Cat. No. W384-E1-1

生産終了
DISCONTINUED

SYSMAC

CS1W-HIO01/HCP22/HCA22

Customizable Counter Units

PROGRAMMING MANUAL

OMRON

Read and Understand this Manual

Please read and understand this manual before using the product. Please consult your OMRON representative if you have any questions or comments.

Warranty and Limitations of Liability

WARRANTY

OMRON's exclusive warranty is that the products are free from defects in materials and workmanship for a period of one year (or other period if specified) from date of sale by OMRON.

OMRON MAKES NO WARRANTY OR REPRESENTATION, EXPRESS OR IMPLIED, REGARDING NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR PARTICULAR PURPOSE OF THE PRODUCTS. ANY BUYER OR USER ACKNOWLEDGES THAT THE BUYER OR USER ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE. OMRON DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED.

LIMITATIONS OF LIABILITY

OMRON SHALL NOT BE RESPONSIBLE FOR SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR COMMERCIAL LOSS IN ANY WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED ON CONTRACT, WARRANTY, NEGLIGENCE, OR STRICT LIABILITY.

In no event shall the responsibility of OMRON for any act exceed the individual price of the product on which liability is asserted.

IN NO EVENT SHALL OMRON BE RESPONSIBLE FOR WARRANTY, REPAIR, OR OTHER CLAIMS REGARDING THE PRODUCTS UNLESS OMRON'S ANALYSIS CONFIRMS THAT THE PRODUCTS WERE PROPERLY HANDLED, STORED, INSTALLED, AND MAINTAINED AND NOT SUBJECT TO CONTAMINATION, ABUSE, MISUSE, OR INAPPROPRIATE MODIFICATION OR REPAIR.

Application Considerations

SUITABILITY FOR USE

OMRON shall not be responsible for conformity with any standards, codes, or regulations that apply to the combination of products in the customer's application or use of the products.

At the customer's request, OMRON will provide applicable third party certification documents identifying ratings and limitations of use that apply to the products. This information by itself is not sufficient for a complete determination of the suitability of the products in combination with the end product, machine, system, or other application or use.

The following are some examples of applications for which particular attention must be given. This is not intended to be an exhaustive list of all possible uses of the products, nor is it intended to imply that the uses listed may be suitable for the products:

- Outdoor use, uses involving potential chemical contamination or electrical interference, or conditions or uses not described in this manual.
- Nuclear energy control systems, combustion systems, railroad systems, aviation systems, medical equipment, amusement machines, vehicles, safety equipment, and installations subject to separate industry or government regulations.
- Systems, machines, and equipment that could present a risk to life or property.

Please know and observe all prohibitions of use applicable to the products.

NEVER USE THE PRODUCTS FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESIGNED TO ADDRESS THE RISKS, AND THAT THE OMRON PRODUCTS ARE PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.

PROGRAMMABLE PRODUCTS

OMRON shall not be responsible for the user's programming of a programmable product, or any consequence thereof.

Disclaimers

CHANGE IN SPECIFICATIONS

Product specifications and accessories may be changed at any time based on improvements and other reasons.

It is our practice to change model numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the products may be changed without any notice. When in doubt, special model numbers may be assigned to fix or establish key specifications for your application on your request. Please consult with your OMRON representative at any time to confirm actual specifications of purchased products.

DIMENSIONS AND WEIGHTS

Dimensions and weights are nominal and are not to be used for manufacturing purposes, even when tolerances are shown.

PERFORMANCE DATA

Performance data given in this manual is provided as a guide for the user in determining suitability and does not constitute a warranty. It may represent the result of OMRON's test conditions, and the users must correlate it to actual application requirements. Actual performance is subject to the OMRON Warranty and Limitations of Liability.

ERRORS AND OMISSIONS

The information in this manual has been carefully checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical, or proofreading errors, or omissions.

CS1W-HIO01/HCP22/HCA22

Customizable Counter Units

Programming Manual

Produced January 2001

Notice:

OMRON products are manufactured for use according to proper procedures by a qualified operator and only for the purposes described in this manual.

The following conventions are used to indicate and classify precautions in this manual. Always heed the information provided with them. Failure to heed precautions can result in injury to people or damage to property.

⚠ **DANGER** Indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠ **WARNING** Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.

⚠ **Caution** Indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury, or property damage.

OMRON Product References

All OMRON products are capitalized in this manual. The word “Unit” is also capitalized when it refers to an OMRON product, regardless of whether or not it appears in the proper name of the product.

The abbreviation “Ch,” which appears in some displays and on some OMRON products, often means “word” and is abbreviated “Wd” in documentation in this sense.

The abbreviation “PC” means Programmable Controller and is not used as an abbreviation for anything else.

Visual Aids

The following headings appear in the left column of the manual to help you locate different types of information.

Note Indicates information of particular interest for efficient and convenient operation of the product.

Reference Indicates supplementary information on related topics that may be of interest to the user.

1, 2, 3... 1. Indicates lists of one sort or another, such as procedures, checklists, etc.

No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice. Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

TABLE OF CONTENTS

PRECAUTIONS . . . .

1 Intended Audience . . . .
2 General Precautions . . . .
3 Safety Precautions . . . .
4 Operating Environment Precautions . . . .
5 Application Precautions . . . .
6 Data Backup . . . .
7 Conformance to EC Directives . . . .

SECTION 1
Memory Areas . . . .

SECTION 2
Instruction Set . . . .

2-1 Instruction Tables . . . .
2-2 Sequence Input Instructions . . . .
2-3 Sequence Output Instructions . . . .
2-4 Sequence Control Instructions . . . .
2-5 Timer and Counter Instructions . . . .
2-6 Data Shift Instructions . . . .
2-7 Data Movement Instructions . . . .
2-8 Comparison Instructions . . . .
2-9 Conversion Instructions . . . .
2-10 Data Control Instructions . . . .
2-11 Special Instructions . . . .
2-12 Symbol Math Instructions . . . .
2-13 Table Data Processing Instructions . . . .
2-14 Special Math Instructions . . . .
2-15 Logic Instructions . . . .
2-16 Increment/Decrement Instructions . . . .
2-17 Subroutine Instructions . . . .
2-18 Interrupt Control Instructions . . . .
2-19 High-Speed Counter/Pulse Output Instructions . . . .
2-20 I/O Instructions . . . .
2-21 Step Instructions:
STEP DEFINE and STEP START–STEP(08)/SNXT(09) . . . .
2-22 User Error Instructions:
FAILURE ALARM AND RESET – FAL(06) and
SEVERE FAILURE ALARM – FALS(07) . . . .

Index . . . .

Revision History . . . .

About this Manual:

This manual describes the memory areas and ladder programming instructions of the CS1W-HIO01, CS1W-HCP22, and CS1W-HCA22 Customizable Counter Units and includes the sections described below. The Customizable Counter Units provide both normal contact I/O points with special I/O points to provide ideal control capabilities for many applications. The Customizable Counter Units are classified as CS1 Special I/O Units.

Please read this manual and all other manuals for the Customizable Counter Units listed below carefully and be sure you understand the information provided before attempting to program and or operate a Customizable Counter Unit.

Manual	Cat. No.	Contents
CS1W-HIO01/HCP22/HCA22 Customizable Counter Units Programming Manual (this manual)	W384	Describes the memory areas and programming instructions of the Customizable Counter Units.
CS1W-HIO01/HCP22/HCA22 Customizable Counter Unit Operation Manual	W378	Describes the hardware and software operation of the Customizable Counter Units.
SYSMAC WS02-CXP□□-E CX-Programmer User Manual	W361	Provide information on how to use the CX-Programmer, a Windows-based Programming Device that supports the CQM1H-series PCs.
CQM1H Series Programmable Controllers Operation Manual	W363	Describes Programming Console operations that can be used connected to the Customizable Counter Units.

Section 1 describes the memory areas that can be used in the Customizable Counter Units.

Section 2 describes the ladder programming instructions that can be used in the Customizable Counter Units.

⚠ **WARNING** Failure to read and understand the information provided in this manual may result in personal injury or death, damage to the product, or product failure. Please read each section in its entirety and be sure you understand the information provided in the section and related sections before attempting any of the procedures or operations given.

PRECAUTIONS

This section provides general precautions for using the CS1W-HIO01, CS1W-HCP22, and CS1W-HCA22 Customizable Counter Units.

The information contained in this section is important for the safe and reliable application of the Customizable Counter Units. You must read this section and understand the information contained before attempting to set up or operate a Customizable Counter Unit.

1 Intended Audience . . . .

2 General Precautions . . . .

3 Safety Precautions . . . .

4 Operating Environment Precautions . . . .

5 Application Precautions . . . .

6 Data Backup . . . .

6-1 Automatic Backup . . . .

6-2 User Programming . . . .

6-3 Backing Up DM Area to Flash Memory . . . .

7 Conformance to EC Directives . . . .

7-1 Applicable Directives . . . .

7-2 Concepts . . . .

7-3 Conformance to EC Directives . . . .

1 Intended Audience

This manual is intended for the following personnel, who must also have knowledge of electrical systems (an electrical engineer or the equivalent).

- Personnel in charge of installing FA systems.
- Personnel in charge of designing FA systems.
- Personnel in charge of managing FA systems and facilities.

2 General Precautions

The user must operate the product according to the performance specifications described in the operation manuals.

Before using the product under conditions which are not described in the manual or applying the product to nuclear control systems, railroad systems, aviation systems, vehicles, combustion systems, medical equipment, amusement machines, safety equipment, and other systems, machines, and equipment that may have a serious influence on lives and property if used improperly, consult your OMRON representative.

Make sure that the ratings and performance characteristics of the product are sufficient for the systems, machines, and equipment, and be sure to provide the systems, machines, and equipment with double safety mechanisms.

This manual provides information for programming and operating the Unit. Be sure to read this manual before attempting to use the Unit and keep this manual close at hand for reference during operation.

⚠ **WARNING** It is extremely important that a PC and all PC Units be used for the specified purpose and under the specified conditions, especially in applications that can directly or indirectly affect human life. You must consult with your OMRON representative before applying a PC System to the above-mentioned applications.

3 Safety Precautions

⚠ **WARNING** Do not attempt to take any Unit apart while the power is being supplied. Doing so may result in electric shock.

⚠ **WARNING** Do not touch any of the terminals or terminal blocks while the power is being supplied. Doing so may result in electric shock.

⚠ **WARNING** Do not attempt to disassemble, repair, or modify any Units. Any attempt to do so may result in malfunction, fire, or electric shock.

⚠ **WARNING** Do not touch the Power Supply Unit while power is being supplied or immediately after power has been turned OFF. Doing so may result in electric shock.

⚠ **WARNING** Provide safety measures in external circuits, i.e., not in the Programmable Controller (CPU Unit including associated Units; referred to as “PC”), in order to ensure safety in the system if an abnormality occurs due to malfunction of the PC or another external factor affecting the PC operation. Not doing so may result in serious accidents.

- Emergency stop circuits, interlock circuits, limit circuits, and similar safety measures must be provided in external control circuits.
- The PC will turn OFF all outputs when its self-diagnosis function detects any error or when a severe failure alarm (FALS) instruction is executed. As a countermeasure for such errors, external safety measures must be provided to ensure safety in the system.
- The PC outputs may remain ON or OFF due to deposition or burning of the output relays or destruction of the output transistors. As a countermeasure for such problems, external safety measures must be provided to ensure safety in the system.
- When the 24-VDC output (service power supply to the PC) is overloaded or short-circuited, the voltage may drop and result in the outputs being turned OFF. As a countermeasure for such problems, external safety measures must be provided to ensure safety in the system.

⚠ **Caution** Execute online edit only after confirming that no adverse effects will be caused by extending the cycle time. Otherwise, the input signals may not be readable.

⚠ **Caution** Confirm safety at the destination node before transferring a program to another node or changing contents of the I/O memory area. Doing either of these without confirming safety may result in injury.

⚠ **Caution** Tighten the screws on the terminal block of the AC power supply to the torque specified in the operation manual. The loose screws may result in burning or malfunction.

4 Operating Environment Precautions

⚠ **Caution** Do not operate the control system in the following locations:

- Locations subject to direct sunlight.
- Locations subject to temperatures or humidity outside the range specified in the specifications.
- Locations subject to condensation as the result of severe changes in temperature.
- Locations subject to corrosive or flammable gases.
- Locations subject to dust (especially iron dust) or salts.
- Locations subject to exposure to water, oil, or chemicals.
- Locations subject to shock or vibration.

⚠ **Caution** Take appropriate and sufficient countermeasures when installing systems in the following locations:

- Locations subject to static electricity or other forms of noise.
- Locations subject to strong electromagnetic fields.
- Locations subject to possible exposure to radioactivity.
- Locations close to power supplies.

⚠ **Caution** The operating environment of the PC System can have a large effect on the longevity and reliability of the system. Improper operating environments can lead to malfunction, failure, and other unforeseeable problems with the PC System. Be sure that the operating environment is within the specified conditions at installation and remains within the specified conditions during the life of the system.

5 Application Precautions

⚠ **WARNING** Always heed these precautions. Failure to abide by the following precautions could lead to serious or possibly fatal injury.

- Always connect to a ground of 100 Ω or less when installing the Units. Not connecting to a ground of 100 Ω or less may result in electric shock.
- A ground of 100 Ω or less must be installed when shorting the GR and LG terminals on the Power Supply Unit.
- Always turn OFF the power supply to the PC before attempting any of the following. Not turning OFF the power supply may result in malfunction or electric shock.
  - Mounting or dismounting Power Supply Units, I/O Units, CPU Units, Inner Boards, or any other Units.
  - Assembling the Units.
  - Setting DIP switches or rotary switches.
  - Connecting cables or wiring the system.
  - Connecting or disconnecting the connectors.

⚠ **Caution** Failure to abide by the following precautions could lead to faulty operation of the PC or the system, or could damage the PC or PC Units. Always heed these precautions.

- Always turn ON power to the PC before turning ON power to the control system. If the PC power supply is turned ON after the control power supply, temporary errors may result in control system signals because the output terminals on DC Output Units and other Units will momentarily turn ON when power is turned ON to the PC.
- Fail-safe measures must be taken by the customer to ensure safety in the event that outputs from Output Units remain ON as a result of internal circuit failures, which can occur in relays, transistors, and other elements.
- Fail-safe measures must be taken by the customer to ensure safety in the event of incorrect, missing, or abnormal signals caused by broken signal lines, momentary power interruptions, or other causes.
- Interlock circuits, limit circuits, and similar safety measures in external circuits (i.e., not in the Programmable Controller) must be provided by the customer.
- Always use the power supply voltages specified in the operation manuals. An incorrect voltage may result in malfunction or burning.
- Take appropriate measures to ensure that the specified power with the rated voltage and frequency is supplied in places where the power supply is unstable. An incorrect power supply may result in malfunction.
- Install external breakers and take other safety measures against short-circuiting in external wiring. Insufficient safety measures against short-circuiting may result in burning.
- Do not apply voltages to the Input Units in excess of the rated input voltage. Excess voltages may result in burning.
- Do not apply voltages or connect loads to the Output Units in excess of the maximum switching capacity. Excess voltage or loads may result in burning.
- Disconnect the functional ground terminal when performing withstand voltage tests. Not disconnecting the functional ground terminal may result in burning.
- Install the Units properly as specified in the operation manuals. Improper installation of the Units may result in malfunction.

- Be sure that all the mounting screws, terminal screws, and cable connector screws are tightened to the torque specified in the relevant manuals. Incorrect tightening torque may result in malfunction.
- Leave the label attached to the Unit when wiring. Removing the label may result in malfunction if foreign matter enters the Unit.
- Remove the label after the completion of wiring to ensure proper heat dissipation. Leaving the label attached may result in malfunction.
- Use crimp terminals for wiring. Do not connect bare stranded wires directly to terminals. Connection of bare stranded wires may result in burning.
- Wire all connections correctly.
- Double-check all wiring and switch settings before turning ON the power supply. Incorrect wiring may result in burning.
- Mount Units only after checking terminal blocks and connectors completely.
- Be sure that the terminal blocks, Memory Units, expansion cables, and other items with locking devices are properly locked into place. Improper locking may result in malfunction.
- Check switch settings, the contents of the DM Area, and other preparations before starting operation. Starting operation without the proper settings or data may result in an unexpected operation.
- Check the user program for proper execution before actually running it on the Unit. Not checking the program may result in an unexpected operation.
- Confirm that no adverse effect will occur in the system before attempting any of the following. Not doing so may result in an unexpected operation.
    - Changing the operating mode of the PC.
    - Force-setting/force-resetting any bit in memory.
    - Changing the present value of any word or any set value in memory.
- Resume operation only after transferring to the new CPU Unit the contents of the DM Area, HR Area, and other data required for resuming operation. Not doing so may result in an unexpected operation.
- Do not pull on the cables or bend the cables beyond their natural limit. Doing either of these may break the cables.
- Do not place objects on top of the cables or other wiring lines. Doing so may break the cables.
- When replacing parts, be sure to confirm that the rating of a new part is correct. Not doing so may result in malfunction or burning.
- Before touching a Unit, be sure to first touch a grounded metallic object in order to discharge any static build-up. Not doing so may result in malfunction or damage.
- When transporting or storing circuit boards, cover them in antistatic material to protect them from static electricity and maintain the proper storage temperature.
- Do not touch circuit boards or the components mounted to them with your bare hands. There are sharp leads and other parts on the boards that may cause injury if handled improperly.
- Data in the DM Area, error history, EM Area, or Timer/Counter Area may become corrupted if power is not supplied for an extended period of time. Program the PC to check SR 24914 before starting operation. If SR 24914 is ON, the memory areas that are normally held during power interruptions will not have been held properly (i.e., the data will be corrupted). (The data in the DM Area can be backed up to flash memory by turning ON SR 25200.)

6 Data Backup

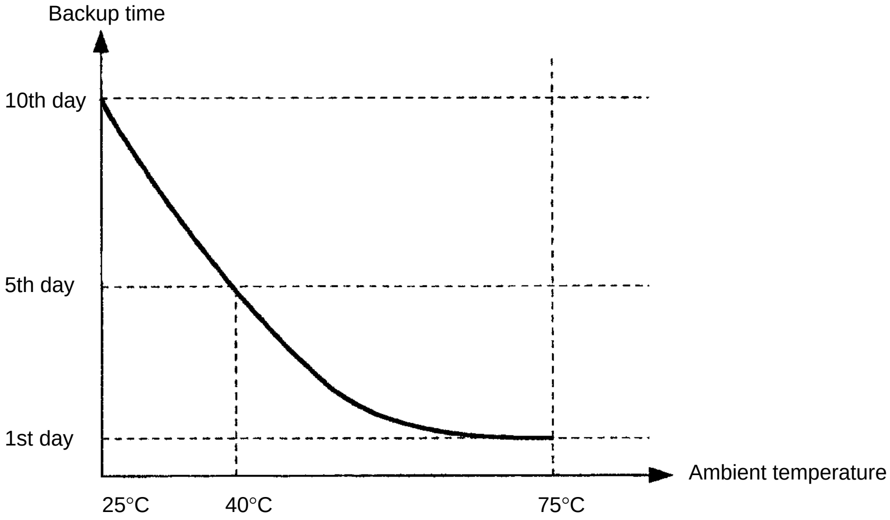
6-1 Automatic Backup

Data in the Customizable Counter Units is backed up either by a super capacitor or flash memory, as listed in the following table.

Data	Data backup
DM Area (DM 0000 to DM 6143), EM Area (EM 0000 to EM 2047), error history (DM 6144 to DM 6199), and counter present values. A setting is provided to either enable or disable holding EM Area data. The default is to not hold the data.	RAM with super capacitor
User program, read-only DM Area (DM 6200 to DM 6599), Unit Setup Area (DM 6600 to DM 6655), expansion instructions information, read/write portion of DM Area (DM 0000 to DM 6143, see note.)	Flash memory

Note The contents of DM 0000 to DM 6143 are written to flash memory only when SR 25200 (DM Area Backup Bit) is turned ON.

The data in RAM is backed up by the super capacitor for 10 days at 25°C. The backup time varies with the ambient temperature as shown in the following graph.



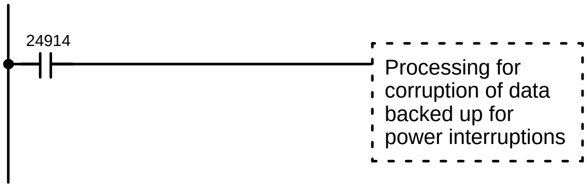
Note The times give above assume that the capacitor is completely charged. Power must be supply to the Unit for at least 15 minutes to completely charge the capacitor.

The data backed up by the capacitor will become unstable or corrupted if the backup time is exceeded.

6-2 User Programming

If the power supply is turned OFF for longer than the data backup time (10 days at 25°C), the data in the DM Area, EM Area, and Error Log, as well as counter present values, will be lost and any data that is read will be unstable.

If the power supply is to be turned OFF for an extended period of time, the contents of DM 0000 to DM 6143 can be backed up in flash memory. The Backup Data Corrupted Flag (SR 24914) can also be used as shown below to detect when backup data (i.e., data in the DM Area, EM Area, and Error Log, as well as counter present values) has become corrupted to perform appropriate error processing.

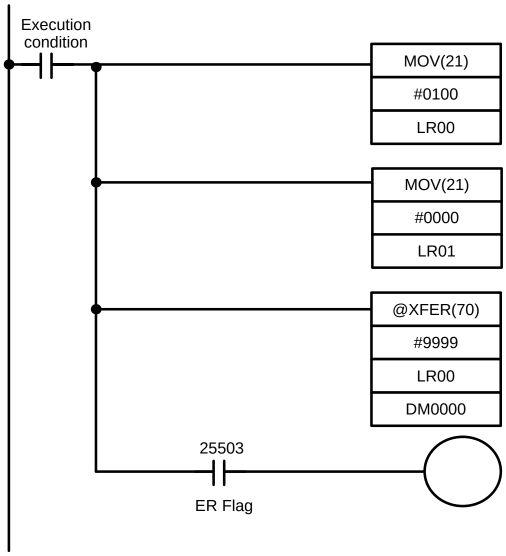

24914
Processing for corruption of data backed up for power interruptions

DM 0000 to DM 6143 (read/write portion of DM Area) can be backed up in flash memory by the user as described in the next section.

6-3 Backing Up DM Area to Flash Memory

The contents of DM 0000 to DM 6143 can be written to flash memory by turning ON SR 25200 (DM Flash Memory Backup Bit) in PROGRAM mode. (SR 25200 will turn OFF automatically when transfer has been completed.)

The data stored in flash memory can be read back to DM 0000 to DM 6143 by using the following type of programming.



7 Conformance to EC Directives

7-1 Applicable Directives

- EMC Directives
- Low Voltage Directive

7-2 Concepts

EMC Directives

OMRON devices that comply with EC Directives also conform to the related EMC standards so that they can be more easily built into other devices or machines. The actual products have been checked for conformity to EMC standards (see the following note). Whether the products conform to the standards in the system used by the customer, however, must be checked by the customer.

EMC-related performance of the OMRON devices that comply with EC Directives will vary depending on the configuration, wiring, and other conditions of the equipment or control panel in which the OMRON devices are installed. The customer must, therefore, perform final checks to confirm that devices and the overall machine conform to EMC standards.

Note Applicable EMC (Electromagnetic Compatibility) standards are as follows:

EMS (Electromagnetic Susceptibility): EN50082-2
EMI (Electromagnetic Interference): EN50081-2
(Radiated emission: 10-m regulations)

Low Voltage Directive

Always ensure that devices operating at voltages of 50 to 1,000 VAC or 75 to 1,500 VDC meet the required safety standards for the PC (EN61131-2).

7-3 Conformance to EC Directives

The CS1W-HIO01, CS1W-HCP22, and CS1W-HCA22 Customizable Counter Units comply with EC Directives. To ensure that the machine or device in which a CS1W-HIO01, CS1W-HCP22, or CS1W-HCA22 Customizable Counter Unit is used complies with EC directives, the Unit must be installed as follows:

1, 2, 3...

1. The CS1W-HIO01, CS1W-HCP22, and CS1W-HCA22 Customizable Counter Unit must be installed within a control panel.
2. Reinforced insulation or double insulation must be used for the CS1W-HIO01, CS1W-HCP22, or CS1W-HCA22 Customizable Counter Unit DC power supplies used for the communications and I/O power supplies.
3. CS1W-HIO01, CS1W-HCP22, and CS1W-HCA22 Customizable Counter Units complying with EC Directives also conform to the Common Emission Standard (EN50081-2). When a CS1W-HIO01, CS1W-HCP22, and CS1W-HCA22 Customizable Counter Unit is built into a machine, however, changes can occur, particularly for the radiated emission (10-m regulations), due to the structure of the machine, other connected devices, wiring, etc. The customer must, therefore, perform final checks to confirm that devices and the overall machine using a CS1W-HIO01, CS1W-HCP22, or CS1W-HCA22 Customizable Counter Unit conform to EC standards.

SECTION 1
Memory Areas

This section describes the memory areas that can be used in the Customizable Counter Units.

The following memory areas can be used with the Customizable Counter Units. Addresses not listed in the following table cannot be used as operations in the ladder programming instructions for the Customizable Counter Units.

Data area	Size	Words	Bits	Function
Input Area	12 bits	IR 000	IR 00000 to IR 00011	Bits in the Input Area are allocated to input terminals. These allocations are fixed and cannot be changed. IR 00000 to IR 00003 can be used either as normal inputs or as interrupt inputs. Interrupt inputs are used in Input Interrupt Mode or Counter Mode.
Output Area	8 bits	IR 001	IR 00100 to IR 00107	Bits in the Output Area are allocated to output terminals. These allocations are fixed and cannot be changed. IR 00108 to IR 00115 can also be used as work bits in programming.
Work Area	1,088 bits	IR 002 to IR 049	IR 00200 to IR 04915	Work bits do not have any specific function, and they can be freely used within the program.
		IR 200 to IR 219	IR 20000 to IR 21915	
SR Area	568 bits	SR 220 to SR 255	SR 22000 to SR 25507	These bits serve specific functions such as flags and control bits. SR 230 to SR 239 are used to exchange data with the I/O memory in the CPU Unit.
AR Area	448 bits	AR 00 to AR 27	AR 0000 to AR 2715	These bits serve specific functions such as flags and control bits.
TR Area	8 bits	---	TR 0 to TR 7	These bits are used to temporarily store ON/OFF status at program branches.
LR Area	256 bits	LR 00 to LR 31	LR 0000 to LR 3115	These bits are used to exchange data with the CPU Unit. Cyclic data transfers can be set up with user-specified words in the CPU Unit. Up to 32 I/O words of data can be exchanged. The settings for the LR Area links are made in DM 6601 to DM 6604 of the Customizable Counter Unit.
Timer/Counter Area	256 bits	TIM/CNT 000 to TIM/CNT 255 (timer/counter numbers)		The timer numbers in the Timer/Counter Area are allocated to create timers and counters. The same numbers are used for both timers and counters.
Read/Write portion of DM Area	6,144 words	DM 0000 to DM 6143	---	DM Area data can be read and written in word units only. Word values are retained when power is turned OFF or when the operating mode is switched. The contents of the DM Area can be backed up in flash memory by turning on a control bit (SR 25200). Data can be read from flash memory using XFER(70).
EM Area	2,048 words	EM 0000 to EM 2047	---	EM area data can be read and written in word units only. It is possible to set whether the EM Area is retained or cleared when power is turned OFF or when the operating mode is switched.

SECTION 2
Instruction Set

The Customizable Counter Units have a large programming instruction set that allows for easy programming for many applications. This section explains instructions individually and provides the ladder diagram symbol, data areas, and flags used with each.

2-1 Instruction Tables . . . .
  2-1-1 Instructions with Fixed Function Codes . . . .
  2-1-2 Expansion Instructions . . . .
  2-1-3 Alphabetic List by Mnemonic . . . .
2-2 Sequence Input Instructions . . . .
  2-2-1 LOAD, LOAD NOT, AND, AND NOT, OR, and OR NOT . . . .
  2-2-2 AND LOAD and OR LOAD . . . .
2-3 Sequence Output Instructions . . . .
  2-3-1 OUTPUT and OUTPUT NOT – OUT and OUT NOT . . . .
  2-3-2 SET and RESET – SET and RSET . . . .
  2-3-3 KEEP – KEEP(11) . . . .
  2-3-4 DIFFERENTIATE UP and DOWN – DIFU(13) and DIFD(14) . . . .
2-4 Sequence Control Instructions . . . .
  2-4-1 NO OPERATION – NOP(00) . . . .
  2-4-2 END – END(01) . . . .
  2-4-3 INTERLOCK and INTERLOCK CLEAR – IL(02) and ILC(03) . . . .
  2-4-4 JUMP and JUMP END – JMP(04) and JME(05) . . . .
2-5 Timer and Counter Instructions . . . .
  2-5-1 TIMER – TIM . . . .
  2-5-2 COUNTER – CNT . . . .
  2-5-3 REVERSIBLE COUNTER – CNTR(12) . . . .
  2-5-4 HIGH-SPEED TIMER – TIMH(15) . . . .
  2-5-5 ONE-MS TIMER – TMHH(—) . . . .
2-6 Data Shift Instructions . . . .
  2-6-1 SHIFT REGISTER – SFT(10) . . . .
  2-6-2 WORD SHIFT – WSFT(16) . . . .
  2-6-3 ARITHMETIC SHIFT LEFT – ASL(25) . . . .
  2-6-4 ARITHMETIC SHIFT RIGHT – ASR(26) . . . .
  2-6-5 ROTATE LEFT – ROL(27) . . . .
  2-6-6 ROTATE RIGHT – ROR(28) . . . .
  2-6-7 ONE DIGIT SHIFT LEFT – SLD(74) . . . .
  2-6-8 ONE DIGIT SHIFT RIGHT – SRD(75) . . . .
  2-6-9 REVERSIBLE SHIFT REGISTER – SFTR(84) . . . .
  2-6-10 ASYNCHRONOUS SHIFT REGISTER – ASFT(17) . . . .
2-7 Data Movement Instructions . . . .
  2-7-1 MOVE – MOV(21) . . . .
  2-7-2 MOVE NOT – MVN(22) . . . .
  2-7-3 DOUBLE MOVE – MOVL(—) . . . .
  2-7-4 BLOCK TRANSFER – XFER(70) . . . .
  2-7-5 BLOCK SET – BSET(71) . . . .
  2-7-6 DATA EXCHANGE – XCHG(73) . . . .
  2-7-7 SINGLE WORD DISTRIBUTE – DIST(80) . . . .
  2-7-8 DATA COLLECT – COLL(81) . . . .
  2-7-9 MOVE BIT – MOVB(82) . . . .
  2-7-10 MOVE DIGIT – MOVD(83) . . . .

2-8 Comparison Instructions . . . .
2-8-1 COMPARE – CMP(20) . . . .
2-8-2 TABLE COMPARE – TCMP(85) . . . .
2-8-3 BLOCK COMPARE – BCMP(68) . . . .
2-8-4 DOUBLE COMPARE – CMPL(60) . . . .
2-8-5 SIGNED BINARY COMPARE – CPS(—) . . . .
2-8-6 DOUBLE SIGNED BINARY COMPARE – CPSL(—) . . . .
2-8-7 AREA RANGE COMPARE – ZCP(—) . . . .
2-8-8 DOUBLE AREA RANGE COMPARE – ZCPL(—) . . . .
2-9 Conversion Instructions . . . .
2-9-1 BCD-TO-BINARY – BIN(23) . . . .
2-9-2 BINARY-TO-BCD – BCD(24) . . . .
2-9-3 DOUBLE BCD-TO-DOUBLE BINARY – BINL(58) . . . .
2-9-4 DOUBLE BINARY-TO-DOUBLE BCD – BCDL(59) . . . .
2-9-5 2'S COMPLEMENT – NEG(—) . . . .
2-9-6 DOUBLE 2'S COMPLEMENT – NEGL(—) . . . .
2-10 Data Control Instructions . . . .
2-10-1 SCALING – SCL(66) . . . .
2-10-2 SIGNED BINARY TO BCD SCALING – SCL2(—) . . . .
2-10-3 BCD TO SIGNED BINARY SCALING – SCL3(—) . . . .
2-10-4 AVERAGE VALUE – AVG(—) . . . .
2-11 Special Instructions . . . .
2-11-1 SET CARRY – STC(40) . . . .
2-11-2 CLEAR CARRY – CLC(41) . . . .
2-12 Symbol Math Instructions . . . .
2-12-1 BCD ADD – ADD(30) . . . .
2-12-2 BCD SUBTRACT – SUB(31) . . . .
2-12-3 BCD MULTIPLY – MUL(32) . . . .
2-12-4 BCD DIVIDE – DIV(33) . . . .
2-12-5 DOUBLE BCD ADD – ADDL(54) . . . .
2-12-6 DOUBLE BCD SUBTRACT – SUBL(55) . . . .
2-12-7 DOUBLE BCD MULTIPLY – MULL(56) . . . .
2-12-8 DOUBLE BCD DIVIDE – DIVL(57) . . . .
2-12-9 BINARY ADD – ADB(50) . . . .
2-12-10 BINARY SUBTRACT – SBB(51) . . . .
2-12-11 BINARY MULTIPLY – MLB(52) . . . .
2-12-12 BINARY DIVIDE – DVB(53) . . . .
2-12-13 DOUBLE BINARY ADD – ADBL(47) . . . .
2-12-14 DOUBLE BINARY SUBTRACT – SBBL(48) . . . .
2-12-15 SIGNED BINARY MULTIPLY – MBS(—) . . . .
2-12-16 DOUBLE SIGNED BINARY MULTIPLY – MBSL(—) . . . .
2-12-17 SIGNED BINARY DIVIDE – DBS(—) . . . .
2-12-18 DOUBLE SIGNED BINARY DIVIDE – DBSL(—) . . . .
2-13 Table Data Processing Instructions . . . .
2-13-1 FIND MAXIMUM – MAX(—) . . . .
2-13-2 FIND MINIMUM – MIN(—) . . . .
2-14 Special Math Instructions . . . .
2-14-1 ARITHMETIC PROCESS – APR(—) . . . .
2-14-2 BIT COUNTER – BCNT(67) . . . .
2-15 Logic Instructions . . . .
2-15-1 COMPLEMENT – COM(29) . . . .
2-15-2 LOGICAL AND – ANDW(34) . . . .
2-15-3 LOGICAL OR – ORW(35) . . . .
2-15-4 EXCLUSIVE OR – XORW(36) . . . .
2-15-5 EXCLUSIVE NOR – XNRW(37) . . . .

