  K100    K100000  Y0    Y1      Using absolute positioning instruction to
<S>                                                      travel to absolute position 100 (Y1 = On).
Backward                   Absolute   Output pulse Pulse output Output point for
positioning                position   frequency   device    reverse signals

       M1029 M4    SET    M13    Backward positioning completed flag
       ┤/├──┤├

       Execution
       completed
       M1336 M3    S0     Backward positioning is completed
       ┤/├──┤├

       M1000       M3     Time-out: 1 scan period
       ┤├

                   RET

                   END
```

API	Mnemonic		Operands					Function	Controllers
160	TCMP	P	S_1 S_2 S_3 S D					Time Compare	ES/EX/SS **SA/SX/SC** EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: "Hour" for comparison   **S₂**: "Minute" for comparison   **S₃**: "Second" for comparison   **S**: Current time of RTC   **D**: Comparison result

Explanations:

1. Range of **S₁**: K0 ~ K23; range of **S₂** and **S₃**: K0 ~ K59

2. **S** will occupy 3 consecutive devices; **D** will occupy 3 consecutive points.

3. See the specifications of each model for their range of use.

4. **S₁**, **S₂** and **S₃** are compared with the present values of "hour", "minute" and "second" starting from **S**. The comparison result is stored in **D**.

5. **S** is the "hour" of the current time (K0 ~ K23) in RTC; **S** + 1 is the "minute" (K0 ~ K59) and **S** + 2 is the "second" (K0 ~ K59).

6. **S** is read by TRD instruction and the comparison is started by TCMP instruction. If **S** exceeds the range, the program will regard this as an operation error and the instruction will not be executed, M1067 and M1068 = On and D1067 will record the error code 0E1A (hex).

Program Example:

1. When X10= On, the instruction will compare the current time in RTC (D20 ~ D22) with the set value 12:20:45 and display the result in M10 ~ M12. When X10 goes from On to Off, the instruction will not be executed, but the On/Off stauts prior to M10 ~ M12 will remain.

2. Connect M10 ~ M12 in series or in parallel to obtain the result of $\geq$, $\leq$, and $\neq$.
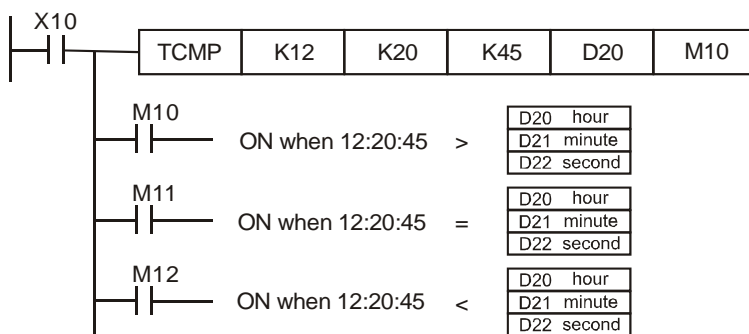
API	Mnemonic		Operands				Function	Controllers
161	TZCP	P	S₁ S₂ S D				Time Zone Compare	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TZCP, TZCPP: 9 steps
S₁											*	*	*			
S₂											*	*	*			
S											*	*	*			
D		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Lower bound of the time for comparison   **S₂**: Upper bound of the time for comparison   **S**: Current time of RTC   **D**: Comparison result

Explanations:

1. **S₁**, **S₂**, and **S** will occupy 3 consecutive devices.

2. The content in **S₁** must be less than the content in **S₂**.

3. **D** will occupy 3 consecutive points.

4. See the specifications of each model for their range of use.

5. **S** is compared with **S₁** and **S₂**. The comparsion result is stored in **D**.

6. **S₁** , **S₁** +1, **S₁** +2: The "hour", "minute" and "second" of the lower bound of the time for comparison.

7. **S₂**, **S₂** +1, **S₂** +2: The "hour", "minute" and "second"ond" of the upper bound of the time for comparison.

8. **S** , **S** +1, **S** +2: The "hour", "minute" and "second" of the current time of RTC.

9. D0 designated by **S** is read by TRD instruction and the comparison is started by TZCP instruction. If **S₁**, **S₂**, and **S** exceed their ranges, the program will regard this as an operation error and the instruction will not be executed, M1067 and M1068 = On and D1067 will record the error code 0E1A (hex).

10. When **S** < **S₁** and **S** < **S₂**, **D** will be On.   When **S** > **S₁** and **S** > **S₂**, **D** + 2 will be On. In other occasions, **D** + 1 will be On.

Program Example:

When X10= On, TZCP instruction will be executed and one of M10 ~ M12 will be On. When X10 = Off, TZCP instruction will not be executed and the status of M10 ~ M12 prior to X10 = Off will remain unchanged.

API		Mnemonic		Operands			Function		Controllers
162		TADD	P	S₁ S₂ D			Time Addition		ES/EX/SS **SA/SX/SC** EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TADD, TADDP: 7 steps
S₁											*	*	*			
S₂											*	*	*			
D											*	*	*			

PULSE	16-bit	32-bit
ES EX SS **SA SX SC** EH SV	ES EX SS **SA SX SC** EH SV	ES EX SS SA SX SC EH SV

Operands:

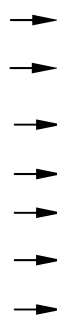**S₁**: Time summand   **S₂**: Time addend   **D**: Time sum

Explanations:

1. **S₁**, **S₂**, and **D** will occupy 3 consecutive devices.

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag); M1022 (carry flag)

4. **S₁** + **S₂** = **D**. The hour, minute, and second of the RTC designated in **S₁** plus the hour, minute, and second designated in **S₂**. The result is stored in the hour, minute, and second of the register designated in **D**.

5. If **S₁** and **S₂** exceed their ranges, the program will regard this as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 record the error code 0E1A (hex).

6. If the sum 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".

7. If the sum equals 0 (00:00:00), the zero flag M1020 will be On.

Program Example:

1. When X10= On, TADD instruction will be executed and the hour, minute and second in RTC designated in D0 ~ D2 will plus the hour, minute and second in RTC designated in D10 ~ D12. The sum is stored in the hour, minute and second of the register designated in D20 ~ D22.

```
    X10
  ─┤ ├──────┬───────┬───────┬───────┐
              │ TADD  │  D0   │  D10  │  D20  │
              └───────┴───────┴───────┴───────┘

  ┌──────────────┐     ┌──────────────┐     ┌──────────────┐
  │ D0   8 (hours)│     │ D10  6 (hours)│     │ D20 14(hours)│
  │ D1  10 (mins) │  +  │ D11 40 (mins) │ →   │ D21 50 (mins)│
  │ D2  20 (secs) │     │ D12  6 (secs) │     │ D22 26 (secs)│
  └──────────────┘     └──────────────┘     └──────────────┘
     08:10:20             06:40:06             14:50:26
```
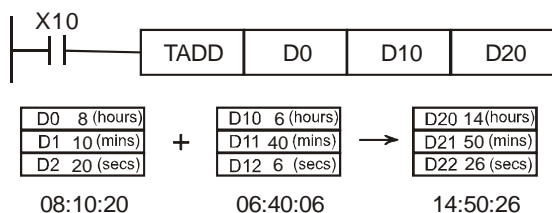
2. If the sum is larger than 24 hours, M1022 will be On.

```
    X10
  ─┤ ├──────┬───────┬───────┬───────┐
              │ TADD  │  D0   │  D10  │  D20  │
              └───────┴───────┴───────┴───────┘

  ┌──────────────┐     ┌──────────────┐     ┌──────────────┐
  │ D0  18 (hours)│     │ D10 11(hours) │     │ D20  6 (hours)│
  │ D1  40 (mins) │  +  │ D11 30 (mins) │ →   │ D21 10 (mins)│
  │ D2  30 (secs) │     │ D12  8 (secs) │     │ D22 38 (secs)│
  └──────────────┘     └──────────────┘     └──────────────┘
     18:40:30             11:30:08             06:10:38
```
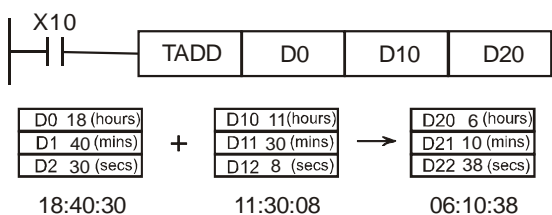
API	Mnemonic		Operands			Function			Controllers
163	TSUB	P	**S₁** **S₂** **D**			Time Subtraction			ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TSUB, TSUBP: 7 steps
S₁											*	*	*			
S₂											*	*	*			
D											*	*	*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Time minuend **S₂**: Time subtrahend **D**: Time remainder

Explanations:

1. **S₁**, **S₂**, and **D** will occupy 3 consecutive devices.

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag); M1021 (borrow flag)

4. **S₁** − **S₂** = **D**. The hour, minute, and second of the RTC designated in **S₁** minus the hour, minute, and second designated in **S₂**. The result is stored in the hour, minute, and second of the register designated in **D**.

5. If **S₁** and **S₂** exceed their ranges, the program will regard this as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 record the error code 0E1A (hex).

6. If the remainder is a negative value, the borrow flag M1021 will be On. The value in **D** will be the result of "the negative value pluses 24 hours".

7. If the remainder equals 0 (00:00:00), the zero flag M1020 will be On.

Program Example:

1. When X10= On, TADD instruction will be executed and the hour, minute and second in RTC designated in D0 ~ D2 will minus the hour, minute and second in RTC designated in D10 ~ D12. The remainder is stored in the hour, minute and second of the register designated in D20 ~ D22.
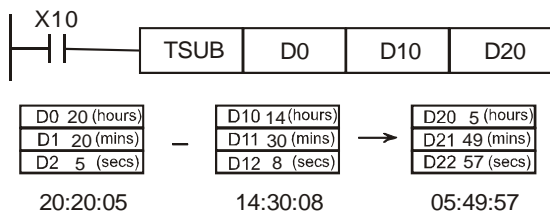
2. If the subtraction result is a negative value, M1021 will be On.
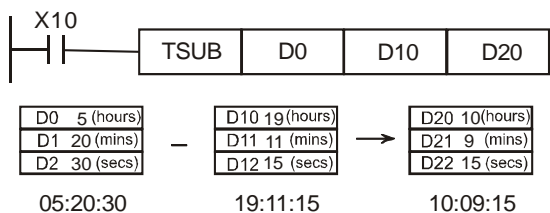
API	Mnemonic		Operands	Function	Controllers
166	TRD	P	**D**	Time Read	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

D: The device for storing the current time read in RTC

Explanations:

1. **D** will occupy 7 consecutive devices.

2. See the specifications of each model for their range of use.

3. Flags: M1016, M1017, M1076. See remarks for more details.

4. The built-in RTC in EH/EH2/SV/SA/SX/SC series MPU offers 7 data (year, week, month, day, hour, minute, second) stored in D1319 ~ D1313. TRD instruction is for program designers to read the current data in RTC and store the data to the 7 registers designated.

5. D1319 only reads the 2-digit year in A.D. If you wish D1319 to read the 4-digit year, see remarks for more information.

Program Example:

1. When X0 = On, the instruction will read the current time in RTC to the designated registers D0 ~ D6.

2. The content of D1318: 1 = Monday; 2 = Tuesday … 7 = Sunday.

```
      X0
    ──┤├──────┌─────┬─────┐
              │ TRD │ D0  │
              └─────┴─────┘
```

Special D	Item	Content		General 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	Date
D1315	Hour	0~23	→	D4	Hour
D1314	Minute	0~59	→	D5	Minute
D1313	Second	0~59	→	D6	Second

Remarks:

1. Flags and special registers for the built-in RTC in EH/EH2/SV/SA/SX/SC series MPU.

Device	Name	Function
M1016	Displaying year in A.D. in RTC	When Off, D1319 will display 2-digit year in A.D. When On, D1319 will display "2-digit year in A.D + 2,000".
M1017	±30 seconds correction	Correction takes place when M1017 goes from Off to On (reset to 0 when in 0 ~ 29 second; minute pluses 1 and second resets to 0 in 30 ~ 59 second)
M1076	Malfunction of RTC	On when the set value exceeds the range. (only available when the power is being switched on).
D1313	Second	0 ~ 59
D1314	Minue	0 ~ 59
D1315	Hour	0 ~ 23
D1316	Day	1 ~ 31
D1317	Month	1 ~ 12
D1318	Week	1 ~ 7
D1319	Year	0 ~ 99 (2-digit year in A.D.)

2. How to correct RTC:

    There are 2 ways to correct the built-in RTC.

    a) By a specific instruction. (See API 167 TWR instruction)

    b) By peripheral devices, WPLSoft, the ladder diagram editing software.

3. How to display 4-digit year in A.D.:

    a) Normally, the year is only displayed in 2 digits (e.g. 2003 displayed as 03). If you wish the year to be displayed in 4 digits, please key in the following program at the start of the program.

    b) The original 2-digit year will be switched to a 4-digit year, i.e. the 2-digit year will pluses 2,000.

    c) If you wish to write in new time in the 4-digit year display mode, you can only write in a 2-digit year (0 ~ 99, indicating year 2000 ~ 2099). For example, 00=year 2000, 50=year 2050 and 99=year 2099.

API	Mnemonic		Operands	Function		Controllers
167	TWR	P	Ⓢ	Time Write		ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Device for storing the new time to be written into RTC

Explanations:

1. **S** will occupy 7 consecutive devices.

2. See the specifications of each model for their range of use.

3. Flags: M1016, M1017, M1076. See remarks of API 166 TRD for more details.

4. To make adjustment on the RTC built in EH/EH2/SV/SA/SX/SC series MPU, use this instruction to write the correct time into the RTC.

5. When this instruction is executed, the new set time will be written in the RTC built in PLC immediately. Therefore, please be noted that the new set time has to match the current time then when the instruction is executed.

6. If **S** exceeds its range, the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code 0E1A (hex).

Program Example 1:

When X0= On, write the correct current time into the RTC.

```
   X0
 ──┤├──┤ TWRP │ D20 │
```
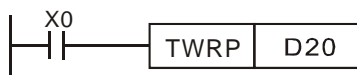
General D	Item	Content		Special D	Item	
D20	Year (A.D.)	00~99	→	D1319	Year (A.D.)	
D21	Day (Mon ~ Sun)	1~7	→	D1318	Day (Mon ~ Sun)	
D22	Month	1~12	→	D1317	Month	Real Time Clock
D23	Date	1~31	→	D1316	Date	
D24	Hour	0~23	→	D1315	Hour	
D25	Minute	0~59	→	D1314	Minute	
D26	Second	0~59	→	D1313	Second	

New set time (label for General D column)

Program Example 2:

1. Set the current time in the RTC as 15:27:30, Tuesday, August 19, 2003.

2. D0 ~ D6 indicate the new set time in the RTC.

3. X10 = On for changing the current time in the RTC and make the changed value the new set value.

4. Whenever X11 = On, RTC will perform a ±30 second correction. The correction is performed according to the rules: When the second hand of RTC locates at 1 ~ 29, the second will be automatically reset to "0" and the minute hand will remain at its location. When the second hand locates at 30 ~ 59, the second will be automatically reset to "0" and the minute hand will increase by 1 minute.