2-16 Increment/Decrement Instructions . . . .
2-16-1 BCD INCREMENT – INC(38) . . . .
2-16-2 BCD DECREMENT – DEC(39) . . . .
2-17 Subroutine Instructions . . . .
2-17-1 SUBROUTINE ENTER – SBS(91) . . . .
2-17-2 SUBROUTINE DEFINE and RETURN – SBN(92)/RET(93) . . . .
2-17-3 MACRO – MCRO(99) . . . .
2-18 Interrupt Control Instructions . . . .
2-18-1 INTERRUPT CONTROL – INT(89) . . . .
2-18-2 INTERVAL TIMER – STIM(69) . . . .
2-19 High-Speed Counter/Pulse Output Instructions . . . .
2-19-1 SET PULSES – PULS(65) . . . .
2-19-2 SPEED OUTPUT– SPED(64) . . . .
2-19-3 PULSE OUTPUT – PLS2(—) . . . .
2-19-4 ACCELERATION CONTROL – ACC(—) . . . .
2-19-5 REGISTER COMPARISON TABLE – CTBL(63) . . . .
2-19-6 MODE CONTROL – INI(61) . . . .
2-19-7 HIGH-SPEED COUNTER PV READ – PRV(62) . . . .
2-20 I/O Instructions . . . .
2-20-1 I/O REFRESH – IORF(97) . . . .
2-21 Step Instructions:
STEP DEFINE and STEP START–STEP(08)/SNXT(09) . . . .
2-22 User Error Instructions:
FAILURE ALARM AND RESET – FAL(06) and
SEVERE FAILURE ALARM – FALS(07) . . . .

2-1 Instruction Tables

This section provides tables of the instructions available in the Customizable Counter Unit. The first two tables can be used to find instructions by function code. The last table can be used to find instructions by mnemonic.

2-1-1 Instructions with Fixed Function Codes

The following table lists the instructions that have fixed function codes. Each instruction is listed by mnemonic and by instruction name. Use the numbers in the leftmost column as the left digit and the number in the column heading as the right digit of the function code. The @ symbol indicates instructions with differentiated forms.

Expansion instructions without default function codes must be allocated function codes to enable using them. Even the expansion instructions with default function codes have been omitted from the following table and space has been provided so that you can write in the ones you will be using. Refer to the next page for details on expansion instructions.

Left digit	Right digit									
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

0	**NOP**
	NO OPERATION
		END
		END
			IL
			INTERLOCK
				ILC
				INTERLOCK CLEAR
					JMP
					JUMP
						JME
						JUMP END
							(@) FAL
							FAILURE ALARM AND RESET
								FALS
								SEVERE FAILURE ALARM
									STEP
									STEP DEFINE
										SNXT
										STEP START

1	**SFT** SHIFT REGISTER	**KEEP** KEEP	**CNTR** REVERSIBLE COUNTER	**DIFU** DIFFERENTIATE UP	**DIFD** DIFFERENTIATE DOWN	**TIMH** HIGH-SPEED TIMER	**(@) WSFT** WORD SHIFT	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)
2	**CMP** COMPARE	**(@) MOV** MOVE	**(@) MVN** MOVE NOT	**(@) BIN** BCD TO BINARY	**(@) BCD** BINARY TO BCD	**(@) ASL** SHIFT LEFT	**(@) ASR** SHIFT RIGHT	**(@) ROL** ROTATE LEFT	**(@) ROR** ROTATE RIGHT	**(@) COM** COMPLEMENT
3	**(@) ADD** BCD ADD	**(@) SUB** BCD SUBTRACT	**(@) MUL** BCD MULTIPLY	**(@) DIV** BCD DIVIDE	**(@) ANDW** LOGICAL AND	**(@) ORW** LOGICAL OR	**(@) XORW** EXCLUSIVE OR	**(@) XNRW** EXCLUSIVE NOR	**(@) INC** INCREMENT	**(@) DEC** DECREMENT
4	**(@) STC** SET CARRY	**(@) CLC** CLEAR CARRY	---	---	---	---	---	(Expansion Instruction)	(Expansion Instruction)	---
5	**(@) ADB** BINARY ADD	**(@) SBB** BINARY SUBTRACT	**(@) MLB** BINARY MULTIPLY	**(@) DVB** BINARY DIVIDE	**(@) ADDL** DOUBLE BCD ADD	**(@) SUBL** DOUBLE BCD SUBTRACT	**(@) MULL** DOUBLE BCD MULTIPLY	**(@) DIVL** DOUBLE BCD DIVIDE	**(@) BINL** DOUBLE BCD-TO-DOUBLE BINARY	**(@) BCDL** DOUBLE BINARY-TO-DOUBLE BCD
6	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)
7	**(@) XFER** BLOCK TRANSFER	**(@) BSET** BLOCK SET	**(@) ROOT** SQUARE ROOT	**(@) XCHG** DATA EXCHANGE	**(@) SLD** ONE DIGIT SHIFT LEFT	**(@) SRD** ONE DIGIT SHIFT RIGHT	---	---	---	---
8	**(@) DIST** SINGLE WORD DISTRIBUTE	**(@) COLL** DATA COLLECT	**(@) MOVB** MOVE BIT	**(@) MOVD** MOVE DIGIT	**(@) SFTR** REVERSIBLE SHIFT REGISTER	**(@) TCMP** TABLE COMPARE	---	(Expansion Instruction)	(Expansion Instruction)	(Expansion Instruction)
9	**(@) SEND** NETWORK SEND	**(@) SBS** SUBROUTINE ENTRY	**SBN** SUBROUTINE DEFINE	**RET** SUBROUTINE RETURN	---	---	---	**(@) IORF** I/O REFRESH	---	**(@) MCRO** MACRO

2-1-2 Expansion Instructions

The expansion instructions that can be used are listed below, along with the default function codes that are assigned when the Customizable Counter Unit is shipped.

Name	Mnemonic	Default function code
ASYNCHRONOUS SHIFT REGISTER	ASFT	17
Not used.	FUN (See note 2.)	18
Not used.	FUN (See note 2.)	19
DOUBLE BINARY ADD	ADBL	47
DOUBLE BINARY SUBTRACT	SBBL	48
DOUBLE COMPARE	CMPL	60
MODE CONTROL	INI (See note 1.)	61
HIGH-SPEED COUNTER PV READ	PRV (See note 1.)	62
COMPARISON TABLE LOAD	CTBL (See note 1.)	63
SPEED OUTPUT	SPED (See note 1.)	64
SET PULSES	PULS (See note 1.)	65
SCALING	SCL	66
BIT COUNTER	BCNT	67
BLOCK COMPARE	BCMP	68
INTERVAL TIMER	STIM	69
DOUBLE 2’S COMPLEMENT	NEGL (See note 1.)	87
Not used.	FUN (See note 2.)	88
INTERRUPT CONTROL	INT	89

Name	Mnemonic	Default function code
ACCELERATION CONTROL	ACC	---
ARITHMETIC PROCESS	APR	---
AVERAGE VALUE	AVG	---
SIGNED BINARY COMPARE	CPS	---
DOUBLE SIGNED BINARY COMPARE	CPSL	---
SIGNED BINARY DIVIDE	DBS	---
DOUBLE SIGNED BINARY DIVIDE	DBSL	---
FIND MAXIMUM	MAX	---
SIGNED BINARY MULTIPLY	MBS	---
DOUBLE SIGNED BINARY MULTIPLY	MBSL	---
FIND MINIMUM	MIN	---
DOUBLE MOVE	MOVL	---
2’S COMPLEMENT	NEG	---
PULSE OUTPUT	PLS2	---
SIGNED BINARY TO BCD SCALING	SCL2	---
BCD TO SIGNED BINARY SCALING	SCL3	---
AREA RANGE COMPARE	ZCP	---
DOUBLE AREA RANGE COMPARE	ZCPL	---

Note
1. The default values depend on the model of the Customizable Counter Unit.
2. These instructions are supported by the CQM1H, but not by the Customizable Counter Unit. These instructions will be processed as NOPs if they are transferred to a Customizable Counter Unit.

2-1-3 Alphabetic List by Mnemonic

Dashes (“–”) in the *Code* column indicate expansion instructions, which do not have fixed function codes. “None” indicates instructions for which function codes are not used. The @ symbol indicates instructions with differentiated forms.

Mnemonic	Code	Words	Name	Page
ACC (@)	—	4	ACCELERATION CONTROL	117
ADB (@)	50	4	BINARY ADD	76
ADBL (@)	47	4	DOUBLE BINARY ADD	80
ADD (@)	30	4	BCD ADD	66
ADDL (@)	54	4	DOUBLE BCD ADD	72
AND	None	1	AND	10
AND LD	None	1	AND LOAD	11
AND NOT	None	1	AND NOT	10

Mnemonic	Code	Words	Name	Page
ANDW (@)	34	4	LOGICAL AND	93
APR (@)	—	4	ARITHMETIC PROCESS	89
ASFT(@)	17	4	ASYNCHRONOUS SHIFT REGISTER	31
ASL (@)	25	2	ARITHMETIC SHIFT LEFT	26
ASR (@)	26	2	ARITHMETIC SHIFT RIGHT	26
AVG	—	4	AVERAGE VALUE	64
BCD (@)	24	3	BINARY TO BCD	55
BCDL (@)	59	3	DOUBLE BINARY-TO-DOUBLE BCD	56
BCMP (@)	68	4	BLOCK COMPARE	46
BCNT (@)	67	4	BIT COUNTER	92
BIN (@)	23	3	BCD-TO-BINARY	54
BINL (@)	58	3	DOUBLE BCD-TO-DOUBLE BINARY	55
BSET (@)	71	4	BLOCK SET	37
CLC (@)	41	1	CLEAR CARRY	66
CMP	20	3	COMPARE	44
CMPL	60	4	DOUBLE COMPARE	48
CNT	None	2	COUNTER	20
CNTR	12	3	REVERSIBLE COUNTER	21
COLL (@)	81	4	DATA COLLECT	40
COM (@)	29	2	COMPLEMENT	92
CPS	—	4	SIGNED BINARY COMPARE	49
CPSL	—	4	DOUBLE SIGNED BINARY COMPARE	50
CTBL(@)	63	4	COMPARISON TABLE LOAD	121
DBS (@)	—	4	SIGNED BINARY DIVIDE	85
DBSL (@)	—	4	DOUBLE SIGNED BINARY DIVIDE	86
DEC (@)	39	2	BCD DECREMENT	96
DIFD	14	2	DIFFERENTIATE DOWN	13
DIFU	13	2	DIFFERENTIATE UP	13
DIST (@)	80	4	SINGLE WORD DISTRIBUTE	38
DIV (@)	33	4	BCD DIVIDE	70
DIVL (@)	57	4	DOUBLE BCD DIVIDE	75
DVB (@)	53	4	BINARY DIVIDE	79
END	01	1	END	14
FAL (@)	06	2	FAILURE ALARM AND RESET	131
FALS	07	2	SEVERE FAILURE ALARM	131
IL	02	1	INTERLOCK	15
ILC	03	1	INTERLOCK CLEAR	15
INC (@)	38	2	INCREMENT	95
INI (@)	61	4	MODE CONTROL	125
INT (@)	89	4	INTERRUPT CONTROL	103
IORF (@)	97	3	I/O REFRESH	128
JME	05	2	JUMP END	17
JMP	04	2	JUMP	17
KEEP	11	2	KEEP	13
LD	None	1	LOAD	10
LD NOT	None	1	LOAD NOT	10
MAX (@)	—	4	FIND MAXIMUM	87
MBS (@)	—	4	SIGNED BINARY MULTIPLY	83
MBSL (@)	—	4	DOUBLE SIGNED BINARY MULTIPLY	84

Mnemonic	Code	Words	Name	Page
MCRO (@)	99	4	MACRO	99
MIN (@)	—	4	FIND MINIMUM	88
MLB (@)	52	4	BINARY MULTIPLY	78
MOV (@)	21	3	MOVE	32
MOVB (@)	82	4	MOVE BIT	42
MOVD (@)	83	4	MOVE DIGIT	43
MOVL	—	4	DOUBLE MOVE	34
MUL (@)	32	4	BCD MULTIPLY	69
MULL (@)	56	4	DOUBLE BCD MULTIPLY	74
MVN (@)	22	3	MOVE NOT	33
NEG (@)	—	4	2’S COMPLEMENT	57
NEGL (@)	87	4	DOUBLE 2’S COMPLEMENT	58
NOP	00	1	NO OPERATION	14
OR	None	1	OR	10
OR LD	None	1	OR LOAD	11
OR NOT	None	1	OR NOT	10
ORW (@)	35	4	LOGICAL OR	93
OUT	None	2	OUTPUT	11
OUT NOT	None	2	OUTPUT NOT	11
PLS2 (@)	—	4	PULSE OUTPUT	113
PRV (@)	62	4	HIGH-SPEED COUNTER PV READ	127
PULS (@)	65	4	SET PULSES	107
RET	93	1	SUBROUTINE RETURN	98
ROL (@)	27	2	ROTATE LEFT	27
ROR (@)	28	2	ROTATE RIGHT	27
RSET	None	2	RESET	12
SBB (@)	51	4	BINARY SUBTRACT	77
SBBL (@)	48	4	DOUBLE BINARY SUBTRACT	81
SBN	92	2	SUBROUTINE DEFINE	98
SBS (@)	91	2	SUBROUTINE ENTRY	96
SCL (@)	66	4	SCALING	59
SCL2 (@)	—	4	SIGNED BINARY TO BCD SCALING	60
SCL3 (@)	—	4	BCD TO SIGNED BINARY SCALING	62
SET	None	2	SET	12
SFT	10	3	SHIFT REGISTER	24
SFTR (@)	84	4	REVERSIBLE SHIFT REGISTER	29
SLD (@)	74	3	ONE DIGIT SHIFT LEFT	28
SNXT	09	2	STEP START	129
SPED (@)	64	4	SPEED OUTPUT	110
SRD (@)	75	3	ONE DIGIT SHIFT RIGHT	29
STC (@)	40	1	SET CARRY	66
STEP	08	2	STEP DEFINE	129
STIM (@)	69	4	INTERVAL TIMER	104
SUB (@)	31	4	BCD SUBTRACT	67
SUBL (@)	55	4	DOUBLE BCD SUBTRACT	73
TCMP (@)	85	4	TABLE COMPARE	45
TIM	None	2	TIMER	19
TIMH	15	3	HIGH-SPEED TIMER	22
TMHH	—	3	ONE-MS TIMER	23

Mnemonic	Code	Words	Name	Page
WSFT (@)	16	3	WORD SHIFT	25
XCHG (@)	73	3	DATA EXCHANGE	38
XFER (@)	70	4	BLOCK TRANSFER	35
XNRW (@)	37	4	EXCLUSIVE NOR	95
XORW (@)	36	4	EXCLUSIVE OR	94
ZCP	—	4	AREA RANGE COMPARE	52
ZCPL	—	4	DOUBLE AREA RANGE COMPARE	53

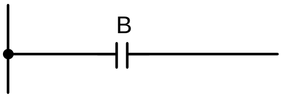
2-2 Sequence Input Instructions

2-2-1 LOAD, LOAD NOT, AND, AND NOT, OR, and OR NOT

Ladder Symbols

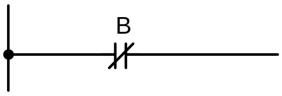
Operand Data Areas

LOAD – LD



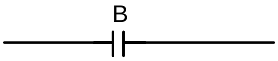
B: Bit
IR, SR, AR, TIM/CNT, LR, TR

LOAD NOT – LD NOT



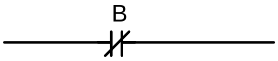
B: Bit
IR, SR, AR, TIM/CNT, LR

AND – AND



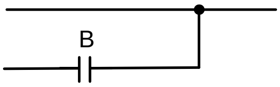
B: Bit
IR, SR, AR, TIM/CNT, LR

AND NOT – AND NOT



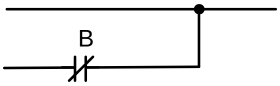
B: Bit
IR, SR, AR, TIM/CNT, LR

OR – OR



B: Bit
IR, SR, AR, TIM/CNT, LR

OR NOT – OR NOT



B: Bit
IR, SR, AR, TIM/CNT, LR

Limitations

There is no limit to the number of any of these instructions, or restrictions in the order in which they must be used, as long as the operands are in the appropriate address ranges.

Description

The status of the bit operand (B) assigned to LD or LD NOT determines the first execution condition. AND takes the logical AND between the execution condition and the status of its bit operand; AND NOT, the logical AND between the execution condition and the inverse of the status of its bit operand. OR takes the logical OR between the execution condition and the status of its bit operand; OR NOT, the logical OR between the execution condition and the inverse of the status of its bit operand.

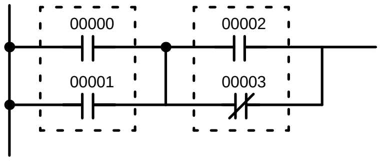
Flags

There are no flags affected by these instructions.

2-2-2 AND LOAD and OR LOAD

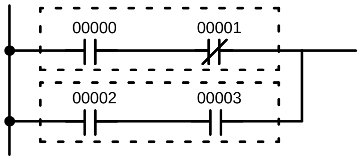
AND LOAD – AND LD

Ladder Symbol



OR LOAD – OR LD

Ladder Symbol



Description

When instructions are combined into blocks that cannot be logically combined using only OR and AND operations, AND LD and OR LD are used. Whereas AND and OR operations logically combine a bit status and an execution condition, AND LD and OR LD logically combine two execution conditions, the current one and the last unused one.

In order to draw ladder diagrams, it is not necessary to use AND LD and OR LD instructions, nor are they necessary when inputting ladder diagrams directly, as is possible from the CX-Programmer. They are required, however, to convert the program to and input it in mnemonic form.

Flags

There are no flags affected by these instructions.

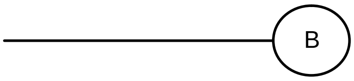
2-3 Sequence Output Instructions

There are seven instructions that can be used generally to control individual bit status. These are OUT, OUT NOT, DIFU(13), DIFD(14), SET, RSET, and KEEP(11). These instructions are used to turn bits ON and OFF in different ways.

2-3-1 OUTPUT and OUTPUT NOT – OUT and OUT NOT

OUTPUT – OUT

Ladder Symbol

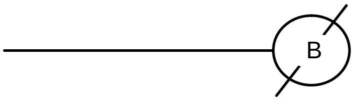


Operand Data Areas

B: Bit
IR, SR, AR, LR, TR

OUTPUT NOT – OUT NOT

Ladder Symbol



Operand Data Areas

B: Bit
IR, SR, AR, LR

Limitations

Any output bit can generally be used in only one instruction that controls its status.

Description

OUT and OUT NOT are used to control the status of the designated bit according to the execution condition.

OUT turns ON the designated bit for an ON execution condition, and turns OFF the designated bit for an OFF execution condition. With a TR bit, OUT appears at a branching point rather than at the end of an instruction line.

OUT NOT turns ON the designated bit for a OFF execution condition, and turns OFF the designated bit for an ON execution condition.

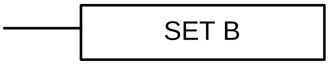
OUT and OUT NOT can be used to control execution by turning ON and OFF bits that are assigned to conditions on the ladder diagram, thus determining execution conditions for other instructions. This is particularly helpful and allows a complex set of conditions to be used to control the status of a single work bit, and then that work bit can be used to control other instructions.

The length of time that a bit is ON or OFF can be controlled by combining the OUT or OUT NOT with TIM. Refer to Examples under *2-5-1 TIMER – TIM* for details.

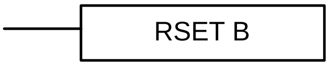
Flags There are no flags affected by these instructions.

2-3-2 SET and RESET – SET and RSET

Ladder Symbols **Operand Data Areas**



B: Bit
IR, SR, AR, LR



B: Bit
IR, SR, AR, LR

Description SET turns the operand bit ON when the execution condition is ON, and does not affect the status of the operand bit when the execution condition is OFF. RSET turns the operand bit OFF when the execution condition is ON, and does not affect the status of the operand bit when the execution condition is OFF.

The operation of SET differs from that of OUT because the OUT instruction turns the operand bit OFF when its execution condition is OFF. Likewise, RSET differs from OUT NOT because OUT NOT turns the operand bit ON when its execution condition is OFF.

Precautions The status of operand bits for SET and RSET programmed between IL(02) and ILC(03), or JMP(04) and JME(05), will not change when the interlock or jump condition is met (i.e., when IL(02) or JMP(04) is executed with an OFF execution condition).

Flags There are no flags affected by these instructions.

Examples The following examples demonstrate the difference between OUT and SET/RSET. In the first example (Diagram A), IR 01000 will be turned ON or OFF whenever IR 00000 goes ON or OFF.

In the second example (Diagram B), IR 01000 will be turned ON when IR 00001 goes ON and will remain ON (even if IR 00001 goes OFF) until IR 00002 goes ON.

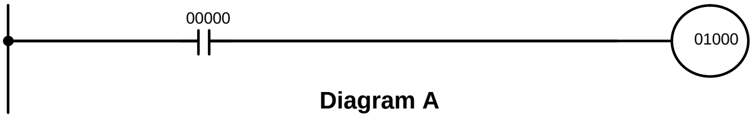


Diagram A

Address	Instruction	Operands
00000	LD	00000
00001	OUT	01000

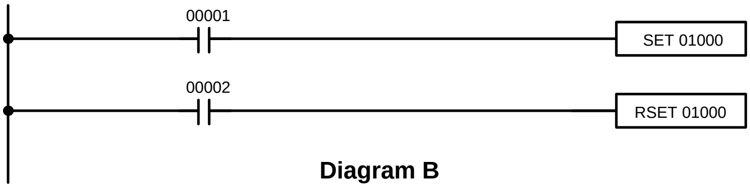
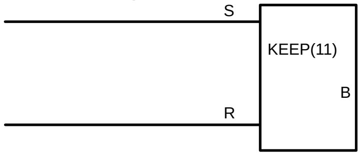


Diagram B

Address	Instruction	Operands
00000	LD	00001
00001	SET	01000
00002	LD	00002
00003	RSET	01000

2-3-3 KEEP – KEEP(11)

Ladder Symbol



Operand Data Areas

B: Bit
IR, SR, AR, LR

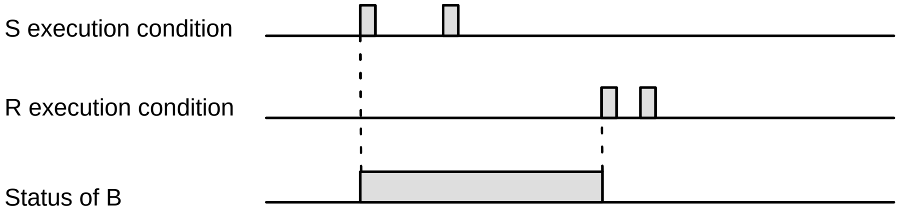
Limitations

Any output bit can generally be used in only one instruction that controls its status.

Description

KEEP(11) is used to maintain the status of the designated bit based on two execution conditions. These execution conditions are labeled S and R. S is the set input; R, the reset input. KEEP(11) operates like a latching relay that is set by S and reset by R.

When S turns ON, the designated bit will go ON and stay ON until reset, regardless of whether S stays ON or goes OFF. When R turns ON, the designated bit will go OFF and stay OFF until reset, regardless of whether R stays ON or goes OFF. The relationship between execution conditions and KEEP(11) bit status is shown below.

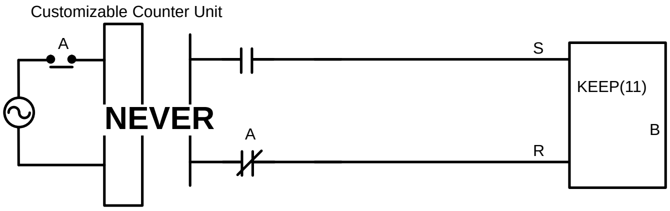


Flags

There are no flags affected by this instruction.

Precautions

Exercise caution when using a KEEP reset line that is controlled by an external normally closed device. Never use an input bit in an inverse condition on the reset (R) for KEEP(11) when the input device uses an AC power supply. The delay in shutting down the PC's DC power supply (relative to the AC power supply to the input device) can cause the designated bit of KEEP(11) to be reset. This situation is shown below.



Bits used in KEEP are not reset in interlocks. Refer to the *2-4-3 INTERLOCK – and INTERLOCK CLEAR IL(02) and ILC(03)* for details.

2-3-4 DIFFERENTIATE UP and DOWN – DIFU(13) and DIFD(14)

Ladder Symbols

Operand Data Areas

B: Bit
IR, SR, AR, LR

B: Bit
IR, SR, AR, LR

Limitations

Any output bit can generally be used in only one instruction that controls its status.

Description

DIFU(13) and DIFD(14) are used to turn the designated bit ON for one cycle only.

Whenever executed, DIFU(13) compares its current execution with the previous execution condition. If the previous execution condition was OFF and the current one is ON, DIFU(13) will turn ON the designated bit. If the previous execution condition was ON and the current execution condition is either ON or OFF, DIFU(13) will either turn the designated bit OFF or leave it OFF (i.e., if the designated bit is already OFF). The designated bit will thus never be ON for longer than one cycle, assuming it is executed each cycle (see *Precautions*, below).

Whenever executed, DIFD(14) compares its current execution with the previous execution condition. If the previous execution condition was ON and the current one is OFF, DIFD(14) will turn ON the designated bit. If the previous execution condition was OFF and the current execution condition is either ON or OFF, DIFD(14) will either turn the designated bit OFF or leave it OFF. The designated bit will thus never be ON for longer than one cycle, assuming it is executed each cycle (see *Precautions*, below).

These instructions are used when differentiated instructions (i.e., those prefixed with an @) are not available and single-cycle execution of a particular instruction is desired. They can also be used with non-differentiated forms of instructions that have differentiated forms when their use will simplify programming. Examples of these are shown below.

Flags

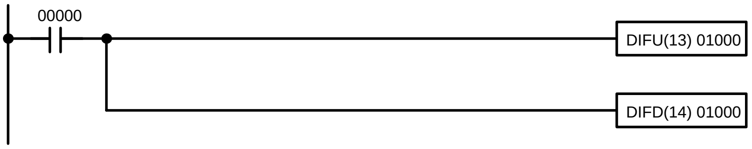
There are no flags affected by these instructions.

Precautions

DIFU(13) and DIFD(14) operation can be uncertain when the instructions are programmed between IL and ILC, between JMP and JME, or in subroutines. Refer to *2-4-3 INTERLOCK and INTERLOCK CLEAR – IL(02) and ILC(03)*, *2-4-4 JUMP and JUMP END – JMP(04) and JME(05)*, *2-17 Subroutine Instructions*, and *2-18-1 INTERRUPT CONTROL – INT(89).*

Example

In this example, IR 00100 will be turned ON for one cycle when IR 00000 goes from OFF to ON. IR 01000 will be turned ON for one cycle when IR 00000 goes from ON to OFF.



Address	Instruction	Operands
00000	LD	00000
00001	DIFU(13)	01000
00002	DIFD(14)	01000

2-4 Sequence Control Instructions

2-4-1 NO OPERATION – NOP(00)

Description

NOP(00) is not generally required in programming and there is no ladder symbol for it. When NOP(00) is found in a program, nothing is executed and the program execution moves to the next instruction. When memory is cleared prior to programming, NOP(00) is written at all addresses. NOP(00) can be input through the 00 function code.

Flags

There are no flags affected by NOP(00).

2-4-2 END – END(01)

Ladder Symbol

Description

END(01) is required as the last instruction in any program. If there are subroutines, END(01) is placed after the last subroutine. No instruction written after

END(01) will be executed. END(01) can be placed anywhere in the program to execute all instructions up to that point, as is sometimes done to debug a program, but it must be removed to execute the remainder of the program.

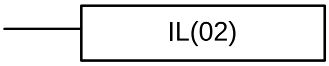
If there is no END(01) in the program, no instructions will be executed and the error message “NO END INST” will appear.

Flags

END(01) turns OFF the ER, CY, GR, EQ, LE, OF, and UF Flags.

2-4-3 INTERLOCK and INTERLOCK CLEAR – IL(02) and ILC(03)

Ladder Symbol



Ladder Symbol

Description

IL(02) is always used in conjunction with ILC(03) to create interlocks. Interlocks are used to enable branching in the same way as can be achieved with TR bits, but treatment of instructions between IL(02) and ILC(03) differs from that with TR bits when the execution condition for IL(02) is OFF. If the execution condition of IL(02) is ON, the program will be executed as written, with an ON execution condition used to start each instruction line from the point where IL(02) is located through the next ILC(03).

If the execution condition for IL(02) is OFF, the interlocked section between IL(02) and ILC(03) will be treated as shown in the following table:

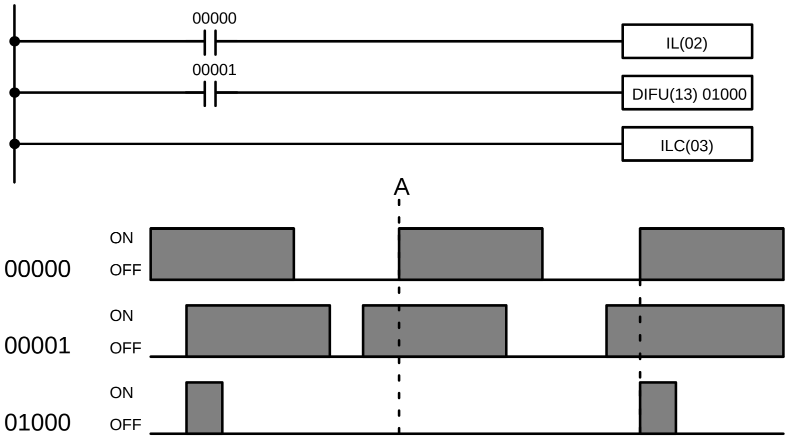
Instruction	Treatment
OUT and OUT NOT	Designated bit turned OFF.
TIM, TIMH(15), and TMHH(—)	Reset.
CNT, CNTR(12)	PV maintained.
KEEP(11)	Bit status maintained.
DIFU(13) and DIFD(14)	Not executed (see below).
All other instructions	The instructions are not executed, and all IR, AR, LR, and SR bits and words written to as operands in the instructions are turned OFF.

IL(02) and ILC(03) do not necessarily have to be used in pairs. IL(02) can be used several times in a row, with each IL(02) creating an interlocked section through the next ILC(03). ILC(03) cannot be used unless there is at least one IL(02) between it and any previous ILC(03).

DIFU(13) and DIFD(14) in Interlocks

Changes in the execution condition for a DIFU(13) or DIFD(14) are not recorded if the DIFU(13) or DIFD(14) is in an interlocked section and the execution condition for the IL(02) is OFF. When DIFU(13) or DIFD(14) is execution in an interlocked section immediately after the execution condition for the IL(02) has gone ON, the execution condition for the DIFU(13) or DIFD(14) will be compared to the execution condition that existed before the interlock became effective (i.e., before the interlock condition for IL(02) went OFF). The ladder diagram and bit status changes for this are shown below. The interlock is in effect while 00000 is

OFF. Notice that 01000 is not turned ON at the point labeled A even though 00001 has turned OFF and then back ON.



Address	Instruction	Operands
00000	LD	00000
00001	IL(02)	
00002	LD	00001
00003	DIFU(13)	01000
00004	ILC(03)	

Precautions

There must be an ILC(03) following any one or more IL(02).

Although as many IL(02) instructions as are necessary can be used with one ILC(03), ILC(03) instructions cannot be used consecutively without at least one IL(02) in between, i.e., nesting is not possible. Whenever a ILC(03) is executed, all interlocks between the active ILC(03) and the preceding ILC(03) are cleared.

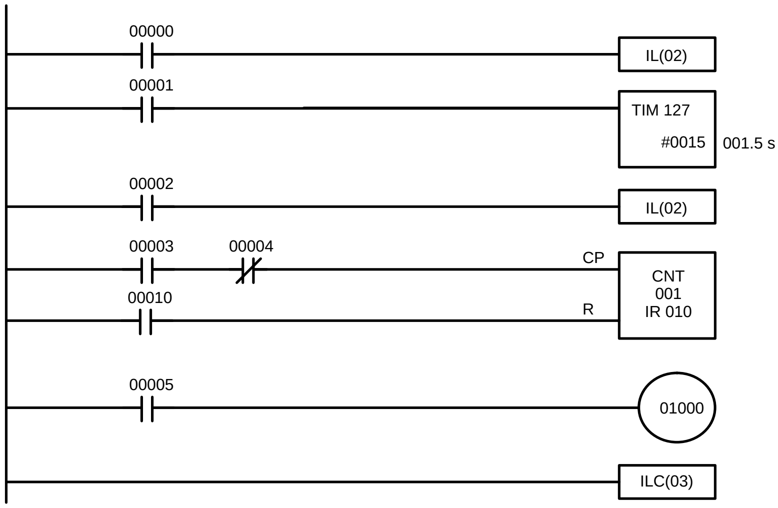
When more than one IL(02) is used with a single ILC(03), an error message will appear when the program check is performed, but execution will proceed normally.

Flags

There are no flags affected by these instructions.

Example

The following diagram shows IL(02) being used twice with one ILC(03).

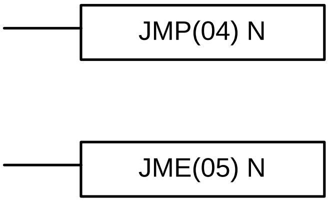


Address	Instruction	Operands
00000	LD	00000
00001	IL(02)	
00002	LD	00001
00003	TIM	127
		# 0015
00004	LD	00002
00005	IL(02)	
00006	LD	00003
00007	AND NOT	00004
00008	LD	00010
00009	CNT	001
		010
00010	LD	00005
00011	OUT	01000
00012	ILC(03)	

When the execution condition for the first IL(02) is OFF, TIM 127 will be reset to 1.5 s, CNT 001 will not be changed, and 01000 will be turned OFF. When the execution condition for the first IL(02) is ON and the execution condition for the second IL(02) is OFF, TIM 127 will be executed according to the status of 00001, CNT 001 will not be changed, and 01000 will be turned OFF. When the execution conditions for both the IL(02) are ON, the program will execute as written.

2-4-4 JUMP and JUMP END – JMP(04) and JME(05)

Ladder Symbols



Definer Values

N: Jump number
#

N: Jump number
#

Limitations

Jump numbers 01 through 99 may be used only once in JMP(04) and once in JME(05), i.e., each can be used to define one jump only. Jump number 00 can be used as many times as desired.

Jump numbers run from 00 through 99.

Description

JMP(04) is always used in conjunction with JME(05) to create jumps, i.e., to skip from one point in a ladder diagram to another point. JMP(04) defines the point from which the jump will be made; JME(05) defines the destination of the jump. When the execution condition for JMP(04) is ON, no jump is made and the program is executed consecutively as written. When the execution condition for JMP(04) is OFF, a jump is made to the JME(05) with the same jump number and the instruction following JME(05) is executed next.

If the jump number for JMP(04) is between 01 and 99, jumps, when made, will go immediately to JME(05) with the same jump number without executing any instructions in between. The status of timers, counters, bits used in OUT, bits used in OUT NOT, and all other status bits controlled by the instructions between JMP(04) and JMP(05) will not be changed. Each of these jump numbers can be used to define only one jump. Because all of instructions between JMP(04) and JME(05) are skipped, jump numbers 01 through 99 can be used to reduce cycle time.

Jump Number 00

If the jump number for JMP(04) is 00, the Customizable Counter Unit will look for the next JME(05) with a jump number of 00. To do so, it must search through the program, causing a longer cycle time (when the execution condition is OFF) than for other jumps.

The status of timers, counters, bits used in OUT, bits used in OUT NOT, and all other status controlled by the instructions between JMP(04) 00 and JMP(05) 00 will not be changed. Jump number 00 can be used as many times as desired. A jump from JMP(04) 00 will always go to the next JME(05) 00 in the program. It is thus possible to use JMP(04) 00 consecutively and match them all with the same JME(05) 00. It makes no sense, however, to use JME(05) 00 consecutively, because all jumps made to them will end at the first JME(05) 00.

DIFU(13) and DIFD(14) in Jumps

Although DIFU(13) and DIFD(14) are designed to turn ON the designated bit for one cycle, they will not necessarily do so when written between JMP(04) and JMP (05). Once either DIFU(13) or DIFD(14) has turned ON a bit, it will remain ON until the next time DIFU(13) or DIFD(14) is executed again. In normal programming, this means the next cycle. In a jump, this means the next time the jump from JMP(04) to JME(05) is not made, i.e., if a bit is turned ON by DIFU(13) or DIFD(14) and then a jump is made in the next cycle so that DIFU(13) or DIFD(14) are skipped, the designated bit will remain ON until the next time the execution condition for the JMP(04) controlling the jump is ON.

Precautions

When JMP(04) and JME(05) are not used in pairs, an error message will appear when the program check is performed. This message also appears if JMP(04) 00 and JME(05) 00 are not used in pairs, but the program will execute properly as written.

Flags

There are no flags affected by these instructions.

2-5 Timer and Counter Instructions

TIM, TIMH(15), and TMHH(—) are decrementing ON-delay timer instructions which require a TIM/CNT number and a set value (SV). STIM(69) is used to control the interval timers, which are used to activate interrupt routines.

CNT is a decrementing counter instruction and CNTR(12) is a reversible counter instruction. Both require a TIM/CNT number and a SV. Both are also connected to multiple instruction lines which serve as an input signal(s) and a reset. CTBL(63), INI(61), and PRV(62) are used to manage the high-speed counter. INI(61) is also used to stop pulse output.

Any one TIM/CNT number cannot be defined twice, i.e., once it has been used as the definer in any of the timer or counter instructions, it cannot be used again. Once defined, TIM/CNT numbers can be used as many times as required as operands in instructions other than timer and counter instructions.

TIM/CNT numbers run from 000 through 255. No prefix is required when using a TIM/CNT number as a definer in a timer or counter instruction. Once defined as a timer, a TIM/CNT number can be prefixed with TIM for use as an operand in certain instructions. The TIM prefix is used regardless of the timer instruction that was used to define the timer. Once defined as a counter, a TIM/CNT number can be prefixed with CNT for use as an operand in certain instructions. The CNT is also used regardless of the counter instruction that was used to define the counter.

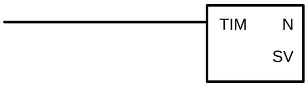
TIM/CNT numbers can be designated as operands that require either bit or word data. When designated as an operand that requires bit data, the TIM/CNT number accesses a bit that functions as a ‘Completion Flag’ that indicates when the time/count has expired, i.e., the bit, which is normally OFF, will turn ON when the designated SV has expired. When designated as an operand that requires word data, the TIM/CNT number accesses a memory location that holds the present value (PV) of the timer or counter. The PV of a timer or counter can thus be used as an operand in CMP(20), or any other instruction for which the TIM/CNT area is allowed. This is done by designating the TIM/CNT number used to define that timer or counter to access the memory location that holds the PV.

Note that “TIM 000” is used to designate the TIMER instruction defined with TIM/CNT number 000, to designate the Completion Flag for this timer, and to designate the PV of this timer. The meaning of the term in context should be clear, i.e., the first is always an instruction, the second is always a bit operand, and the third is always a word operand. The same is true of all other TIM/CNT numbers prefixed with TIM or CNT.

An SV can be input as a constant or as a word address in a data area. If an IR Area word assigned to an input is designated as the word address, the input can be wired so that the SV can be set externally through thumbwheel switches or similar devices. Timers and counters wired in this way can only be set externally during RUN or MONITOR mode. All SVs, including those set externally, must be in BCD.

2-5-1 TIMER – TIM

Ladder Symbol



Definer Values

N: TIM/CNT number
#

Operand Data Areas

SV: Set value (word, BCD)
IR, SR, AR, DM, EM, LR, #

Limitations

SV is between 000.0 and 999.9. The decimal point is not entered.

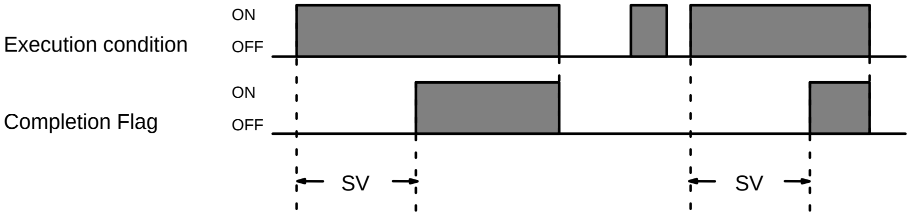
Each TIM/CNT number can be used as the definer in only one TIMER or COUNTER instruction.

Description

A timer is activated when its execution condition goes ON and is reset (to SV) when the execution condition goes OFF. Once activated, TIM measures in units of 0.1 second from the SV.

If the execution condition remains ON long enough for TIM to time down to zero, the Completion Flag for the TIM/CNT number used will turn ON and will remain ON until TIM is reset (i.e., until its execution condition is goes OFF).

The following figure illustrates the relationship between the execution condition for TIM and the Completion Flag assigned to it.



Precautions

Timers in interlocked program sections are reset when the execution condition for IL(02) is OFF. Power interruptions also reset timers. If a timer that is not reset under these conditions is desired, SR area clock pulse bits can be counted to produce timers using CNT. Refer to *2-5-2 COUNTER – CNT* for details.

Note The timer set value must be BCD between #0000 and #9999. Operation will be as follows if #0000 or #0001 is set.

- If #0000 is set, the Completion Flag will turn ON as soon as the timer's execution condition turns ON.
- If #0001 is set, the Completion Flag may turn ON as soon as the timer's execution condition turns ON because timer accuracy is 0 to –0.1 s.

Consider the timer accuracy (0 to –0.1 s) when determining the proper set value.

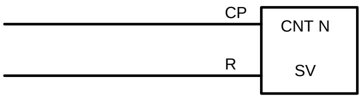
Flags

ER: SV is not in BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-5-2 COUNTER – CNT

Ladder Symbol



Definer Values

N: TIM/CNT number
#

Operand Data Areas

SV: Set value (word, BCD)
IR, SR, AR, DM, EM, LR, #

Limitations

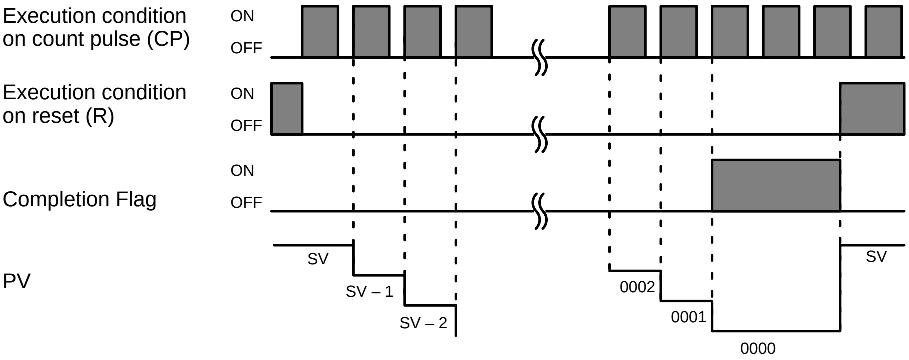
Each TIM/CNT number can be used as the definer in only one TIMER or COUNTER instruction.

Description

CNT is used to count down from SV when the execution condition on the count pulse, CP, goes from OFF to ON, i.e., the present value (PV) will be decremented by one whenever CNT is executed with an ON execution condition for CP and the execution condition was OFF for the last execution. If the execution condition has not changed or has changed from ON to OFF, the PV of CNT will not be changed. The Completion Flag for a counter is turned ON when the PV reaches zero and will remain ON until the counter is reset.

CNT is reset with a reset input, R. When R goes from OFF to ON, the PV is reset to SV. The PV will not be decremented while R is ON. Counting down from SV will begin again when R goes OFF. The PV for CNT will not be reset in interlocked program sections or by power interruptions.

Changes in execution conditions, the Completion Flag, and the PV are illustrated below. PV line height is meant only to indicate changes in the PV.


Execution condition on count pulse (CP)
ON
OFF
Execution condition on reset (R)
ON
OFF
Completion Flag
ON
OFF
PV
SV
SV – 1
SV – 2
0002
0001
0000
SV

Precautions

Program execution will continue even if a non-BCD SV is used, but the SV will not be correct.

Flags

ER: SV is not in BCD.

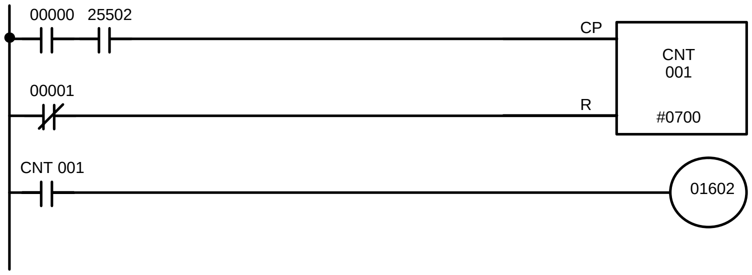
Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

Example

In the following example, CNT is used to create extended timers by counting SR area clock pulse bits.

CNT 001 counts the number of times the 1-second clock pulse bit (SR 25502) goes from OFF to ON. Here again, IR 00000 is used to control the times when CNT is operating.

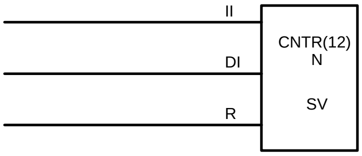
Because in this example the SV for CNT 001 is 700, the Completion Flag for CNT 002 turns ON when 1 second x 700 times, or 11 minutes and 40 seconds have expired. This would result in IR 01602 being turned ON.



Address	Instruction	Operands	
00000	LD		00000
00001	AND		25502
00002	LD NOT		00001
00003	CNT		001
		#	0700
00004	LD	CNT	001
00005	OUT		01602

2-5-3 REVERSIBLE COUNTER – CNTR(12)

Ladder Symbol



Definer Values

N: TIM/CNT number
#

Operand Data Areas

SV: Set value (word, BCD)
IR, SR, AR, DM, EM, LR, #

Limitations

Each TIM/CNT number can be used as the definer in only one TIMER or COUNTER instruction.

Description

The CNTR(12) is a reversible, up/down circular counter, i.e., it is used to count between zero and SV according to changes in two execution conditions, those in the increment input (II) and those in the decrement input (DI).

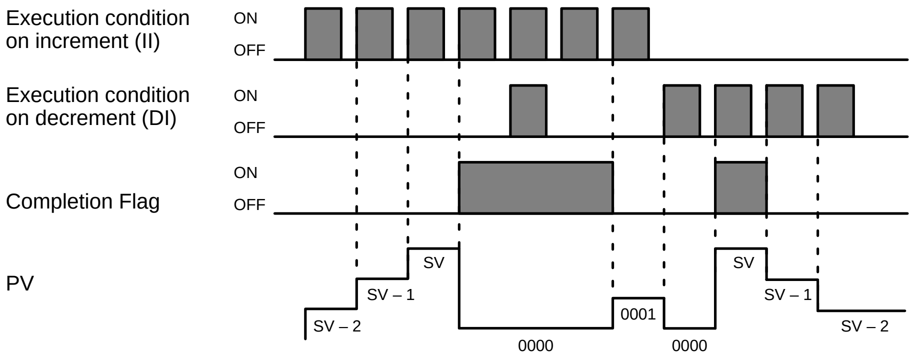
The present value (PV) will be incremented by one whenever CNTR(12) is executed with an ON execution condition for II and the last execution condition for II was OFF. The present value (PV) will be decremented by one whenever CNTR(12) is executed with an ON execution condition for DI and the last execution condition for DI was OFF. If OFF to ON changes have occurred in both II and DI since the last execution, the PV will not be changed.

If the execution conditions have not changed or have changed from ON to OFF for both II and DI, the PV of CNT will not be changed.

When decremented from 0000, the present value is set to SV and the Completion Flag is turned ON until the PV is decremented again. When incremented past the SV, the PV is set to 0000 and the Completion Flag is turned ON until the PV is incremented again.

CNTR(12) is reset with a reset input, R. When R goes from OFF to ON, the PV is reset to zero. The PV will not be incremented or decremented while R is ON. Counting will begin again when R goes OFF. The PV for CNTR(12) will not be reset in interlocked program sections or by the effects of power interruptions.

Changes in II and DI execution conditions, the Completion Flag, and the PV are illustrated below starting from part way through CNTR(12) operation (i.e., when reset, counting begins from zero). PV line height is meant to indicate changes in the PV only.


Execution condition on increment (II)
ON
OFF
Execution condition on decrement (DI)
ON
OFF
ON
Completion Flag
OFF
PV
SV
SV
SV – 1
SV – 1
0001
SV – 2
SV – 2
0000
0000

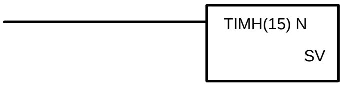
Precautions Program execution will continue even if a non-BCD SV is used, but the SV will not be correct.

Flags **ER:** SV is not in BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-5-4 HIGH-SPEED TIMER – TIMH(15)

Ladder Symbol



Definer Values

N: TIM/CNT number
#

Operand Data Areas

SV: Set value (word, BCD)
IR, SR, AR, DM, EM, LR, #

Limitations SV is between 00.00 and 99.99. (Although 00.00 and 00.01 may be set, 00.00 will disable the timer, i.e., turn ON the Completion Flag immediately, and 00.01 is not reliably scanned.) The decimal point is not entered.

Each TIM/CNT number can be used as the definer in only one TIMER or COUNTER instruction.

Description TIMH(15) operates in the same way as TIM except that TIMH measures in units of 0.01 second. Refer to *2-5-1 TIMER – TIM* for operational details.

Precautions Timers in interlocked program sections are reset when the execution condition for IL(02) is OFF. Power interruptions also reset timers. If a timer that is not reset under these conditions is desired, SR area clock pulse bits can be counted to produce timers using CNT. Refer to *2-5-2 COUNTER – CNT* for details.

Timers in jumped program sections will not be reset and the timers will stop timing when the execution condition for JMP(04) is OFF regardless of the jump number.

Note When TIMH(15) is between JMP(04) and JME(05) and the execution condition for JMP(04) is OFF, timing will be stopped and the PV held for TIMH(15) regardless of the jump number that is used.

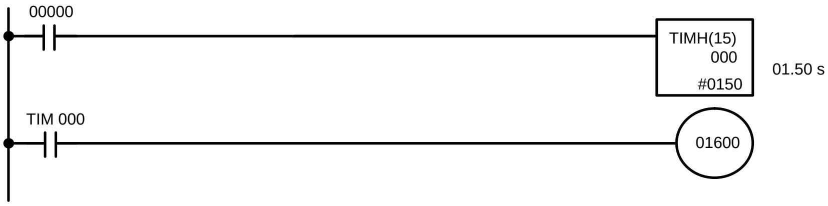
Flags

ER: SV is not in BCD.

Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

Example

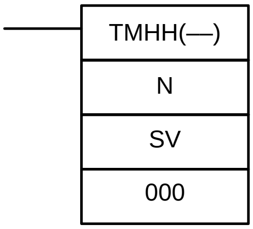
The following example shows a timer set with a constant. 01600 will be turned ON after 00000 goes ON and stays ON for at least 1.5 seconds. When 00000 goes OFF, the timer will be reset and 01600 will be turned OFF.



Address	Instruction	Operands
00000	LD	00000
00001	TIMH(15)	000
		# 0150
00002	LD	TIM 000
00003	OUT	01600

2-5-5 ONE-MS TIMER – TMHH(––)

Ladder Symbol



Definer Values

N: TIM/CNT number
TIM/CNT

Operand Data Areas

SV: Set value (word, BCD)
IR, SR, AR, DM, EM, LR, #

Limitations

SV is between 0000 and 9999 (ms). (Although 0000 and 0001 may be set, 0000 will disable the timer, i.e., turn ON the Completion Flag immediately (although a delay may occur for TIM 000 to TIM 003), and 0001 is not reliably scanned.)

Each TIM/CNT number can be used as the definer in only one TIMER or COUNTER instruction.

Note When using this instruction from the CX-Programmer, set bits 08 to 11 of DM 6600 in the Unit Setup Area to 1 to enable user-specified expansion instruction settings before uploading and downloading the program. On the CX-Programmer, TMHH(—) will be displayed as PMCR.

Description

TMHH(—) operates in the same way as TIM except that TMHH(—) measures in units of 1 ms. Refer to *2-5-1 TIMER – TIM* for operational details.

Precautions

Timers in interlocked program sections are reset when the execution condition for IL(02) is OFF. Power interruptions also reset timers. If a timer that is not reset under these conditions is desired, SR area clock pulse bits can be counted to produce timers using CNT. Refer to *2-5-2 COUNTER – CNT* for details.

Timers in jumped program sections will not be reset when the execution condition for JMP(04) is OFF, but the timer will stop timing if TIM/CNT 004 to TIM/CNT 255 is used. The timers will continue timing if TIM/CNT 000 to TIM/CNT 003 is used.

TMHH(—) timers with timer numbers TIM/CNT 000 to TIM/CNT 003 will be accurate even if the cycle time is greater than 1 ms. TIM/CNT 004 through TIM/CNT 255 may not be accurate if the cycle time is greater than 1 ms.

Flags

ER: SV is not in BCD.

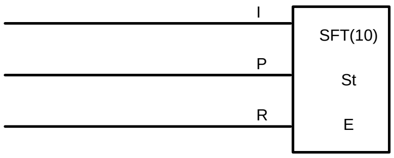
N is not between 000 and 255.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-6 Data Shift Instructions

2-6-1 SHIFT REGISTER – SFT(10)

Ladder Symbol



Operand Data Areas

St: Starting word
IR, SR, AR, LR

E: End word
IR, SR, AR, LR

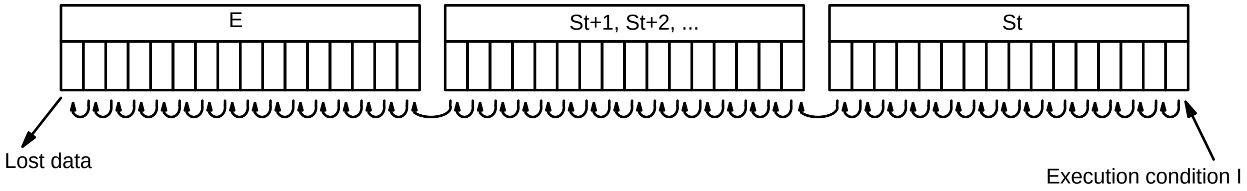
Limitations

E must be greater than or equal to St, and St and E must be in the same data area.

If a bit address in one of the words used in a shift register is also used in an instruction that controls individual bit status (e.g., OUT, KEEP(11)), an error (“COIL/OUT DUPL”) will be generated when program syntax is checked on the Programming Console or another Programming Device. The program, however, will be executed as written. See *Example 2: Controlling Bits in Shift Registers* for a programming example that does this.

Description

SFT(10) is controlled by three execution conditions, I, P, and R. If SFT(10) is executed and 1) execution condition P is ON and was OFF in the last execution, and 2) R is OFF, then execution condition I is shifted into the rightmost bit of a shift register defined between St and E, i.e., if I is ON, a 1 is shifted into the register; if I is OFF, a 0 is shifted in. When I is shifted into the register, all bits previously in the register are shifted to the left and the leftmost bit of the register is lost.



The execution condition on P functions like a differentiated instruction, i.e., I will be shifted into the register only when P is ON and was OFF the last time SFT(10) was executed. If execution condition P has not changed or has gone from ON to OFF, the shift register will remain unaffected.

St designates the rightmost word of the shift register; E designates the leftmost. The shift register includes both of these words and all words between them. The same word may be designated for St and E to create a 16-bit (i.e., 1-word) shift register.

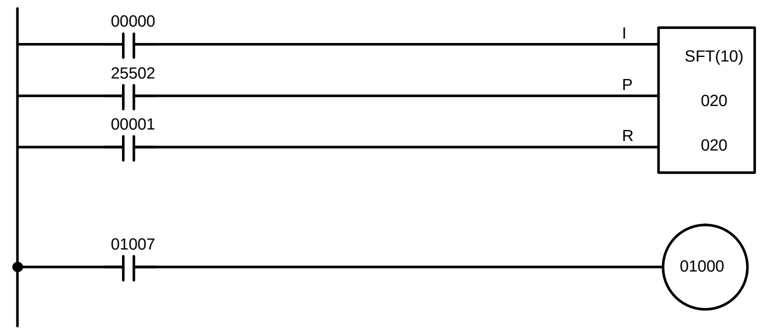
When execution condition R goes ON, all bits in the shift register will be turned OFF (i.e., set to 0) and the shift register will not operate until R goes OFF again.

Flags

There are no flags affected by SFT(10).

Example

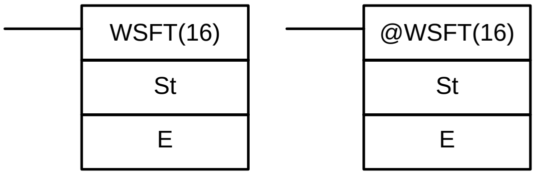
The following example uses the 1-second clock pulse bit (25502) so that the execution condition produced by 00000 is shifted into IR 010 every second. Output 01000 is turned ON whenever a “1” is shifted into 01007.



Address	Instruction	Operands
00000	LD	00000
00001	LD	25502
00002	LD	00001
00003	SFT(10)	020
		020
00004	LD	01007
00005	OUT	01000

2-6-2 WORD SHIFT – WSFT(16)

Ladder Symbols



Operand Data Areas

St: Starting word
IR, SR, AR, DM, EM, LR

E: End word
IR, SR, AR, DM, EM, LR

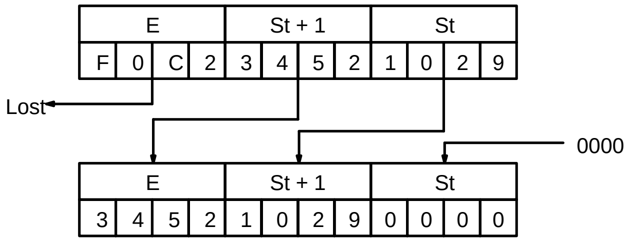
Limitations

St and E must be in the same data area, and E must be greater than or equal to St.

DM 6144 to DM 6655 cannot be used for St or E.

Description

When the execution condition is OFF, WSFT(16) is not executed. When the execution condition is ON, WSFT(16) shifts data between St and E in word units. Zeros are written into St and the content of E is lost.



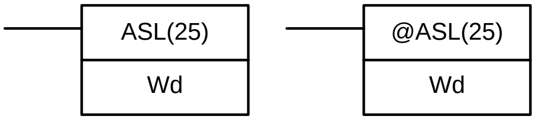
Flags

ER: The St and E words are in different areas, or St is greater than E.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-6-3 ARITHMETIC SHIFT LEFT – ASL(25)

Ladder Symbols

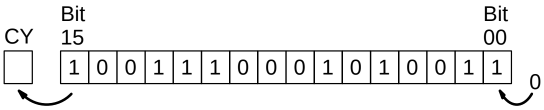


Operand Data Areas

Wd: Shift word
IR, SR, AR, DM, EM, LR

Limitations DM 6144 to DM 6655 cannot be used for Wd.

Description When the execution condition is OFF, ASL(25) is not executed. When the execution condition is ON, ASL(25) shifts a 0 into bit 00 of Wd, shifts the bits of Wd one bit to the left, and shifts the status of bit 15 into CY.



Precautions A 0 will be shifted into bit 00 every cycle if the undifferentiated form of ASL(25) is used. Use the differentiated form (@ASL(25)) or combine ASL(25) with DIFU(13) or DIFD(14) to shift just one time.

Flags

N: ON when the leftmost bit is 1 as a result of the shift.

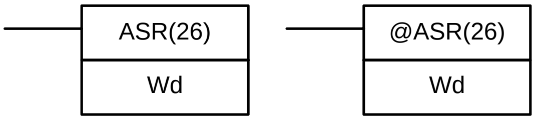
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: Receives the status of bit 15.

EQ: ON when the content of Wd is zero; otherwise OFF.

2-6-4 ARITHMETIC SHIFT RIGHT – ASR(26)

Ladder Symbols

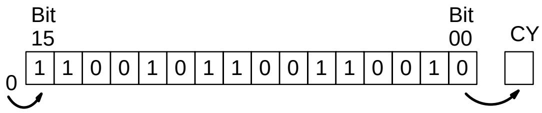


Operand Data Areas

Wd: Shift word
IR, SR, AR, DM, EM, LR

Limitations DM 6144 to DM 6655 cannot be used for Wd.

Description When the execution condition is OFF, ASR(25) is not executed. When the execution condition is ON, ASR(25) shifts a 0 into bit 15 of Wd, shifts the bits of Wd one bit to the right, and shifts the status of bit 00 into CY.



Precautions A 0 will be shifted into bit 15 every cycle if the undifferentiated form of ASR(26) is used. Use the differentiated form (@ASR(26)) or combine ASR(26) with DIFU(13) or DIFD(14) to shift just one time.

Flags

N: ON when the leftmost bit is 1 as a result of the shift.

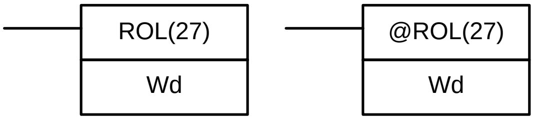
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: Receives the data of bit 00.

EQ: ON when the content of Wd is zero; otherwise OFF.

2-6-5 ROTATE LEFT – ROL(27)

Ladder Symbols



Operand Data Areas

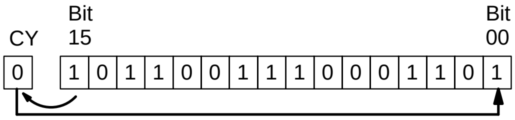
Wd: Rotate word
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for Wd.

Description

When the execution condition is OFF, ROL(27) is not executed. When the execution condition is ON, ROL(27) shifts all Wd bits one bit to the left, shifting CY into bit 00 of Wd and shifting bit 15 of Wd into CY.



Precautions

Use STC(41) to set the status of CY or CLC(41) to clear the status of CY before doing a rotate operation to ensure that CY contains the proper status before executing ROL(27).

CY will be shifted into bit 00 every cycle if the undifferentiated form of ROL(27) is used. Use the differentiated form (@ROL(27)) or combine ROL(27) with DIFU(13) or DIFD(14) to shift just one time.

Flags

N: ON when the leftmost bit is 1 as a result of the shift.

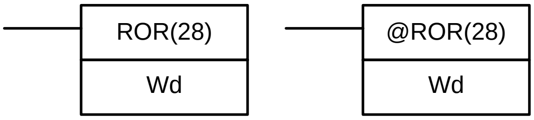
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: Receives the data of bit 15.

EQ: ON when the content of Wd is zero; otherwise OFF.

2-6-6 ROTATE RIGHT – ROR(28)

Ladder Symbols



Operand Data Areas

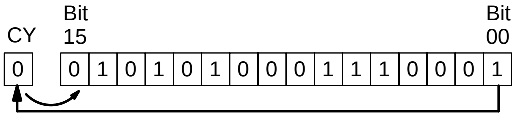
Wd: Rotate word
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for Wd.

Description

When the execution condition is OFF, ROR(28) is not executed. When the execution condition is ON, ROR(28) shifts all Wd bits one bit to the right, shifting CY into bit 15 of Wd and shifting bit 00 of Wd into CY.



Precautions

Use STC(41) to set the status of CY or CLC(41) to clear the status of CY before doing a rotate operation to ensure that CY contains the proper status before execution ROR(28).

CY will be shifted into bit 15 every cycle if the undifferentiated form of ROR(28) is used. Use the differentiated form (@ROR(28)) or combine ROR(28) with DIFU(13) or DIFD(14) to shift just one time.

Flags

N: ON when the leftmost bit is 1 as a result of the shift.

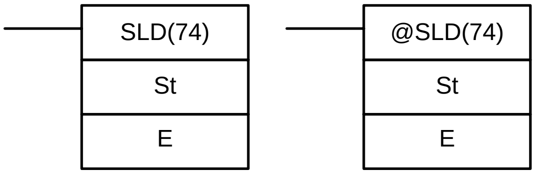
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: Receives the data of bit 00.

EQ: ON when the content of Wd is zero; otherwise OFF.

2-6-7 ONE DIGIT SHIFT LEFT – SLD(74)

Ladder Symbols



Operand Data Areas

St: Starting word
IR, SR, AR, DM, EM, LR

E: End word
IR, SR, AR, DM, EM, LR

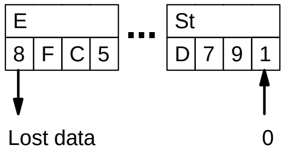
Limitations

St and E must be in the same data area, and E must be greater than or equal to St.

DM 6144 to DM 6655 cannot be used for St or E.

Description

When the execution condition is OFF, SLD(74) is not executed. When the execution condition is ON, SLD(74) shifts data between St and E (inclusive) by one digit (four bits) to the left. 0 is written into the rightmost digit of the St, and the content of the leftmost digit of E is lost.



Precautions

If a power failure occurs during a shift operation across more than 50 words, the shift operation might not be completed.

A 0 will be shifted into the least significant digit of St every cycle if the undifferentiated form of SLD(74) is used. Use the differentiated form (@SLD(74)) or combine SLD(74) with DIFU(13) or DIFD(14) to shift just one time.

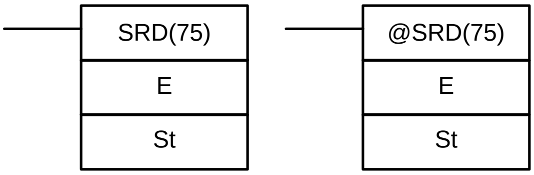
Flags

ER: The St and E words are in different areas, or St is greater than E.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-6-8 ONE DIGIT SHIFT RIGHT – SRD(75)

Ladder Symbols



Operand Data Areas

E: End word
IR, SR, AR, DM, EM, LR

St: Starting word
IR, SR, AR, DM, EM, LR

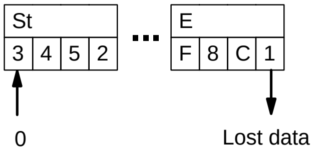
Limitations

St and E must be in the same data area, and E must be less than or equal to St.

DM 6144 to DM 6655 cannot be used for St or E.

Description

When the execution condition is OFF, SRD(75) is not executed. When the execution condition is ON, SRD(75) shifts data between St and E (inclusive) by one digit (four bits) to the right. 0 is written into the leftmost digit of St and the rightmost digit of E is lost.



Precautions

If a power failure occurs during a shift operation across more than 50 words, the shift operation might not be completed.

A 0 will be shifted into the most significant digit of St every cycle if the undifferentiated form of SRD(75) is used. Use the differentiated form (@SRD(75)) or combine SRD(75) with DIFU(13) or DIFD(14) to shift just one time.

Flags

ER: The St and E words are in different areas, or St is less than E.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-6-9 REVERSIBLE SHIFT REGISTER – SFTR(84)

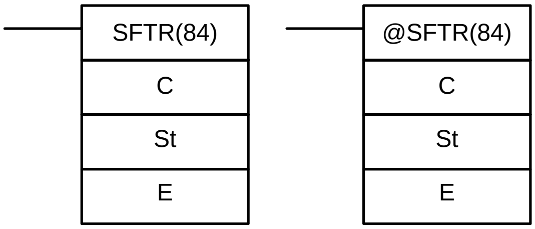
Operand Data Areas

C: Control word
IR, SR, AR, DM, EM, LR

St: Starting word
IR, SR, AR, DM, EM, LR

E: End word
IR, SR, AR, DM, EM, LR

Ladder Symbols



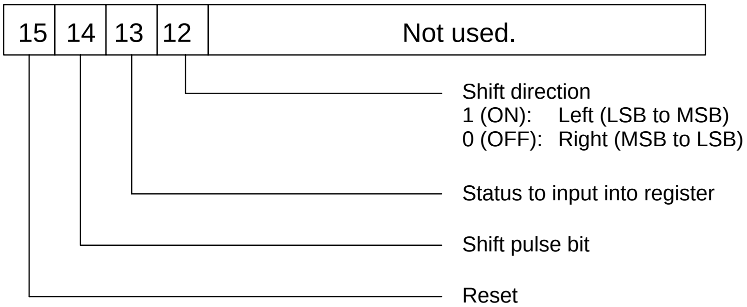
Limitations

St and E must be in the same data area and St must be less than or equal to E.