```
      X10
      ─┤↑├──┬──── MOV  K03  D0 │ Year (2003)
            │
            ├──── MOV  K2   D1 │ Day (Tuesday)
            │
            ├──── MOV  K8   D2 │ Month (August)
            │
            ├──── MOV  K19  D3 │ Date
            │
            ├──── MOV  K15  D4 │ Hour
            │
            ├──── MOV  K27  D5 │ Minute
            │
            ├──── MOV  K30  D6 │ Second
            │
            └──── TWR  D0      │ Write the set time into RTC
      X11
      ─┤ ├──────────(M1017)      ± 30 seconds correction
```
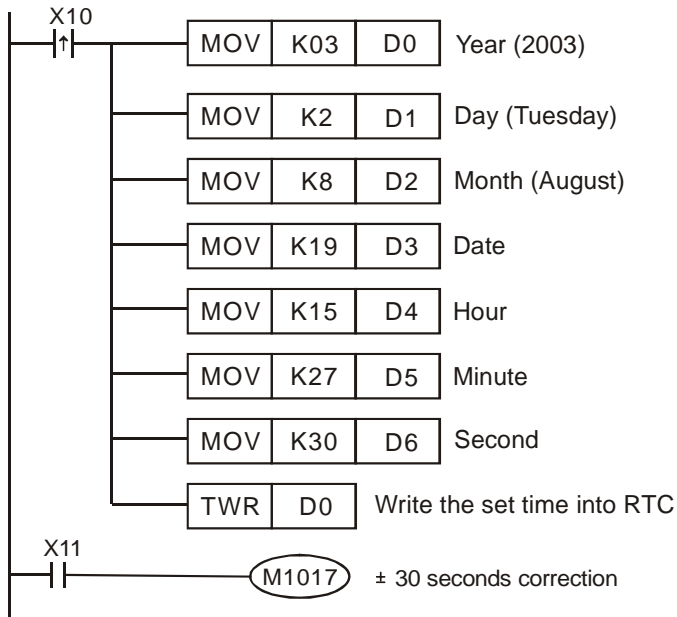
API	Mnemonic		Operands			Function		Controllers
169	D	HOUR	$\textcircled{S}$ $\textcircled{D_1}$ $\textcircled{D_2}$			Hour Meter		ES/EX/SS **SA/SX/SC EH/SV**

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	HOUR: 7 steps
S					*	*	*	*	*	*	*	*	*	*	*	DHOUR: 13 steps
D₁													*			
D₂		*	*	*												

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS **SA SX SC EH SV**	ES EX SS **SA SX SC EH SV**

Operands:

S: Period of time when **D₂** is On (in hour)   **D₁**: Current value being measured (in hour)   **D₂**: Output device

Explanations:

1. If **S** is used in device F, only 16-bit instruction is applicable.

2. **D₁** will occupy 2 consecutive points. **D₁** + 1 uses 16-bit register in 16-bit or 32-bit instruction.

3. See the specifications of each model for their range of use.

4. HOUR instruction can be used 4 times of SA/SX/SC.

5. Range of **S**: K1 ~ K32,767 (unit: hour); range of **D₁**: K0 ~ K32,767 (unit: hour). **D₁** + 1 refers to the current time that is less than an hour (range: K0 ~K3,599; unit: second).

6. This instruction times the time and when the time reaches the set time (in hour), **D₂** will be On. This function allows the user to time the operation of the machine or conduct maintenance works.

7. After **D₂** is On, the timer will resume the timing.

8. In the 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, **D₁** and **D₁** + 1 have to be reset to "0".

9. In the 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, **D₁** ~ **D₁** + 2 have to be reset to "0".

10. There is no limitations on the times of using this instruction in the program for EH series MPU; however, only 4 instructions can be executed at the same time.

Program Example 1:

In 16-bit instruction, when X0 = On, Y10 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) and D1 will record the current time that is less than an hour (0 ~ 3,599; unit: second).
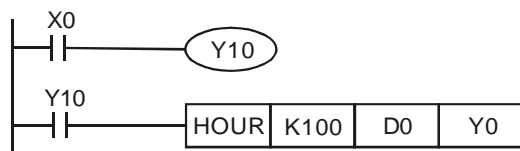
Program Example 2:

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 that is less than an hour (0 ~ 3,599; unit: second).
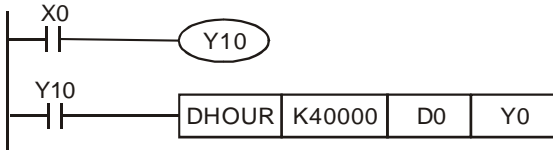
API		Mnemonic		Operands		Function	Controllers
170	D	GRY	P	$\boxed{S}$ $\boxed{D}$		BIN → Gray Code	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Source device for BIN value   **D**: Device for storing Gray code

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. This instruction converts the BIN value in the device designated in **S** into Gray code and stores the value in **D**.

4. See the ranges of **S** as indicated below. If **S** exceeds the ranges, the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code 0E1A (hex).

   In 16-bit instruction: 0 ~ 32,767

   In 32-bit instruction: 0 ~ 2,147,483,647

Program Example:

When X0 = On, the instruction will convert constant K6,513 into Gray code and store the result in K4Y20.
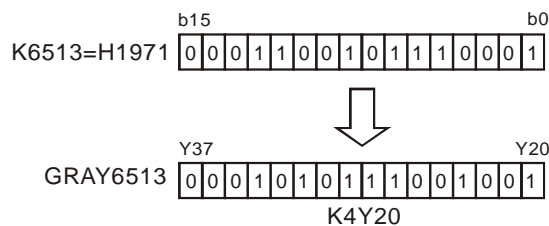
API		Mnemonic		Operands		Function			Controllers
171	D	GBIN	P	**S** **D**		Gray Code → BIN			ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices										Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	GBIN, GBINP: 5 steps
S					*	*	*	*	*	*	*	*	*	*	*	DGBIN, DGBINP: 9 steps
D								*	*	*	*	*	*	*	*	

	PULSE									16-bit									32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV		

Operands:

S: Source device for Gray code   **D**: Device for storing BIN value

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.

2. See the specifications of each model for their range of use.

3. This instruction converts the Gray code in the device designated in **S** into BIN value and stores the value in **D**.

4. This instruction converts the content (in Gray code) in the absolute position encoder connected at the PLC input terminal into BIN value and store the result in the designated register.

5. See the ranges of **S** as indicated below. If **S** exceeds the ranges, the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code 0E1A (hex).

   In 16-bit instruction: 0 ~ 32,767

   In 32-bit instruction: 0 ~ 2,147,483,647

Program Example:

When X20 = On, the Gray code in the absolute position encoder connected at X0 ~ X17 will be converted into BIN value and stored in D10.
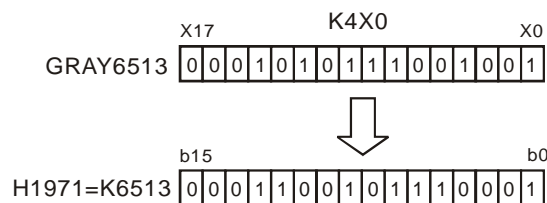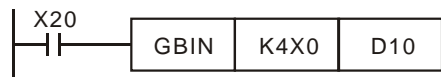
API	Mnemonic		Operands			Function	Controllers
172	D ADDR P		**S₁** **S₂** **D**			Floating Point Addition	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DADDR, DADDRP: 13 steps
S₁													*			
S₂													*			
D													*			

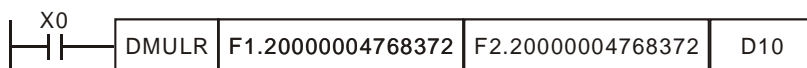| PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Floating point summand   **S₂**: Floating point addend   **D**: Sum

Explanations:

1. **S₁** and **S₂** can be floating point values (FX.XX).

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)

4. In DADDR instruction, floating point values (e.g. F1.2) can be entered directly into **S₁** and **S₂** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.

5. When **S₁** and **S₂** stores the floating point values in register D, their functions are the same as API 120 EADD.

6. **S₁** and **S₂** can designate the same register. In this case, if the "continuous execution" type instruction is in use and during the On period of the drive contact, the register will be added once in every scan by a "pulse execution" type instruction (DADDRP).

7. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.

8. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.

9. If the operation result is "0", the zero flag M1020 will be On.

Program Example 1:

When X0 = On, the floating point F1.20000004768372 will plus F2.20000004768372 and the result F3.40000009536743 will be stored in the data registers (D10, D11).

```
  X0
──┤ ├──┤ DADDR │ F1.20000004768372 │ F2.20000004768372 │ D10 │
```

Program Example 2:

When X0 = On, the floating point value (D1, D0) + floating point value (D3, D2) and the result will be stored in the registers designated in (D11, D10).

```
  X0
──┤ ├──┤ DADDR │ D0 │ D2 │ D10 │
```
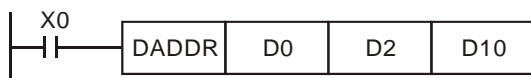
Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series.

DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic		Operands	Function	Controllers
173	D SUBR	P	S_1 S_2 D	Floating Point Subtraction	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSUBR, DSUBRP: 13 steps
S_1													*			
S_2													*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Floating point minuend   **S₂**: Floating point subtrahend   **D**: Remainder

Explanations:

1. **S₁** and **S₂** can be floating point values (FX.XX).

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)

4. In DSUBR instruction, floating point values (e.g. F1.2) can be entered directly into **S₁** and **S₂** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.

5. When **S₁** and **S₂** stores the floating point values in register D, their functions are the same as API 121 ESUB.

6. **S₁** and **S₂** can designate the same register. In this case, if the "continuous execution" type instruction is in use and during the On period of the drive contact, the register will be subtracted once in every scan by a "pulse execution" type instruction (DSUBRP).

7. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.

8. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.

9. If the operation result is "0", the zero flag M1020 will be On.

Program Example 1:

When X0 = On, the floating point F1.20000004768372 will minus F2.20000004768372 and the result F-1 will be stored in the data registers (D10, D11).

X0	DSUBR	F1.20000004768372	F2.20000004768372	D10

Program Example 2:

When X0 = On, the floating point value (D1, D0) – floating point value (D3, D2) and the result will be stored in the registers designated in (D11, D10).

X0	DSUBR	D0	D2	D10
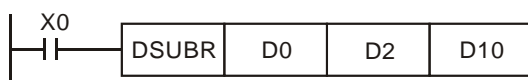
Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series.

DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic		Operands			Function	Controllers
174	D MULR P		S_1 S_2 D			Floating Point Multiplication	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DMULR, DMULRP: 13 steps
S_1													*			
S_2													*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Floating point multiplicand   **S₂**: Floating point multiplicator   **D**: Product

Explanations:

1. **S₁** and **S₂** can be floating point values (FX.XX).

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)

4. In DMULR instruction, floating point values (e.g. F1.2) can be entered directly into **S₁** and **S₂** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.

5. When **S₁** and **S₂** stores the floating point values in register D, their functions are the same as API 122 EMUL.

6. **S₁** and **S₂** can designate the same register. In this case, if the "continuous execution" type instruction is in use and during the On period of the drive contact, the register will be multiplied once in every scan by a "pulse execution" type instruction (DMULRP).

10. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.

11. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.

12. If the operation result is "0", the zero flag M1020 will be On.

Program Example 1:

When X0 = On, the floating point F1.20000004768372 will multiply F2.20000004768372 and the result F2.64000010490417 will be stored in the data registers (D10, D11).

```
 X0
─┤├──┬────────┬──────────────────┬──────────────────┬──────┐
     │ DMULR  │ F1.20000004768372│ F2.20000004768372│ D10  │
     └────────┴──────────────────┴──────────────────┴──────┘
```

Program Example 2:

When X1 = On, the floating point value (D1, D0) × floating point value (D11, D10) and the result will be stored in the registers designated in (D21, D20).

```
 X1
─┤├──┬────────┬─────┬─────┬─────┐
     │ DMULR  │ D0  │ D10 │ D20 │
     └────────┴─────┴─────┴─────┘
```
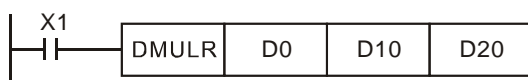
Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series. DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic		Operands			Function	Controllers
175	D DIVR P		(S₁) (S₂) (D)			Floating Point Division	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices												Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		DDIVR, DDIVRP: 13 steps
S₁													*				
S₂													*				
D													*				

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Floating point dividend   **S₂**: Floating point divisor   **D**: Quotient

Explanations:

1. **S₁** and **S₂** can be floating point values.

2. See the specifications of each model for their range of use.

3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)

4. In DDIVR instruction, floating point values (e.g. F1.2) can be entered directly into **S₁** and **S₂** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.

5. When **S₁** and **S₂** stores the floating point values in register D, their functions are the same as API 123 EDIV.

6. If **S₂** is "0", the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code H'0E19.

7. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.

8. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.

9. If the operation result is "0", the zero flag M1020 will be On.

Program Example 1:

When X0 = On, the floating point F1.20000004768372 will be divided by F2.20000004768372 and the result F0.545454561710358 will be stored in the data registers (D10, D11).
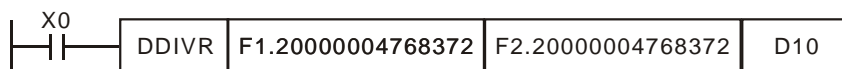
```
  X0
──┤├──┬─────────┬────────────────────┬────────────────────┬──────┐
      │ DDIVR   │ F1.20000004768372  │ F2.20000004768372  │ D10  │
      └─────────┴────────────────────┴────────────────────┴──────┘
```

Program Example 2:

When X1 = On, the floating point value (D1, D0) ÷ floating point value (D11, D10) and the quotient will be stored in the registers designated in (D21, D20).