DM 6144 to DM 6655 cannot be used for C, St, or E.

Description

SFTR(84) is used to create a single- or multiple-word shift register that can shift data to either the right or the left. To create a single-word register, designate the same word for St and E. The control word provides the shift direction, the status to be put into the register, the shift pulse, and the reset input. The control word is allocated as follows:


15
14
13
12
Not used.
Shift direction
1 (ON): Left (LSB to MSB)
0 (OFF): Right (MSB to LSB)
Status to input into register
Shift pulse bit
Reset

The data in the shift register will be shifted one bit in the direction indicated by bit 12, shifting one bit out to CY and the status of bit 13 into the other end whenever SFTR(84) is executed with an ON execution condition as long as the reset bit is OFF and as long as bit 14 is ON. If SFTR(84) is executed with an OFF execution condition or if SFTR(84) is executed with bit 14 OFF, the shift register will remain unchanged. If SFTR(84) is executed with an ON execution condition and the reset bit (bit 15) is OFF, the entire shift register and CY will be set to zero.

Flags

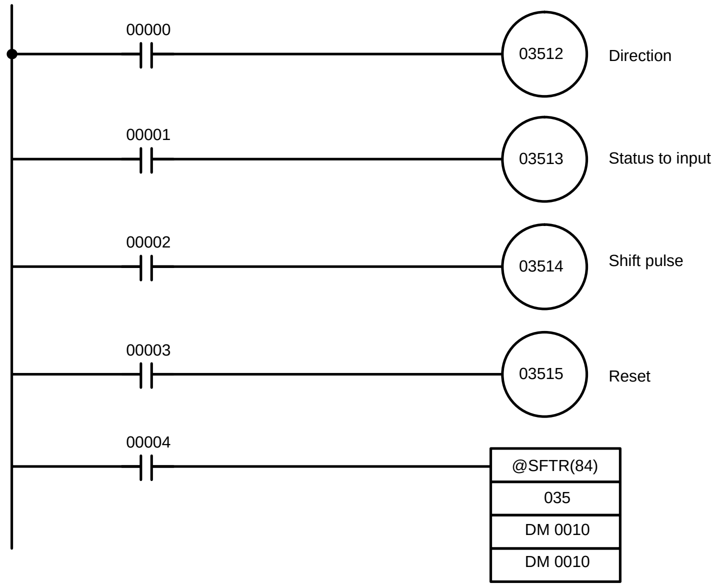
ER: St and E are not in the same data area or ST is greater than E.

Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: Receives the status of bit 00 of St or bit 15 of E, depending on the shift direction.

Example

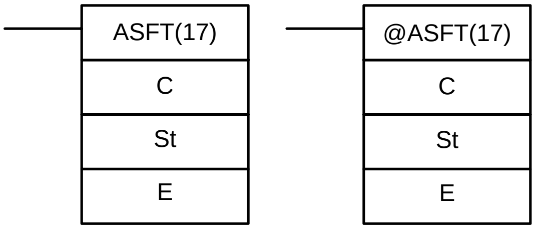
In the following example, IR 00000, IR 00001, IR 00002, and IR 00003 are used to control the bits of C used in @SFTR(84). The shift register is in DM 0010, and it is controlled through IR 00004.



Address	Instruction	Operands
00000	LD	00000
00001	OUT	03512
00002	LD	00001
00003	OUT	03513
00004	LD	00002
00005	OUT	03514
00006	LD	00003
00007	OUT	03515
00008	LD	00004
00009	@SFT(10)	
		035
		DM 0010
		DM 0010

2-6-10 ASYNCHRONOUS SHIFT REGISTER – ASFT(17)

Ladder Symbols



Operand Data Areas

C: Control word

IR, SR, AR, DM, EM, LR, #

St: Starting word

IR, SR, AR, DM, EM, LR

E: End word

IR, SR, AR, DM, EM, LR

Limitations

St and E must be in the same data area, and E must be greater than or equal to St.

DM 6144 to DM 6655 cannot be used for St or E.

Description

When the execution condition is OFF, ASFT(17) does nothing and the program moves to the next instruction. When the execution condition is ON, ASFT(17) is used to create and control a reversible asynchronous word shift register between St and E. This register only shifts words when the next word in the register is zero, e.g., if no words in the register contain zero, nothing is shifted. Also, only one word is shifted for each word in the register that contains zero. When the contents of a word are shifted to the next word, the original word's contents are set to zero. In essence, when the register is shifted, each zero word in the register trades places with the next word. (See *Example* below.)

The shift direction (i.e. whether the "next word" is the next higher or the next lower word) is designated in C. C is also used to reset the register. All of any portion of the register can be reset by designating the desired portion with St and E.

Control Word

Bits 00 through 12 of C are not used. Bit 13 is the shift direction: turn bit 13 ON to shift down (toward lower addressed words) and OFF to shift up (toward higher addressed words). Bit 14 is the Shift Enable Bit: turn bit 14 ON to enable shift register operation according to bit 13 and OFF to disable the register. Bit 15 is the Reset bit: the register will be reset (set to zero) between St and E when ASFT(17) is executed with bit 15 ON. Turn bit 15 OFF for normal operation.

Note If the non-differentiated form of ASFT(17) is used, data will be shifted every cycle while the execution condition is ON. Use the differentiated form to prevent this.

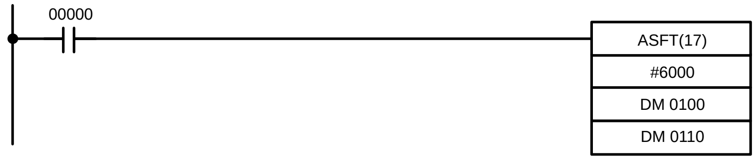
Flags

ER: The St and E words are in different areas, or St is greater than E.

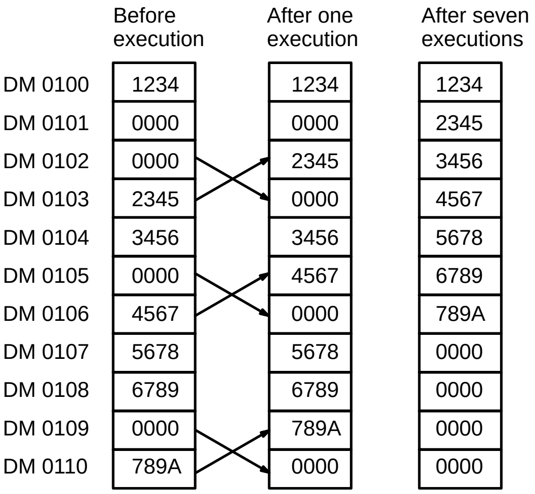
Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

Example

The following example shows instruction ASFT(17) used to shift words in an 11-word shift register created between DM 0100 and DM 0110 with C=#6000. Non-zero data is shifted towards St (DM 0110).



Address	Instruction	Operands	
00000	LD		00000
00001	ASFT(17)		
		#	6000
		DM	0100
		DM	0110



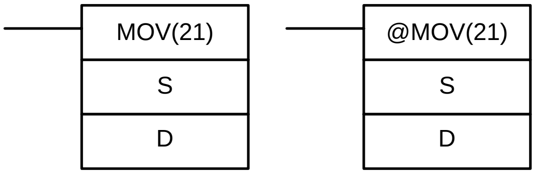
	Before execution	After one execution	After seven executions
DM 0100	1234	1234	1234
DM 0101	0000	0000	2345
DM 0102	0000	2345	3456
DM 0103	2345	0000	4567
DM 0104	3456	3456	5678
DM 0105	0000	4567	6789
DM 0106	4567	0000	789A
DM 0107	5678	5678	0000
DM 0108	6789	6789	0000
DM 0109	0000	789A	0000
DM 0110	789A	0000	0000

Note The zeroes are shifted “upward” if C=4000, and the entire shift register is set to zero if C=8000.

2-7 Data Movement Instructions

2-7-1 MOVE – MOV(21)

Ladder Symbols



Operand Data Areas

S: Source word
IR, SR, AR, DM, EM, TIM/CNT, LR, #

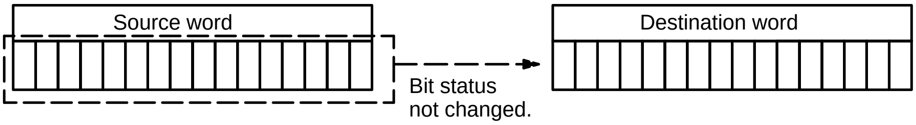
D: Destination word
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for D.

Description

When the execution condition is OFF, MOV(21) is not executed. When the execution condition is ON, MOV(21) copies the content of S to D.



Precautions

TIM/CNT numbers cannot be designated as D to change the PV of the timer or counter. You can, however, easily change the PV of a timer or a counter by using BSET(71).

Flags

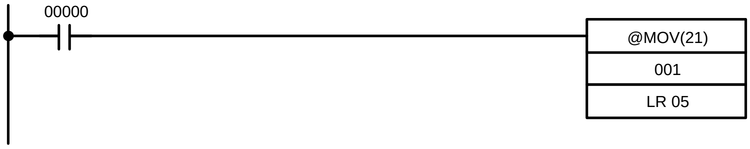
N: ON when the leftmost bit of the data being transferred is 1.

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

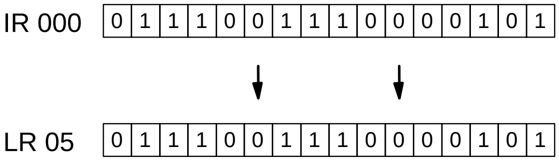
EQ: ON when all zeros are transferred to D.

Example

The following example shows @MOV(21) being used to copy the content of IR 001 to LR 05 when IR 00000 goes from OFF to ON.

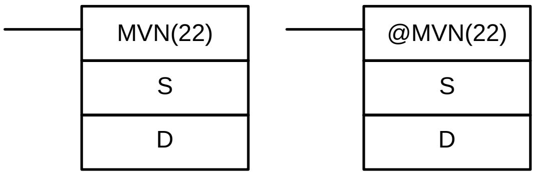


Address	Instruction	Operands
00000	LD	00000
00001	@MOV(21)	
		001
		LR 05



2-7-2 MOVE NOT – MVN(22)

Ladder Symbols



Operand Data Areas

S: Source word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

D: Destination word

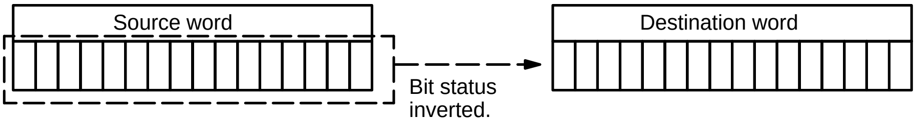
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for D.

Description

When the execution condition is OFF, MVN(22) is not executed. When the execution condition is ON, MVN(22) transfers the inverted content of S (specified word or four-digit hexadecimal constant) to D, i.e., for each ON bit in S, the corresponding bit in D is turned OFF, and for each OFF bit in S, the corresponding bit in D is turned ON.



Precautions

TIM/CNT numbers cannot be designated as D to change the PV of the timer or counter. However, these can be easily changed using BSET(71).

Flags

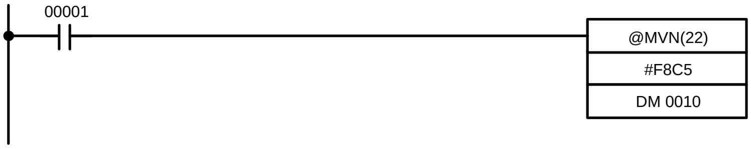
N: ON when the leftmost bit of the data being transferred is 1.

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when all zeros are transferred to D.

Example

The following example shows @MVN(22) being used to copy the complement of #F8C5 to DM 0010 when IR 00001 goes from OFF to ON.

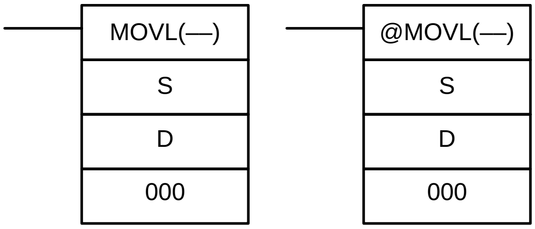


Address	Instruction	Operands
00000	LD	00001
00001	@MOV(21)	
		# F8C5
		DM 0010

#F8C5	1	1	1	1	1	0	0	0	1	1	0	0	0	1	0	1
DM 0010	0	0	0	0	0	1	1	1	0	0	1	1	1	0	1	0

2-7-3 DOUBLE MOVE – MOVL(––)

Ladder Symbols



Operand Data Areas

S: Starting source word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

D: Starting destination word

IR, SR, AR, DM, EM, LR, #

000: Always “000”

000

Limitations

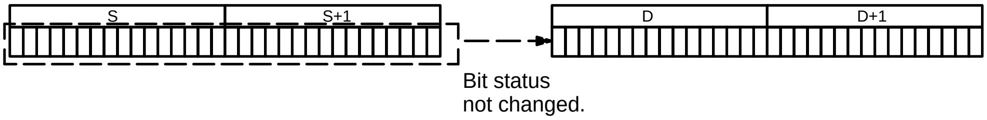
S and S+1 must be in the same data area, as must D and D+1.

DM 6144 to DM 6655 cannot be used for D.

Note When using this instruction from the CX-Programmer, set bits 08 to 11 of DM 6600 in the Unit Setup Area to 1 to enable user-specified expansion instruction settings before uploading and downloading the program. On the CX-Programmer, 7SEG. (@ cannot be attached to 7SEG, so use DIFU(13)/DIFD(14) for differential treatment).

Description

When the execution condition is OFF, MOVL(—) is not executed. When the execution condition is ON, MOVL(—) copies the content of S and S+1 to D and D+1.



Precautions

TIM/CNT numbers cannot be designated as D to change the PV of the timer or counter. You can, however, easily change the PV of a timer or a counter by using BSET(71).

Flags

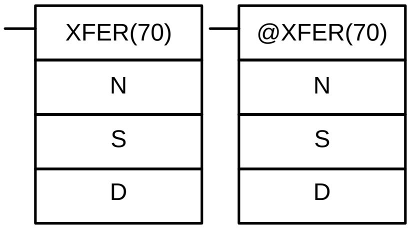
N: ON when the leftmost bit of the data being transferred is 1.

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when all zeros are transferred to D.

2-7-4 BLOCK TRANSFER – XFER(70)

Ladder Symbols



Operand Data Areas (N ≠ 9999)	Operand Data Areas (N = 9999)
N: Number of words (BCD)	**N**: Flash memory designation
IR, SR, AR, DM, EM, TIM/CNT, LR, #	IR, SR, AR, DM, EM, TIM/CNT, LR, #
S: Starting source word	**S**: Flash memory word source
IR, SR, AR, DM, EM, TIM/CNT, LR	IR, SR, AR, DM, EM, TIM/CNT, LR
D: Starting destination word	**D**: Starting destination word
IR, SR, AR, DM, EM, TIM/CNT, LR	IR, SR, AR, DM, EM, TIM/CNT, LR

Limitations

When N is not 9999, S and S+N must be in the same data area, as must D and D+N.

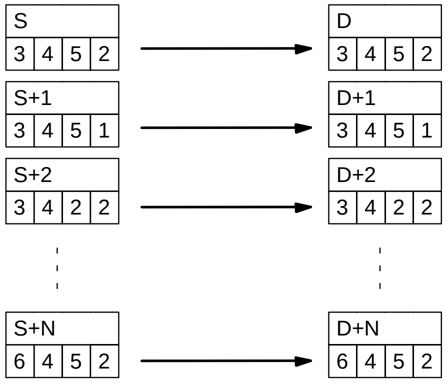
DM 6144 to DM 6655 cannot be used for D.

Description

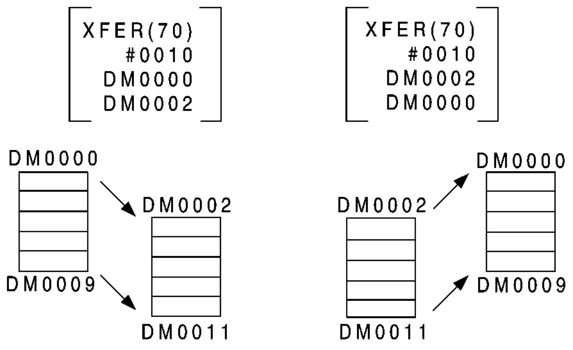
The operation of XFER(70) depends on the value of N. If N is not 9999, then XFER(70) transfers data between two areas of memory. If N is 9999, then XFER(70) transfers the specified data from flash memory (i.e., all or part of the data previously backed up from DM 0000 to DM 6143) to specified words in the DM Area or another data area.

N Not Equal to 9999

When the execution condition is OFF, XFER(70) is not executed. When the execution condition is ON, XFER(70) copies the contents of S, S+1, ..., S+N to D, D+1, ..., D+N.



Note XFER(70) can be used to shift data by setting the operands so that data is moved between overlapping areas.



N Equal to 9999

When the execution condition is OFF, XFER(70) is not executed. When the execution condition is ON, XFER(70) copies the contents flash memory to words starting at D. The portion of flash memory that is copied is specified by S and S+1.

S specifies the number of words in BCD between 0000 and 6144.
S+1 specifies the first word in flash memory in BCD between 0000 and 6143. (Here, 0000 to 6144 are the offsets from the beginning of flash memory, but they would have corresponded to DM 0000 to DM 6143 when the DM Area was backed up to flash memory.)

Flags

ER: N is not BCD.

S and S+N or D and D+N are not in the same data area.

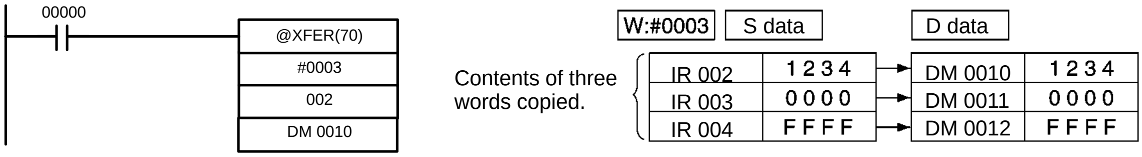
Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

SR 24904:
A checksum error has occurred in flash memory data. If this bit turns ON, the data in flash memory will not be copied to the destination words.

Examples

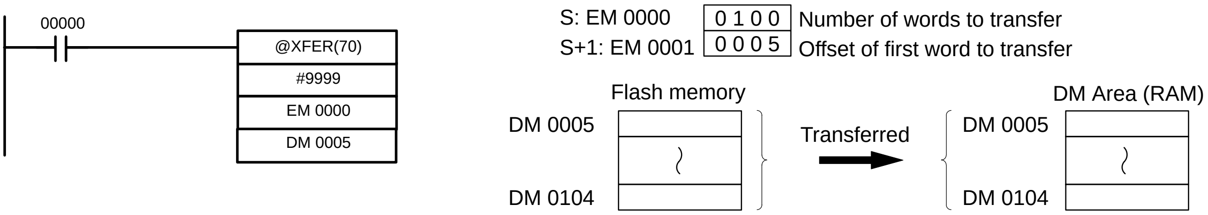
Normal Data Transfer (N Not Equal to 9999)

When IR 00000 turns ON in the following example, the contents of IR 002 to IR 004 will be copied to DM 0010 to DM 0012.



Flash Memory Data Transfer (N Equal to 9999)

When IR 00000 turns ON in the following example, the contents of 100 words of flash memory starting at an offset of 5 (i.e., the backed up contents of DM 0005 to DM 0104) will be copied to DM 0005 to DM 0104.



Note The following steps are used to back up DM Area data to flash memory.

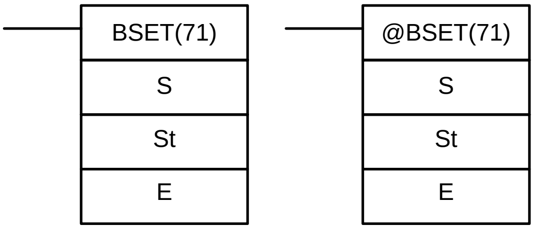
1, 2, 3...

1. Change the Customizable Counter Unit to PROGRAM mode.
2. Make sure that DM 0000 to DM 6143 contain the data to be backed up.
3. Turn ON SR 25200 (the DM Area Backup Bit).

   The contents of DM 0000 to DM 6143 will be copied to flash memory and SR 25200 will turn OFF when the transfer operation has been completed.

2-7-5 BLOCK SET – BSET(71)

Ladder Symbols



Operand Data Areas

S: Source data

IR, SR, AR, DM, EM, TIM/CNT, LR, #

St: Starting word

IR, SR AR, DM, EM, TIM/CNT, LR

E: End Word

IR, SR, AR, DM, EM, TIM/CNT, LR

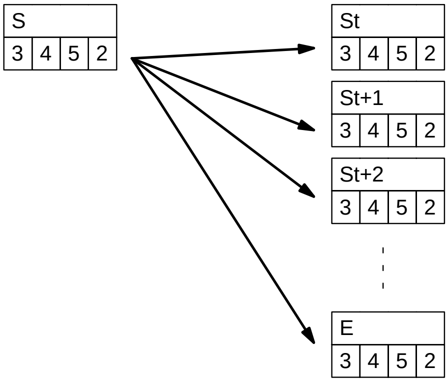
Limitations

St must be less than or equal to E, and St and E must be in the same data area.

DM 6144 to DM 6655 cannot be used for St or E.

Description

When the execution condition is OFF, BSET(71) is not executed. When the execution condition is ON, BSET(71) copies the content of S to all words from St through E.



BSET(71) can be used to change timer/counter PV. (This cannot be done with MOV(21) or MVN(22).) BSET(71) can also be used to clear sections of a data area, i.e., the DM area, to prepare for executing other instructions. It can also be used to clear words by transferring all zeros.

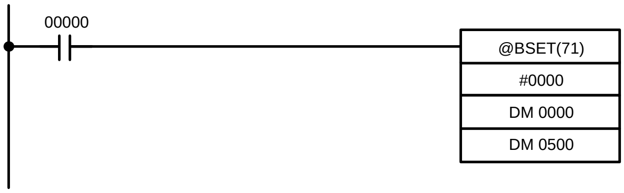
Flags

ER: St and E are not in the same data area or St is greater than E.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

Example

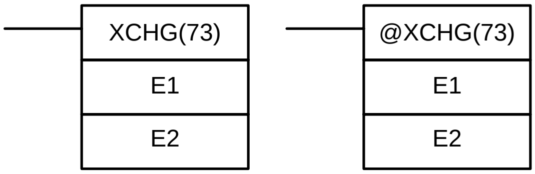
The following example shows how to use BSET(71) to copy a constant (#0000) to a block of the DM area (DM 0000 to DM 0500) when IR 00000 is ON.



Address	Instruction	Operands	
00000	LD		00000
00001	@BSET(71)		
		#	0000
		DM	0000
		DM	0500

2-7-6 DATA EXCHANGE – XCHG(73)

Ladder Symbols



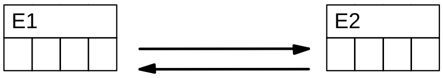
Operand Data Areas

E1: Exchange word 1
IR, SR, AR, DM, EM, TIM/CNT, LR

E2: Exchange word 2
IR, SR, AR, DM, EM, TIM/CNT, LR

Limitations DM 6144 to DM 6655 cannot be used for E1 or E2.

Description When the execution condition is OFF, XCHG(73) is not executed. When the execution condition is ON, XCHG(73) exchanges the content of E1 and E2.



If you want to exchange content of blocks whose size is greater than 1 word, use work words as an intermediate buffer to hold one of the blocks using XFER(70) three times.

Flags **ER:** Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-7-7 SINGLE WORD DISTRIBUTE – DIST(80)

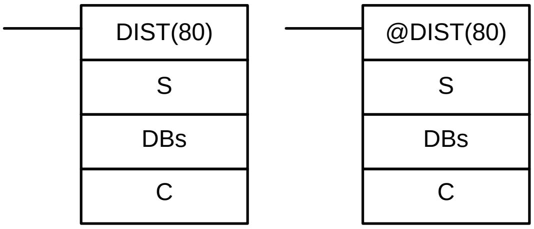
Operand Data Areas

S: Source data
IR, SR, AR, DM, EM, TIM/CNT, LR, #

DBs: Destination base word
IR, SR, AR, DM, EM, TIM/CNT, LR

C: Control word (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR, #

Ladder Symbols



Limitations C must be BCD.

DM 6144 to DM 6655 cannot be used for DBs or C.

Description DIST(80) can be used for single-word distribution or for a stack operation depending on the content of the control word, C.

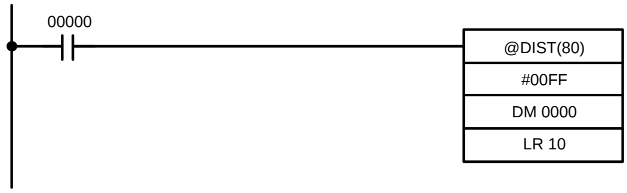
Single-word Distribution When bits 12 to 15 of C=0 to 8, DIST(80) can be used for a single word distribute operation. The entire contents of C specifies an offset, Of.

When the execution condition is OFF, DIST(80) is not executed. When the execution condition is ON, DIST(80) copies the content of S to DBs+Of, i.e., Of is added to DBs to determine the destination word.

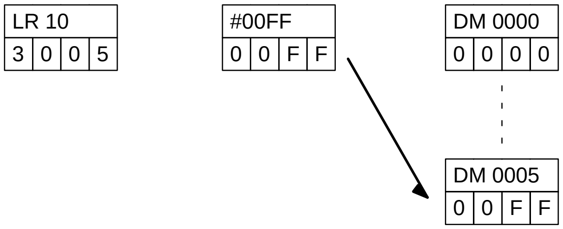
Note DBs and DBs+Of must be in the same data area and cannot be between DM 6144 and DM 6655.

Example
The following example shows how to use DIST(80) to copy #00FF to DM 0000 +

Of. The content of LR 10 is #3005, so #00FF is copied to DM 0005 (DM 0000 + 5) when IR 00000 is ON.



Address	Instruction	Operands
00000	LD	00000
00001	@DIST(80)	
		# 00FF
		DM 0000
		LR 10



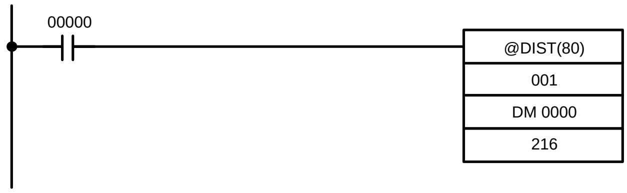
Stack Operation

When bits 12 to 15 of C=9, DIST(80) can be used for a stack operation. The other 3 digits of C specify the number of words in the stack (000 to 999). The content of DBs is the stack pointer.

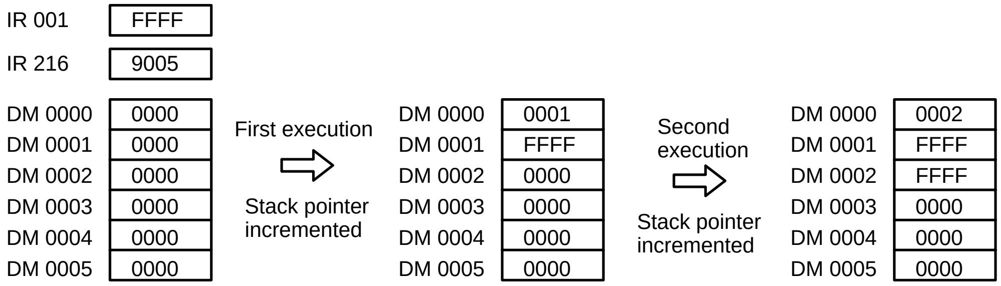
When the execution condition is OFF, DIST(80) is not executed. When the execution condition is ON, DIST(80) copies the content of S to DBs+1+the content of DBs. In other words, 1 and the content of DBs are added to DBs to determine the destination word. The content of DBs is then incremented by 1.

Note
1. DIST(80) will be executed every cycle unless the differentiated form (@DIST(80)) is used or DIST(80) is used with DIFU(13) or DIFD(14).
2. Be sure to initialize the stack pointer before using DIST(80) as a stack operation.

Example
The following example shows how to use DIST(80) to create a stack between DM 0001 and DM 0005. DM 0000 acts as the stack pointer.



Address	Instruction	Operands
00000	LD	00000
00001	@DIST(80)	
		001
		DM 0000
		216



Flags

N: ON when the leftmost bit of the data being transferred is 1.

ER: The offset or stack length in the control word is not BCD.
Indirectly addressed EM/DM word is non-existent.

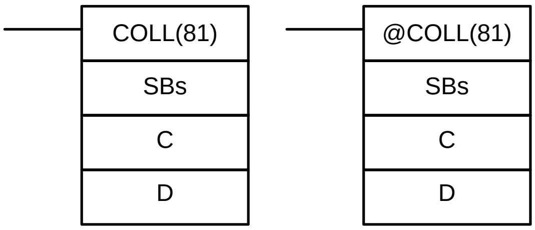
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

During stack operation, the value of the stack pointer+1 exceeds the length of the stack.

EQ: ON when the content of S is zero; otherwise OFF.

2-7-8 DATA COLLECT – COLL(81)

Ladder Symbols



Operand Data Areas

SBs: Source base word

IR, SR, AR, DM, EM, TIM/CNT, LR

C: Control word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

D: Destination word

IR, SR, AR, DM, EM, TIM/CNT, LR

Limitations C must be BCD.

DM 6144 to DM 6655 cannot be used for D.

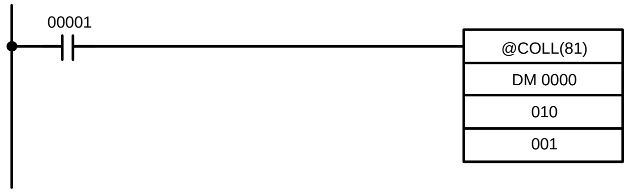
Description COLL(81) can be used for data collection, an FIFO stack operation, or an LIFO stack operation depending on the content of the control word, C.

Data Collection When bits 12 to 15 of C=0 to 7, COLL(81) is used for data collection. The entire contents of C specifies an offset, Of.

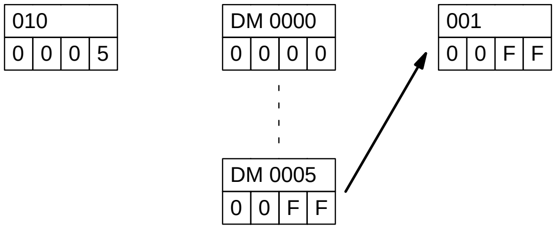
When the execution condition is OFF, COLL(81) is not executed. When the execution condition is ON, COLL(81) copies the content of SBs + Of to D, i.e., Of is added to SBs to determine the source word.

Note SBs and SBs+Of must be in the same data area.

Example
The following example shows how to use COLL(81) to copy the content of DM 0000+Of to IR 001. The content of 010 is #0005, so the content of DM 0005 (DM 0000 + 5) is copied to IR 001 when IR 00001 is ON.



Address	Instruction	Operands
00000	LD	00001
00001	@DIST(80)	
		DM 0000
		010
		001



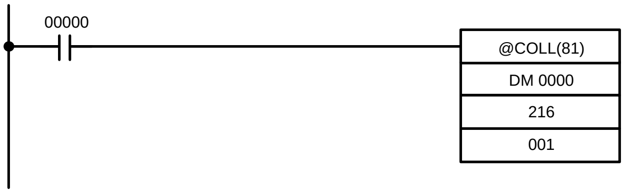
FIFO Stack Operation When bits 12 to 15 of C=9, COLL(81) can be used for an FIFO stack operation. The other 3 digits of C specify the number of words in the stack (000 to 999). The content of SBs is the stack pointer.

When the execution condition is ON, COLL(81) shifts the contents of each word within the stack down by one address, finally shifting the data from SBs+1 (the first value written to the stack) to the destination word (D). The content of the stack pointer (SBs) is then decremented by one.

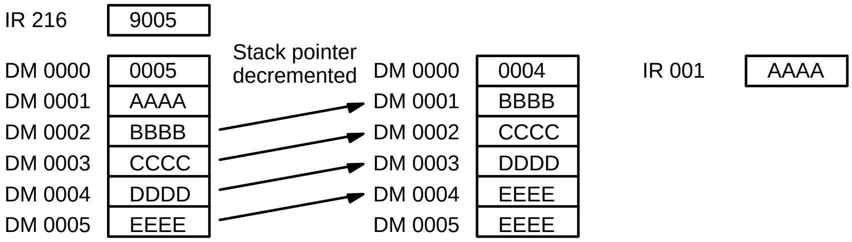
Note COLL(81) will be executed every cycle unless the differentiated form (@COLL(81)) is used or COLL(81) is used with DIFU(13) or DIFD(14).

Example
The following example shows how to use COLL(81) to create a stack between DM 0001 and DM 0005. DM 0000 acts as the stack pointer.

When IR 00000 goes from OFF to ON, COLL(81) shifts the contents of DM 0002 to DM 0005 down by one address, and shifts the data from DM 0001 to IR 001. The content of the stack pointer (DM 0000) is then decremented by one.



Address	Instruction	Operands
00000	LD	00000
00001	@COLL(81)	
		DM 0000
		216
		001



LIFO Stack Operation

When bits 12 to 15 of C=8, COLL(81) can be used for an LIFO stack operation. The other 3 digits of C specify the number of words in the stack (000 to 999). The content of SBs is the stack pointer.

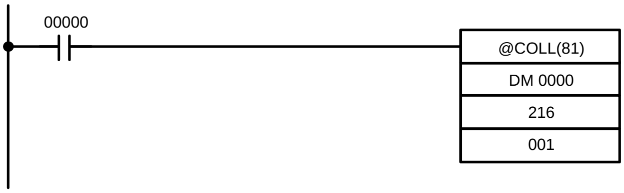
When the execution condition is ON, COLL(81) copies the data from the word indicated by the stack pointer (SBs+the content of SBs) to the destination word (D). The content of the stack pointer (SBs) is then decremented by one.

The stack pointer is the only word changed in the stack.

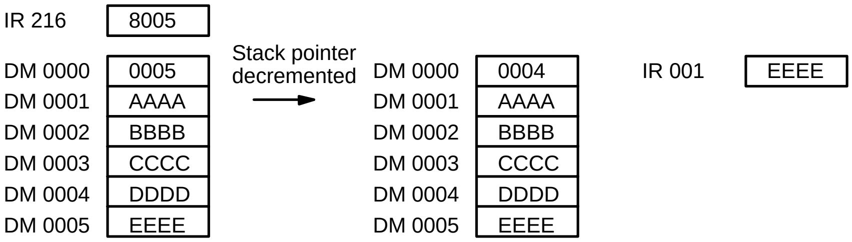
Note COLL(81) will be executed every cycle unless the differentiated form (@DIST(80)) is used or DIST(80) is used with DIFU(13) or DIFD(14).

Example
The following example shows how to use COLL(81) to create a stack between DM 0001 and DM 0005. DM 0000 acts as the stack pointer.

When IR 00000 goes from OFF to ON, COLL(81) copies the content of DM 0005 (DM 0000 + 5) to IR 001. The content of the stack pointer (DM 0000) is then decremented by one.



Address	Instruction	Operands
00000	LD	00000
00001	@COLL(81)	
		DM 0000
		216
		001



Flags

N: ON when the leftmost bit of the data being transferred is 1.

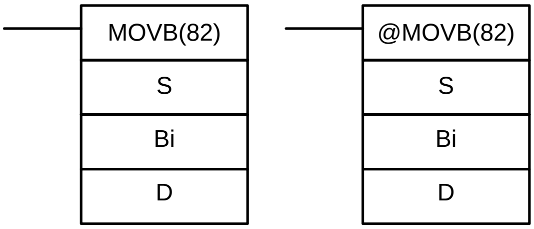
ER: The offset or stack length in the control word is not BCD.
Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

During stack operation, the value of the stack pointer exceeds the length of the stack; an attempt was made to write to a word beyond the end of the stack.

EQ: ON when the content of S is zero; otherwise OFF.

2-7-9 MOVE BIT – MOVB(82)

Ladder Symbols



Operand Data Areas

S: Source word
IR, SR, AR, DM, EM, LR, #

Bi: Bit designator (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR, #

D: Destination word
IR, SR, AR, DM, EM, LR

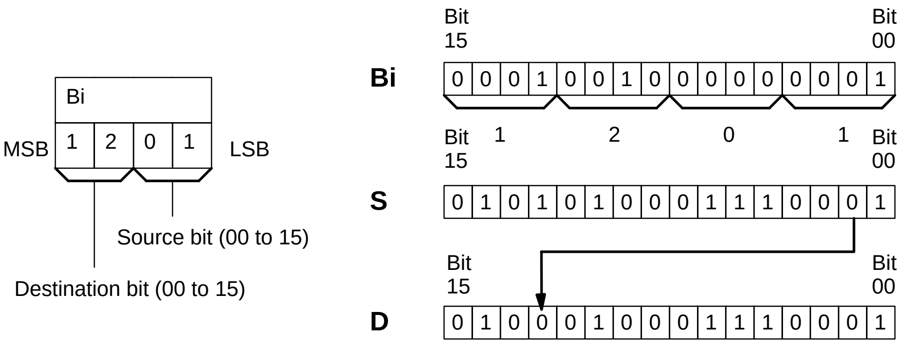
Limitations

The rightmost two digits and the leftmost two digits of Bi must each be between 00 and 15.

DM 6144 to DM 6655 cannot be used for Bi or D.

Description

When the execution condition is OFF, MOVB(82) is not executed. When the execution condition is ON, MOVB(82) copies the specified bit of S to the specified bit in D. The bits in S and D are specified by Bi. The rightmost two digits of Bi designate the source bit; the leftmost two bits designate the destination bit.



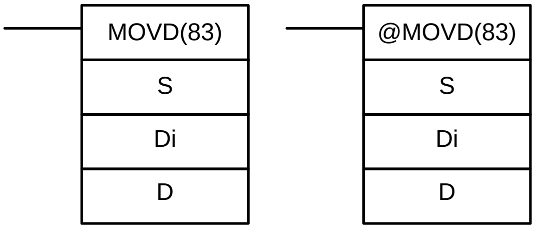
Flags

ER: Bi is not BCD, or it is specifying a non-existent bit (i.e., bit specification must be between 00 and 15).

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-7-10 MOVE DIGIT – MOVD(83)

Ladder Symbols



Operand Data Areas

S: Source word
IR, SR, AR, DM, EM, TIM/CNT, LR, #

Di: Digit designator (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR, #

D: Destination word
IR, SR, AR, DM, EM, TIM/CNT, LR

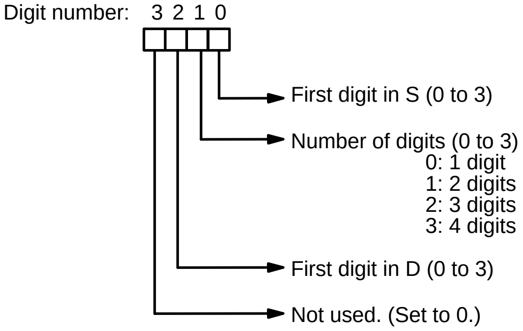
Limitations

The rightmost three digits of Di must each be between 0 and 3.

DM 6144 to DM 6655 cannot be used for Di or D.

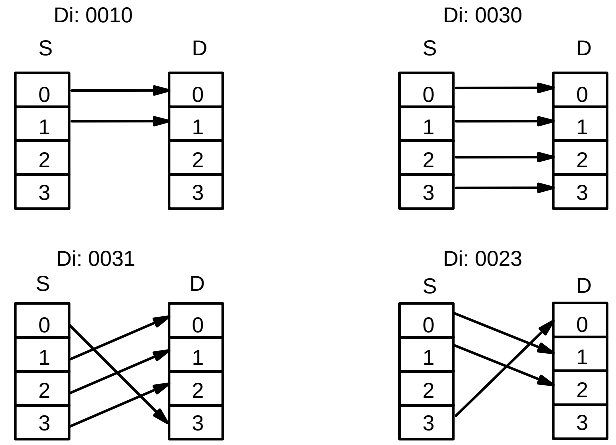
Description

When the execution condition is OFF, MOVD(83) is not executed. When the execution condition is ON, MOVD(83) copies the content of the specified digit(s) in S to the specified digit(s) in D. Up to four digits can be transferred at one time. The first digit to be copied, the number of digits to be copied, and the first digit to receive the copy are designated in Di as shown below. Digits from S will be copied to consecutive digits in D starting from the designated first digit and continued for the designated number of digits. If the last digit is reached in either S or D, further digits are used starting back at digit 0.



Digit Designator

The following show examples of the data movements for various values of Di.



Flags

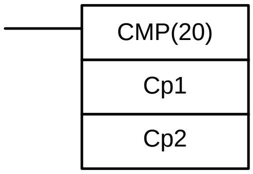
ER: At least one of the rightmost three digits of Di is not between 0 and 3.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

2-8 Comparison Instructions

2-8-1 COMPARE – CMP(20)

Ladder Symbols



Operand Data Areas

Cp1: First compare word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Cp2: Second compare word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Limitations

When comparing a value to the PV of a timer or counter, the value must be in BCD.

Description

When the execution condition is OFF, CMP(20) is not executed. When the execution condition is ON, CMP(20) compares Cp1 and Cp2 and outputs the result to the GR, EQ, and LE flags in the SR area.

Precautions

Placing other instructions between CMP(20) and the operation which accesses the EQ, LE, and GR flags may change the status of these flags. Be sure to access them before the desired status is changed.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON if Cp1 equals Cp2.

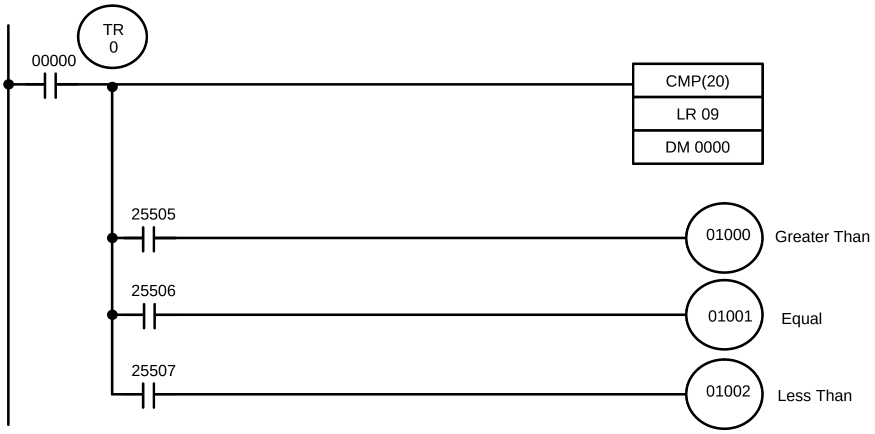
LE: ON if Cp1 is less than Cp2.

GR: ON if Cp1 is greater than Cp2.

Flag	Address	C1 < C2	C1 = C2	C1 > C2
GR	25505	OFF	OFF	ON
EQ	25506	OFF	ON	OFF
LE	25507	ON	OFF	OFF

Example:
Saving CMP(20) Results

The following example shows how to save the comparison result immediately. If the content of LR 09 is greater than that of DM 0000, IR 01000 is turned ON; if the two contents are equal, IR 01001 is turned ON; if content of LR 09 is less than that of IR 010, IR 01002 is turned ON. In some applications, only one of the three OUTs would be necessary, making the use of TR 0 unnecessary. With this type of programming, IR 01000, IR 01001, and IR 01002 are changed only when CMP(20) is executed.

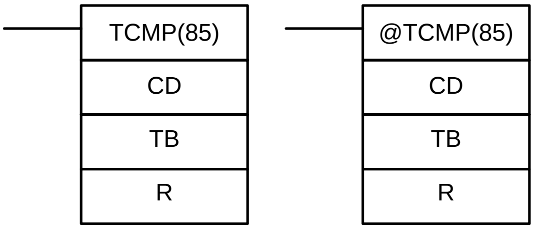


Address	Instruction	Operands	
00000	LD		00000
00001	OUT	TR	0
00002	CMP(20)		
		LR	09
		DM	0000
00003	AND		25505
00004	OUT		01000

Address	Instruction	Operands	
00005	LD	TR	0
00006	AND		25506
00007	OUT		01001
00008	LD	TR	0
00009	AND		25507
00010	OUT		01002

2-8-2 TABLE COMPARE – TCMP(85)

Ladder Symbols



Operand Data Areas

CD: Compare data

IR, SR, DM, EM, TIM/CNT, LR, #

TB: First comparison table word

IR, SR, DM, EM, TIM/CNT, LR

R: Result word

IR, SR, DM, EM, TIM/CNT, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, TCMP(85) is not executed. When the execution condition is ON, TCMP(85) compares CD to the content of TB, TB+1, TB+2, ..., and TB+15. If CD is equal to the content of any of these words, the corresponding bit in R is set, e.g., if the CD equals the content of TB, bit 00 is turned ON, if it equals that of TB+1, bit 01 is turned ON, etc. The rest of the bits in R will be turned OFF.

Flags

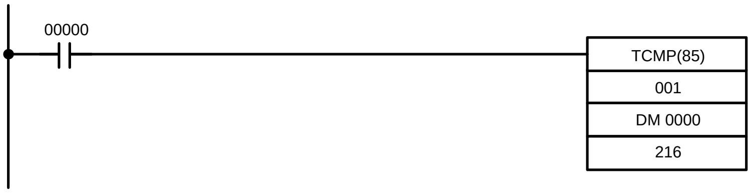
ER: The comparison table (i.e., TB through TB+15) exceeds the data area.

Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

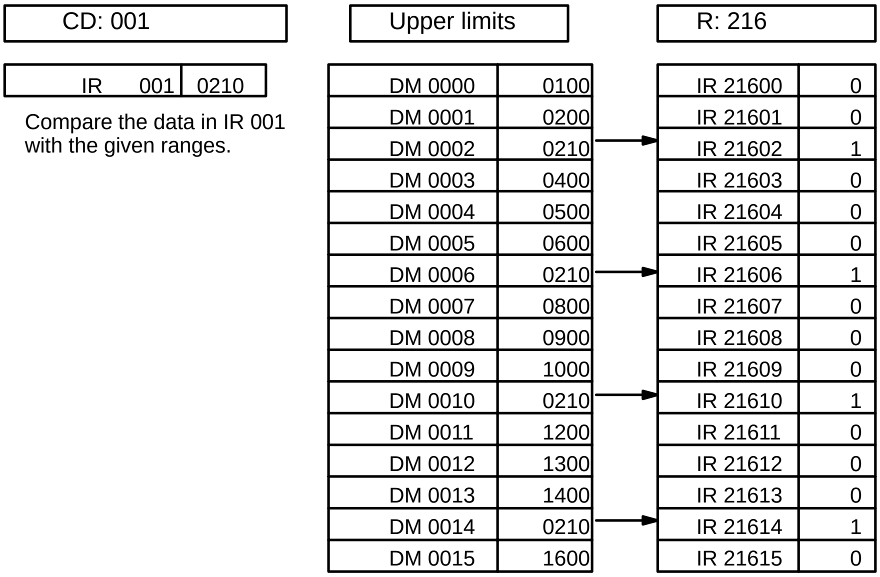
EQ: ON if the result word contains 0000 (i.e., none of the 16 words in the table equals CD).

Example

The following example shows the comparisons made and the results provided for TCMP(85). Here, the comparison is made during each cycle when IR 00000 is ON.



Address	Instruction	Operands	
00000	LD		00000
00001	TCMP(85)		
			001
		DM	0000
			216



CD: 001

IR	001	0210

Compare the data in IR 001 with the given ranges.

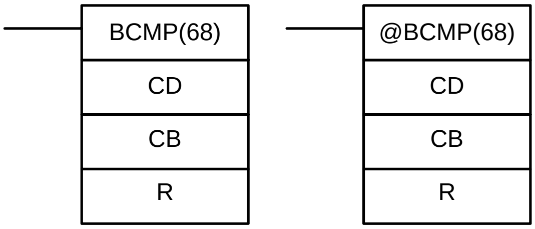
Upper limits

R: 216

Upper limits		R: 216	
DM 0000	0100	IR 21600	0
DM 0001	0200	IR 21601	0
DM 0002	0210	IR 21602	1
DM 0003	0400	IR 21603	0
DM 0004	0500	IR 21604	0
DM 0005	0600	IR 21605	0
DM 0006	0210	IR 21606	1
DM 0007	0800	IR 21607	0
DM 0008	0900	IR 21608	0
DM 0009	1000	IR 21609	0
DM 0010	0210	IR 21610	1
DM 0011	1200	IR 21611	0
DM 0012	1300	IR 21612	0
DM 0013	1400	IR 21613	0
DM 0014	0210	IR 21614	1
DM 0015	1600	IR 21615	0

2-8-3 BLOCK COMPARE – BCMP(68)

Ladder Symbols



Operand Data Areas

CD: Compare data

IR, SR, AR, DM, EM, TIM/CNT, LR, #

CB: First comparison block word

IR, SR, DM, EM, TIM/CNT, LR

R: Result word

IR, SR, AR, DM, EM, TIM/CNT, LR

Limitations

Each lower limit word in the comparison block must be less than or equal to the upper limit.

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, BCMP(68) is not executed. When the execution condition is ON, BCMP(68) compares CD to the ranges defined by a block consisting of CB, CB+1, CB+2, ..., CB+31. Each range is defined by two words, the first one providing the lower limit and the second word providing the upper limit. If CD is found to be within any of these ranges (inclusive of the upper and lower limits), the corresponding bit in R is set. The comparisons that are made and the corresponding bit in R that is set for each true comparison are shown below. The rest of the bits in R will be turned OFF.

CB ≤ CD ≤ CB+1	→	Bit 00
CB+2 ≤ CD ≤ CB+3	→	Bit 01
CB+4 ≤ CD ≤ CB+5	→	Bit 02
CB+6 ≤ CD ≤ CB+7	→	Bit 03
CB+8 ≤ CD ≤ CB+9	→	Bit 04
CB+10 ≤ CD ≤ CB+11	→	Bit 05
CB+12 ≤ CD ≤ CB+13	→	Bit 06
CB+14 ≤ CD ≤ CB+15	→	Bit 07
CB+16 ≤ CD ≤ CB+17	→	Bit 08
CB+18 ≤ CD ≤ CB+19	→	Bit 09
CB+20 ≤ CD ≤ CB+21	→	Bit 10
CB+22 ≤ CD ≤ CB+23	→	Bit 11
CB+24 ≤ CD ≤ CB+25	→	Bit 12
CB+26 ≤ CD ≤ CB+27	→	Bit 13
CB+28 ≤ CD ≤ CB+29	→	Bit 14
CB+30 ≤ CD ≤ CB+31	→	Bit 15

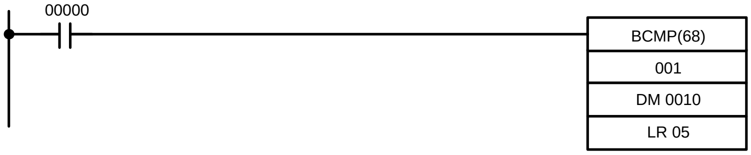
Flags

ER: The comparison block (i.e., CB through CB+31) exceeds the data area.

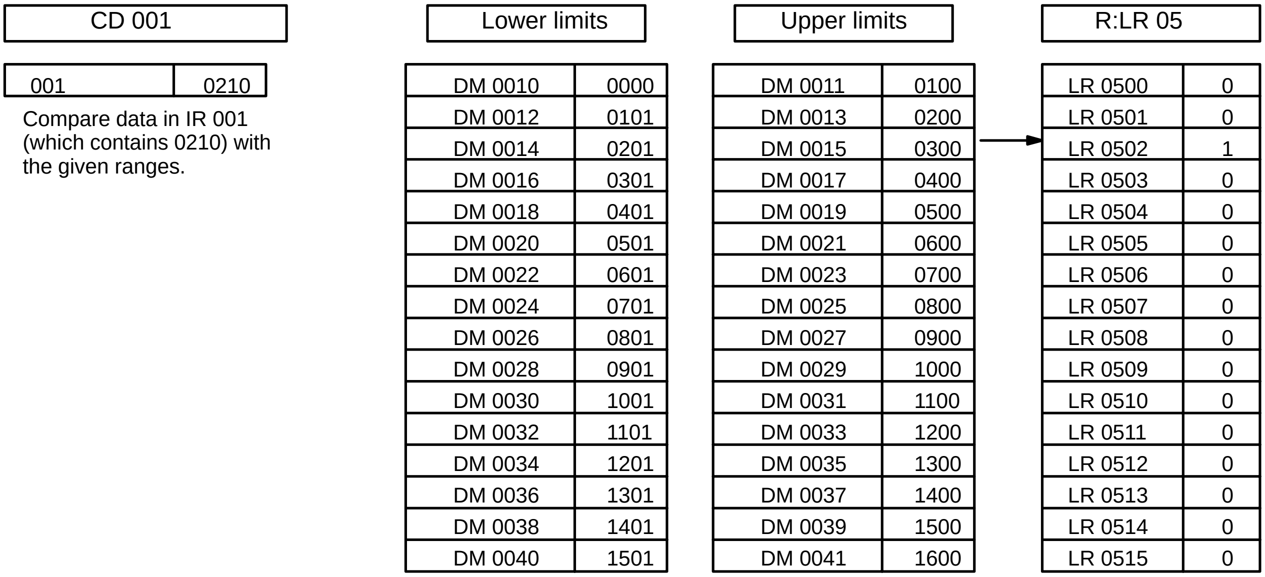
Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

Example

The following example shows the comparisons made and the results provided for BCMP(68). Here, the comparison is made during each cycle when IR 00000 is ON.



Address	Instruction	Operands	
00000	LD		00000
00001	BCMP(68)		
			001
		DM	0010
		LR	05



CD 001

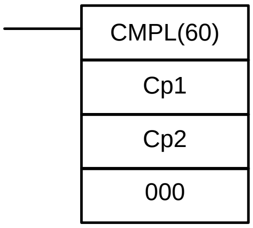
001	0210

Compare data in IR 001 (which contains 0210) with the given ranges.

Lower limits		Upper limits		R:LR 05	
DM 0010	0000	DM 0011	0100	LR 0500	0
DM 0012	0101	DM 0013	0200	LR 0501	0
DM 0014	0201	DM 0015	0300	LR 0502	1
DM 0016	0301	DM 0017	0400	LR 0503	0
DM 0018	0401	DM 0019	0500	LR 0504	0
DM 0020	0501	DM 0021	0600	LR 0505	0
DM 0022	0601	DM 0023	0700	LR 0506	0
DM 0024	0701	DM 0025	0800	LR 0507	0
DM 0026	0801	DM 0027	0900	LR 0508	0
DM 0028	0901	DM 0029	1000	LR 0509	0
DM 0030	1001	DM 0031	1100	LR 0510	0
DM 0032	1101	DM 0033	1200	LR 0511	0
DM 0034	1201	DM 0035	1300	LR 0512	0
DM 0036	1301	DM 0037	1400	LR 0513	0
DM 0038	1401	DM 0039	1500	LR 0514	0
DM 0040	1501	DM 0041	1600	LR 0515	0

2-8-4 DOUBLE COMPARE – CMPL(60)

Ladder Symbols



Operand Data Areas

Cp1: First word of first compare word pair
IR, SR, AR, DM, EM, TIM/CNT, LR

Cp2: First word of second compare word pair
IR, SR, AR, DM, EM, TIM/CNT, LR

Limitations

Cp1 and Cp1+1 must be in the same data area.

Cp2 and Cp2+1 must be in the same data area.

Set the third operand to 000.

Description

When the execution condition is OFF, CMPL(60) is not executed. When the execution condition is ON, CMPL(60) joins the 4-digit hexadecimal content of Cp1+1 with that of Cp1, and that of Cp2+1 with that of Cp2 to create two 8-digit hexadecimal numbers, Cp+1,Cp1 and Cp2+1,Cp2. The two 8-digit numbers are then compared and the result is output to the GR, EQ, and LE flags in the SR area.

Precautions

Placing other instructions between CMPL(60) and the operation which accesses the EQ, LE, and GR flags may change the status of these flags. Be sure to access them before the desired status is changed.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

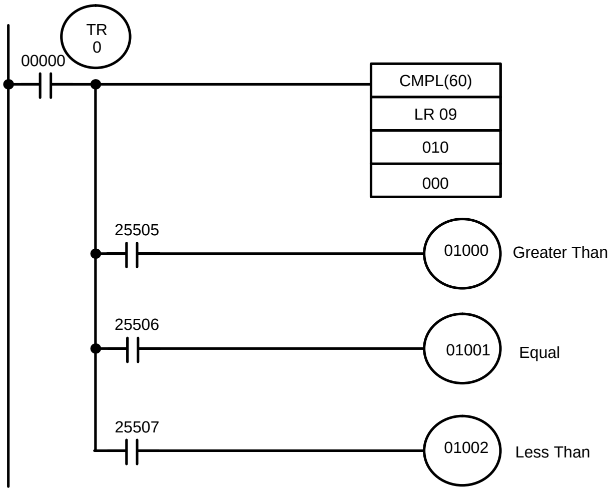
GR: ON if Cp1+1,Cp1 is greater than Cp2+1,Cp2.

EQ: ON if Cp1+1,Cp1 equals Cp2+1,Cp2.

LE: ON if Cp1+1,Cp1 is less than Cp2+1,Cp2.

Example: Saving CMPL(60) Results

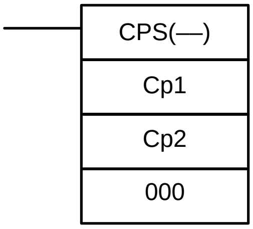
The following example shows how to save the comparison result immediately. If the content of LR 10, LR 09 is greater than that of IR 011, IR 010, then IR 01000 is turned ON; if the two contents are equal, IR 01001 is turned ON; if content of LR 10, LR 09 is less than that of IR 011, IR 010, then IR 01002 is turned ON. In some applications, only one of the three OUTs would be necessary, making the use of TR 0 unnecessary. With this type of programming, IR 01000, IR 01001, and IR 01002 are changed only when CMPL(60) is executed.



Address	Instruction	Operands	
00000	LD		00000
00001	OUT	TR	0
00002	CMPL(60)		
		LR	09
			010
00003	AND		25505
00004	OUT		01000
00005	LD	TR	0
00006	AND		25506
00007	OUT		01001
00008	LD	TR	0
00009	AND		25507
00010	OUT		01002

2-8-5 SIGNED BINARY COMPARE – CPS(––)

Ladder Symbols



Operand Data Areas

Cp1: First compare word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Cp2: Second compare word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

000

Not used. Set to 000.

Description

When the execution condition is OFF, CPS(—) is not executed. When the execution condition is ON, CPS(—) compares the 16-bit (4-digit) signed binary contents in Cp1 and Cp2 and outputs the result to the GR, EQ, and LE flags in the SR area.

Precautions

Placing other instructions between CPS(—) and the operation which accesses the EQ, LE, and GR flags may change the status of these flags. Be sure to access them before the desired status is changed.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON if Cp1 equals Cp2.

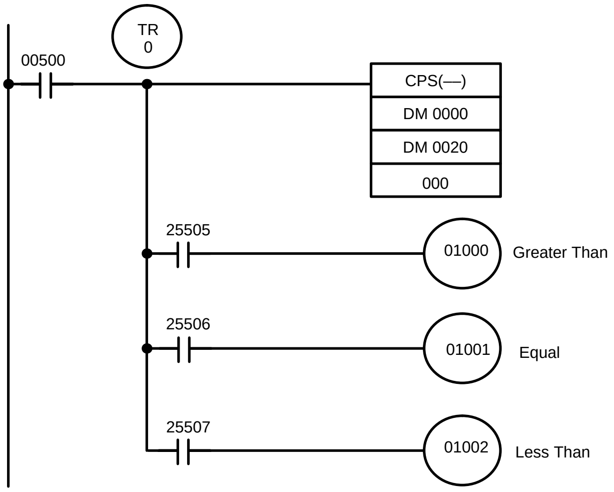
LE: ON if Cp1 is less than Cp2.

GR: ON if Cp1 is greater than Cp2.

Comparison result	**Flag status**		
	GR (SR 25505)	**EQ (SR 25506)**	**LE (SR 25507)**
Cp1 < Cp2	0	0	1
Cp1 = Cp2	0	1	0
Cp1 > Cp2	1	0	0

Example

In the following example, the content of DM 0000 is greater than that of DM 0020, so IR 01000 is turned ON and the other bits, IR 01001 and IR 01002, are turned OFF.



Address	**Instruction**	**Operands**	
00000	LD		00500
00001	OUT	TR	0
00002	CPS(—)		
		DM	0000
		DM	0020
			000
00003	AND		25505
00004	OUT		10000
00005	LD	TR	0
00006	AND		25506
00007	OUT		10001
00008	LD	TR	0
00009	AND		25507
00010	OUT		10002

Cp1: DM0000			
6	F	A	4

(28,580 decimal)

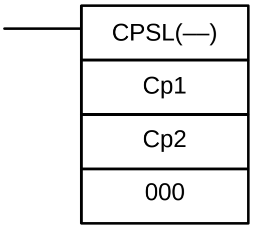
>

Cp2: DM 0020			
A	E	3	5

(–20,939 decimal)

2-8-6 DOUBLE SIGNED BINARY COMPARE – CPSL(––)

Ladder Symbols



Operand Data Areas

Cp1: First compare word

IR, SR, AR, DM, EM, TIM/CNT, LR

Cp2: Second compare word

IR, SR, AR, DM, EM, TIM/CNT, LR

000

Not used. Set to 000.

Description

When the execution condition is OFF, CPSL(—) is not executed. When the execution condition is ON, CPSL(—) compares the 32-bit (8-digit) signed binary contents in Cp1+1, Cp1 and Cp2+1, Cp2 and outputs the result to the GR, EQ, and LE flags in the SR area.

Precautions

Placing other instructions between CPSL(—) and the operation which accesses the EQ, LE, and GR flags may change the status of these flags. Be sure to access them before the desired status is changed.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON if Cp1+1, Cp1 equals Cp2+1, Cp2.

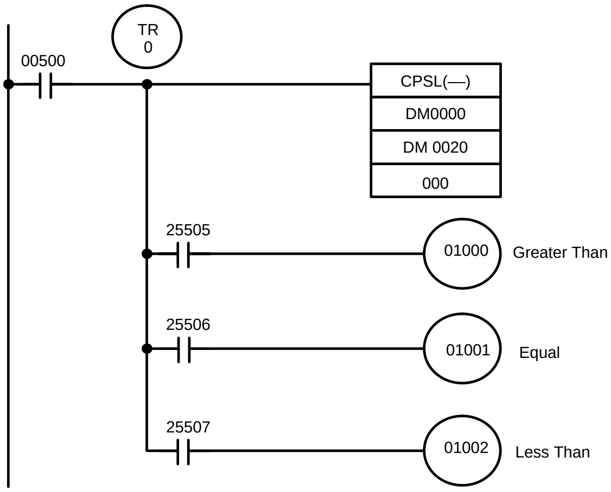
LE: ON if Cp1+1, Cp1 is less than Cp2+1, Cp2.

GR: ON if Cp1+1, Cp1 is greater than Cp2+1, Cp2.

Comparison result	**Flag status**		
	GR (SR 25505)	**EQ (SR 25506)**	**LE (SR 25507)**
Cp1+1, Cp1 < Cp2+1, Cp2	0	0	1
Cp1+1, Cp1 = Cp2+1, Cp2	0	1	0
Cp1+1, Cp1 > Cp2+1, Cp2	1	0	0

Example

In the following example, the content of DM 0001, DM 0000 is less than that of DM 0021, DM 0020, so IR 01002 is turned ON and the other bits, IR 01000 and IR 01001, are turned OFF.



Address	**Instruction**	**Operands**	
00000	LD		00500
00001	OUT	TR	0
00002	CPSL(—)		
		DM	0000
		DM	0020
			000
00003	AND		25505
00004	OUT		01000
00005	LD	TR	0
00006	AND		25506
00007	OUT		01001
00008	LD	TR	0
00009	AND		25507
00010	OUT		01002

Cp1+1: DM 0001				Cp1: DM 0000			
8	2	B	6	F	5	7	B

(–2,101,938,823 decimal)

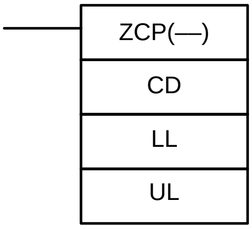
<

Cp2+1: DM 0021				Cp2: DM 0020			
0	5	6	A	9	9	D	B

(90,872,283 decimal)

2-8-7 AREA RANGE COMPARE – ZCP(––)

Ladder Symbol



Operand Data Areas

CD: Compare data

IR, SR, AR, DM, EM, TIM/CNT, LR, #

LL: Lower limit of range

IR, SR, AR, DM, EM, TIM/CNT, LR, #

UL: Upper limit of range

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Limitations

LL must be less than or equal to UL.

Description

When the execution condition is OFF, ZCP(—) is not executed. When the execution condition is ON, ZCP(—) compares CD to the range defined by lower limit LL and upper limit UL and outputs the result to the GR, EQ, and LE flags in the SR area. The resulting flag status is shown in the following table.

Comparison result	Flag status		
	GR (SR 25505)	**EQ (SR 25506)**	**LE (SR 25507)**
CD < LL	0	0	1
LL ≤ CD ≤ UL	0	1	0
UL < CD	1	0	0

Precautions

Placing other instructions between ZCP(—) and the operation which accesses the EQ, LE, and GR flags may change the status of these flags. Be sure to access them before the desired status is changed.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

LL is greater than UL.

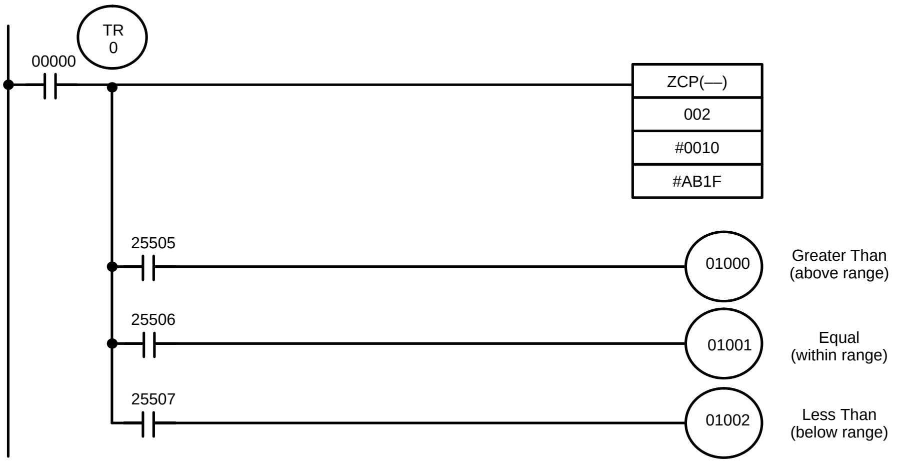
EQ: ON if LL ≤ CD ≤ UL

LE: ON if CD < LL.

GR: ON if CD > UL.

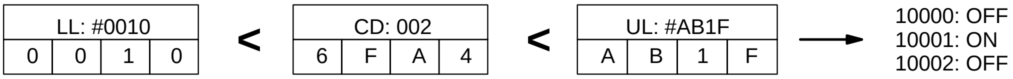
Example

In the following example, the content of IR 002 (#6FA4) is compared to the range #0010 to #AB1F. Since #0010 ≤ #6FA4 ≤ #AB1F, the EQ flag and IR 01001 are turned ON.



Address	Instruction	Operands	
00000	LD		00000
00001	OUT	TR	0
00002	ZCP(—)		
			002
		#	0010
		#	AB1F
00003	AND		25505

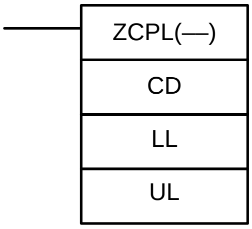
Address	Instruction	Operands	
00004	OUT		01000
00005	LD	TR	0
00006	AND		25506
00007	OUT		01001
00008	LD	TR	0
00009	AND		25507
00010	OUT		01002



LL: #0010					CD: 002					UL: #AB1F					
0	0	1	0	<	6	F	A	4	<	A	B	1	F	→	10000: OFF 10001: ON 10002: OFF

2-8-8 DOUBLE AREA RANGE COMPARE – ZCPL(––)

Ladder Symbol



Operand Data Areas

CD: Compare data

IR, SR, AR, DM, EM, LR

LL: Lower limit of range

IR, SR, AR, DM, EM, LR

UL: Upper limit of range

IR, SR, AR, DM, EM, LR

Limitations

The 8-digit value in LL+1,LL must be less than or equal to UL+1,UL.