```
  X1
──┤├──┬─────────┬──────┬──────┬──────┐
      │ DDIVR   │ D0   │ D10  │ D20  │
      └─────────┴──────┴──────┴──────┘
```
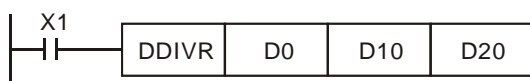
Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series.

DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic		Operands				Function	Controllers
180	MAND	P	S_1	S_2	D	n	Matrix 'AND' Operation	ES/EX/SS SA/SX/SC EH/SV

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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

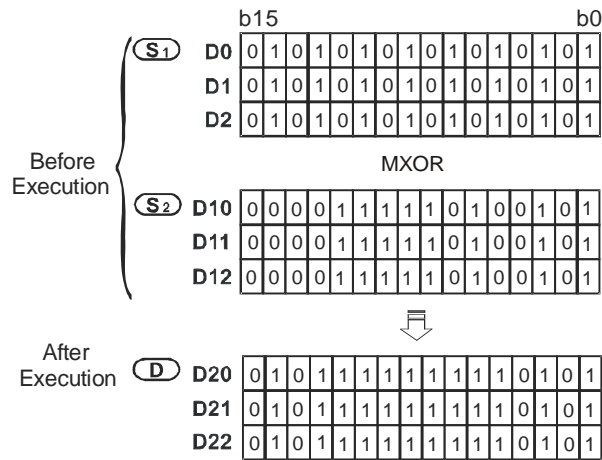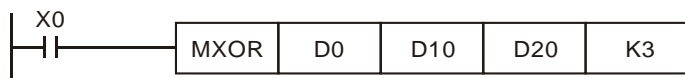**Operands:**

S₁: Matrix source device 1   **S₂**: Matrix source device 2   **D**: Operation result   **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256

2. **S₁**, and **S₂** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n ≦ 4.

4. See the specifications of each model for their range of use.

5. The two matrix sources **S₁** and **S₂** perform matrix 'AND' operation according to the array length **n**. The result is stored in **D**.

6. Operation rule of matix 'AND' : The result will be 1 if both two bits are 1; otherwise the result will be 0.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'AND' operation. The result will be stored in the 3 arrays of 16-bitd registers D20 ~ D22.
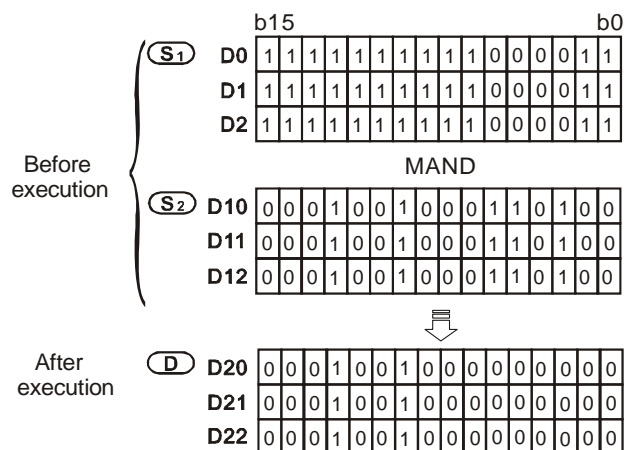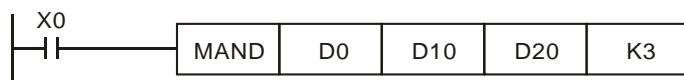
Remarks:

1. Explanations on the matrix instruction:

    a) A matix consists of more than 1 consecutive 16-bit registers. The number of registers in the matrix is the

length of the array (n). A matrix contains 16 × n bits (points) and there is only 1 bit (point) offered for an operand at a time.

b) The matrix instruction gathers a series of 16 × n bits ($b_0 \sim b_{16n-1}$) and designates a single point for operation. The point will not be seen as a value.

c) The matrix instruction processes the moving, copying, comparing and searching of one-to-many or many-to-many matrix status, which is a very handy and important application instruction.

d) The matrix operation will need a 16-bit register to designate a point among the 16n points in the matrix for the operation. The register is the Pointer (Pr) of the matrix, designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix.

e) There are left displacement, right displacement and rotation in a matrix operation. The bit number decreases from left to right (see the figure below).
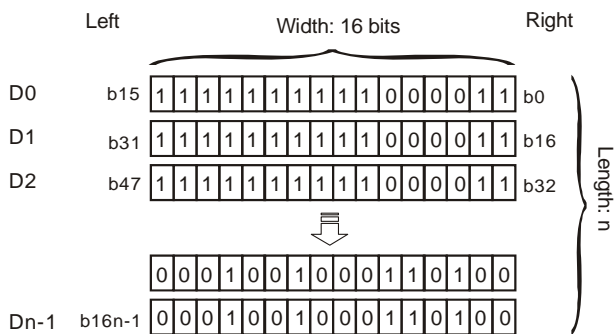
f) The matrix width (C) is fixed at 16 bits.

g) Pr: matrix pointer. E.g. if Pr is 15, the designated point will be b15.

h) Array length (R) is n: n = 1 ~ 256.

Example: The 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: The matrix is composed of K2X0, n = 3; K2X0 = H37, K2X10 = H68, K2X20 = 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	$X_0 \sim X_7$
R_1	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	$X_{10} \sim X_{17}$
R_2	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	$X_{20} \sim X_{27}$

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	S_1 S_2	D	n		Matrix 'OR' Operation	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MOR, MORP: 9 steps
S_1							*	*	*	*	*	*	*			
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Matrix source device 1   **S₂**: Matrix source device 2.   **D**: Operation result   **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256

2. **S₁**, and **S₂** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leq$ 4.

4. See the specifications of each model for their range of use.

5. The two matrix sources **S₁** and **S₂** perform matrix 'OR' operation according to the array length **n**. The result is stored in **D**.

6. Operation rule of matrix 'OR': The result will be 1 if either of the two bits is 1. The result is 0 only when both two bits are 0.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'OR' operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.
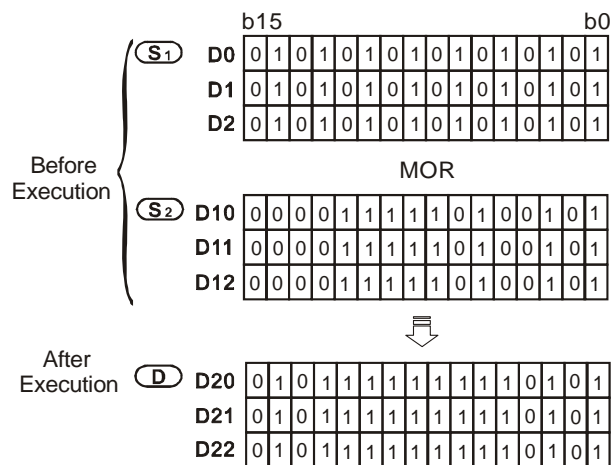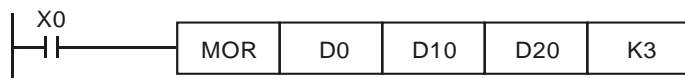
API	Mnemonic		Operands				Function		Controllers
182	MXOR	P	S_1 S_2 D n				Matrix 'XOR' Operation		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MXOR, MXORP: 9 steps
S_1							*	*	*	*	*	*	*			
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Matrix source device 1   **S₂**: Matrix source device 2   **D**: Operation result   **n**: Array length
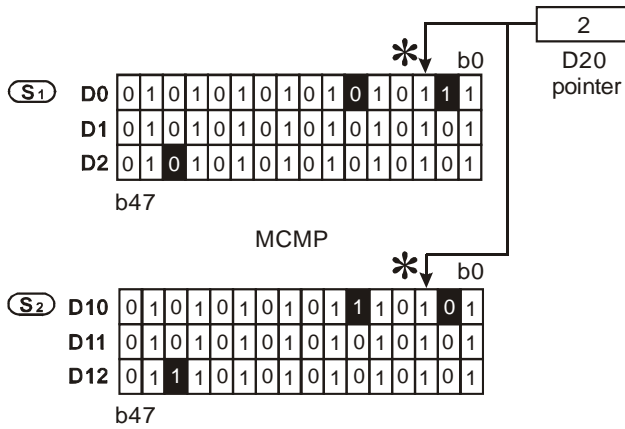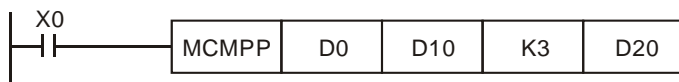
Explanations:

1. Range of **n**: K1 ~ K256

2. **S₁**, and **S₂** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.

4. See the specifications of each model for their range of use.

5. The two matrix sources **S₁** and **S₂** perform matrix 'XOR' operation according to the array length **n**. The result is stored in **D**.

6. Operation rule of matrix 'XOR': The result will be 1 if the two bits are different. The result will be 0 if the two bits are the same.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'XOR' operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.

API		Mnemonic		Operands			Function		Controllers
183		MXNR	P	S_1 S_2 D n			Matrix 'XNR' Operation		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices										Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MXNR, MXNRP: 9 steps
S_1							*	*	*	*	*	*	*			
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Matrix source device 1   **S₂**: Matrix source device 2   **D**: Operation result   **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256

2. **S₁**, and **S₂** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n ≦ 4.

4. See the specifications of each model for their range of use.

5. The two matrix sources **S₁** and **S₂** perform matrix 'XNR' operation according to the array length **n**. The result is stored in **D**.

6. Operation rule of matrix 'XNR': The result will be 1 if the two bits are the same. The result will be 0 if the two bits are different.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'XNR' operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.
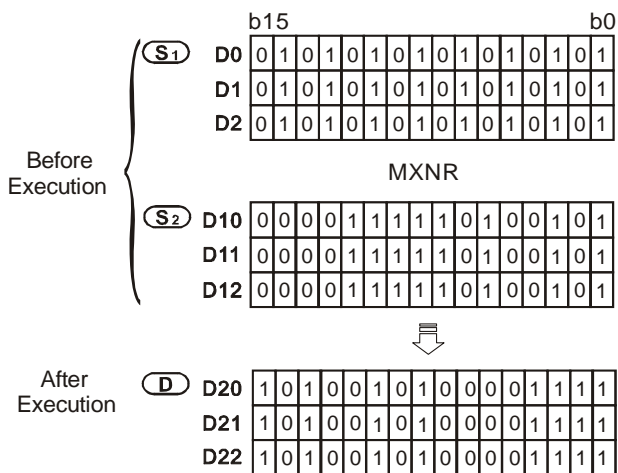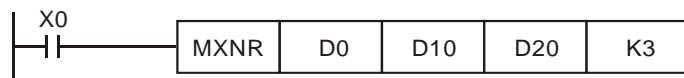
API	Mnemonic		Operands			Function	Controllers
184	MINV	P	S	D	n	Matrix Inverse Operation	ES/EX/SS **SA/SX/SC** EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MINV, MINVP: 7 steps
S							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS **SA SX SC** EH SV	ES EX SS **SA SX SC** EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Matrix source device   **D**: Operation result   **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256

2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leq$ 4.

4. See the specifications of each model for their range of use.

5. **S** performs an inverse matrix operation according to the array length **n**. The result is stored in **D**.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 perform a matrix inverse operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.

API	Mnemonic		Operands				Function		Controllers	
185	MCMP	P	S₁ S₂ n D				Matrix Compare		ES/EX/SS	SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MCMP, MCMPP: 9 steps
S₁							*	*	*	*	*	*	*			
S₂							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Matrix source device 1   **S₂**: Matrix source device 2   **n**: Array length

**D**: Pointer (Pr), for storing the value of target location

Explanations:

1. Range of **n**: K1 ~ K256

2. **S₁**, and **S₂** designate KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n ≦ 4.

4. See the specifications of each model for their range of use.

5. Flags: M1088 ~ M1092. See remarks for more details.

6. This instruction compares every bit in **S₁** with every bit in **S₂** starting from location **D** and finds out the location of different bits. The location will be stored in **D**.

7. The matrix comparison flag (M1088) decides to compare between equivalent values (M1088 = 1) or different values (M1088 = 0). When the comparison is completed, it will stop immediately and the matrix bit search flag will turn "On" (M1091 = 1). When the comparison progresses to the last bit, the matrix search end flag (M1089) will turn "On" and the No. where the comparison is completed is stored in **D**. The comparison will start from the 0th bit in the next scan period and the matrix search start flag will turn "On" (M1090 = 1). When **D** exceeds the range, the pointer error flag will turn "On" (M1092 = 1).

8. The matrix operation will need a 16-bit register to designate a point among the 16n points in the matrix for the operation. The register is the Pointer (Pr) of the matrix, designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix. Please avoid changing the Pr value during the operation in case the comparing and searching will not ne correct. If the Pr value exceeds its range, M1092 will be On and the instruction will not be executed.

9. When M1089 and M1091 take place at the same time, both flags will be "1" at the same time.

Program Example:

1. When X0 goes from Off to On, the matrix search start falg M1090 = 0. The searching will start from the bit marked with "*" (current Pr value +1) for bits of different status (M1088 = 0).

2. Set the Pr value D20 = 2. When X0 goes from Off to On for 4 times, we can obtain the 4 execution results ❶, ❷, ❸, ❹.

   ❶ D20 = 5, M1091 = 1, M1089 = 0.

   ❷ D20 = 45, M1091 = 1, M1089 = 0.

❸ D20 = 47, M1091 = 0, M1089 = 1.

❹ D20 = 1, M1091 = 1, M1089 = 0.

Remarks:

Flags explanations:

Flags	Function
M1088	Matrix comparison flag. Comparing between equivalent values (M1088 = 1) or different values (M1088 = 0).
M1089	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.
M1090	Matrix search start flag. Comparing from bit 0 (M1090 = 1).
M1091	Matrix bit search flag. When the comparison is completed, the comparison will stop immediately (M1091=1).
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.

API	Mnemonic		Operands			Function		Controllers
186	MBRD	P	S	n	D	Read Matrix Bit		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MBRD, MBRDP: 7 steps
S							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE	16-bit	32-bit
ES EX SS **SA SX SC** EH SV	ES EX SS **SA SX SC** EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Matrix source device   **n**: Array length   **D**: Pointer (Pr), for storing the value of target location

Explanations:

1. Range of **n**: K1 ~ K256

2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leq$ 4.

4. See the specifications of each model for their range of use.

5. Flags: M1089 ~ M1095. See remarks for more details.

6. When this instruction is executed, it first determines if the matrix pointer clear flag (M1094) is "On". If it is "On", pointer D is cleared as 0. The instruction then reads the On/Off status from the 0th bit of **S** to the matrix rotation/displacement/output carry flag (M1095). Whenever finishing reading 1 bit, the instruction determines whether the matrix pointer increasing flag (M1093) is "On". If it is "On", the value of pointer D will plus 1. When the reading is processed to the last bit, the matrix search end flag (M1089) will turn "On" and pointer D record the No. of read bits.

7. The Pointer (Pr) of the matrix is designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b_0 ~ b_{16n-1} in the matrix. If the Pr value exceeds its range, M1092 will be On and the instruction will not be executed.

Program Example:

1. When X0 goes from Off to On, M1094 will be set to "0" and M1093 to "1". Therefore, the Pr will plus 1 after every reading.

2. Set the Pr value D20 = 45. When X0 goes from Off to On for 3 times, we can obtain the 3 execution results ❶, ❷, ❸.

   ❶ D20 = 46, M1095 = 0, M1089 = 0.

   ❷ D20 = 47, M1095 = 1, M1089 = 0.

   ❸ D20 = 47, M1095 = 1, M1089 = 1.
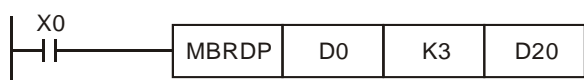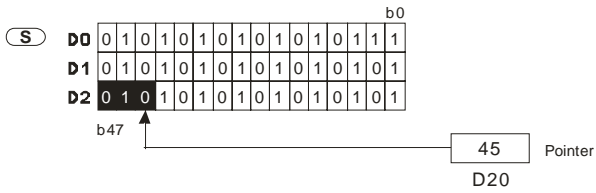
```
   X0
   ├─┤├──────┌────────┬──────┬──────┬──────┐
              │ MBRDP  │  D0  │  K3  │ D20  │
              └────────┴──────┴──────┴──────┘
```

Remarks:

Flag explanations:

Flags	Function
M1088	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.
M1093	Matrix pointer increasing flag. Adding 1 to the current value of the Pr.
M1094	Matrix pointer clear flag. Clearing the current value of the Pr to 0.
M1095	Matrix rotation/displacement/output carry flag.

API	Mnemonic		Operands			Function		Controllers
187	MBWR	P	S	n	D	Write Matrix Bit		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Matrix source device   **n**: Array length   **D**: Pointer (Pr), for storing the value of target location

Explanations:

1. Range of **n**: K1 ~ K256

2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.

4. See the specifications of each model for their range of use.

5. Flags: M1089 ~ M1096. See remarks for more details.

6. When this instruction is executed, if first determines if the matrix pointer clear flag (M1094) is "On", If it is "On", pointer D is cleared as 0. The instruction then writes the value in the matrix displacement/input complement flag (M1096) into the location starting from the 0th bit of **S**. Whenever finishing writing 1 bit, the instruction determines whether the matrix pointer increasing flag (M1093) is "On". If it is "On", the value of pointer D will plus 1. When the writing is processed to the last bit, the matrix search end flag (M1089) will turn "On" and pointer D records the No. of written bits. If D exceeds its range, M1092 will be On.

7. The Pointer (Pr) of the matrix is designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b_0 ~ b_{16n-1} in the matrix. If the Pr value exceeds its range, M1092 will be On and the instruction will not be executed.

Program Example:

1. When X0 goes from Off to On, M1094 will be set to "0" and M1093 to "1". Therefore, the Pr will plus 1 after every writing.

2. Set the Pr value D20 = 45 and M1096 = 1. When X0 goes from Off to On for 1 time, we can obtain the execution results: D20 = 45, M1096 = 1, M1089 = 0.
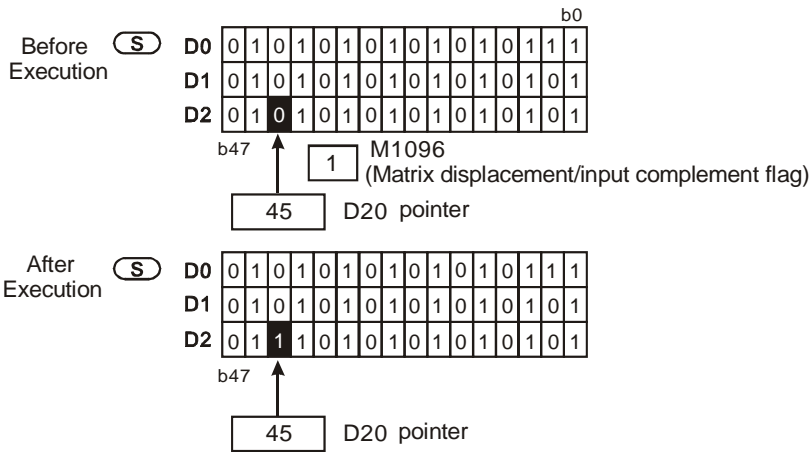
```
 X0
 ─┤├─────────[ MBWRP │ D0 │ K3 │ D20 ]
```

Remarks:

Flag explanations:

Flags	Function
M1088	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.
M1093	Matrix pointer increasing flag. Adding 1 to the current value of the Pr.
M1094	Matrix pointer clear flag. Clearing the current value of the Pr to 0.
M1096	Matrix displacement/input complement flag.

API	Mnemonic		Operands			Function	Controllers
188	MBS	P	S	D	n	Matrix Bit Displacement	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MBS, MBSP: 7 steps
S							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Matrix source device   **D**: Operation result   **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256

2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leq$ 4.

4. See the specifications of each model for their range of use.

5. Flags: M1095 ~ M1097. See remarks for more details.

6. This instruction performs left-right displacement on the matrix bits in **S** according to array length **n**. M1097 determines the left (M1097 = 0) or right (M1097 = 1) displacement of matrix bits. The empty bits derived from every displacement of 1 bit (when left displacement: b_0; when right displacement: b_{16n-1}) is filled by the status of the complement flag (M1096). The spare bits (when left displacement: b_{16n-1}; when right displacement: b_0) are sent to the carry flag (M1095). The result is stored in **D**.

7. The pulse execution instruction MBSP is generally adopted.

Program Example 1:

When X0 = On, M1097 = Off, indicating a left matrix displacement is performed. Set M1096 = 0 and the 16-bit registers D0 ~ D2 will perform a left matrix displacement and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The carry flag M1095 will be "1".