Description

When the execution condition is OFF, ZCPL(—) is not executed. When the execution condition is ON, ZCPL(—) compares the 8-digit value in CD, CD+1 to the range defined by lower limit LL+1,LL and upper limit UL+1,UL and outputs the result to the GR, EQ, and LE flags in the SR area. The resulting flag status is shown in the following table.

Comparison result	Flag status		
	GR (SR 25505)	**EQ (SR 25506)**	**LE (SR 25507)**
CD , CD+1< LL+1,LL	0	0	1
LL+1,LL ≤ CD, CD+1 ≤ UL+1,UL	0	1	0
UL+1,UL < CD, CD+1	1	0	0

Precautions

Placing other instructions between ZCPL(—) and the operation which accesses the EQ, LE, and GR flags may change the status of these flags. Be sure to access them before the desired status is changed.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

LL+1,LL is greater than UL+1,UL.

EQ: ON if LL+1,LL ≤ CD, CD+1 ≤ UL+1,UL

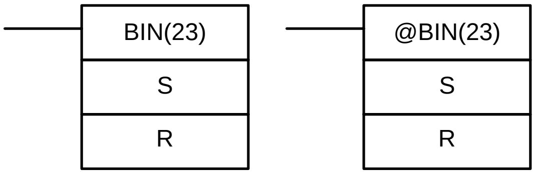
LE: ON if CD, CD+1 < LL+1,LL.

GR: ON if CD, CD+1 > UL+1,UL.

2-9 Conversion Instructions

2-9-1 BCD-TO-BINARY – BIN(23)

Ladder Symbols



Operand Data Areas

S: Source word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: Result word

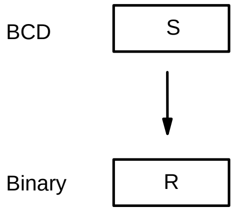
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, BIN(23) is not executed. When the execution condition is ON, BIN(23) converts the BCD content of S into the numerically equivalent binary bits, and outputs the binary value to R. Only the content of R is changed; the content of S is left unchanged.



BIN(23) can be used to convert BCD to binary so that displays on the Programming Console or any other programming device will appear in hexadecimal rather than decimal. It can also be used to convert to binary to perform binary arithmetic operations rather than BCD arithmetic operations, e.g., when BCD and binary values must be added.

Flags

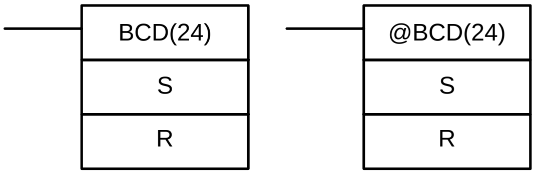
ER: The content of S is not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is zero.

2-9-2 BINARY-TO-BCD – BCD(24)

Ladder Symbols



Operand Data Areas

S: Source word (binary)

IR, SR, AR, DM, EM, LR

R: Result word

IR, SR, AR, DM, EM, LR

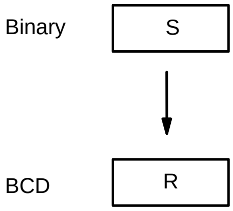
Limitations

If the content of S exceeds 270F, the converted result would exceed 9999 and BCD(24) will not be executed. When the instruction is not executed, the content of R remains unchanged.

DM 6144 to DM 6655 cannot be used for R.

Description

BCD(24) converts the binary (hexadecimal) content of S into the numerically equivalent BCD bits, and outputs the BCD bits to R. Only the content of R is changed; the content of S is left unchanged.



BCD(24) can be used to convert binary to BCD so that displays on the Programming Console or any other programming device will appear in decimal rather than hexadecimal. It can also be used to convert to BCD to perform BCD arithmetic operations rather than binary arithmetic operations, e.g., when BCD and binary values must be added.

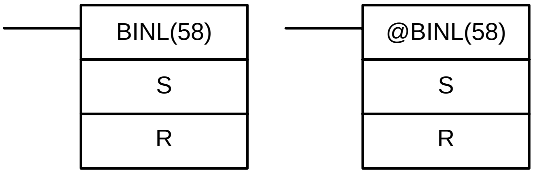
Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is zero.

2-9-3 DOUBLE BCD-TO-DOUBLE BINARY – BINL(58)

Ladder Symbols



Operand Data Areas

S: First source word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

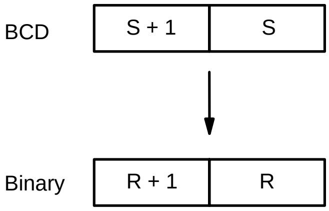
IR, SR, AR, DM, EM, LR

Limitations

DM 6143 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, BINL(58) is not executed. When the execution condition is ON, BINL(58) converts an eight-digit number in S and S+1 into 32-bit binary data, and outputs the converted data to R and R+1.



Flags

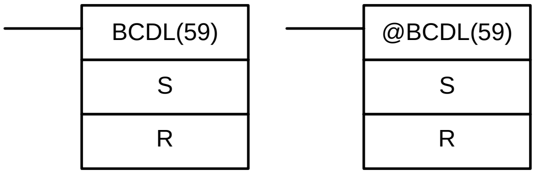
ER: The contents of S and/or S+1 words are not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is zero.

2-9-4 DOUBLE BINARY-TO-DOUBLE BCD – BCDL(59)

Ladder Symbols



Operand Data Areas

S: First source word (binary)

IR, SR, AR, DM, EM, LR

R: First result word

IR, SR, AR, DM, EM, LR

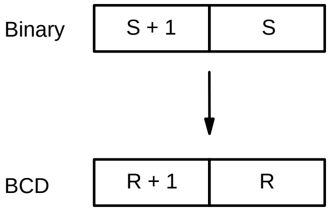
Limitations

If the content of S exceeds 05F5E0FF, the converted result would exceed 99999999 and BCDL(59) will not be executed. When the instruction is not executed, the content of R and R+1 remain unchanged.

DM 6143 to DM 6655 cannot be used for R.

Description

BCDL(59) converts the 32-bit binary content of S and S+1 into eight digits of BCD data, and outputs the converted data to R and R+1.



Flags

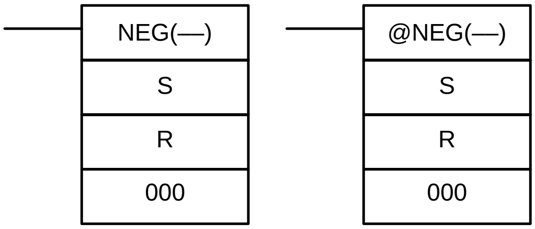
ER: Content of R and R+1 exceeds 99999999.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is zero.

2-9-5 2'S COMPLEMENT – NEG(––)

Ladder Symbols



Operand Data Areas

S: Source word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word

IR, SR, AR, DM, EM, LR

000

Not used. Set to 000.

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

Converts the four-digit hexadecimal content of the source word (S) to its 2's complement and outputs the result to the result word (R). This operation is effectively the same as subtracting S from 0000 and outputting the result to R; it will calculate the absolute value of negative signed binary data.

If the content of S is 0000, the content of R will also be 0000 after execution and EQ (SR 25506) will be turned on.

If the content of S is 8000, the content of R will also be 8000 after execution and UF (SR 25405) will be turned on.

Flags

N: ON when the leftmost bit of the result is 1.

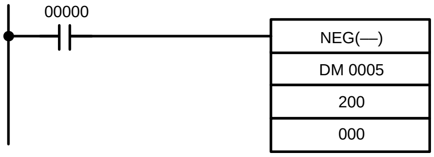
ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the content of R is zero after execution; otherwise OFF.

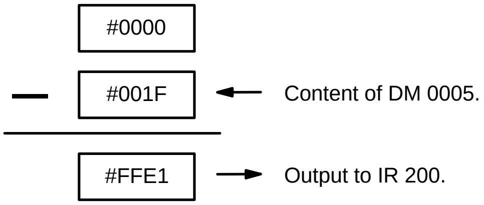
UF: ON when the content of S is 8000; otherwise OFF.

Example

The following example shows how to use NEG(––) to find the 2's complement of the content of DM 0005 and output the result to IR 200.

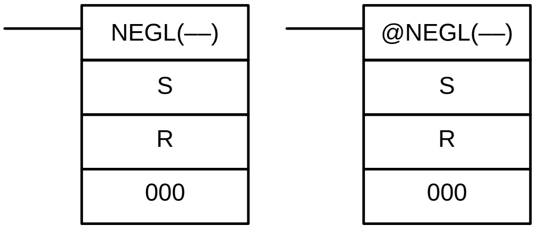


Address	Instruction	Operands
00000	LD	00000
00001	NEG(––)	
		DM 0005
		200
		000



2-9-6 DOUBLE 2’S COMPLEMENT – NEGL(––)

Ladder Symbols



Operand Data Areas

S: First source word

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

IR, SR, AR, DM, EM, LR

000

Not used. Set to 000.

Limitations

DM 6143 to DM 6655 cannot be used for R.

S and S+1 must be in the same data area, as must R and R+1.

Description

Converts the eight-digit hexadecimal content of the source words (S and S+1) to its 2’s complement and outputs the result to the result words (R and R+1). This operation is effectively the same as subtracting the eight-digit content S and S+1 from $0000 0000 and outputting the result to R and R+1; it will calculate the absolute value of negative signed binary data.

If the content of S is 0000 0000, the content of R will also be 0000 0000 after execution and EQ (SR 25506) will be turned on.

If the content of S is 8000 0000, the content of R will also be 8000 0000 after execution and UF (SR 25405) will be turned on.

Flags

N: ON when the leftmost bit of the result is 1.

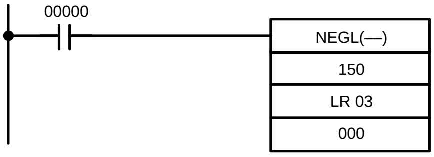
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the content of R+1, R is zero after execution; otherwise OFF.

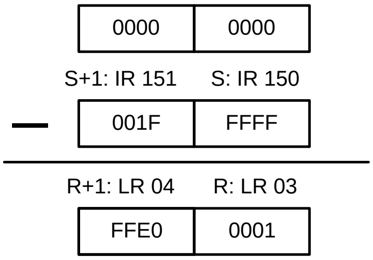
UF: ON when the content of S+1, S is 8000 0000; otherwise OFF.

Example

The following example shows how to use NEGL(—) to find the 2’s complement of the hexadecimal value in IR 151, IR 150 (001F FFFF) and output the result to LR 04, LR 03.



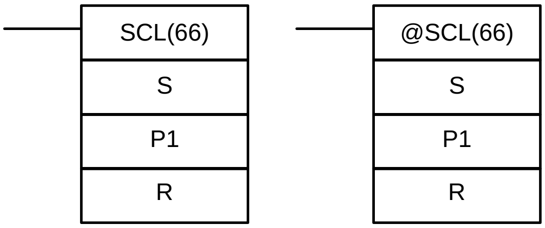
Address	Instruction	Operands	
00000	LD		00000
00001	NEGL(—)		
			150
		LR	03
			000



2-10 Data Control Instructions

2-10-1 SCALING – SCL(66)

Ladder Symbols



Operand Data Areas

S: Source word
IR, SR, AR, DM, EM, TIM/CNT, LR, #

P1: First parameter word
IR, SR, AR, DM, EM, TIM/CNT, LR

R: Result word
IR, SR, AR, DM, EM, LR

Limitations

S must be BCD.

P1 through P1+3 must be in the same data area.

DM 6144 to DM 6655 cannot be used for P1 through P1+3 or R.

Description

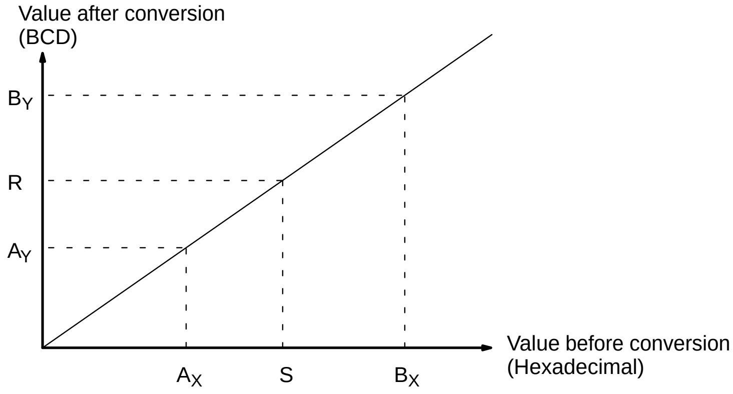
SCL(66) is used to linearly convert a 4-digit hexadecimal value to a 4-digit BCD value. Unlike BCD(24), which converts a 4-digit hexadecimal value to its 4-digit BCD equivalent ($S_{hex} \rightarrow S_{BCD}$), SCL(66) can convert the hexadecimal value according to a specified linear relationship. The conversion line is defined by two points specified in the parameter words P1 to P1+3.

When the execution condition is OFF, SCL(66) is not executed. When the execution condition is ON, SCL(66) converts the 4-digit hexadecimal value in S to the 4-digit BCD value on the line defined by points (P1, P1+1) and (P1+2, P1+3), and places the results in R. The results is rounded off to the nearest integer. If the results is less than 0000, then 0000 is written to R, and if the result is greater than 9999, then 9999 is written to R.

The following table shows the functions and ranges of the parameter words:

Parameter	Function	Range	Comments
P1	BCD point #1 (A_Y)	0000 to 9999	---
P1+1	Hex. point #1 (A_X)	0000 to FFFF	Do not set P1+1=P1+3.
P1+2	BCD point #2 (B_Y)	0000 to 9999	---
P1+3	Hex. point #2 (B_X)	0000 to FFFF	Do not set P1+3=P1+1.

The following diagram shows the source word, S, converted to D according to the line defined by points (A_Y, A_X) and (B_Y, B_X).



The results can be calculated by first converting all values to BCD and then using the following formula.

$\text{Results} = B_Y - [(B_Y - A_Y)/(B_X - A_X) \text{ X } (B_X - S)]$

Flags

ER: The value in P1+1 equals that in P1+3.

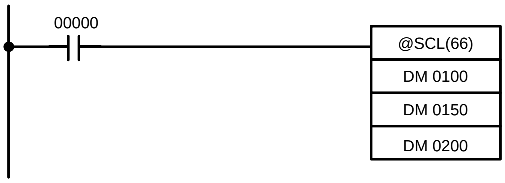
Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

P1 and P1+3 are not in the same data area, or other setting error.

EQ: ON when the result, R, is 0000.

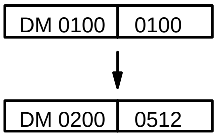
Example

When IR 00000 is turned ON in the following example, the BCD source data in DM 0100 (#0100) is converted to hexadecimal according to the parameters in DM 0150 to DM 0153. The result (#0512) is then written to DM 0200.



Address	Instruction	Operands	
00000	LD		00000
00001	@SCL(66)		
		DM	0100
		DM	0150
		DM	0200

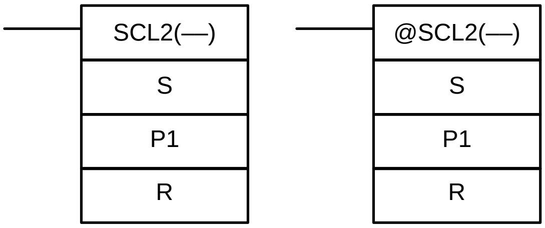
DM 0150	0010
DM 0151	0005
DM 0152	0050
DM 0153	0019


DM 0100 0100

DM 0200 0512

2-10-2 SIGNED BINARY TO BCD SCALING – SCL2(––)

Ladder Symbols



Operand Data Areas

S: Source word
IR, SR, AR, DM, EM, LR

P1: First parameter word
IR, SR, AR, DM, EM, LR

R: Result word
IR, SR, AR, DM, EM, LR

Limitations

S must be BCD.

P1 through P1+2 must be in the same data area.

DM 6144 to DM 6655 cannot be used for R.

Description

SCL2(—) is used to linearly convert a 4-digit signed hexadecimal value to a 4-digit BCD value. Unlike BCD(24), which converts a 4-digit hexadecimal value to its 4-digit BCD equivalent ($S_{hex} \rightarrow S_{BCD}$), SCL2(—) can convert the signed hexadecimal value according to a specified linear relationship. The conversion line is defined by the x-intercept and the slope of the line specified in the parameter words P1 to P1+2.

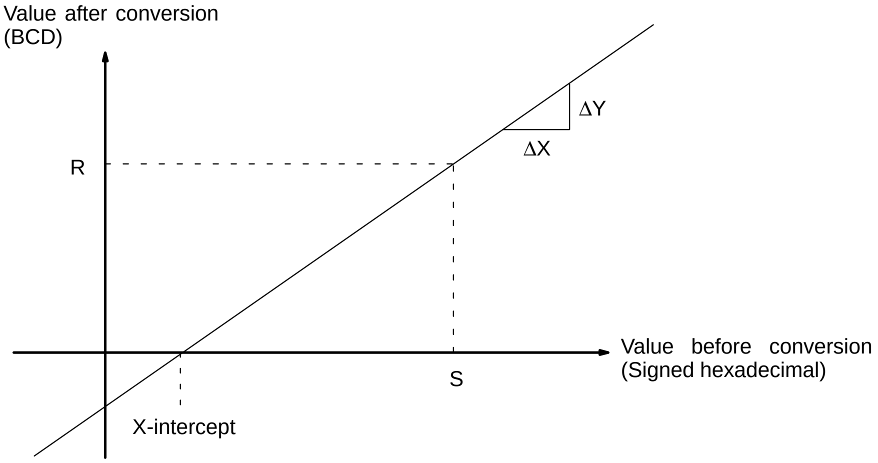
When the execution condition is OFF, SCL2(—) is not executed. When the execution condition is ON, SCL2(—) converts the 4-digit signed hexadecimal value in S to the 4-digit BCD value on the line defined by the x-intercept (P1, 0) and the slope (P1+2 ÷ P1+1) and places the results in R. The result is rounded off to the nearest integer.

If the result is negative, then CY is set to 1. If the result is less than –9999, then –9999 is written to R. If the result is greater than 9999, then 9999 is written to R.

The following table shows the functions and ranges of the parameter words:

Parameter	Function	Range
P1	x-intercept (signed hex.)	8000 to 7FFF (–32,768 to 32,767)
P1+1	ΔX (signed hex.)	8000 to 7FFF (–32,768 to 32,767)
P1+2	ΔY (BCD)	0000 to 9999

The following diagram shows the source word, S, converted to R according to the line defined by the point (P1, 0) and slope ΔY/ΔX.


Value after conversion (BCD)
ΔY
ΔX
R
Value before conversion (Signed hexadecimal)
S
X-intercept

The result can be calculated by first converting all signed hexadecimal values to BCD and then using the following formula.

$$R - \frac{\Delta Y}{\Delta X} \times (S–P1)$$

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

P1 and P1+2 are not in the same data area, or other setting error.

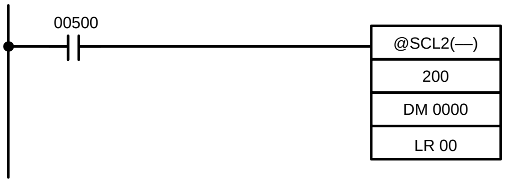
CY: ON when the result, R, is negative.

EQ: ON when the result, R, is 0000.

Example

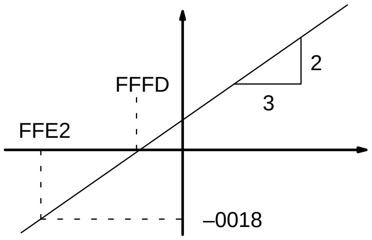
When IR 00500 is turned ON in the following example, the signed binary source data in 001 (#FFE2) is converted to BCD according to the parameters in

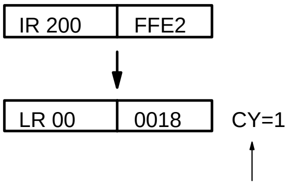
DM 0000 to DM 0002. The result (#0018) is then written to LR 00 and CY is turned ON because the result is negative.



Address	Instruction	Operands
00000	LD	00500
00001	@SCL2(—)	
		200
		DM 0000
		LR 00

DM 0000	FFFD
DM 0001	0003
DM 0002	0002





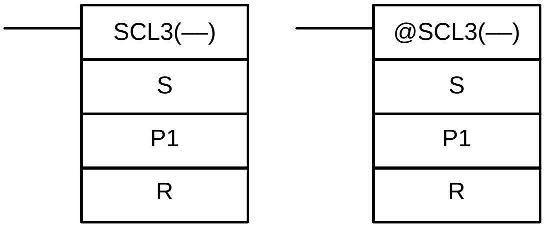
CY flag is turned ON because the conversion result is negative.

$$R - \frac{0002}{0003} \times (FFE2–FFFD)$$

$$- \frac{2}{3} \times (–1B) - –18$$

2-10-3 BCD TO SIGNED BINARY SCALING – SCL3(––)

Ladder Symbols



Operand Data Areas

S: Source word

IR, SR, AR, DM, EM, LR

P1: First parameter word

IR, SR, AR, DM, EM, LR

R: Result word

IR, SR, AR, DM, EM, LR

Limitations

P1+1 must be BCD.

P1 through P1+4 must be in the same data area.

DM 6144 to DM 6655 cannot be used for R.

Description

SCL3(—) is used to linearly convert a 4-digit BCD value to 4-digit signed hexadecimal. SCL3(—) converts the BCD value according to a specified linear relationship. The conversion line is defined by the y-intercept and the slope of the line specified in the parameter words P1 to P1+2.

When the execution condition is OFF, SCL3(—) is not executed. When the execution condition is ON, SCL3(—) converts the 4-digit BCD value in S to the 4-digit signed hexadecimal value on the line defined by the y-intercept (0, P1) and the slope (P1+2 ÷ P1+1) and places the result in R. The result is rounded off to the nearest integer.

The content of S can be 0000 to 9999, but S will be treated as a negative value if CY=1, so the effective range of S is actually –9999 to 9999. Be sure to set the desired sign in CY using STC(40) or CLC(41).

Parameter words P1+3 and P1+4 define upper and lower limits for the result. If the result is greater than the upper limit in P1+3, then the upper limit is written to

R. If the result is less than the lower limit in P1+4, then the lower limit is written to R.

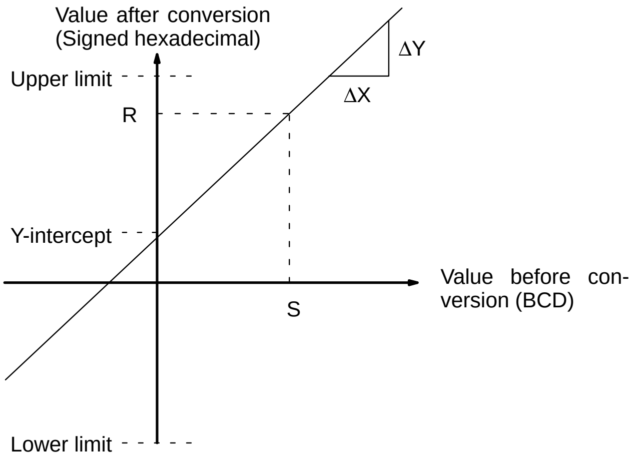
Note The upper and lower limits for a 12-bit Analog Input Unit would be 07FF and F800.

The following table shows the functions and ranges of the parameter words:

Parameter	Function	Range
P1	x-intercept (signed hex.)	8000 to 7FFF (–32,768 to 32,767)
P1+1	ΔX (BCD)	0001 to 9999
P1+2	ΔY (signed hex.)	8000 to 7FFF (–32,768 to 32,767)
P1+3	Upper limit (signed hex.)	8000 to 7FFF (–32,768 to 32,767)
P1+4	Lower limit (signed hex.)	8000 to 7FFF (–32,768 to 32,767)

Note Do not set 0000 for ΔX (4 digits BCD) in the second word (P1+1). The contents of P1+1 is used for division and correct conversion cannot be obtained when dividing by 0000. Correct results also cannot be obtained if a hexadecimal value is used. Always use BCD data between 0001 and 9999 for P1+1.

The following diagram shows the source word, S, converted to R according to the line defined by the point (0, P1) and slope ΔY/ΔX.



The result can be calculated by first converting all BCD values to signed binary and then using the following formula.

$$R - \frac{\Delta Y}{\Delta X} \times S \cong + \; P1$$

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

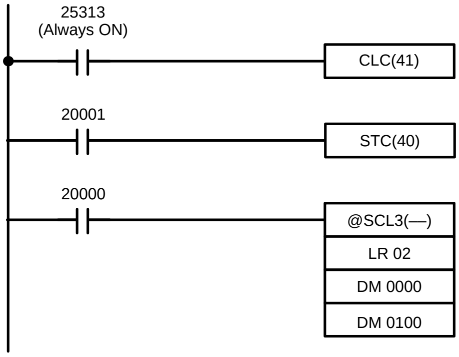
The content of S is not BCD.

CY: CY is not changed by SCL3(—). (CY shows the sign of S before execution.)

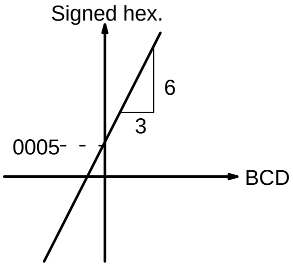
EQ: ON when the result, R, is 0000.

Example

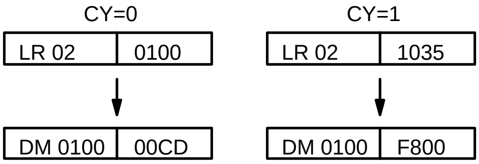
The status of IR 20001 determines the sign of the BCD source word in the following example. If IR 20001 is ON, then the source word is negative. When IR 20000 is turned ON, the BCD source data in LR 02 is converted to signed binary according to the parameters in DM 0000 to DM 0004. The result is then written to DM 0100. (In the second conversion, the signed binary equivalent of –1035 is less than the lower limit specified in DM 0004, so the lower limit is written to DM 0100.)



Address	Instruction	Operands	
00000	LD		25313
00001	CLC(41)		
00002	LD		20001
00101	STC(40)		
00004	LD		20000
00005	SCL3(—)		
		LR	02
		DM	0000
		DM	0100

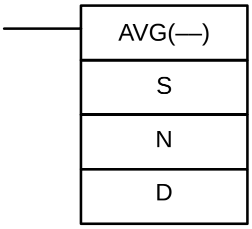


DM 0000	0005
DM 0001	0003
DM 0002	0006
DM 0003	07FF
DM 0004	F800



2-10-4 AVERAGE VALUE – AVG(––)

Ladder Symbols



Operand Data Areas

S: Source word
IR, SR, AR, DM, EM, TIM/CNT, LR

N: Number of cycles
IR, SR, AR, DM, EM, TIM/CNT, LR, #

D: First destination word
IR, SR, AR, DM, EM, LR

Limitations

S must be hexadecimal.

N must be BCD from #0001 to #0064.

D and D+N+1 must be in the same data area.

DM 6144 to DM 6655 cannot be used for S, N, or D to D+N+1.

Description

AVG(—) is used to calculate the average value of S over N cycles.

When the execution condition is OFF, AVG(—) is not executed.

Each time that AVG(—) is executed, the content of S is stored in words D+2 to D+N+1. On the first execution, AVG(—) writes the content of S to D+2; on the second execution it writes the content of S to D+3, etc. On the Nth execution, AVG(—) writes the content of S stored in D+N+1, AVG(—) calculates the average value of the values stored in D+2 to D+N+1, and writes the average to D.

The following diagram shows the function of words D to D+N+1.

D	Average value (after N or more executions)
D+1	Used by the system.
D+2	Content of S from the 1st execution of AVG(—)
D+3	Content of S from the 2nd execution of AVG(—)
⋮	⋮ ⋮ ⋮
D+N+1	Content of S from the Nth execution of AVG(—)

Precautions

The average value is calculated in binary. Be sure that the content of S is in binary.

N must be BCD from #0001 to #0064. If the content of N ≥ #0065, AVG(—) will operate with N=64.

The average value will be rounded off to the nearest integer value. (0.5 is rounded up to 1.)

Leave the contents of D+1 set to #0000 after the first execution of AVG(—).

Flags

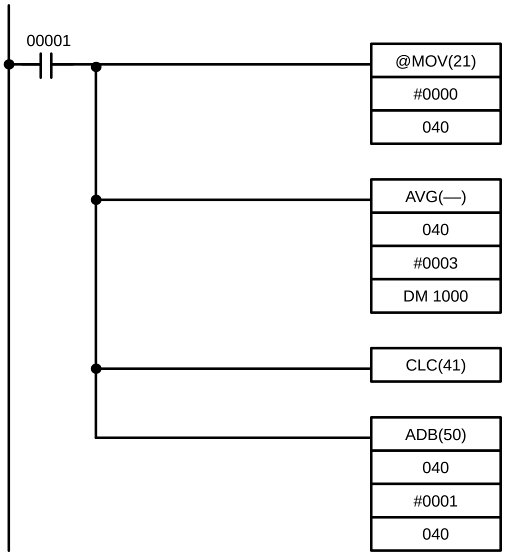
ER: Indirectly addressed EM/DM word is non-existent. (Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

One or more operands have been set incorrectly.

D and D+N+1 are not in the same data area.

Example

In the following example, the content of IR 040 is set to #0000 and then incremented by 1 each cycle. For the first two cycles, AVG(—) moves the content of IR 040 to DM 1002 and DM 1003. On the third and later cycles AVG(—) calculates the average value of the contents of DM 1002 to DM 1004 and writes that average value to DM 1000.



Address	Instruction	Operands
00000	LD	00001
00001	@MOV(21)	
		# 0000
		040
00002	AVG(—)	
		040
		# 0003
		DM 1000
00003	CLC(41)	
00004	ADB(50)	
		040
		# 0001
		040

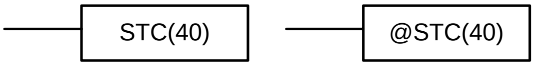
	1st cycle	2nd cycle	3rd cycle	4th cycle
IR 040	0000	0001	0002	0003

	1st cycle	2nd cycle	3rd cycle	4th cycle	
DM 1000	0000	0001	0001	0002	Average
DM 1001					Used by the system.
DM 1002	0000	0000	0000	0003	Previous values of IR 40
DM 1003	---	0001	0001	0001	
DM 1004	---	---	0002	0002	

2-11 Special Instructions

2-11-1 SET CARRY – STC(40)

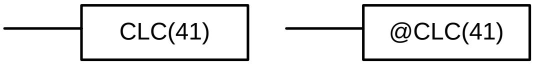
Ladder Symbols



When the execution condition is OFF, STC(40) is not executed. When the execution condition is ON, STC(40) turns ON CY (SR 25504).

2-11-2 CLEAR CARRY – CLC(41)

Ladder Symbols



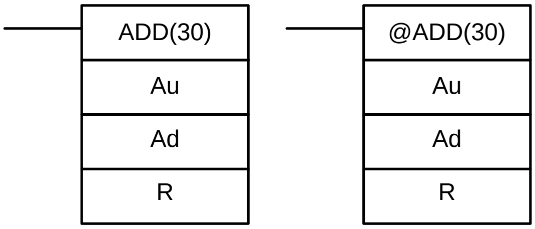
When the execution condition is OFF, CLC(41) is not executed. When the execution condition is ON, CLC(41) turns OFSF CY (SR 25504).

CLEAR CARRY is used to reset (turn OFF) CY (SR 25504) to “0.”

2-12 Symbol Math Instructions

2-12-1 BCD ADD – ADD(30)

Ladder Symbols



Operand Data Areas

Au: Augend word (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR, #
Ad: Addend word (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR, #
R: Result word
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, ADD(30) is not executed. When the execution condition is ON, ADD(30) adds the contents of Au, Ad, and CY, and places the result in R. CY will be set if the result is greater than 9999.

$$\boxed{Au} + \boxed{Ad} + \boxed{CY} \rightarrow \boxed{CY} \quad \boxed{R}$$

Flags

ER: Au and/or Ad is not BCD.

Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

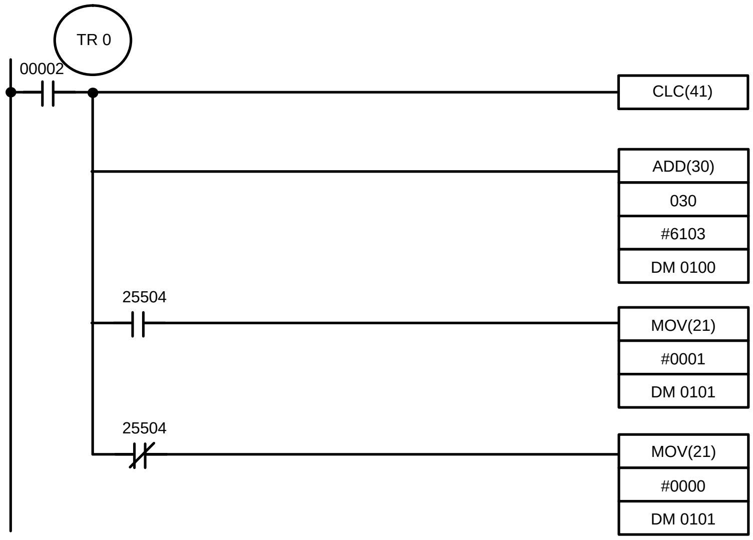
CY: ON when there is a carry in the result.

EQ: ON when the result is 0.

Example

If 00002 is ON, the program represented by the following diagram clears CY with CLC(41), adds the content of IR 030 to a constant (6103), places the result in DM

0100, and then moves either all zeros or 0001 into DM 0101 depending on the status of CY (25504). This ensures that any carry from the last digit is preserved in R+1 so that the entire result can be later handled as eight-digit data.

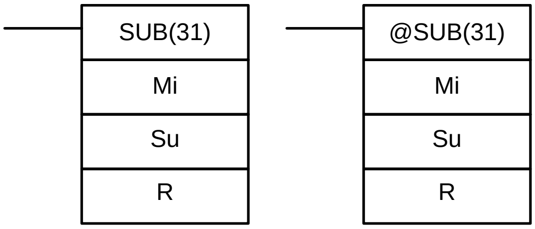


Address	Instruction	Operands
00000	LD	00002
00001	OUT	TR 0
00002	CLC(41)	
00003	ADD(30)	
		030
		# 6103
		DM 0100
00004	AND	25504
00005	MOV(21)	
		# 0001
		DM 0101
00006	LD	TR 0
00007	AND NOT	25504
00008	MOV(21)	
		# 0000
		DM 0101

Although two ADD(30) can be used together to perform eight-digit BCD addition, ADDL(54) is designed specifically for this purpose.