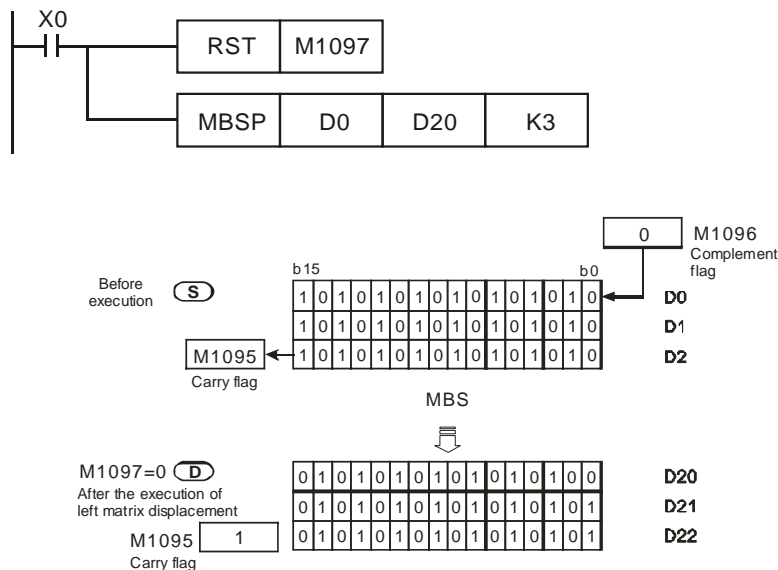
Program Example 2:

When X1 = On, M1097 = On, indicating a right matrix displacement is performed. Set M1096 = 1 and the 16-bit registers D0 ~ D2 will perform a right matrix displacement and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The carry flag M1095 will be "0".
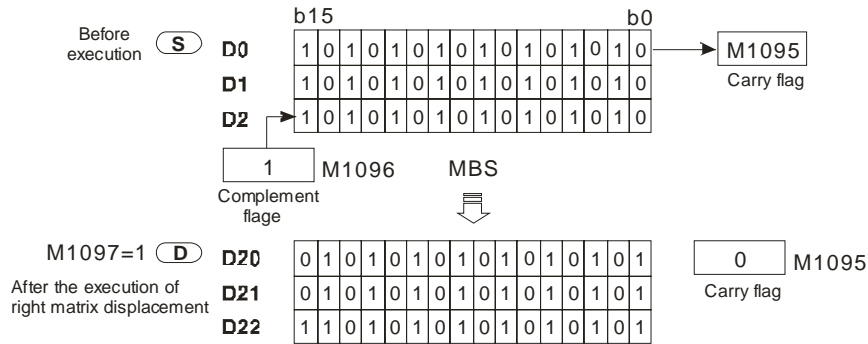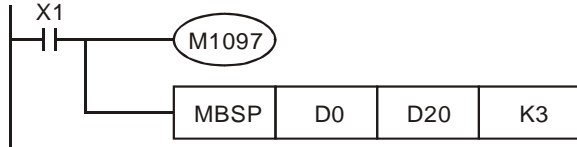
Explanations:

Flag explanations:

Flags	Function
M1095	Matrix rotation/displacement/output carry flag.
M1096	Matrix displacement/input complement flag.
M1097	Matrix rotation/displacement direction flag.

API	Mnemonic		Operands			Function	Controllers
189	MBR	P	S D	n		Matrix Bit Rotation	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MBR, MBRP: 7 steps
S							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Matrix source device   **D**: Operation result   **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256

2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leq$ 4.

4. See the specifications of each model for their range of use.

5. Flags: M1095, M1097. See remarks for more details.

6. This instruction performs left-right rotation on the matrix bits in **S** according to array length **n**. M1097 determines the left (M1097 = 0) or right (M1097 = 1) rotation of matrix bits. The empty bits derived from every rotation of 1 bit (when left rotation: b_0; when right rotation: b_{16n-1}) is filled by rotation bits (when left rotation: b_{16n-1}; when right rotation: b_0). The result is stored in **D**. Rotation bits not only fill the empty bits but also send the status of bits to the carry flag M1095.

7. The pulse execution instruction MBRP is generally adopted.

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 carry flag M1095 will be "1".
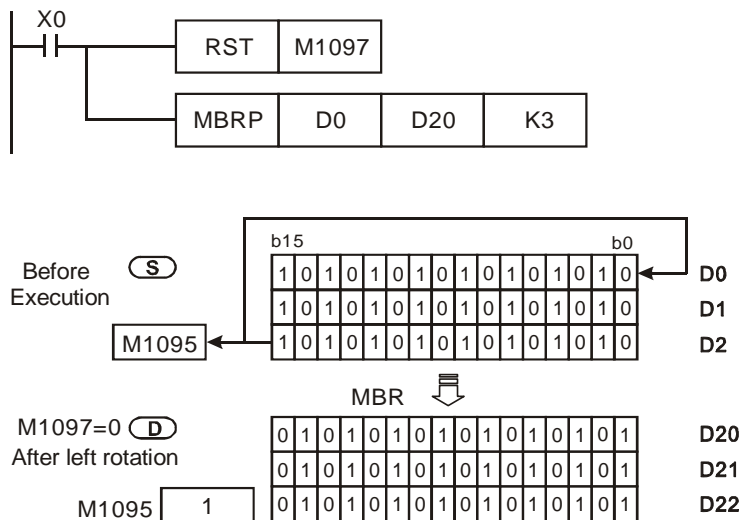
Program Example 2:

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 carry flag M1095 will be "0".
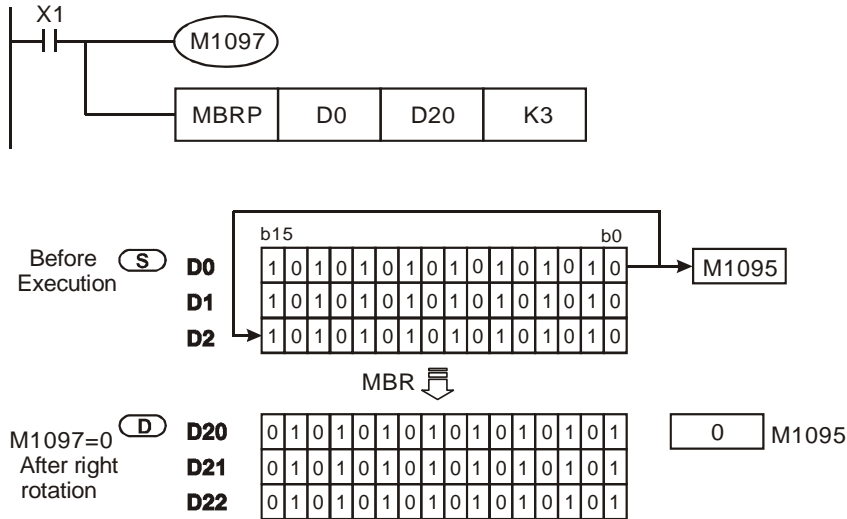
Remarks:

Flag explanations:

Flags	Function
M1095	Matrix rotation/displacement/output carry flag.
M1097	Matrix rotation/displacement direction flag.

API	Mnemonic		Operands			Function		Controllers
190	MBC	P	S n D			Matrix Bit Status Counting		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices										Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MBC, MBCP: 7 steps
S							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Matrix source device   **n**: Array length   **D**: Counting result

Explanations:

1. Range of **n**: K1 ~ K256

2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.

3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leq$ 4.

4. See the specifications of each model for their range of use.

5. Flags: M1098, M1099. See remarks for more details.

6. This instruction counts the number of bits which are "1" or "0" in **S** by array length **n**. The result is stored in **D**.

7. The instruction counts the number of bits which are "1" when M1098 = 1 and counts the number of bits which are "0" when M1098 = 0. When the operation result is "0", M1099 = 1.

Program Example:

When X10 = On, in the matrix of D0 ~ D2, when M1098 = 1, the instruction counts the total number of bits which are "1" and store the number in D10. When M1098 = 0, the instruction counts the total number of bits which are "0" and store the number in D10.

Remarks:

Flag explanations:

Flags	Function
M1098	Counting the number of bits which are "1" or "0"
M1099	On when the counting result is "0".

API	Mnemonic		Operands	Function		Controllers
191	D	PPMR	S_1 S_2 S D	2-Axis Relative Point to Point Motion		ES/EX/SS SA/SX/SC **EH/SV**

Type	Bit Devices				Word Devices											Program Steps
OP	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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	**EH**	**SV**

Operands:

S₁: Number of output pulses of X axis   **S₂**: Number of output pulses of Y axis   **S**: Max. point to point output frequency   **D**: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks for more details.

2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports "pulse + direction" mode.

3. **S₁** and **S₂** are the designated (relative designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.

4. **D** can designate Y0 and Y4.

    When Y0 is designated:

    Y0 refers to 1st group X-axis pulse output device.

    Y1 refers to 1st group X-axis direction signal.

    Y2 refers to 1st group Y-axis pulse output device.

    Y3 refers to 1st group Y-axis direction signal.

    Y4 refers to 2nd group X-axis pulse output device.

    Y5 refers to 2nd group X-axis direction signal.

    Y6 refers to 2nd group Y-axis pulse output device.

    Y7 refers to 2nd group Y-axis direction signal.

    When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.

5. D1340 (D1379) refers to the settings of the start/end frequencies of the 1st/2nd 2-axis motion. D1343 (D1381) refers to the time of the first acceleration segment and last deceleration segment of the 1st/2nd 2-axis motion. The time shall be longer than 10ms. If the time is shorter than 10ms or longer than 10,000ms, the output will be operated at 10ms. Default setting = 100ms.

6. If the maximum output frequency setting is less than 10Hz, the output will be operated at 10Hz. If the setting is more than 200KHz, the output will be operated at 200KHz.

7. When the 2-axis synchronous motion instruction is enabled, the start frequency and acceleration/deceleration time in Y axis will be same as the settings in X axis.

8. The number of output pulses for the 2-axis motion shall not be less than 59; otherwise the line drawn will not be straight enough.

9. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.

Program Example:

1. Draw a rhombus as the figure below.
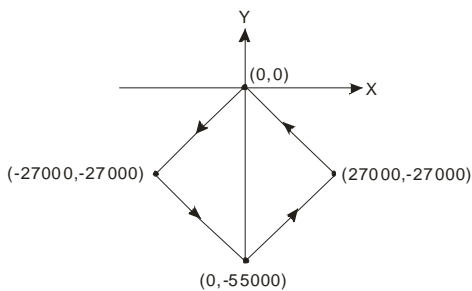
2. Steps:

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 (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Write program codes as follows.

c) PLC RUN. Set M0 as On and start the 2-axis line drawing.
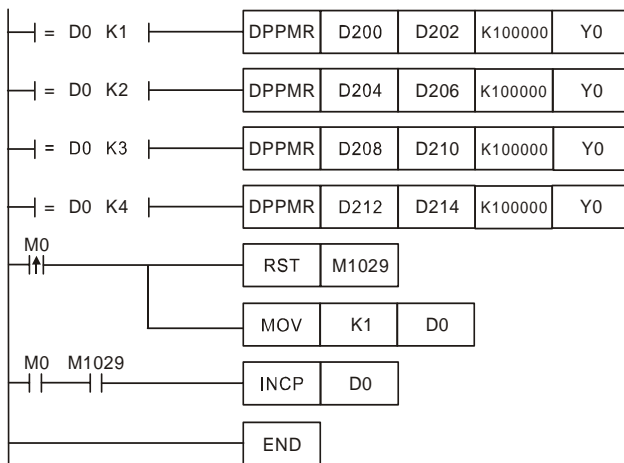
3. Motion explanation:

When PLC RUN 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 same motion will keep executing until the fourth point-to-point motion is completed.

Remarks:

1. Flag explanations:

   M1029： On when the 1st group 2-axis pulse output is completed.

   M1036： On when the 2nd group 2-axis pulse output is completed.

   M1334： On when the 1st group 2-axis pulse output is forbidden.

   M1336： 1st group 2-axis pulse output indication flag

   M1520： On when the 2nd group 2-axis pulse output is forbidden.

   M1522： 2nd group 2-axis pulse output indication flag

2. Special register explanations:

   D1336, D1337 **:** Pulse present value register for Y0 output of the 1st group X-axis motion. The present value increases or decreases following the rotation direction. (D1337 high word; D1336 low word)

   D1338, D1339 **:** Pulse present value register for Y2 output of the 1st group Y-axis motion. The present value increases or decreases following the rotation direction. (D1339 high word; D1338 low word)

   D1340 **:** Frequency settings of the first acceleration and last deceleration segment for the Y0 output of the 1st group X-axis motion and Y2 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.

   D1343 **:** Time settings of the first acceleration and last deceleration segment for the Y0 output of the 1st group X-axis motion and Y2 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.

   D1375, D1376 **:** Pulse present value register for Y4 output of the 2nd group X-axis motion. The present value increases or decreases following the rotation direction. (D1337 high word; D1336 low word)

   D1377, D1378 **:** Pulse present value register for Y6 output of the 2nd group Y-axis motion. The present value increases or decreases following the rotation direction. (D1339 high word; D1338 low word)

   D1379 **:** Frequency settings of the first acceleration and last deceleration segment for the Y4 output of the 2nd group X-axis motion and Y6 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.

   D1381 **:** Time settings of the first acceleration and last deceleration segment for the Y4 output of the 2nd group X-axis motion and Y6 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.

API	Mnemonic		Operands	Function	Controllers
192	D	PPMA	S_1 S_2 S D	2-Axis Absolute Point to Point Motion	ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		DPPMA: 17 steps
S_1					*	*							*				
S_2					*	*							*				
S					*	*							*				
D		*															

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses of X axis   **S_2**: Number of output pulses of Y axis   **S**: Max. point to point output frequency   **D**: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks of API 191 DPPMR for more details.

2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports "pulse + direction" mode.

3. **S_1** and **S_2** are the designated (absolute designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.

4. **D** can designate Y0 and Y4.

    When Y0 is designated:

    Y0 refers to 1st group X-axis pulse output device.

    Y1 refers to 1st group X-axis direction signal.

    Y2 refers to 1st group Y-axis pulse output device.

    Y3 refers to 1st group Y-axis direction signal.

    Y4 refers to 2nd group X-axis pulse output device.

    Y5 refers to 2nd group X-axis direction signal.

    Y6 refers to 2nd group Y-axis pulse output device.

    Y7 refers to 2nd group Y-axis direction signal.

    When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.

5. D1340 (D1379) refers to the settings of the start/end frequencies of the 1st/2nd 2-axis motion. D1343 (D1381) refers to the time of the first acceleration segment and last deceleration segment of the 1st/2nd 2-axis motion. The time shall be longer than 10ms. If the time is shorter than 10ms or longer than 10,000ms, the output will be operated at 10ms. Default setting = 100ms.
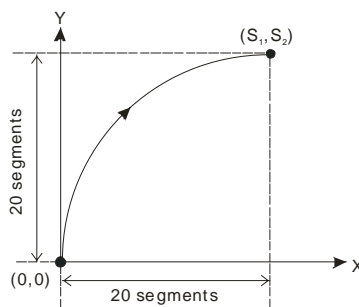
6. If the maximum output frequency setting is less than 10Hz, the output will be operated at 10Hz. If the setting is more than 200KHz, the output will be operated at 200KHz.

7. When the 2-axis synchronous motion instruction is enabled, the start frequency and acceleration/deceleration time in Y axis will be same as the settings in X axis.

8. The number of output pulses for the 2-axis motion shall not be the values within 1 ~ 59; otherwise the line drawn will not be straight enough.

9. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.

Program Example:

1. Draw a rhombus as the figure below.

2. Steps:

a) Set the four coordinates (0,0), (-27000, -27000), (0, -55000), (27000, -27000) (as the figure above). Place them in the 32-bit (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Write program codes as follows.

c) PLC RUN. Set M0 as On and start the 2-axis line drawing.

3. Motion explanation:

    When PLC RUN 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 same motion will keep executing until the fourth point-to-point motion is completed.

API	Mnemonic		Operands				Function		Controllers	
193	D	CIMR	S_1 S_2 S D				2-Axis Relative Position Arc Interpolation		ES/EX/SS SA/SX/SC EH/SV	

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCIMR: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S													*			
D		*														

	PULSE								16-bit								32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

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. Flags: M1029, M1030, M1334, M1335. See remarks of API 191 DPPMR for more details.

2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports "pulse + direction" mode.

3. **S₁** and **S₂** are the designated (relative designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.

4. The lower 16 bits of **S** (settings of direction and resolution): K0 refers to clockwise 10-segment (average resolution) output; K2 refers to clockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 1 and 2). K1 refers to counterclockwise 10-segment (average resolution) output; K3 refers to counterclockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 3 and 4).

5. The higher 16 bits of **S** (settings of motion time): K1 refers to 0.1 second. The setting range for average resolution is K1 ~ K100 (0.1 sec. ~ 10 secs.), for higher resolution is K2 ~ K200 (0.2 sec. ~ 20 secs.) This instruction is restricted by the maximum pulse output frequency; therefore when the set time goes faster than the actual output time, the set time will be automatically modified.

Figure 1   Figure 2

Figure 3   Figure 4

6. **D** can designate Y0 and Y4.

   When Y0 is designated:

   Y0 refers to 1st group X-axis pulse output device.

   Y1 refers to 1st group X-axis direction signal.

   Y2 refers to 1st group Y-axis pulse output device.

   Y3 refers to 1st group Y-axis direction signal.

   When Y4 is designated:

   Y4 refers to 2nd group X-axis pulse output device.

   Y5 refers to 2nd group X-axis direction signal.

   Y6 refers to 2nd group Y-axis pulse output device.

   Y7 refers to 2nd group Y-axis direction signal.

   When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.

7. Draw four 90° arcs.

8. When the direction signal is On, the direction is positive. When the direction signal is Off, the direction is negative. When **S** is set as K0, K2, the arcs will be clockwise (see figure 5). When **S** is set as K1, K3, the arcs will be counterclockwise (see figure 6).

Figure 5   Figure 6

9. When the 2-axis motion is being executed in 10 segments (of average resolution), the operation time of the instruction when the instruction is first enabled is approximately 5ms. The number of output pulses cannot be less than 100 and more than 1,000,000; otherwise, the instruction cannot be enabled.

10. When the 2-axis motion is being executed in 20 segments (of high resolution), the operation time of the instruction when the instruction is first enabled is approximately 10ms. The number of output pulses cannot be less than 1,000 and more than 10,000,000; otherwise, the instruction cannot be enabled.

11. If you wish the number of pulses in 10-segment or 20-segment motion to be off the range, you may adjust the gear ratio of the servo for obtaining your desired number.

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 precise 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 acceleration/deceleration time.

14. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.

15. The settings of direction and resolution in the lower 16 bits of **S** can only be K0 ~ K3.

16. The settings of motion time in the high 16 bits of **S** can be slower than the the fastest suggested time but shall not be faster than the fastest suggested time.

17. The fastest suggested time for the arc interpolation:

Segments	Max. target position (pulse)	Fastest suggested set time (unit:100ms)
Average resolution	100 ~ 10,000	1
	10,001 ~ 19,999	2
	:	:
	Less than 1,000,000	Less than 100
Higher resolution	1,000 ~ 20,000	2
	20,000 ~ 29,999	3
	:	:
	Less than 10,000,000	Less than 200

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 (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Select "draw clockwise arc" and "average resolution" (**S** = K0).

c) Write program codes as follows.
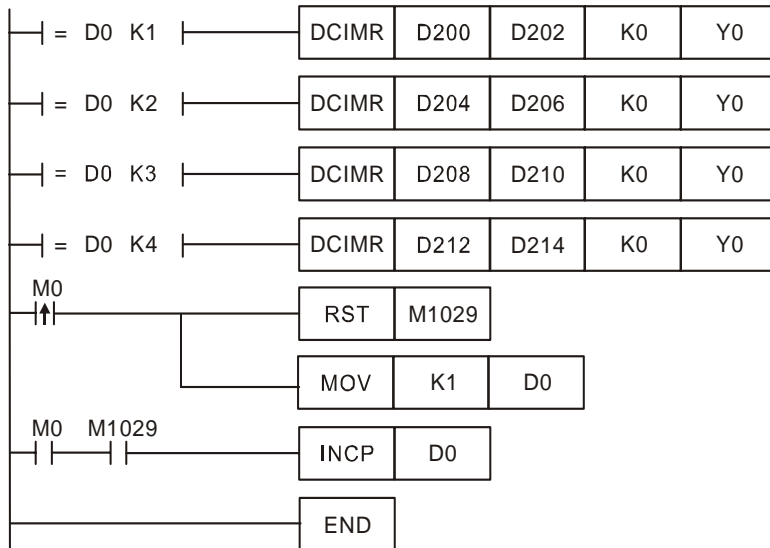
d) PLC RUN. Set M0 as On and start the drawing of the ellipse.

= D0 K1	DCIMR	D200	D202	K0	Y0
= D0 K2	DCIMR	D204	D206	K0	Y0
= D0 K3	DCIMR	D208	D210	K0	Y0
= D0 K4	DCIMR	D212	D214	K0	Y0

```
  M0
 ┤↑├──────────┬──── RST    M1029
              │
              └──── MOV    K1     D0
  M0   M1029
 ┤ ├───┤ ├──────── INCP   D0

 ──────────────── END
```

3. Motion explanation:

    When PLC RUN 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 same motion will keep executing 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 (D200,D202), (D204,D206), (D208,D210) and (D212,D214).

b) Select "draw clockwise arc" and "average resolution" (**S** = K0).

c) Select DCIMR instruction for drawing arc and write program codes as follows.

d) PLC RUN. Set M0 as On and start the drawing of the ellipse.

─┤ = D0 K1 ├─	DCIMR	D200	D202	K0	Y0

─┤ = D0 K2 ├─	DCIMR	D204	D206	K0	Y0

─┤ = D0 K3 ├─	DCIMR	D208	D210	K0	Y0

─┤ = D0 K4 ├─	DCIMR	D212	D214	K0	Y0

M0
─┤↑├─   RST   M1029

            MOV   K1   D0

M0   M1029
─┤ ├──┤ ├─   INCP D0

            END

3. Motion explanation:

    When PLC RUN 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 same motion will keep executing until the fourth segment of arc is completed.

API		Mnemonic	Operands				Function			Controllers	
194	D	CIMA	S_1 S_2 S D				2-Axis Absolute Position Arc Interpolation			ES/EX/SS SA/SX/SC **EH/SV**	

Type		Bit Devices			Word Devices											Program Steps
OP	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							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	**EH**	**SV**

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. Flags: M1029, M1030, M1334, M1335. See remarks of API 191 DPPMR for more details.

2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports "pulse + direction" mode.

3. **S₁** and **S₂** are the designated (absolute designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647. When **S₁** and **S₂** are larger than pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word), and CH3 (D1378 high word, D1377 low word), the output direction will be positive and direction signals Y1, Y3, Y5, Y7 will be On. When **S₁** and **S₂** are less than pulse present value registers, the output direction will be negative and direction signals Y1, Y3, Y5, Y7 will be Off.

4. The lower 16 bits of **S** (settings of direction and resolution): K0 refers to clockwise 10-segment (average resolution) output; K2 refers to clockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 1 and 2). K1 refers to counterclockwise 10-segment (average resolution) output; K3 refers to counterclockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 3 and 4).

5. The higher 16 bits of **S** (settings of motion time): K0 refers to 0.1 second. The setting range for average resolution is K1 ~ K100 (0.1 sec. ~ 10 secs.), for higher resolution is K2 ~ K200 (0.2 sec. ~ 20 secs.) This instruction is restricted by the maximum pulse output frequency; therefore when the set time goes faster than the actual output time, the set time will be automatically modified.

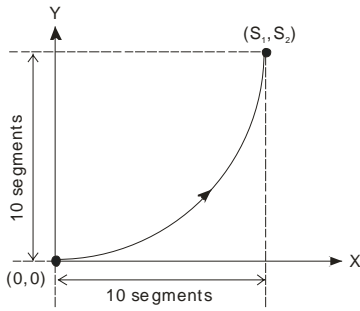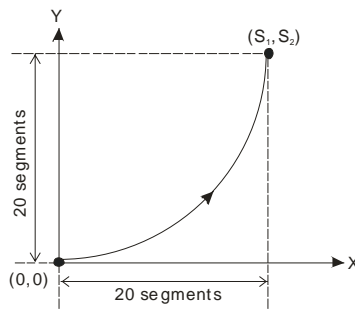Figure 1   Figure 2

Figure 3

Figure 4

6. **D** can designate Y0 and Y4.

   When Y0 is designated:

   Y0 refers to 1st group X-axis pulse output device.

   Y1 refers to 1st group X-axis direction signal.

   Y2 refers to 1st group Y-axis pulse output device.

   Y3 refers to 1st group Y-axis direction signal.

   When Y4 is designated:

   Y4 refers to 2nd group X-axis pulse output device.

   Y5 refers to 2nd group X-axis direction signal.

   Y6 refers to 2nd group Y-axis pulse output device.

   Y7 refers to 2nd group Y-axis direction signal.

   When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.

7. Draw four 90° arcs.

8. When the direction signal is On, the direction is positive. When the direction signal is Off, the direction is negative. When **S** is set as K0, K2, the arcs will be clockwise (see figure 5). When **S** is set as K1, K3, the arcs will be counterclockwise (see figure 6).

Figure 5

Figure 6

9. When the 2-axis motion is being executed in 10 segments (of average resolution), the operation time of the instruction when the instruction is first enabled is approximately 5ms. The number of output pulses cannot be less than 100 and more than 1,000,000; otherwise, the instruction cannot be enabled.

10. When the 2-axis motion is being executed in 20 segments (of high resolution), the operation time of the instruction when the instruction is first enabled is approximately 10ms. The number of output pulses cannot be less than 1,000 and more than 10,000,000; otherwise, the instruction cannot be enabled.

11. If you wish the number of pulses in 10-segment or 20-segment motion to be off the range, you may adjust the gear ratio of the servo for obtaining your desired number.

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 precise 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 acceleration/deceleration time.

14. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.

15. The settings of direction and resolution in the lower 16 bits of **S** can only be K0 ~ K3.

16. The settings of motion time in the high 16 bits of **S** can be slower than the the fastest suggested time but shall not be faster than the fastest suggested time.

17. The fastest suggested time for the arc interpolation:

Segments	Max. target position (pulse)	Fastest suggested set time (unit:100ms)
Average resolution	100 ~ 10,000	1
	10,001 ~ 19,999	2
	:	:
	Less than 1,000,000	Less than 100
Higher resolution	1,000 ~ 20,000	2
	20,000 ~ 29,999	3
	:	:
	Less than 10,000,000	Less than 200

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 (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Select "draw clockwise arc" and "average resolution" (**S** = K0).

c) Select DCIMA instruction for drawing arc and write program codes as follows.

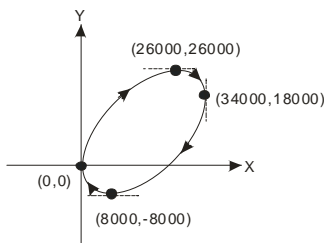d) PLC RUN. Set M0 as On and start the drawing of the ellipse.

3. Motion explanation: ：

When PLC RUN 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 same motion will keep executing until the fourth segment of arc is completed.
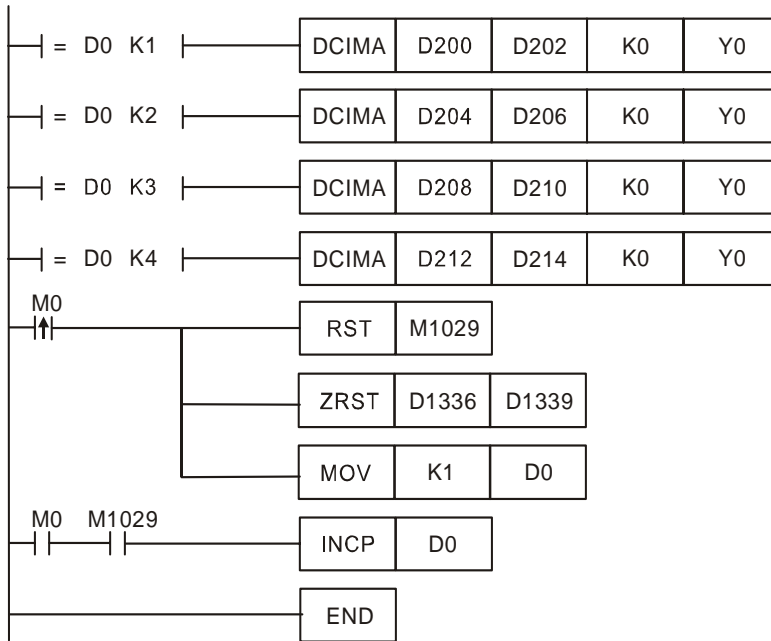
Program Example 2:

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 (D200,D202), (D204,D206), (D208,D210) and (D212,D214).

b) Select "draw clockwise arc" and "average resolution" (**S** = K0).

c) Select DCIMA instruction for drawing arc and write program codes as follows.

d) PLC RUN. Set M0 as On and start the drawing of the ellipse.

= D0 K1	DCIMA	D200	D202	K0	Y0
= D0 K2	DCIMA	D204	D206	K0	Y0
= D0 K3	DCIMA	D208	D210	K0	Y0
= D0 K4	DCIMA	D212	D214	K0	Y0

M0
```
RST    M1029
ZRST   D1336   D1339
MOV    K1      D0
```

M0 M1029
```
INCP   D0
END
```

3. Motion explanation:

    When PLC RUN 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 same motion will keep executing until the fourth segment of arc is completed.

API	Mnemonic		Operands	Function		Controllers
195	D	PTPO	S_1 S_2 D	Single-Axis Pulse Output by Table		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		DPTPO: 13 steps
S_1													*				
S_2													*				
D		*															

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Source start device   **S₂**: Number of segments   **D**: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks for more details.

2. This instruction only supports EH2/SV series MPU, not EH series.

3. According to the value of **S₂** + 0, every segment consecutively occupy four register D. (**S₁** + 0) refers to output frequency. (**S₁** + 2) refers to the number of output pulses.

4. When the output frequency of **S₁** is less than 1, PLC will automatically modify it as 1. When the value is larger than 200,000KHz, PLC will automatically modify it as 200,000KHz.

5. **S₂** + 0: number of segments (range: 1 ~ 60). **S₂** + 1: number of segments being executed. Whenever the program scans to this instruction, the instruction will automatically update the segment No. that is currently being executed.

6. **D** can only designate output devices Y0, Y2, Y4 and Y6 and can only perform pulse output control. For the pin for direction control, the user has to compile other programs to control.

7. This instruction does not offer acceleration and deceleration functions. Therefore, when the instruction is disabled, the output pulses will stop immediately.

8. In every program scan, each channel can only be executed by one instruction. However, there is no limitation on the number of times using this instruction.

9. When the instruction is being executed, the user is not allowed to update the frequency or number of the segments. Changes made will not be able to make changes in the actual output.

Program Example:

1. When X0 = On, the output will be operated according to the set frequency and number of pulses in every segment.

2. Format of the table:

S₂ = D300, number of segments (D300 = K60)	**S₁** = D0, frequency (**S₁** + 0)	**S₁** = D0, number of output pulses (**S₁** + 2)
K1 (1ˢᵗ segment)	D1, D0	D3, D2
K2 (2ⁿᵈ segment)	D5, D4	D7, D6
⋮ ⋮	⋮ ⋮	⋮ ⋮
K60 (60ᵗʰ segment)	D237, D236	D239, D238

3. Monitor the segment No. that is currently being executed in register D301.

```
  X0
 ──┤├──────────────┌────────┬──────┬──────┬──────┐
                   │ DPTPO  │  D0  │ D300 │  Y0  │
                   └────────┴──────┴──────┴──────┘
                   ┌────────┐
                   │  END   │
                   └────────┘
```
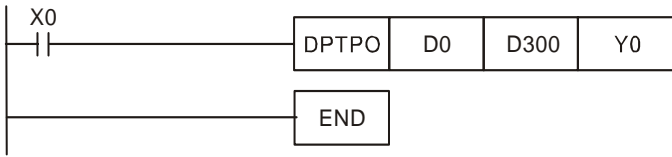
4. The pulse output curve:
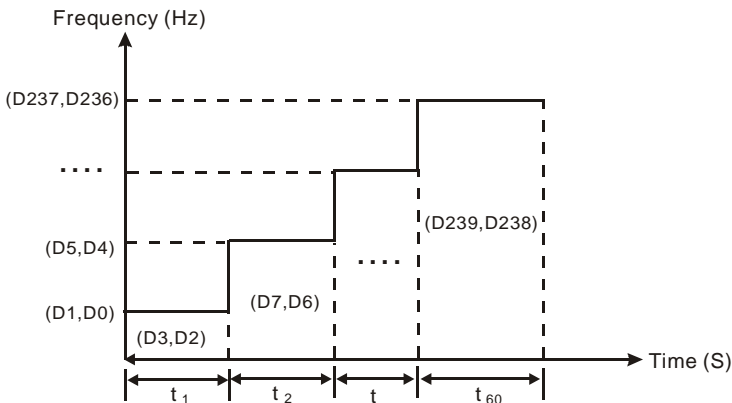
Remarks:

1. Flag explanations:

    M1029： On when CH0 (Y0) pulse output is completed.

    M1030： On when CH1 (Y2) pulse output is completed.

    M1036： On when CH2 (Y4) pulse output is completed.

    M1037： On when CH3 (Y6) pulse output is completed.

    M1334： When On, CH0 (Y0) pulse output will be forbidden.

    M1335： When On, CH1 (Y2) pulse output will be forbidden.

    M1520： When On, CH2 (Y4) pulse output will be forbidden.

    M1521： When On, CH3 (Y6) pulse output will be forbidden.

    M1336： CH0 (Y0) pulse output indication flag

    M1337： CH1 (Y2) pulse output indication flag

    M1522： CH2 (Y4) pulse output indication flag

    M1523： CH3 (Y6) pulse output indication flag

2. Special register explanations:

    D1336, D1337： Pulse present value register of CH0 (Y0) (D1337 high word, D1336 low word)

    D1338, D1339： Pulse present value register of CH1 (Y2) (D1339 high word, D1338 low word)

    D1375, D1376： Pulse present value register of CH2 (Y4) (D1376 high word, D1375 low word)

    D1377, D1378： Pulse present value register of CH3 (Y6) (D1378 high word, D1377 low word)

API		Mnemonic		Operands	Function			Controllers
196		HST	P	**S**	High Speed Timer			ES/EX/SS SA/SX/SC EH/SV

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

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S: Condition to stop the startup of high speed timer

Explanations:

1. Range of **S**: S = K0 (H0), K1 (H1).

2. Flag: M1015

3. When **S** = 1, the high speed timer will be enabled and M1015 = On. The high speed timer starts to time and record the present value in D1015 (min. unit: 100us).

4. Timing range of D1015: K0 ~ K32,767. When the timing reaches K32,767, the next timing will restart from 0.

5. When **S** = 0, the high speed timer will be disabled and M1015 = Off. D1015 will stop the timing immediately.

6. When **S** is neither 1 nor 0, HST instruction will not be executed.

Program Example :

1. When X10 = On, M1015 will be On. The high speed timer will start to time and record the present value in D1015.

2. When X10 = Off, M1015 will be Off. The high speed timer will be shut down.