2-12-2 BCD SUBTRACT – SUB(31)

Ladder Symbols



Operand Data Areas

Mi: Minuend word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Su: Subtrahend word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word

IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, SUB(31) is not executed. When the execution condition is ON, SUB(31) subtracts the contents of Su and CY from Mi, and places the result in R. If the result is negative, CY is set and the 10's complement of the actual result is placed in R. To convert the 10's complement to the true result, subtract the content of R from zero (see example below).

$$\boxed{\text{Mi}} - \boxed{\text{Su}} - \boxed{\text{CY}} \rightarrow \boxed{\text{CY}} \quad \boxed{\text{R}}$$

Flags

ER: Mi and/or Su is not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: ON when the result is negative, i.e., when Mi is less than Su plus CY.

EQ: ON when the result is 0.

⚠ **Caution** Be sure to clear the carry flag with CLC(41) before executing SUB(31) if its previous status is not required, and check the status of CY after doing a subtraction with SUB(31). If CY is ON as a result of executing SUB(31) (i.e., if the result is negative), the result is output as the 10's complement of the true answer. To convert the output result to the true value, subtract the value in R from 0.

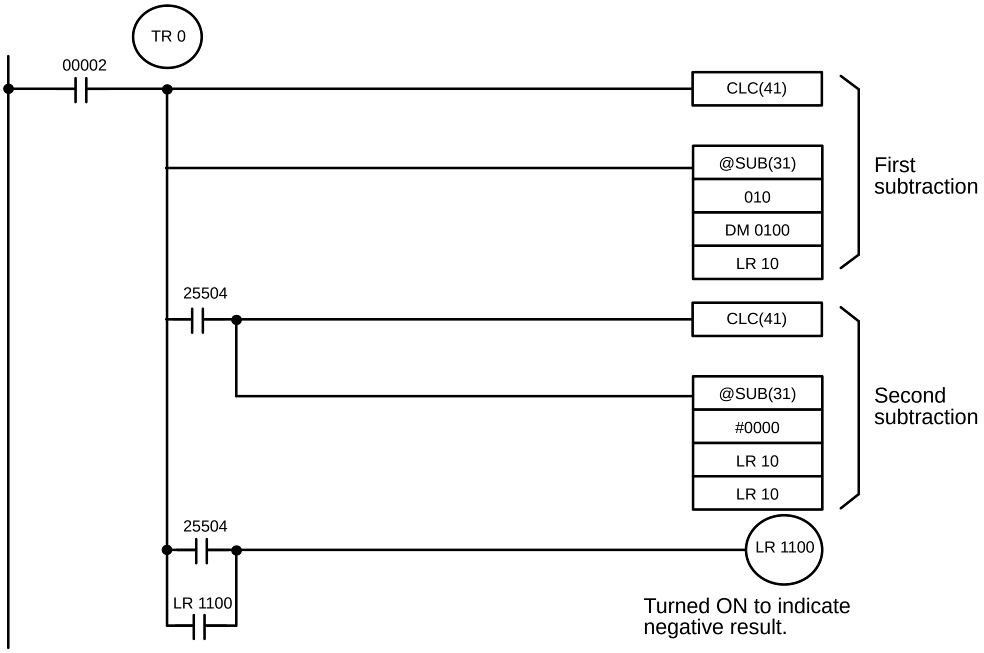
Example

When 00002 is ON, the following ladder program clears CY, subtracts the contents of DM 0100 and CY from the content of 010 and places the result in LR 10.

If CY is set by executing SUB(31), the result in LR 10 is subtracted from zero (note that CLC(41) is again required to obtain an accurate result), the result is placed back in LR 10, and LR 1100 is turned ON to indicate a negative result.

If CY is not set by executing SUB(31), the result is positive, the second subtraction is not performed, and LR 1100 is not turned ON. LR 1100 is programmed as a self-maintaining bit so that a change in the status of CY will not turn it OFF when the program is rescanned.

In this example, differentiated forms of SUB(31) are used so that the subtraction operation is performed only once each time 00002 is turned ON. When another subtraction operation is to be performed, 00002 will need to be turned OFF for at least one cycle (resetting LR 1100) and then turned back ON.



Address	Instruction	Operands	
00000	LD		00002
00001	OUT	TR	0
00002	CLC(41)		
00003	@SUB(31)		
			010
		DM	0100
		LR	10
00004	AND		25504
00005	CLC(41)		
00006	@SUB(31)		
		#	0000
		LR	10
		LR	10
00007	LD	TR	0
00008	LD		25504
00009	OR	LR	1100
00010	AND LD		
00011	OUT	LR	1100

The first and second subtractions for this diagram are shown below using example data for 010 and DM 0100.

Note The actual SUB(31) operation involves subtracting Su and CY from 10,000 plus Mi. For positive results the leftmost digit is truncated. For negative results the 10s complement is obtained. The procedure for establishing the correct answer is given below.

First Subtraction

IR 010 1029
DM 0100 – 3452
CY – 0

LR 10 7577 (1029 + (10000 – 3452))
CY 1 (negative result)

Second Subtraction

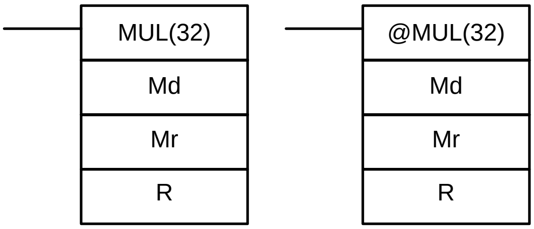
0000
LR 10 –7577
CY –0

LR 10 2423 (0000 + (10000 – 7577))
CY 1 (negative result)

In the above case, the program would turn ON LR 1100 to indicate that the value held in LR 10 is negative.

2-12-3 BCD MULTIPLY – MUL(32)

Ladder Symbols



Operand Data Areas

Md: Multiplicand (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Mr: Multiplier (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

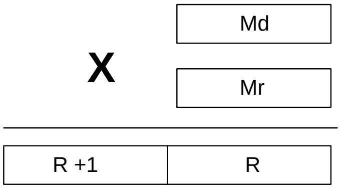
R: First result word

IR, SR, AR, DM, EM, LR

Limitations DM 6143 to DM 6655 cannot be used for R.

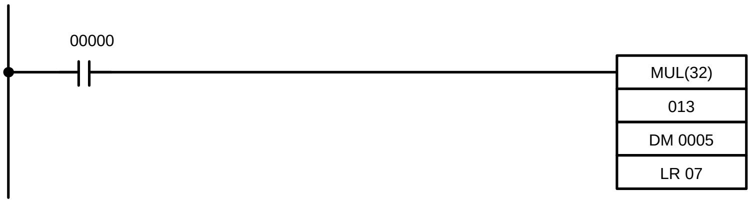
Description

When the execution condition is OFF, MUL(32) is not executed. When the execution condition is ON, MUL(32) multiplies Md by the content of Mr, and places the result In R and R+1.

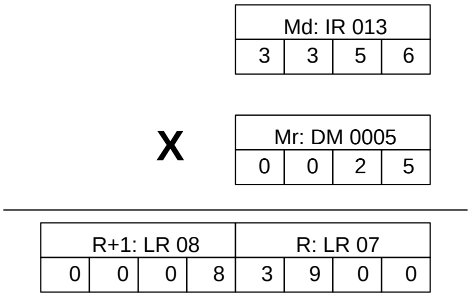


Example

When IR 00000 is ON with the following program, the contents of IR 013 and DM 0005 are multiplied and the result is placed in LR 07 and LR 08. Example data and calculations are shown below the program.



Address	Instruction	Operands
00000	LD	00000
00001	MUL(32)	
		013
		DM 0005
		LR 07



Flags

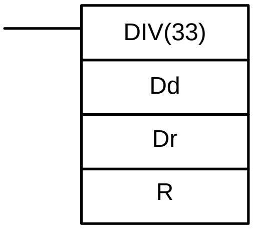
ER: Md and/or Mr is not BCD.

Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

2-12-4 BCD DIVIDE – DIV(33)

Ladder Symbol



Operand Data Areas

Dd: Dividend word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Dr: Divisor word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: First result word (BCD)

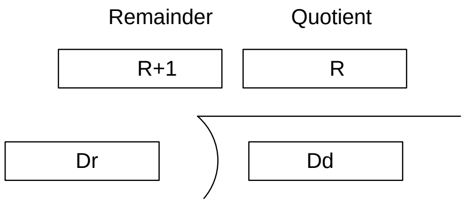
IR, SR, AR, DM, EM, LR

Limitations

R and R+1 must be in the same data area. DM 6143 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, DIV(33) is not executed and the program moves to the next instruction. When the execution condition is ON, Dd is divided by Dr and the result is placed in R and R + 1: the quotient in R and the remainder in R + 1.



Flags

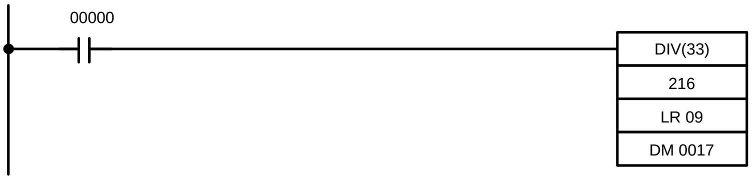
ER: Dd or Dr is not in BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

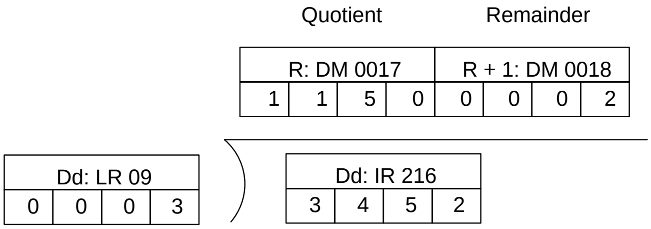
EQ: ON when the result is 0.

Example

When IR 00000 is ON with the following program, the content of IR 216 is divided by the content of LR 09 and the result is placed in DM 0017 and DM 0018. Example data and calculations are shown below the program.

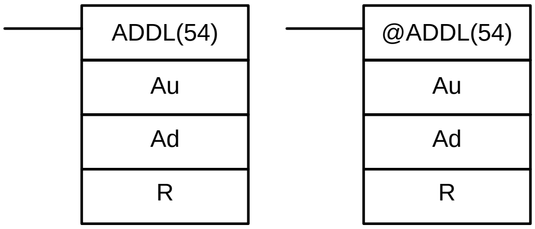


Address	Instruction	Operands	
00000	LD		00000
00001	DIV(33)		
			216
		LR	09
		DM	0017



2-12-5 DOUBLE BCD ADD – ADDL(54)

Ladder Symbols



Operand Data Areas

Au: First augend word (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR

Ad: First addend word (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR

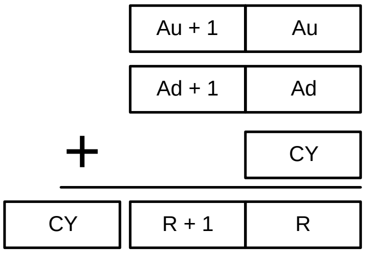
R: First result word
IR, SR, AR, DM, EM, LR

Limitations

DM 6143 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, ADDL(54) is not executed. When the execution condition is ON, ADDL(54) adds the contents of CY to the 8-digit value in Au and Au+1 to the 8-digit value in Ad and Ad+1, and places the result in R and R+1. CY will be set if the result is greater than 99999999.



Flags

ER: Au and/or Ad is not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

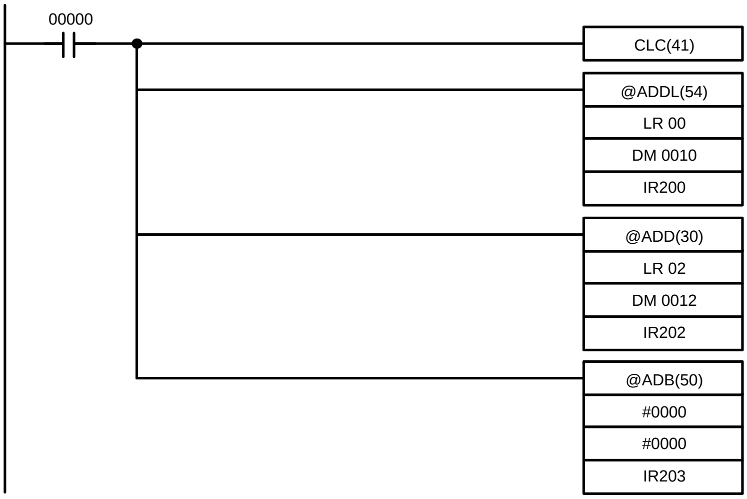
CY: ON when there is a carry in the result.

EQ: ON when the result is 0.

Example

When IR 00000 is ON, the following program section adds two 12-digit numbers, the first contained in LR 00 through LR 02 and the second in DM 0010 through DM 0012. The result is placed in IR 200 through IR 201.

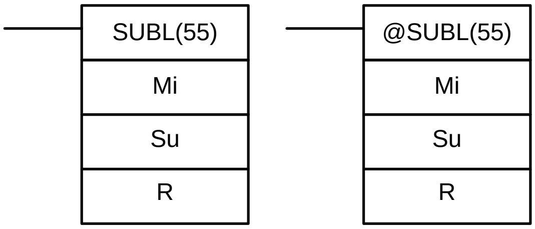
The rightmost 8 digits of the two numbers are added using ADDL(54), i.e., the contents of LR 00 and LR 01 are added to DM 0010 and DM 0011 and the results is placed in IR 200 and IR 201. The second addition adds the leftmost 4 digits of each number using ADD(30), and includes any carry from the first addition. The last instruction, ADB(50) (see *2-12-9 BINARY ADD – ADB(50)*) adds two all-zero constants to place any carry from the second addition into IR 203.



Address	Instruction	Operands	
00000	LD		00000
00001	CLC(41)		
00002	@ADDL(54)		
		LR	00
		DM	0010
			200
00003	@ADD(30)		
		LR	02
		DM	0012
			202
00004	@ADB(50)		
		#	0000
		#	0000
			203

2-12-6 DOUBLE BCD SUBTRACT – SUBL(55)

Ladder Symbols



Operand Data Areas

Mi: First minuend word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

Su: First subtrahend word (BCD)

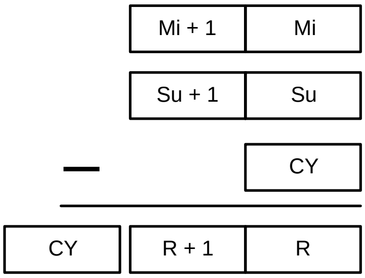
IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

IR, SR, AR, DM, EM, LR

Limitations DM 6143 to DM 6655 cannot be used for R.

Description When the execution condition is OFF, SUBL(55) is not executed. When the execution condition is ON, SUBL(55) subtracts CY and the 8-digit contents of Su and Su+1 from the 8-digit value in Mi and Mi+1, and places the result in R and R+1. If the result is negative, CY is set and the 10’s complement of the actual result is placed in R. To convert the 10’s complement to the true result, subtract the content of R from zero. Since an 8-digit constant cannot be directly entered, use the BSET(71) instruction (see *2-7-5 BLOCK SET – BSET(71)*) to create an 8-digit constant.



Flags

ER: Mi, M+1,Su, or Su+1 are not BCD.

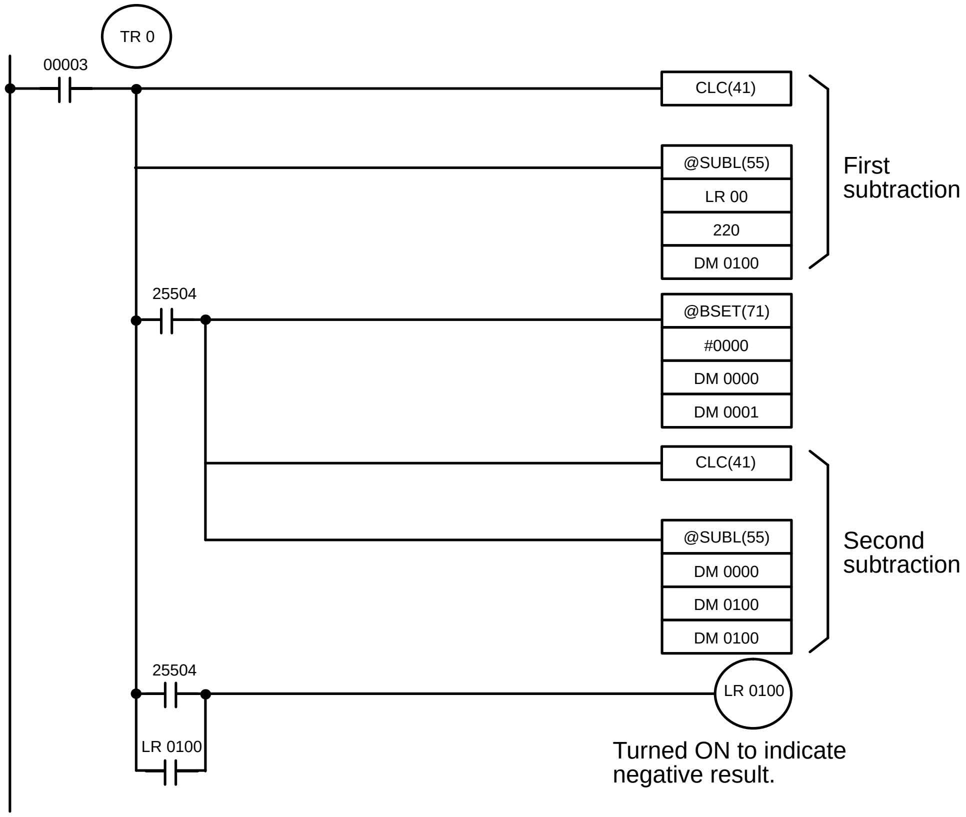
Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: ON when the result is negative, i.e., when Mi is less than Su.

EQ: ON when the result is 0.

Example The following example works much like that for single-word subtraction. In this example, however, BSET(71) is required to clear the content of DM 0000 and

DM 0001 so that a negative result can be subtracted from 0 (inputting an 8-digit constant is not possible).

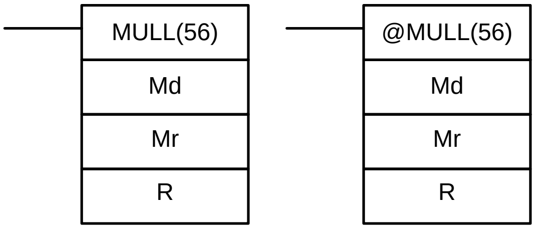


Address	Instruction	Operands
00000	LD	00003
00001	OUT	TR 0
00002	CLC(41)	
00003	@SUBL(55)	
		LR 00
		220
		DM 0100
00004	AND	25504
00005	@BSET(71)	
		# 0000
		DM 0000
		DM 0001

Address	Instruction	Operands
00006	CLC(41)	
00007	@SUBL(55)	
		DM 0000
		DM 0100
		DM 0100
00008	LD	TR 0
00009	LD	25504
00010	OR	LR 0100
00011	AND LD	
00012	OUT	LR 0100

2-12-7 DOUBLE BCD MULTIPLY – MULL(56)

Ladder Symbols



Operand Data Areas

Md: First multiplicand word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

Mr: First multiplier word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

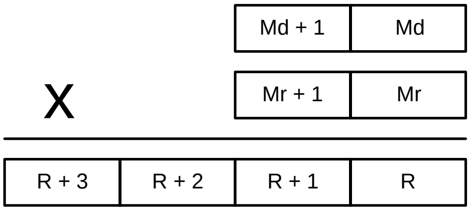
IR, SR, AR, DM, EM, LR

Limitations

DM 6141 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, MULL(56) is not executed. When the execution condition is ON, MULL(56) multiplies the eight-digit content of Md and Md+1 by the content of Mr and Mr+1, and places the result in R to R+3.



Flags

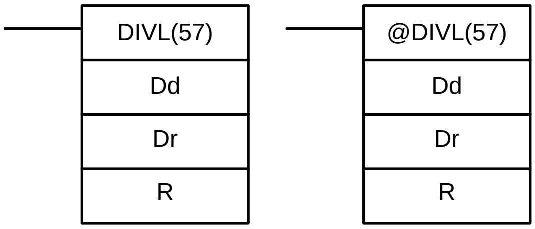
ER: Md, Md+1,Mr, or Mr+1 is not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

2-12-8 DOUBLE BCD DIVIDE – DIVL(57)

Ladder Symbols



Operand Data Areas

Dd: First dividend word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

Dr: First divisor word (BCD)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

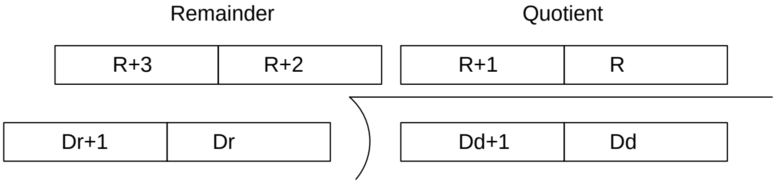
IR, SR, AR, DM, EM, LR

Limitations

DM 6141 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, DIVL(57) is not executed. When the execution condition is ON, DIVL(57) the eight-digit content of Dd and D+1 is divided by the content of Dr and Dr+1 and the result is placed in R to R+3: the quotient in R and R+1, the remainder in R+2 and R+3.



Flags

ER: Dr and Dr+1 contain 0.

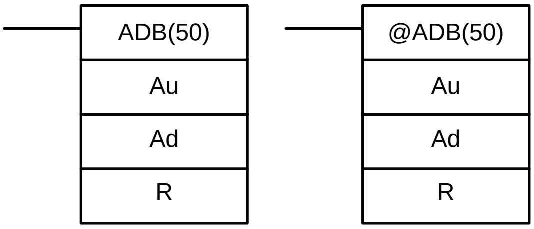
Dd, Dd+1, Dr, or Dr+1 is not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

2-12-9 BINARY ADD – ADB(50)

Ladder Symbols



Operand Data Areas

Au: Augend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Ad: Addend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word

IR, SR, AR, DM, EM, LR

Limitations DM 6144 to DM 6655 cannot be used for R.

Description When the execution condition is OFF, ADB(50) is not executed. When the execution condition is ON, ADB(50) adds the contents of Au, Ad, and CY, and places the result in R. CY will be set if the result is greater than FFFF.

$$\boxed{Au} + \boxed{Ad} + \boxed{CY} \rightarrow \boxed{CY}\ \boxed{R}$$

ADB(50) can also be used to add signed binary data. The Overflow and Underflow Flags (SR 25404 and SR 25405) indicate whether the result has exceeded the lower or upper limits of the 16-bit signed binary data range.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: ON when the result is greater than FFFF.

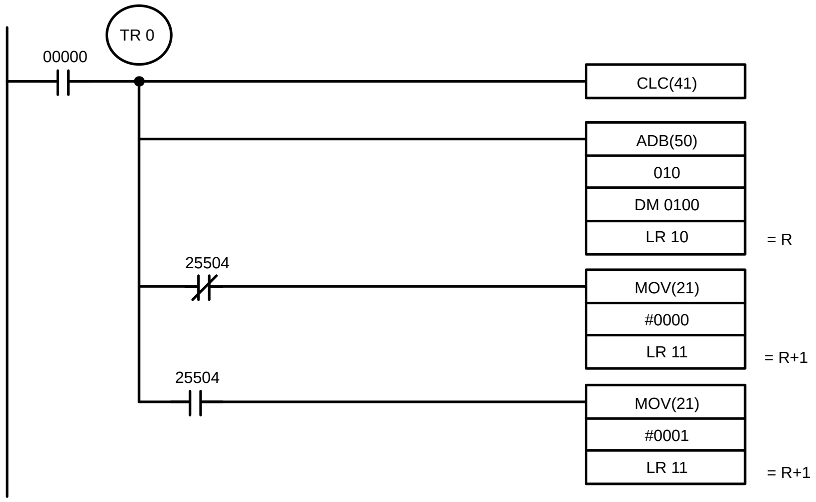
EQ: ON when the result is 0.

N: ON when the leftmost bit of the result is 1.

OF: ON when the result exceeds +32,767 (7FFF).

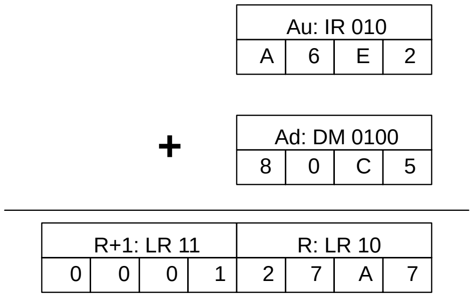
UF: ON when the result is below –32,768 (8000).

Example The following example shows a four-digit addition with CY used to place either #0000 or #0001 into R+1 to ensure that any carry is preserved.



Address	Instruction	Operands	
00000	LD		00000
00001	OUT	TR	0
00002	CLC(41)		
00003	ADB(50)		
			010
		DM	0100
		LR	10
00004	AND NOT		25504
00005	MOV(21)		
		#	0000
		LR	11
00006	LD	TR	0
00007	AND		25504
00008	MOV(21)		
		#	00001
		LR	11

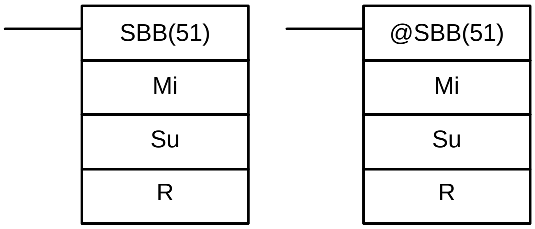
In the case below, A6E2 + 80C5 = 127A7. The result is a 5-digit number, so CY (SR 25504) = 1, and the content of R + 1 becomes #0001.



Note For signed binary calculations, the status of the UF and OF flags indicate whether the result has exceeded the signed binary data range (–32,768 (8000) to +32,767 (7FFF)).

2-12-10 BINARY SUBTRACT – SBB(51)

Ladder Symbols



Operand Data Areas

Mi: Minuend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Su: Subtrahend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word

IR, SR, AR, DM, EM, LR

Limitations DM 6144 to DM 6655 cannot be used for R.

Description When the execution condition is OFF, SBB(51) is not executed. When the execution condition is ON, SBB(51) subtracts the contents of Su and CY from Mi and places the result in R. If the result is negative, CY is set and the 2’s complement of the actual result is placed in R.

$$\boxed{\mathrm{Mi}} - \boxed{\mathrm{Su}} - \boxed{\mathrm{CY}} \rightarrow \boxed{\mathrm{CY}}\quad \boxed{\mathrm{R}}$$

SBB(51) can also be used to subtract signed binary data. The Overflow and Underflow Flags (SR 25404 and SR 25405) indicate whether the result has exceeded the lower or upper limits of the 16-bit signed binary data range.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: ON when the result is negative, i.e., when Mi is less than Su plus CY.

EQ: ON when the result is 0.

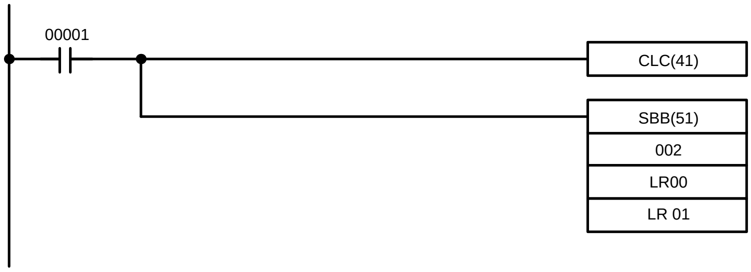
N: ON when the leftmost bit of the result is 1.

OF: ON when the result exceeds +32,767 (7FFF).

UF: ON when the result is below –32,768 (8000).

Example The following example shows a four-digit subtraction. When IR 00001 is ON, the content of LR 00 and CY are subtracted from the content of IR 002 and the result is written to LR 01.

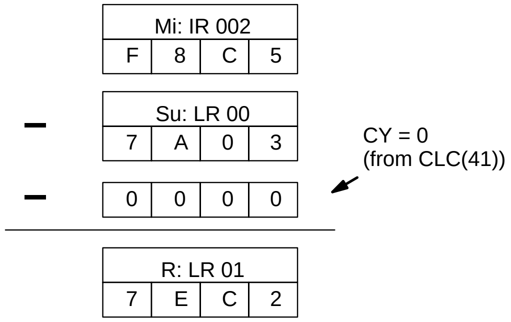
CY is turned ON if the result is negative. If normal data is being used, a negative result (signed binary) must be converted to normal data using NEG(—). Refer to *2-9-5 2's COMPLEMENT – NEG(—)* for details.



Address	Instruction	Operands
00000	LD	00001
00001	OUT	TR 1
00002	CLC(41)	
00003	SBB(51)	
		002
		LR 00
		LR 01

In the case below, the content of LR 00 (#7A03) and CY are subtracted from IR 002 (#F8C5). Since the result is positive, CY is 0.

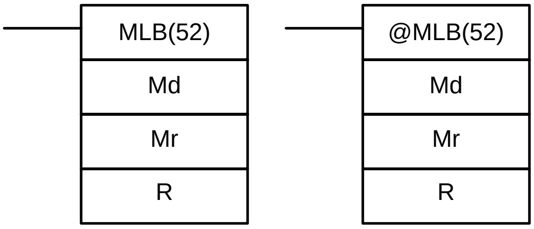
If the result had been negative, CY would have been set to 1. For normal (unsigned) data, the result would have to be converted to its 2's complement.



Note For signed binary calculations, the status of the UF and OF flags indicate whether the result has exceeded the signed binary data range (–32,768 (8000) to +32,767 (7FFF)).

2-12-11 BINARY MULTIPLY – MLB(52)

Ladder Symbols



Operand Data Areas

Md: Multiplicand word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Mr: Multiplier word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: First result word

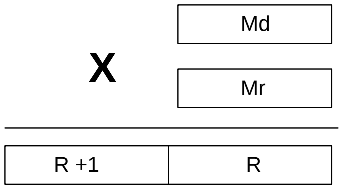
IR, SR, AR, DM, EM, LR

Limitations DM 6143 to DM 6655 cannot be used for R.

MLB(52) cannot be used to multiply signed binary data, but MBS(—) can be used. Refer to *2-12-15 SIGNED BINARY MULTIPLY – MBS(—)*.

Description

When the execution condition is OFF, MLB(52) is not executed. When the execution condition is ON, MLB(52) multiplies the content of Md by the contents of Mr, places the rightmost four digits of the result in R, and places the leftmost four digits in R+1.



Flags

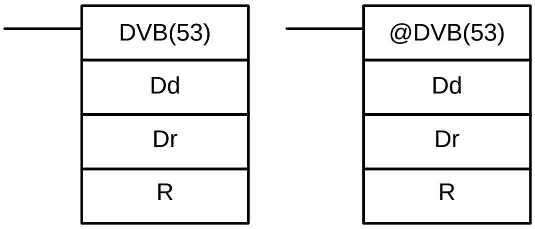
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

N: ON when the leftmost bit of the result is 1.

2-12-12 BINARY DIVIDE – DVB(53)

Ladder Symbols



Operand Data Areas

Dd: Dividend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Dr: Divisor word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: First result word

IR, SR, AR, DM, EM, LR

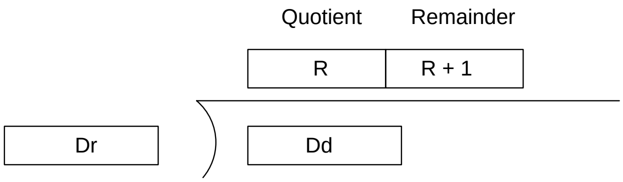
Limitations

DM 6143 to DM 6655 cannot be used for R.

DVB(53) cannot be used to divide signed binary data, but DBS(––) can be used. Refer to *2-12-17 SIGNED BINARY DIVIDE – DBS(––)* for details.

Description

When the execution condition is OFF, DVB(53) is not executed. When the execution condition is ON, DVB(53) divides the content of Dd by the content of Dr and the result is placed in R and R+1: the quotient in R, the remainder in R+1.



Flags

ER: Dr contains 0.

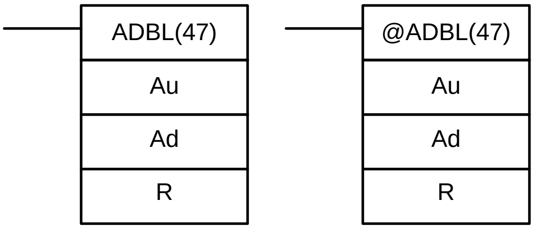
Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

N: ON when the leftmost bit of the result is 1.

2-12-13 DOUBLE BINARY ADD – ADBL(47)

Ladder Symbols



Operand Data Areas

Au: First augend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR

Ad: First addend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

IR, SR, AR, DM, EM, LR

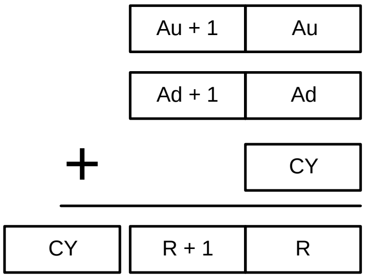
Limitations

Au and Au+1 must be in the same data area, as must Ad and Ad+1, and R and R+1.

DM 6142 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, ADBL(47) is not executed. When the execution condition is ON, ADBL(47) adds the eight-digit contents of Au+1 and Au, the eight-digit contents of Ad+1 and Ad, and CY, and places the result in R. CY will be set if the result is greater than FFFF FFFF.



ADBL(47) can also be used to add signed binary data. The Overflow and Underflow Flags (SR 25404 and SR 25405) indicate whether the result has exceeded the lower or upper limits of the 32-bit signed binary data range.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: ON when the result is greater than FFFF FFFF.

EQ: ON when the result is 0.

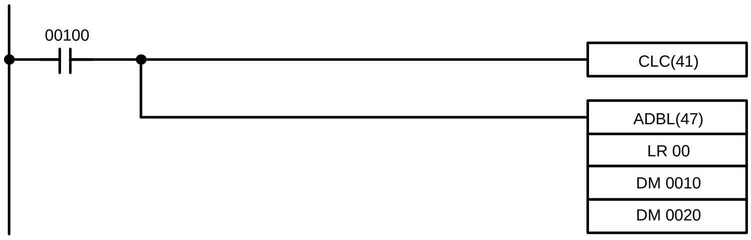
N: ON when the leftmost bit of the result is 1.