```
     X10
     ┤├          ┌──────┬──────┐
                 │ HST  │  K1  │
                 └──────┴──────┘
     X10
     ┤/├         ┌──────┬──────┐
                 │ HST  │  K0  │
                 └──────┴──────┘
```
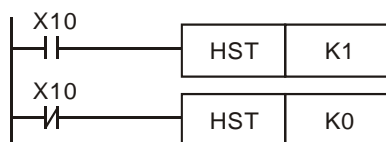
Remarks:

1. Flag explanations:

   M1015: high speed timer start-up flag

   D1015: high speed timer

2. EH/EH2/SV series MPU do not use this instruction and use special M and special D directly for the timer.

   a) Special M and special D are only applicable when PLC RUN.

   b) When M1015 = On and PLC scans to END instruction, the high speed timer D1015 will be enabled. The minimum timing unit of D1015: 100us.

   c) Timing range of D1015: K0 ~ K32,767. When the timing reaches K32,767, the next timing will restart from K0.

   d) When M1015 = Off, D1015 will stop the timing when encountering END or HST instruction.

3. SA/SX/SC series MPU do not use this instruction and use special M and special D directly for the timer.

   a) Special M and special D are applicable when PLC RUN or STOP.

b) When M1015 = On, the high speed timer D1015 will be enabled. The minimum timing unit of D1015: 100us.

c) Timing range of D1015: K0 ~ K32,767. When the timing reaches K32,767, the next timing will restart from K0.

d) When M1015 = Off, D1015 will stop the timing immediately.

API	Mnemonic		Operands	Function		Controllers
197	D	CLLM	S_1 S_2 S_3 D	Close Loop Position Control		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCLLM: 17 steps
S_1	*											*				
S_2					*	*							*			
S_3					*	*							*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Feedback source device   **S₂**: Target number of feedbacks   **S₃**: Target frequency of output

D: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks for more details.

2. This instruction only supports EH2/SV series MPU, not EH series.

3. The corresponding interruption of **S₁**:

Source device	X0	X1	X2	X3	C241 ~ C254			
Corresponding outout	Y0	Y2	Y4	Y6	Y0	Y2	Y4	Y6
Interruption No.	I00□	I10□	I20□	I30□	I010	I020	I030	I040

□ = 1: rising-edige trigger; □ = 0: falling-edge trigger

a) When **S₁** designates X as the input points and the pulse output reaches the set target number of feedbacks in **S₂**, the output will continue to operate by the frequency of the last segment until the interruption of X input points occurs.

b) When **S₁** designates a high speed counter and the pulse output reaches the set target number of feedbacks in **S₂**, the output will continue to operate by the frequency of the last segment until the feedback pulses reaches the target number.

c) **S₁** can be a high speed counter C or an external interruption X. If **S₁** is C, DCNT instruction should be first executed to enable the high-speed counting function and EI and I0x0 interruption service program to enable the high-speed interruption. If **S₁** is X, EI instruction and I0x0 interruption service program should be executed to enable the external interruption function.

4. The range of **S₂**: -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.

5. If **S₃** is lower than 10Hz, the output will operate at 10Hz; if **S₃** is higher than 200KHz, the output will operate at 200KHz.

6. **D** can only designate Y0, Y2, Y4 and Y6 and the direction signals repectively are Y1, Y3, Y5 and Y7. When there is a direction signal output, the direction signal will not be Off immediately after the pulse output is completed. The direction signal will be Off only when the drive contact is Off.

7. D1340, D1352, D1379 and D1380 are the settings of start/end frequencies of CH0 ~ CH3. The minimun frequency is 10Hz and default is 200Hz.

8. D1343, D1353, D1381 and D1382 are the settings of the time of the first segment and the last deceleration segment of CH0 ~ CH3. The acceleration/deceleration time cannot be shorter than 10ms. The outptu will be operated in 10ms if the time set is shorter than 10ms or longer than 10,000ms. The dafault setting is 100ms.

9. D1198, D1199, D1478 and D1479 are the output/input ratio of the close loop control in CH0 ~ CH3. K1 refers to 1 output pulse out of the 100 target feedback input pulses; K200 refers to 200 output pulses out of the 100 target feedback input pulses. D1198, D1199, D1478 and D1479 are the numerators of the ratio (range: K1 ~ K10,000) and the denominator is fixed as K100 (the user does not have to enter a denominator).

10. M1305, M1306, M1532 and M1533 are the direction signal flags for CH0 ~ CH3. When S_2 is a positive value, the output will be in forward direction and the flag will be Off. When S_2 is a negative value, the output will be in backward direction and the flag will be On.

Close Loop Explanations:

1. Function: Immediately stop the high-speed pulse output according to the number of feedback pulses or external interruption signals.

2. The execution:
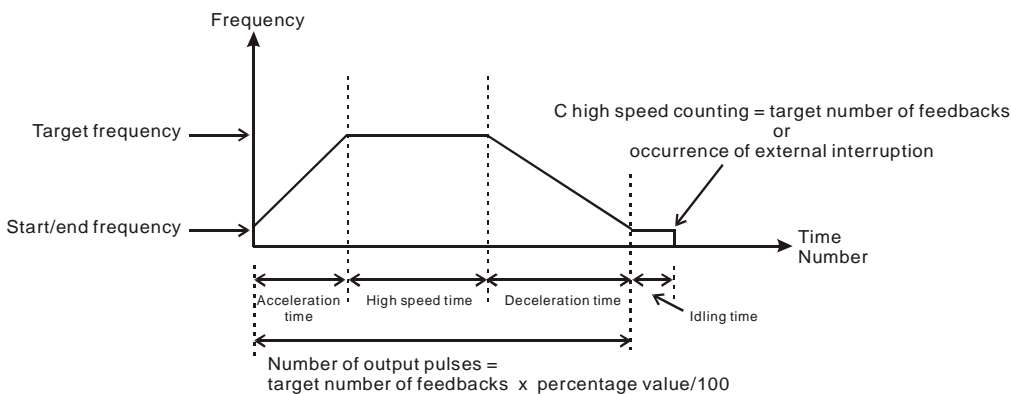
3. How to adjust the time for the completion of the positioning:

   a) The time for the completion of the positioning refers to the time for "acceleration + high speed + deceleration + idling" (see the figure above). For example, you can increase or decrease the entire number of output pulses by making adjustment on the percentage value and further increase or decrease the time required for the positioning.

   b) Among the four segments of time, only the idling time cannot be adjusted directly by the user. However, you 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 idling 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 (taget number of feedbacks x percentage value/100), you can improve the situation by adjusting the percentage value, acceleration/decelartion time or target frequency.

Program Example:

1. Assume we adopt X0 as the external interruption, together with I001 (rising-edge trigger) interruption program; target number of feedbacks = 50,000; target frequency = 10KHz; Y0, Y1 (CH0) as output pulses; start/end frequency (D1340) = 200Hz; acceleration time (D1343) = 300ms; deceleration time (D1348) = 600ms; percentage value (D1198) = 100; current number of output pulses (D1336, D1337) = 0.

2. Write the program codes as follows:
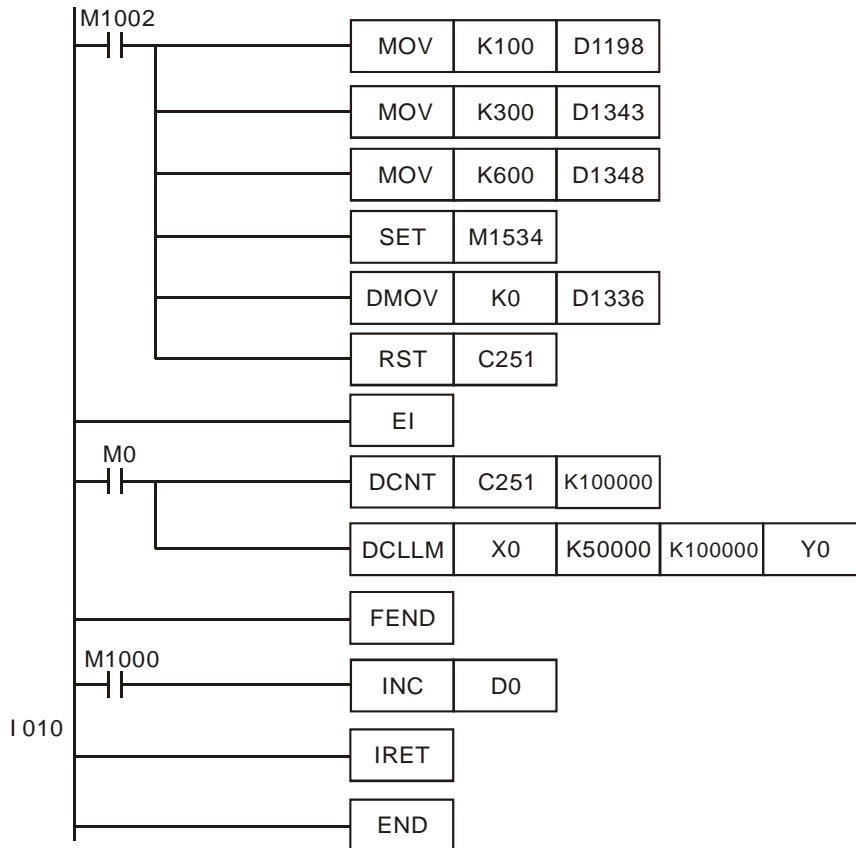
3. Assume the first execution result as:
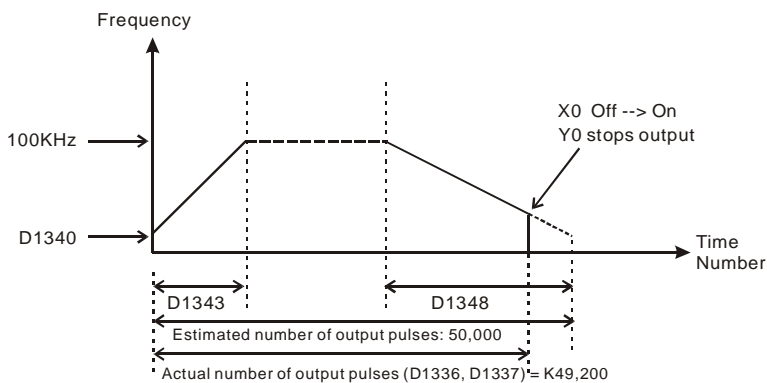
4. Observe the result of the first execution:

   a) The actual output number 49,200 – estimated output number 50,000 = -800 (a negative value). A negative value indicates that the entire execution finishes earlier and has not completed yet.

   b) Try to shorten the acceleration time (D1343) into 250ms and deceleration time (D1348) into 550ms.
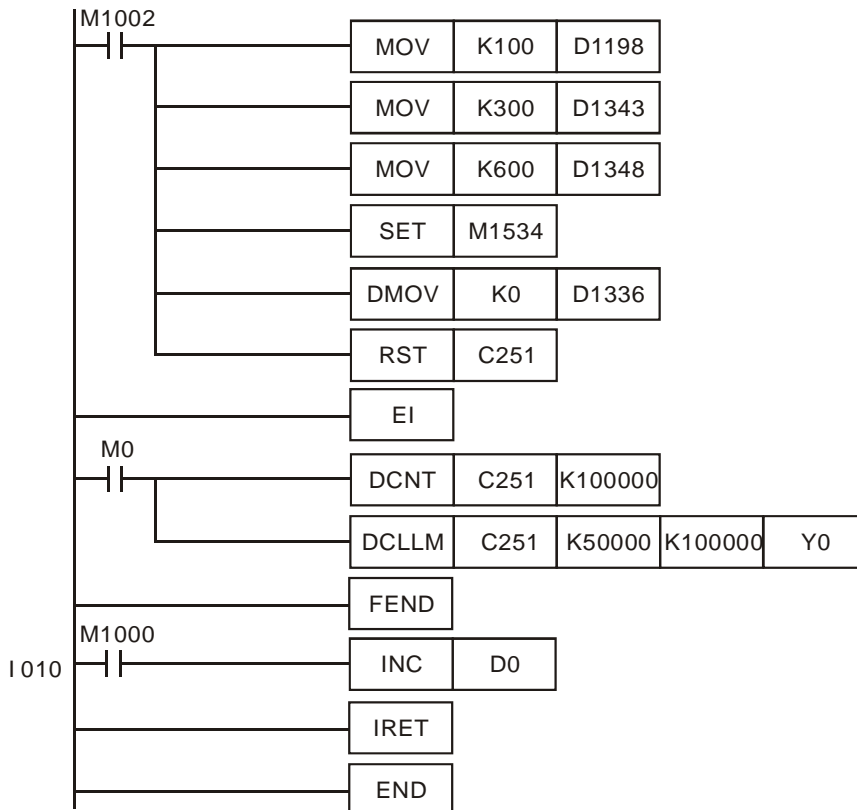
5. Obtain the result of the second execution:
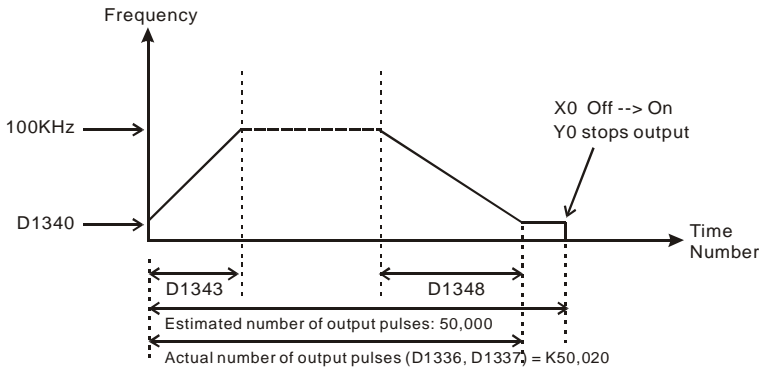
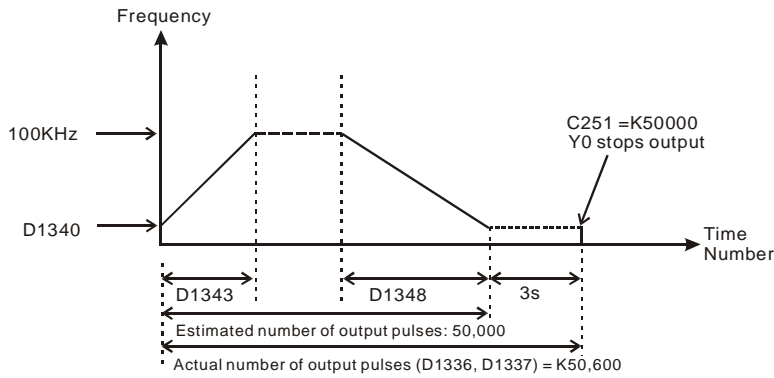6. Observe the result of the second execution:

   a) The actual output number 50,020 – estimated output number 50,000 = 20

   b) 20 x (1/200Hz) = 100ms (idling time)

   c) 100ms is an appropriate value. Therefore, set the acceleration time as 250ms and deceleration time as 550ms to complete the design.

Program Example 2:

1. Assume the feedback of the encoder is an A/B phase input and we adopt C251 timing (we suggust you clear it to 0 before the execution); target number of feedbacks = 50,000; target output frequency = 100KHz; Y0, Y1 (CH0) as output pulses; start/end frequency (D1340) = 200Hz; acceleration time (D1343) = 300ms; deceleration time (D1348) = 600ms; precentage value (D1198) = 100; current number of output pulses (D1336, D1337) = 0.
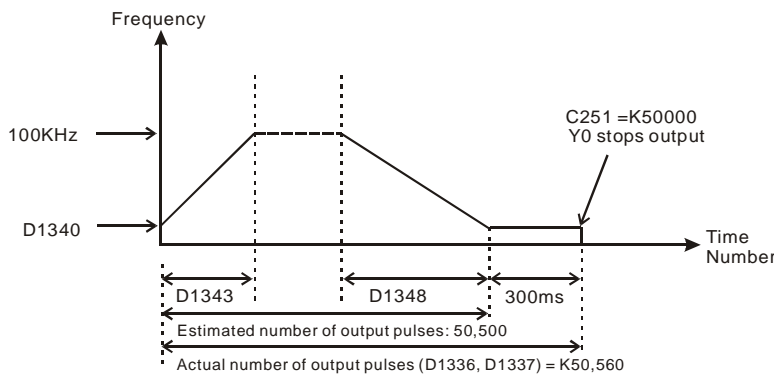
2. Write the program codes as follows:

3. Assume the first execution result as:

4. Observe the result of the first execution:

    a) The actual output number 50,600 – estimated output number 50,000 = 600

    b) 600 x (1/200Hz) = 3s (idling time)

    c) 3 seconds are too long. Therefore, increase the percentage value (D1198) to K101.

5. Obatin the result of the second execution:

6. Observe the result of the second execution:

    a) The actual output number 50,560 – estimated output number 50,500 = 60

    b) 60 x (1/200Hz) = 300ms (idling time)

    c) 300ms is an appropriate value. Therefore, set the percentage value (D1198) as K101 to complete the
        design.

Remarks:

1. Flag explanations:

    M1010： When On, CH0, CH1, CH2 and CH3 will output pulses when encountering END instruction. Off
    when the output starts.

    M1029： On when CH0 pulse output is completed.

    M1030： On when CH1 pulse output is completed.

    M1036： On when CH2 pulse output is completed.

    M1037： On when CH3 pulse output is completed.

    M1334： When On, CH0 pulse output will be forbidden.

    M1335： When On, CH1 pulse output will be forbidden.

M1520：	When On, CH2 pulse output will be forbidden.
M1521：	When On, CH3 pulse output will be forbidden.
M1336：	CH0 pulse output indication flag
M1337：	CH1 pulse output indication flag
M1522：	CH2 pulse output indication flag
M1523：	CH3 pulse output indication flag
M1305：	CH0 direction signal flag
M1306：	CH1 direction signal flag
M1532：	CH2 direction signal flag
M1533：	CH3 direction signal flag
M1534：	Deceleration time of CH0 setup flag (must used with D1348)
M1535：	Deceleration time of CH1 setup flag (must used with D1349)
M1536：	Deceleration time of CH2 setup flag (must used with D1350)
M1537：	Deceleration time of CH3 setup flag (must used with D1351)

2. Special register explanations:

D1198：	Close loop output/input ratio of CH0 (default: K100)
D1199：	Close loop output/input ratio of CH1 (default: K100)
D1478：	Close loop output/input ratio of CH2 (default: K100)
D1479：	Close loop output/input ratio of CH3 (default: K100)
D1220：	Phase setting of CH0 (Y0, Y1): determined by the last 2 digits of D1220; other digits are invalid.

1. K0: Y0 output
2. K1: Y0, Y1 AB-phase output; A ahead of B
3. K2: Y0, Y1 AB-phase output; B ahead of A

| D1221： | Phase setting of CH1 (Y2, Y3): determined by the last 2 digits of D1221; other digits are invalid. |

1. K0: Y2 output
2. K1: Y2, Y3 AB-phase output; A ahead of B
3. K2: Y2, Y3 AB-phase output; B ahead of A

| D1229： | Phase setting of CH2 (Y4, Y5): determined by the last 2 digits of D1229; other digits are invalid. |

1. K0: Y4 output
2. K1: Y4, Y5 AB-phase output; A ahead of B
3. K2: Y4, Y5 AB-phase output; B ahead of A

| D1230： | Phase setting of CH3 (Y6, Y7): determined by the last 2 digits of D1230; other digits are invalid. |

1. K0: Y6 output
2. K1: Y6, Y7 AB-phase output; A ahead of B
3. K2: Y6, Y7 AB-phase output; B ahead of A

D1222：	Time difference between the direction signal and pulse output of CH0
D1223：	Time difference between the direction signal and pulse output of CH1
D1383：	Time difference between the direction signal and pulse output of CH2
D1384：	Time difference between the direction signal and pulse output of CH3

D1336： Low word of the current number of output pulses of CH0

D1337： High word of the current number of output pulses of CH0

D1338： Low word of the current number of output pulses of CH1

D1339： High word of the current number of output pulses of CH1

D1375： Low word of the current number of output pulses of CH2

D1376： High word of the current number of output pulses of CH2

D1377： Low word of the current number of output pulses of CH3

D1378： High word of the current number of output pulses of CH3

D1340： Start/end frequency settings of CH0 (default: K200)

D1352： Start/end frequency settings of CH1 (default: K200)

D1379： Start/end frequency settings of CH2 (default: K200)

D1380： Start/end frequency settings of CH3 (default: K200)

D1348： Deceleration time of CH0 pulse output when M1534 = On (default: K100)

D1349： Deceleration time of CH1 pulse output when M1535 = On (default: K100)

D1350： Deceleration time of CH2 pulse output when M1536 = On (default: K100)

D1351： Deceleration time of CH3 pulse output when M1537 = On (default: K100)

D1343： Acceleration/deceleration time of CH0 pulse output (default: K100)

D1353： Acceleration/deceleration time of CH1 pulse output (default: K100)

D1381： Acceleration/deceleration time of CH2 pulse output (default: K100)

D1382： Acceleration/deceleration time of CH3 pulse output (default: K100)

MEMO

API	Mnemonic		Operands		Function		Controllers
202	SCAL	P	S_1 S_2 S_3 D		Proportional Value Calculation		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SCAL, SCALP: 9 steps
S_1					*	*							*			
S_2					*	*							*			
S_3					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Source value   **S₂**: Slope   **S₃**: Offset   **D**: Destination device

Explanations:

1. Range of S1, S2, S3: -32,768 ~ 32,767

2. Unit of S2: 0.001

3. See the specifications of each model for their range of use.

4. Operation equation in the instruction: $D = (S_1 \times S_2) \div 1000 + S_3$.

   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
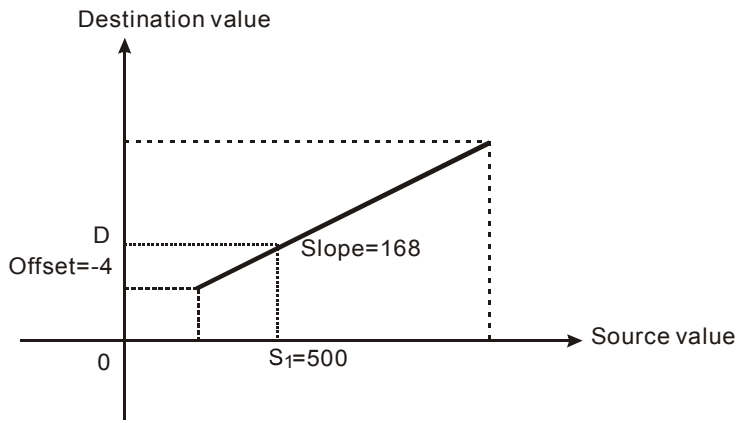
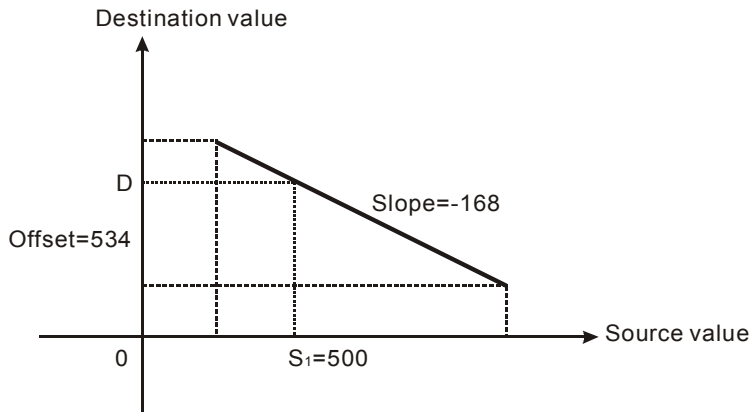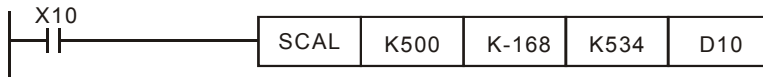   The output curve is shown as the figure:
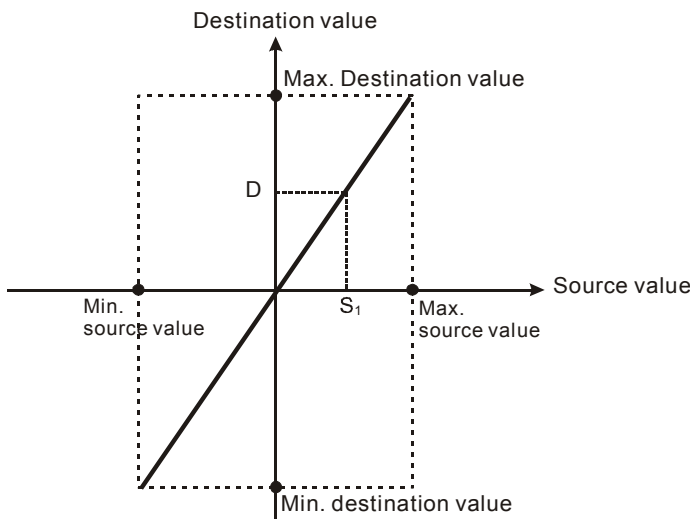
Program Example 1:

1. Assume **S₁** = 500, **S₂** = 168, **S₃** = -4. When X0 = On, SCAL instruction will be executed and obtain the proportional value at D0.

2. Equation: D0 = (500 × 168) ÷ 1,000 + (-4) = 80

Program Example 2:

1. Assume S_1 = 500, S_2 = -168, S_3 = 534. When X10 = On, SCAL instruction will be executed and obtain the proportional value at D10.

2. Equation: D0 = (500 × -168) ÷ 1,000 + 534 = 450

Remarks:

1. This instruction is applicable for known slope and offset. If slope and offset are unknown, use SCLP instruction for the calculation.

2. S_2 has to be within the range -32,768 ~ 32,767. If S_2 falls without the range, use SCLP instruction for the calculation.

3. When using the slope equation, please be aware that the max. source value must > min. source value, but it is not necessary that max. destination value > min. destination value.

4. If the value of **D** > 32,767, **D** = 32,767; if the value of **D** < -32,768, **D** = -32,768.

5. Only ES_V6.2, SA/SX_V1.6, SC_V1.4, EH2/SV_V1.0 and versions above support this instruction. EH series MPU does not support this instruction.

	Mnemonic		Operands	Function		Controllers
203	D SCLP P		S_1 S_2 D	Parameter Proportional Value Calculation		ES/EX/SS SA/SX/SC EH/SV

Type OP	Bit Devices						Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SCLP, SCLPP: 7 steps
S_1					*	*							*			DSCLP, DSCLPP: 13 steps
S_2													*			
D													*			

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Source value **S₂**: Parameter **D**: Destination device

Explanations:

1. See the specifications of each model for the range of operands.

2. Flags: M1162 (decimal integer or binary floating point); M1162 = On -> Binary floating point

3. Settings of **S₂** for 16-bit instruction:

    S₂ occupies 4 consecutive devices in 16-bit instruction.

Device No.	Parameter	Range
S_2	Maximum source value	-32,768 ~ 32,767
S_2 + 1	Minimum source value	-32,768 ~ 32,767
S_2 + 2	Maximum destination value	-32,768 ~ 32,767
S_2 + 3	Minimum destination value	-32,768 ~ 32,767

4. Settings of **S₂** for 32-bit instruction:

    S₂ occupies 8 consecutive devices in 32-bit instruction.

Device No.	Parameter	Range	
		Integer	Floating point
S_2, S_2 + 1	Maximum source value		
S_2 + 2, 3	Minimum source value	-2,147,483,648 ~ 2,147,483,647	Range of 32-bit floating point
S_2 + 4, 5	Maximum destination value		
S_2 + 6, 7	Minimum destination value		

5. Operation equation in the instruction: **D** = [(**S₁** – min. source value) × (max. destination value – min. destination value)] ÷ (max. source value – min. source value) + min. destination value

6. The operational relation between source value and destination value is as stated below:

    y = kx + b

    y= Destination value (**D**)

    k= Slope = (max. destination value – min. destination value) ÷ (max. source value – min. source value)

    x= Source value (**S₁**)

    b= Offset = Min. destination value – Min. source value × slope

   Bring all the parameters into equation y = kx + b and obtain the equation in the instruction:

    y = kx + b = D = k **S₁** + b = slope ×**S₁** + offset = slope ×**S₁** + 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 **S1** > max. source value, **S1** = max. source value

    If **S1** < min. source value, **S1** = min. source value

    When all the input values and parameters are set, the output curve is shown as the figure:
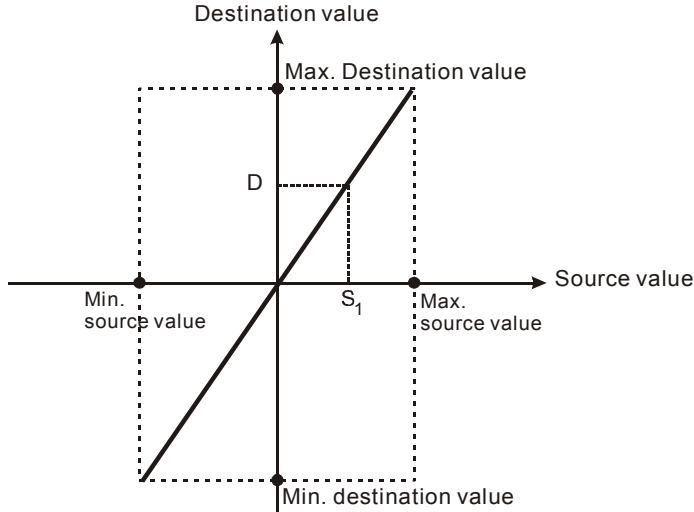
Program Example 1:

1. Assume S_1 = 500, max. source value D0 = 3,000, min. source value D1 = 200, max. destination value D2 = 500, and min. destination value D3 = 30. When X0 = On, SCLP instruction will be executed and obtain the proportional value at D10.

2. Equation: D10 = [(500 – 200) × (500 – 30)] ÷ (3,000 – 200) + 30 = 80.35. Round off the result into an integer D10 = 80.
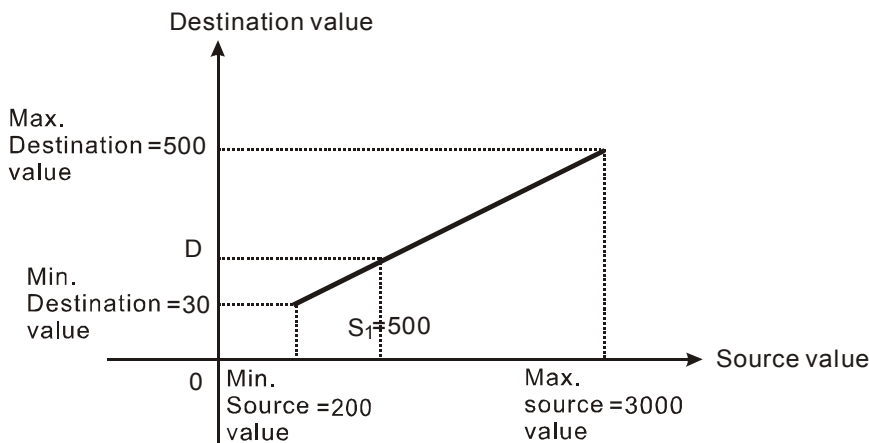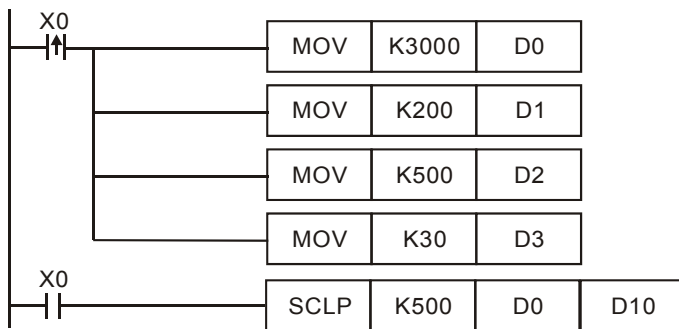
Program Example 2:

1. Assume S_1 = 500, max. source value D0 = 3,000, min. source value D1 = 200, max. destination value D2 = 30, and min. destination value D3 = 500. When X0 = On, SCLP instruction will be executed and obtain the proportional value at D10.

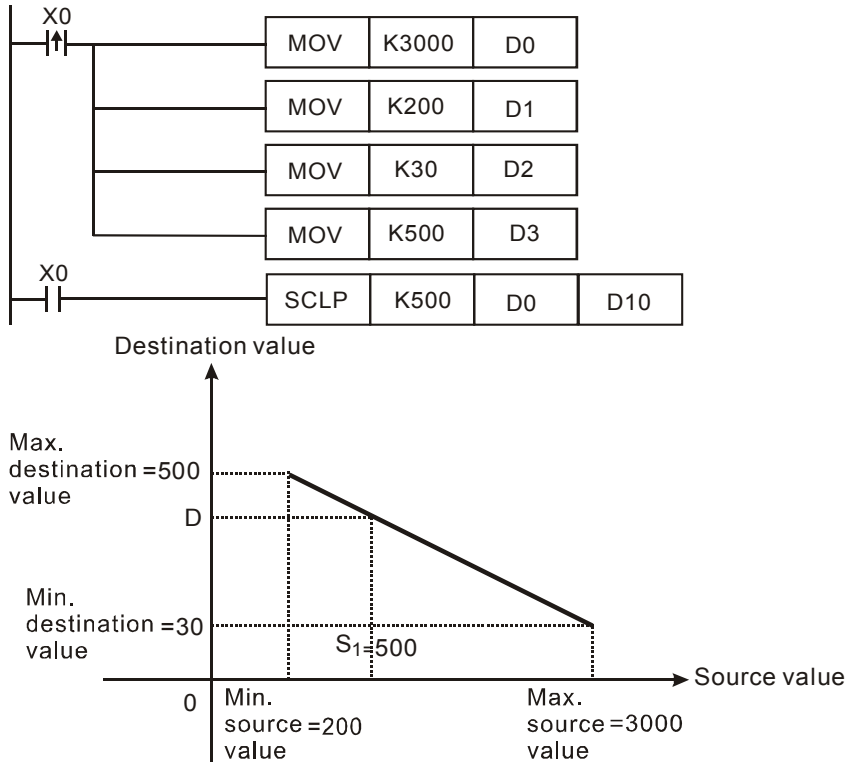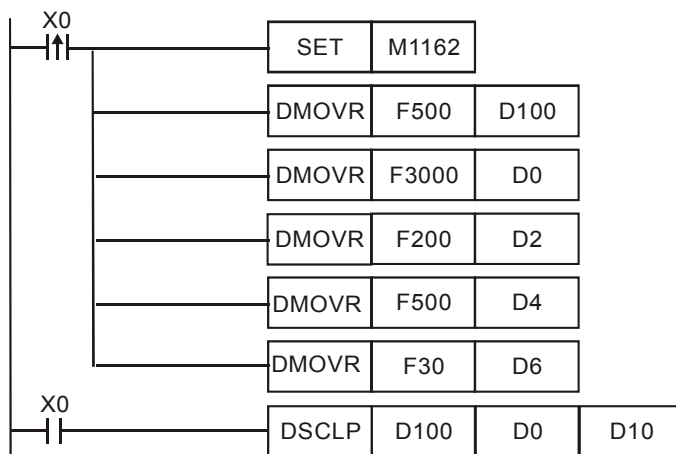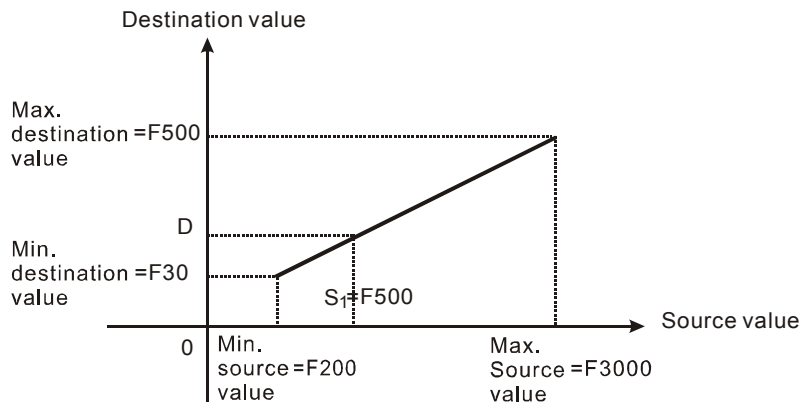2. Equation: D10 = [(500 – 200) × (30 – 500)] ÷ (3,000 – 200) + 500 = 449.64. Round off the result into an integer D10 = 450.

```
  X0
──┤↑├──────┬──── MOV │ K3000 │ D0
           │
           ├──── MOV │ K200  │ D1
           │
           ├──── MOV │ K30   │ D2
           │
           └──── MOV │ K500  │ D3
  X0
──┤ ├──────────── SCLP │ K500 │ D0 │ D10
```

Destination value

Max. destination = 500 value

D

Min. destination = 30 value

S_1=500

Source value

0 │ Min. source = 200 value

Max. source = 3000 value

Program Example 3:

1. Assume the source of 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, set up M1162, adopt floating point operation and execute DSCLP instruction. The proportional value will be obtained at D10.

2. Equation: D10 = [(F500 – F200) × (F500 – F30)] ÷ (F3000 – F200) + F30 = F80.35. Round off the result into an integer D10 = F80.

```
  X0
──┤↑├──────┬──── SET   │ M1162
           │
           ├──── DMOVR │ F500  │ D100
           │
           ├──── DMOVR │ F3000 │ D0
           │
           ├──── DMOVR │ F200  │ D2
           │
           ├──── DMOVR │ F500  │ D4
           │
           └──── DMOVR │ F30   │ D6
  X0
──┤ ├──────────── DSCLP │ D100 │ D0 │ D10
```

Remarks:

1. Range of S_1 for 16-bit instruction: max. source value ≥ S_1 ≥ min. source value; -32,768 ~ 32,767. If the value falls without 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 falls without 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; following the range of 32-bit floating point. If the value falls without the bounds, the bound value will be used for calculation.

4. Please be aware that the max. source value must > min. source value, but it is not necessary that max. destination value > min. destination value.

5. Only ES_V6.2, SA/SX_V1.6, SC_V1.4, EH2/SV_V1.0 and versions above support this instruction. EH series MPU does not support this instruction.

API	Mnemonic		Operands		Function		Controllers	
215~ 217	**D**	LD#	S₁	S₂	Contact Logical Operation LD#		ES/EX/SS	SA/SX/SC EH/SV

Type OP	Bit Devices				Word Devices												Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F		LD#: 5 steps
S₁					*	*	*	*	*	*	*	*	*	*	*		DLD#: 9 steps
S₂					*	*	*	*	*	*	*	*	*	*	*		

	PULSE								16-bit									32-bit							
	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S₁: Data source device 1   **S₂**: Data source device 2

Explanations:

1. See the specifications of each model for the range of operands.

2. This instruction compares the content in **S₁** and **S₂**. 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.

3. LD#   (**#: &, |, ^**) instruction is used for direct connection with BUS.

API No.	16-bit instruction	32-bit instruction	Continuity condition	No-continuity condition
215	LD&	**D**LD&	$S_1 \& S_2 \neq 0$	$S_1 \& S_2 = 0$
216	LD|	**D**LD|	$S_1 \mid S_2 \neq 0$	$S_1 \mid S_2 = 0$
217	LD^	**D**LD^	$S_1 \verb|^| S_2 \neq 0$	$S_1 \verb|^| S_2 = 0$

4. **&:** Logical "AND" operation

5. **|:** Logical "OR" operation

6. **^:** Logical "XOR" operation

7. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DLD#). If 16-bit instructions (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 of C0 and C10 ≠ 0, Y10 = On.

2. When the result of logical OR operation of D200 and D300 ≠ 0 and X1 = On, Y11 = On will be retained.

3. When the result of logical XOR operation of C201 and C200 ≠ 0 or M3 = On, M50 = On.
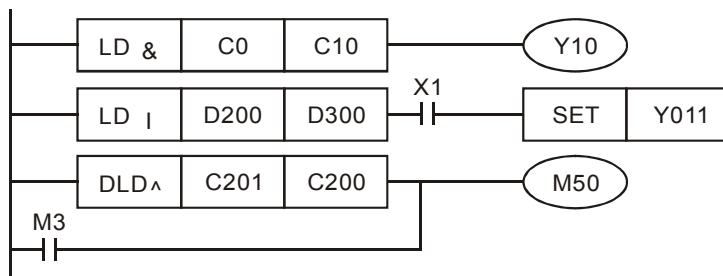
API	Mnemonic		Operands		Function		Controllers	
218~ 220	D	AND#	S_1 S_2		Contact Logical Operation AND#		ES/EX/SS	SA/SX/SC	EH/SV

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	AND#: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DAND#: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE									16-bit									32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV		

Operands:

S₁: Data source device 1   **S₂**: Data source device 2

Explanations:

1. See the specifications of each model for the range of operands.

2. This instruction compares the content in **S₁** and **S₂**. 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.

3. AND# (**#:** &, |, ^) is an operation instruction used on series contacts.

API No.	16-bit instruction	32-bit instruction	Continuity condition	No-continuity condition
218	AND&	DAND&	S1 & S2≠0	S1 & S2=0
219	AND|	DAND|	S1 | S2≠0	S1 | S2=0
220	AND^	DAND^	S1 ^ S2≠0	S1 ^ S2=0

4. **&:** Logical "AND" operation

5. **|:** Logical "OR" operation

6. **^:** Logical "XOR" operation

7. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DAND#). If 16-bit instructions (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 of C0 and C10 ≠ 0, Y10 = On.

2. When X1 = Off and the result of logical OR operation of D10 and D0 ≠ 0 and X1 = On, Y11 = On will be retained.

3. When X2 = On and the result of logical XOR operation of 32-bit register D200 (D201) and 32-bit register D100 (D101) ≠ 0 or M3 = On, M50 = On.
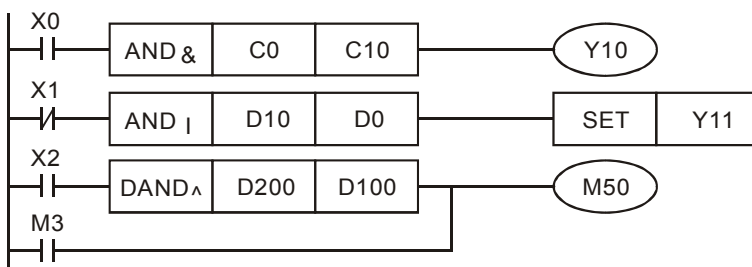
API	Mnemonic		Operands		Function	Controllers
221~ 223	D	OR#	S_1 S_2		Contact Logical operation OR#	ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	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
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Data source device 1 S_2: Data source device 2

Explanations:

1. See the specifications of each model for the range of operands.

2. This instruction compares 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.

3. OR# (**#:** &, |, ^) is an operation instruction used on parallel contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
221	OR&	**D**OR&	S_1 & $S_2 \neq 0$	S_1 & $S_2 = 0$
222	OR|	**D**OR|	S_1 | $S_2 \neq 0$	S_1 | $S_2 = 0$
223	OR^	**D**OR^	S_1 ^ $S_2 \neq 0$	S_1 ^ $S_2 = 0$

4. **&:** Logical "AND" operation

5. **|:** Logical "OR" operation

6. **^:** Logical "XOR" operation

7. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DOR#). If 16-bit instructions (OR#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X1 = On and the result of logical AND operation of C0 and C10 ≠ 0, Y10 = On.

2. M60 will be On when X2 = On and M30 = On, or the result of logical OR operation of 32-bit register D10 (D11) and 32-bit register D20 (D21) ≠ 0, or the result of logical XOR operation of 32-bit register D200 (D201) and 32-bit counter C235 ≠ 0.
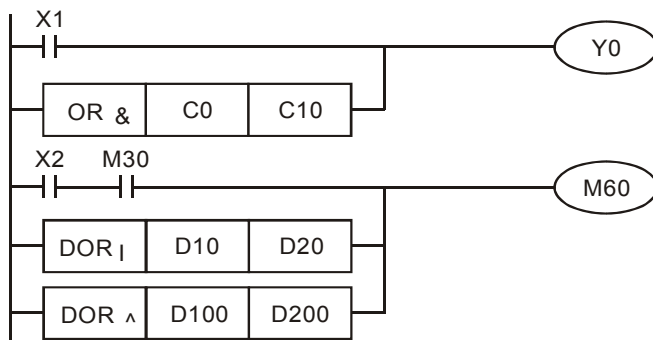
API	Mnemonic		Operands		Function		Controllers	
224~ 230	D	LD※	S_1 S_2		LoaD Compare		ES/EX/SS SA/SX/SC EH/SV	

Type OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	LD※: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DLD※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE	16-bit	32-bit
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S_1: Data source device 1   **S_2**: Data source device 2
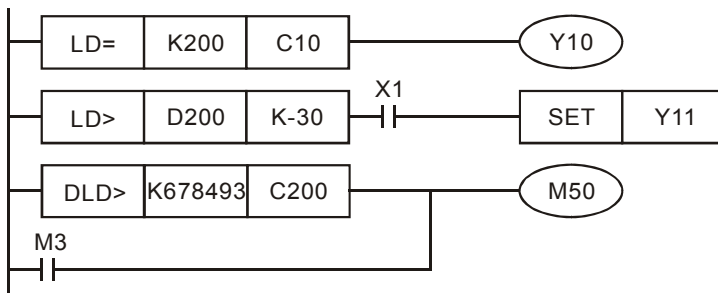
Explanations:

1. See the specifications of each model for the range of operands.

2. 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.

3. LD※ (※**: =, >, <, <>, ≤, ≥**) instruction is used for direct connection with BUS.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
224	LD＝	**D**LD＝	$S_1 = S_2$	$S_1 \neq S_2$
225	LD＞	**D**LD＞	$S_1 > S_2$	$S_1 \leq S_2$
226	LD＜	**D**LD＜	$S_1 < S_2$	$S_1 \geq S_2$
228	LD＜＞	**D**LD＜＞	$S_1 \neq S_2$	$S_1 = S_2$
229	LD＜＝	**D**LD＜＝	$S_1 \leq S_2$	$S_1 > S_2$
230	LD＞＝	**D**LD＞＝	$S_1 \geq S_2$	$S_1 < S_2$

4. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DLD※). If 16-bit instructions (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, Y10 = On.

2. When the content in D200 > K-30 and X1 = On, Y11= On will be retained.

3. When the content in C200 < K678,493 or M3 = On, M50 = On.

```
├──┤ LD= │ K200 │ C10 ├─────────────( Y10 )
│                                  X1
├──┤ LD> │ D200 │ K-30 ├──────┤ ├──┤ SET │ Y11 │
│
├──┤ DLD> │ K678493 │ C200 ├────┬────( M50 )
│  M3                            │
└──┤ ├────────────────────────────┘
```

API	Mnemonic		Operands	Function		Controllers
232~ 238	D AND※		**S₁** **S₂**	AND Compare		ES/EX/SS SA/SX/SC EH/SV

Type	Bit Devices				Word Devices											Program Steps
OP	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	AND※: 5 steps
S₁					*	*	*	*	*	*	*	*	*	*	*	DAND※: 9 steps
S₂					*	*	*	*	*	*	*	*	*	*	*	

	PULSE	16-bit	32-bit
	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Data source device 1   **S₂**: Data source device 2

Explanations:

1. See the specifications of each model for the range of operands.

2. This instruction compares the content in **S₁** and **S₂**. 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.

3. AND※ (※**:** =, >, <, <>, ≤, ≥) is a comparison instruction is used on series contacts.

API No.	16 –bit instruction	32 –bit instruction	Continuity condition	No-continuity condition
232	AND=	**D**AND=	$S_1 = S_2$	$S_1 \neq S_2$
233	AND>	**D**AND>	$S_1 > S_2$	$S_1 \leq S_2$
234	AND<	**D**AND<	$S_1 < S_2$	$S_1 \geq S_2$
236	AND<>	**D**AND<>	$S_1 \neq S_2$	$S_1 = S_2$
237	AND<=	**D**AND<=	$S_1 \leq S_2$	$S_1 > S_2$
238	AND>=	**D**AND>=	$S_1 \geq S_2$	$S_1 < S_2$

4. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DAND※). If 16-bit instructions (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, Y10 = On.

2. When X1 = Off and the content in D0 ≠ K-10, Y11= On will be retained.

3. When X2 = On and the content in 32-bit register D0 (D11) < 678,493 or M3 = On, M50 = On.
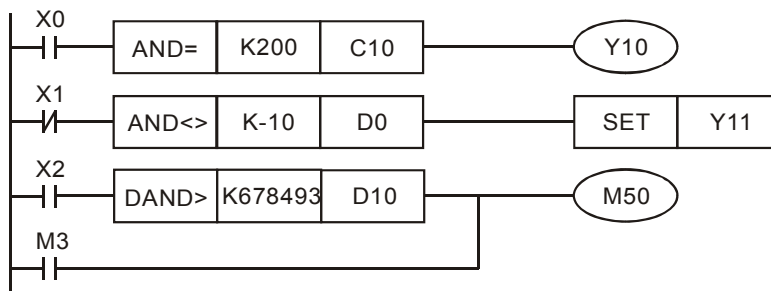
API	Mnemonic		Operands		Function		Controllers
240~ 246	D	OR※	S_1 S_2		OR Compare		ES/EX/SS SA/SX/SC EH/SV

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
ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV	ES EX SS SA SX SC EH SV

Operands:

S₁: Data source device 1   **S₂**: Data source device 2

Explanations:

1. See the specifications of each model for the range of operands.

2. This instruction compares the content in **S₁** and **S₂**. 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.

3. OR※ (※**:** =, >, <, <>, ≤, ≥) is an comparison instruction used on parallel contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
240	OR＝	**D**OR＝	$S_1＝S_2$	$S_1≠S_2$
241	OR＞	**D**OR＞	$S_1＞S_2$	$S_1≦S_2$
242	OR＜	**D**OR＜	$S_1＜S_2$	$S_1≧S_2$
244	OR＜＞	**D**OR＜＞	$S_1≠S_2$	$S_1＝S_2$
245	OR＜＝	**D**OR＜＝	$S_1≦S_2$	$S_1＞S_2$
246	OR＞＝	**D**OR＞＝	$S_1≧S_2$	$S_1＜S_2$

4. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DOR※). If 16-bit instructions (OR※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X1 = On and the present value of C10 = K200, Y0 = On.

2. M60 will be On when X2 = On, M30 = On and the content in 32-bit register D100 (D101) ≥ K100,000.