OF: ON when the result exceeds +2,147,483,647 (7FFF FFFF).

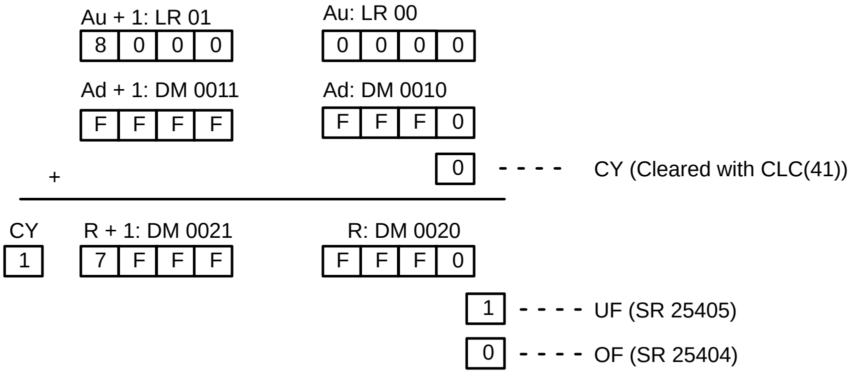
UF: ON when the result is below –2,147,483,648 (8000 0000).

Example

The following example shows an eight-digit addition with CY (SR 25504) used to represent the status of the 9th digit. The status of the UF and OF flags indicate whether the result has exceeded the signed binary data range (–2,147,483,648 (8000 0000) to +2,147,483,647 (7FFF FFFF)).



Address	Instruction	Operands	
00000	LD		00100
00001	CLC(41)		
00002	ADBL(47)		
		LR	20
		DM	0010
		DM	0020

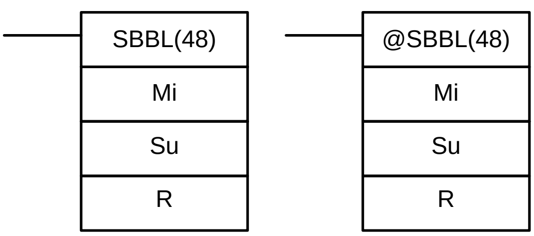


Note

1. For unsigned binary addition, CY indicates that the sum of the two values exceeds FFFF FFFF. (UF and OF can be ignored.)
2. For signed binary addition, the UF flag indicates that the sum of the two values is below –2,147,483,648 (8000 0000). (CY can be ignored.)

2-12-14 DOUBLE BINARY SUBTRACT – SBBL(48)

Ladder Symbols



Operand Data Areas

Mi: First minuend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR

Su: First subtrahend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

IR, SR, AR, DM, EM, LR

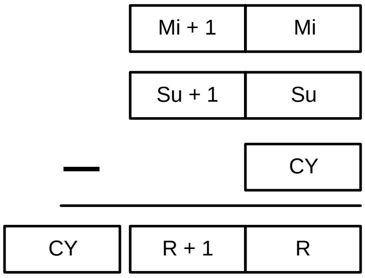
Limitations

Mi and Mi+1 must be in the same data area, as must Su and Su+1, and R and R+1.

DM 6142 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, SBBL(48) is not executed. When the execution condition is ON, SBBL(48) subtracts CY and the eight-digit value in Su and Su+1 from the eight-digit value in Mi and Mi+1, and places the result in R and R+1. If the result is negative, CY is set and the 2's complement of the actual result is placed in R+1 and R. Use NEGL(—) to convert the 2's complement to the true result.



SBBL(48) can also be used to subtract signed binary data. The Overflow and Underflow Flags (SR 25404 and SR 25405) indicate whether the result has exceeded the lower or upper limits of the 32-bit signed binary data range.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CY: ON when the result is negative, i.e., when Mi is less than Su plus CY.

EQ: ON when the result is 0.

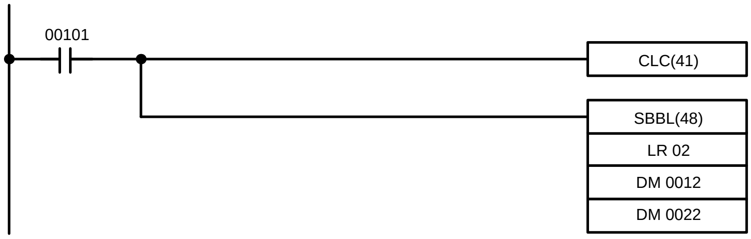
N: ON when the leftmost bit of the result is 1.

OF: ON when the result exceeds +2,147,483,647 (7FFF FFFF).

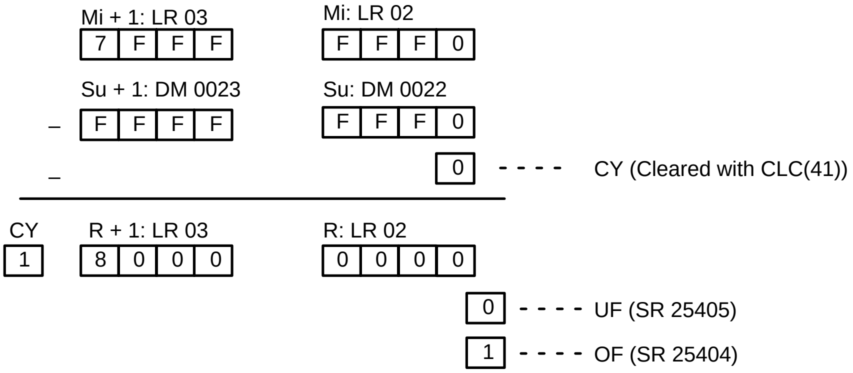
UF: ON when the result is below –2,147,483,648 (8000 0000).

Example

The following example shows an eight-digit subtraction with CY (SR 25504) used to indicate a negative result (with unsigned data). The status of the UF and OF flags indicate whether the result has exceeded the signed binary data range (–2,147,483,648 (8000 0000) to +2,147,483,647 (7FFF FFFF)).



Address	Instruction	Operands
00000	LD	00101
00001	CLC(41)	
00002	SBBL(48)	
		LR 02
		DM 0012
		DM 0022

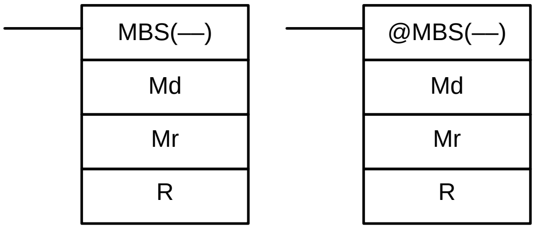


Note

1. For unsigned binary data, CY indicates that the result is negative. Take the 2’s complement using NEGL(—) to obtain the absolute value of the true result. (UF and OF can be ignored.)
2. For signed binary data, the OF flag indicates that the result exceeds +2,147,483,647 (7FFF FFFF). (CY can be ignored.)

2-12-15 SIGNED BINARY MULTIPLY – MBS(––)

Ladder Symbols



Operand Data Areas

Md: Multiplicand word

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Mr: Multiplier word

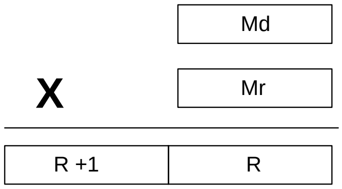
IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: First result word

IR, SR, AR, DM, EM, LR

Limitations DM 6143 to DM 6655 cannot be used for R.

Description MBS(—) multiplies the signed binary content of two words and outputs the 8-digit signed binary result to R+1 and R. The rightmost four digits of the result are placed in R, and the leftmost four digits are placed in R+1.



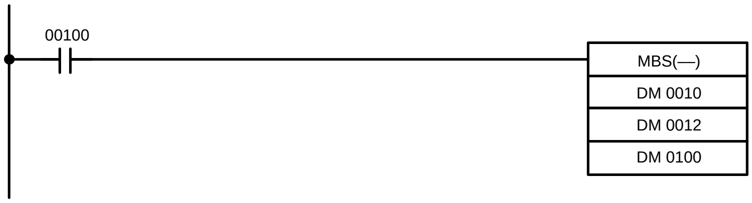
Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

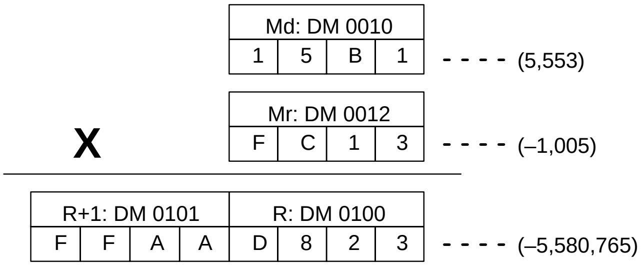
EQ: ON when the result is 0000 0000, otherwise OFF.

N: ON when the leftmost bit of the result is 1.

Example In the following example, MBS(—) is used to multiply the signed binary contents of DM 0010 with the signed binary contents of DM 0012 and output the result to DM 0100 and DM 0101.

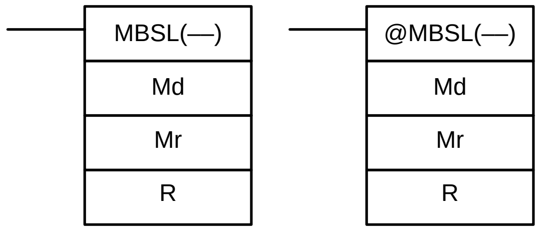


Address	Instruction	Operands
00000	LD	00100
00001	MBS(—)	
		DM 0010
		DM 0012
		DM 0100



2-12-16 DOUBLE SIGNED BINARY MULTIPLY – MBSL(––)

Ladder Symbols



Operand Data Areas

Md: First multiplicand word

IR, SR, AR, DM, EM, TIM/CNT, LR

Mr: First multiplier word

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

IR, SR, AR, DM, EM, LR

Limitations

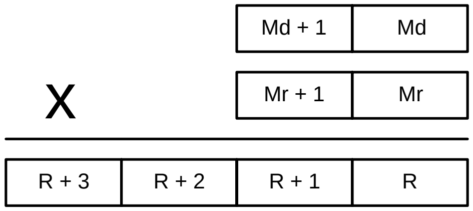
Md and Md+1 must be in the same data area, as must Mr and Mr+1.

R and R+3 must be in the same data area.

DM 6143 to DM 6655 cannot be used for R.

Description

MBSL(—) multiplies the 32-bit (8-digit) signed binary data in Md+1 and Md with the 32-bit signed binary data in Mr+1 and Mr, and outputs the 16-digit signed binary result to R+3 through R.



Flags

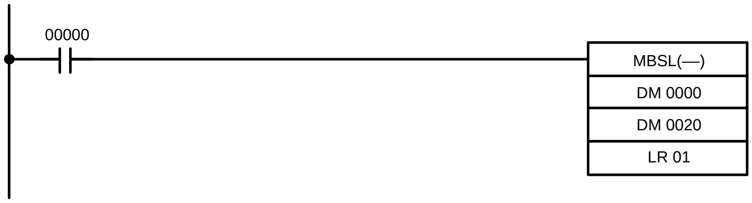
ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is zero (content of R+3 through R all zeroes), otherwise OFF.

N: ON when the leftmost bit of the result is 1.

Example

In the following example, MBSL(—) is used to multiply the signed binary contents of DM 0001 and DM 0000 with the signed binary contents of DM 0021 and DM 0020 and output the result to LR 24 through LR 01.



Address	Instruction	Operands	
00000	LD		00000
00001	MBSL(—)		
		DM	0000
		DM	0020
		LR	01

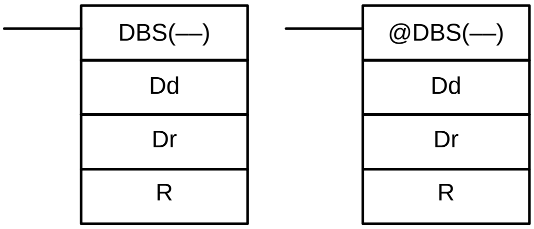
	Md+1: DM 0001				Md: DM 0000				
	0	0	0	8	7	9	3	8	- - - - (555,320)

	Mr+1: DM 0021				Mr: DM 0020				
X	F	F	F	0	A	8	1	2	- - - - (–1,005,550)

R+3: LR 04				R+2: LR 03				R+1: LR 02				R: LR 01				
F	F	F	F	F	F	7	D	F	C	A	5	4	5	F	0	- - - - (–55,840,206,000)

2-12-17 SIGNED BINARY DIVIDE – DBS(––)

Ladder Symbols



Operand Data Areas

Dd: Dividend word
IR, SR, AR, DM, EM, TIM/CNT, LR, #

Dr: Divisor word
IR, SR, AR, DM, EM, TIM/CNT, LR, #

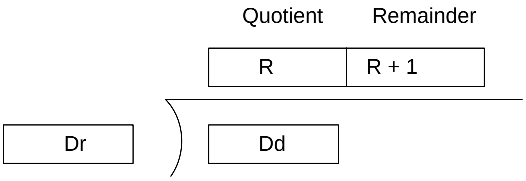
R: First result word
IR, SR, AR, DM, EM, LR

Limitations

DM 6143 to DM 6655 cannot be used for R.

Description

DBS(—) divides the signed binary content of Dd by the signed binary content of Dr, and outputs the 8-digit signed binary result to R+1 and R. The quotient is placed in R, and the remainder is placed in R+1.



Flags

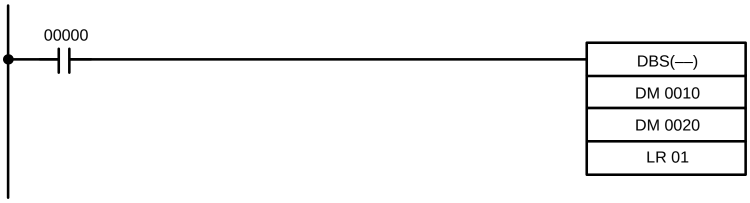
ER: Dr contains 0. Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the content of R (the quotient) is 0000, otherwise OFF.

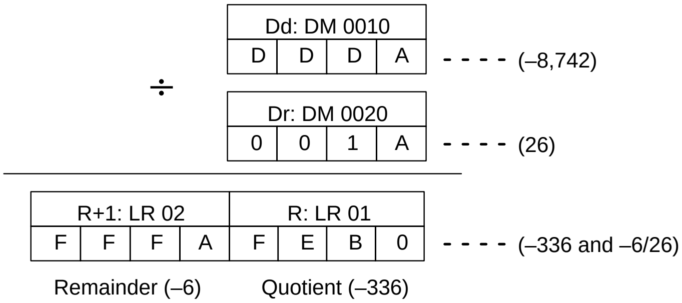
N: ON when the leftmost bit of the result is 1.

Example

In the following example, DBS(—) is used to divide the signed binary contents of DM 0010 with the signed binary contents of DM 0020 and output the result to LR 21 and LR 02.

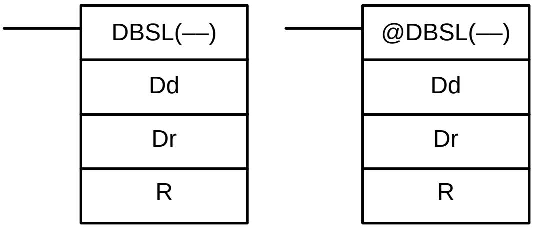


Address	Instruction	Operands	
00000	LD		00000
00001	DBS(—)		
		DM	0010
		DM	0020
		LR	01



2-12-18 DOUBLE SIGNED BINARY DIVIDE – DBSL(––)

Ladder Symbols



Operand Data Areas

Dd: First dividend word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR

Dr: First divisor word (binary)

IR, SR, AR, DM, EM, TIM/CNT, LR

R: First result word

IR, SR, AR, DM, EM, LR

Limitations

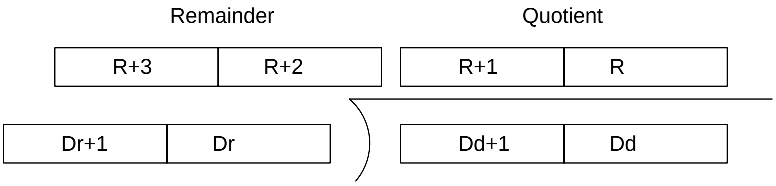
Dd and Dd+1 must be in the same data area, as must Dr and Dr+1.

R and R+3 must be in the same data area.

DM 6143 to DM 6655 cannot be used for R.

Description

DBS(—) divides the 32-bit (8-digit) signed binary data in Dd+1 and Dd by the 32-bit signed binary data in Dr+1 and Dr, and outputs the 16-digit signed binary result to R+3 through R. The quotient is placed in R+1 and R, and the remainder is placed in R+3 and R+2.



Flags

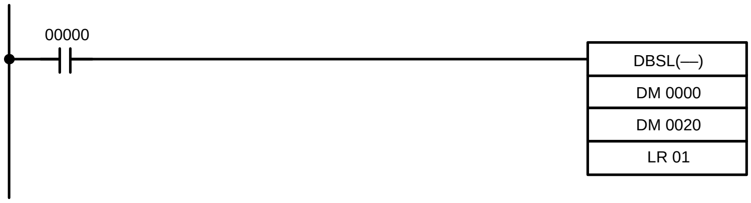
ER: Dr+1 and Dr contain 0.
Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the content of R+1 and R (the quotient) is 0, otherwise OFF.

N: ON when the leftmost bit of the result is 1.

Example

In the following example, DBSL(—) is used to divide the signed binary contents of DM 0001 and DM 0000 with the signed binary contents of DM 0021 and DM 0020 and output the result to LR 24 through LR 01.



Address	Instruction	Operands	
00000	LD		00000
00001	DBSL(—)		
		DM	0000
		DM	0020
		LR	01

Dd+1: DM 0001				Dd: DM 0000				
F	F	7	A	B	1	5	C	- - - - (–8,736,420)

÷

Dr+1: DM 0021				Dr: DM 0020				
0	0	0	0	0	0	1	A	- - - - (26)

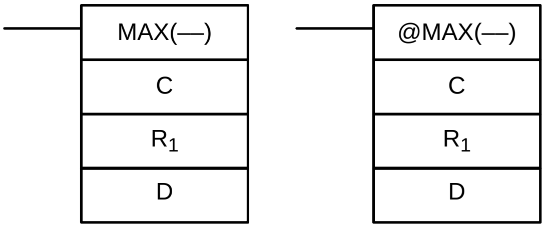
R+3: LR 04				R+2: LR 03				R+1: LR 02				R: LR 01				
F	F	F	F	F	F	F	C	F	F	F	A	D	F	7	0	- - - - (–336,016 and –4/26)

Remainder (–4) Quotient (–336)

2-13 Table Data Processing Instructions

2-13-1 FIND MAXIMUM – MAX(––)

Ladder Symbols



Operand Data Areas

C: Control data
IR, SR, AR, DM, EM, TIM/CNT, LR, #

R_1: First word in range
IR, SR, AR, DM, EM, TIM/CNT, LR

D: Destination word
IR, SR, AR, DM, EM, LR

Limitations

N must be BCD between 0001 to 9999.

R_1 and R_1+N–1 must be in the same data area.

DM 6144 to DM 6655 cannot be used for D.

Description

When the execution condition is OFF, MAX(—) is not executed. When the execution condition is ON, MAX(—) searches the range of memory from R_1 to R_1+N–1 for the address that contains the maximum value and outputs the maximum value to the destination word (D).

If bit 15 of C is ON, MAX(—) identifies the address of the word containing the maximum value in D+1. The address is identified differently for the DM area:

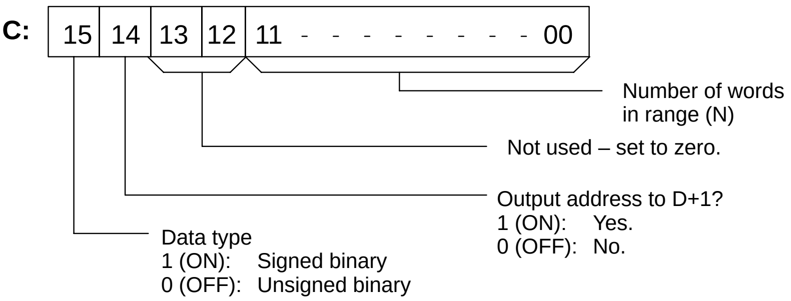
1, 2, 3...

1. For an address in the DM area, the word address is written to C+1. For example, if the address containing the maximum value is DM 0114, then #0114 is written in D+1.
2. For an address in another data area, the number of addresses from the beginning of the search is written to D+1. For example, if the address containing the maximum value is IR 214 and the first word in the search range is IR 014, then #0200 is written in D+1.

If bit 14 of C is ON and more than one address contains the same maximum value, the position of the lowest of the addresses will be output to D+1. The position will be output as the DM address for the DM area, but as an absolute position relative to the first word in the range for all other areas.

The number of words within the range (N) is contained in the 3 rightmost digits of C, which must be BCD between 001 and 999.

When bit 15 of C is OFF, data within the range is treated as unsigned binary and when it is ON the data is treated as signed binary.



⚠ **Caution** If bit 14 of C is ON, values above #8000 are treated as negative numbers, so the results will differ depending on the specified data type. Be sure that the correct data type is specified.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

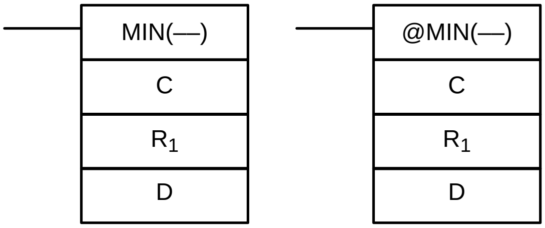
R_1 and R_1+N–1 are not in the same data area.

EQ: ON when the maximum value is #0000.

N: ON when the leftmost bit of the result is 1.

2-13-2 FIND MINIMUM – MIN(––)

Ladder Symbols



Operand Data Areas

C: Control data
IR, SR, AR, DM, EM, TIM/CNT, LR, #

R_1: First word in range
IR, SR, AR, DM, EM, TIM/CNT, LR

D: Destination word
IR, SR, AR, DM, EM, LR

Limitations

N must be BCD between 0001 to 9999.

R_1 and R_1+N–1 must be in the same data area.

DM 6144 to DM 6655 cannot be used for D.

Description

When the execution condition is OFF, MIN(—) is not executed. When the execution condition is ON, MIN(—) searches the range of memory from R_1 to R_1+N–1 for the address that contains the minimum value and outputs the minimum value to the destination word (D).

If bit 15 of C is ON, MIN(—) identifies the address of the word containing the minimum value in D+1. The address is identified differently for the DM area:

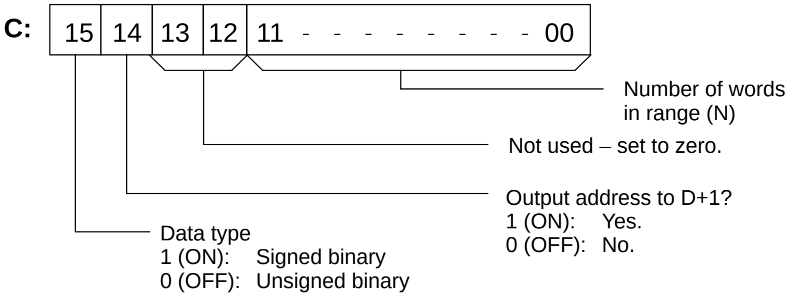
1, 2, 3...

1. For an address in the DM area, the word address is written to C+1. For example, if the address containing the minimum value is DM 0114, then #0114 is written in D+1.
2. For an address in another data area, the number of addresses from the beginning of the search is written to D+1. For example, if the address containing the minimum value is IR 214 and the first word in the search range is IR 014, then #0200 is written in D+1.

If bit 14 of C is ON and more than one address contains the same minimum value, the position of the lowest of the addresses will be output to D+1. The position will be output as the DM address for the DM area, but as an absolute position relative to the first word in the range for all other areas.

The number of words within the range (N) is contained in the 3 rightmost digits of C, which must be BCD between 001 and 999.

When bit 15 of C is OFF, data within the range is treated as unsigned binary and when it is ON the data is treated as signed binary.



⚠ **Caution** If bit 14 of C is ON, values above #8000 are treated as negative numbers, so the results will differ depending on the specified data type. Be sure that the correct data type is specified.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

R_1 and R_1+N–1 are not in the same data area.

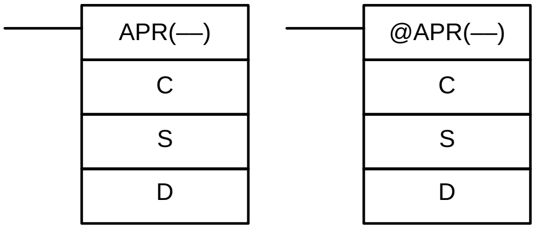
EQ: ON when the minimum value is #0000.

N: ON when the leftmost bit of the result is 1.

2-14 Special Math Instructions

2-14-1 ARITHMETIC PROCESS – APR(––)

Ladder Symbols



Operand Data Areas

C: Control word and linear data table
IR, SR, AR, DM, EM, TIM/CNT, LR

S: Input data source word
IR, SR, AR, DM, EM, TIM/CNT, LR

D: Starting result destination word
IR, SR, AR, DM, EM, IM/CNT, LR

Limitations APR(—) is supported by the CS1W-HCP22 and CS1W-HCA22 only.

DM 6144 to DM 6655 cannot be used for D.

Description When the execution condition is OFF, APR(—) is not executed. When the execution condition is ON, APR(—) computes f(x) of the linear function entered as a point table beginning at word C and outputs the result to D (rightmost digits) and D+1 (leftmost digits). The function is a series of line segments (which can approximate a curve). The input data (a 4-digit hexadecimal value), x, is specified by S or specified as the present value of a high-speed counter. The specification is made in the control word, C, described below. The linear data table consists of points (X, Y), where each X is a 4-digit hexadecimal value and each Y is an 8-digit hexadecimal value. The structure of the linear data table is given below.

Assuming that the input data, x, is between first X_n and X_{n+1}, the result output to D and D+1 is calculated using the following formula:

$$\text{Result} = Y_n + [(Y_{n+1} - Y_n)/(X_{n+1} - X_n) \text{ x } \{(\text{Input data}) - X_n\}]$$

Word C+1 is the first word of the continuous block of memory containing the linear data table. The content of word C specifies the number of line segments in the approximation, and the source of the input data. Bits 00 to 07 contain the number of line segments less 1, m–1, as a hexadecimal value (256 points maximum). Bits 08 to 11 specify the source of the input data, i.e., as the contents of a word in memory or as the present value of one of the high-speed counters.

C:	Always 1Hex	11	10	09	08	07	06	05	04	03	02	01	00

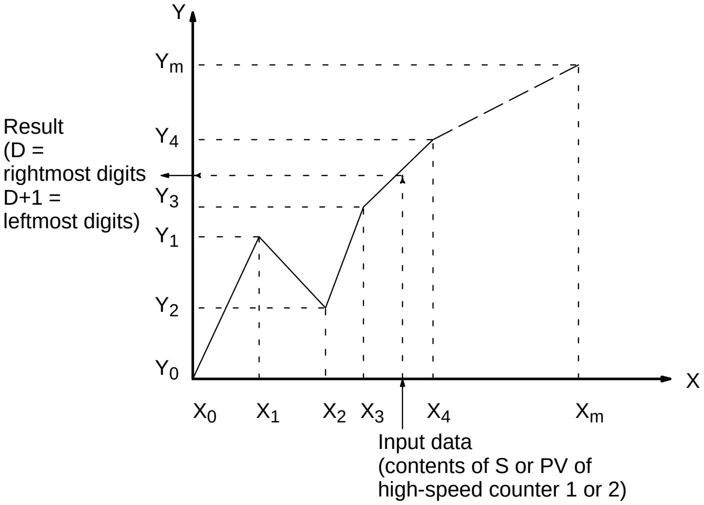
Input data designation

0 Hex:	Use data from S
1 Hex:	Use high-speed counter 1 PV
2 Hex:	Use high-speed counter 2 PV

(S is ignored if 1 or 2 Hex is set.)

Number of coordinates minus one (m–1) (256 points max.)

The coordinates of the m+1 points, which define m line segments, are entered in a linear data table beginning from C+1 as shown below. Enter all coordinates in hexadecimal form. X_0 is always 0000, and does not have to be entered.



Word	Coordinate
C+1	X_m (max. X value)
C+2	Rightmost digits of Y_0
C+3	Leftmost digits of Y_0
C+4	X_1
C+5	Rightmost digits of Y_1
C+6	Leftmost digits of Y_1
↓	↓
C+(3m+1)	X_m
C+(3m+2)	Rightmost digits of Y_m
C+(3m+3)	Leftmost digits of Y_m

Note When inputting the PV of a high-speed counter, the rightmost 16 bits of the most recent counter PV is used as the input data.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

C bits 08 to 15 contain a value other than 10, 11, or 12.

The linear approximation data is incorrect.

The input data is not in the linear data table.

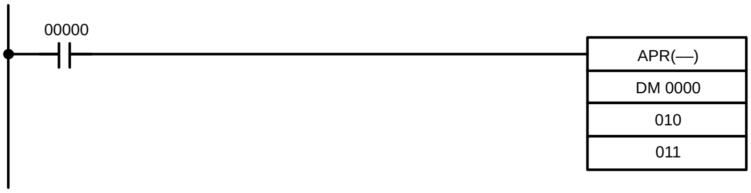
EQ: The result is 0000 0000.

N: ON when the leftmost bit of the result is 1.

Example

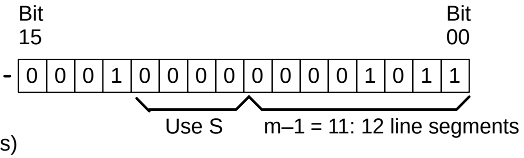
The following example demonstrates the construction of a linear approximation with 12 line segments. The block of data is continuous, as it must be, from

DM 0000 to DM 0039 (C to C + (3 × 12 + 3)). The input data is taken from IR 010, and the result is output to IR 011 and IR 012.

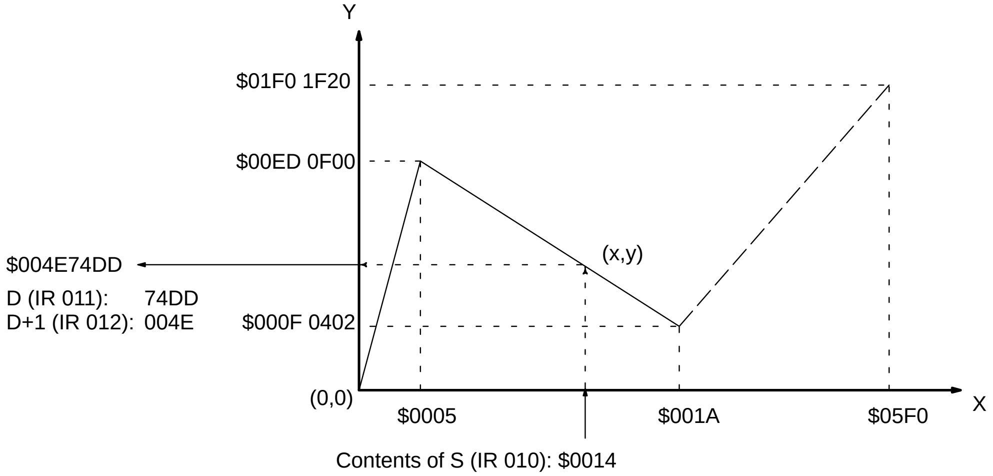


Address	Instruction	Operands
00000	LD	00000
00001	APR(—)	
		DM 0000
		010
		011

	Content	Coordinate
DM 0000	$100B	
DM 0001	$05F0	X_{12}
DM 0002	$0000	Y_0 (Rightmost 4 digits)
DM 0003	$0000	Y_0 (Leftmost 4 digits)
DM 0004	$0005	X_1
DM 0005	$0F00	Y_1 (Rightmost 4 digits)
DM 0006	$00DE	Y_1 (Leftmost 4 digits)
DM 0007	$001A	X_2
DM 0008	$0402	Y_2 (Rightmost 4 digits)
DM 0009	$000F	Y_2 (Leftmost 4 digits)
↓	↓	↓
DM 0037	$05F0	X_{12}
DM 0038	$1F20	Y_{12} (Rightmost 4 digits)
DM 0039	$01F0	Y_{12} (Leftmost 4 digits)

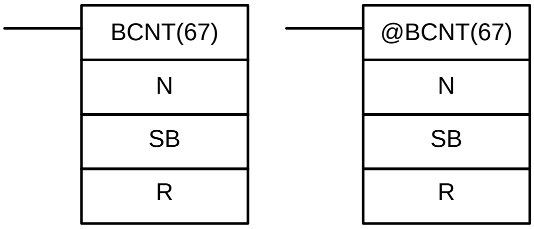


In this case, the input data word, IR 010, contains #0014, and f(0014) = #004E74DD is output to D and D+1, IR 011 and IR 012.



2-14-2 BIT COUNTER – BCNT(67)

Ladder Symbols



Operand Data Areas

N: Number of words (BCD)
IR, SR, AR, DM, EM, TIM/CNT, LR, #

SB: Source beginning word
IR, SR, AR, DM, EM, TIM/CNT, LR

R: Destination word
IR, SR, AR, DM, EM, TIM/CNT, LR

Limitations

N cannot be 0.

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, BCNT(67) is not executed. When the execution condition is ON, BCNT(67) counts the total number of bits that are ON in all words between SB and SB+(N–1) and places the result in R.

Flags

ER: N is not BCD, or N is 0; SB and SB+(N–1) are not in the same area.

The resulting count value exceeds 9999.

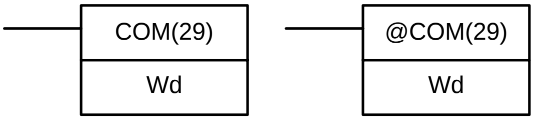
Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

2-15 Logic Instructions

2-15-1 COMPLEMENT – COM(29)

Ladder Symbols



Operand Data Areas

Wd: Complement word
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for Wd.

Description

When the execution condition is OFF, COM(29) is not executed. When the execution condition is ON, COM(29) clears all ON bits and sets all OFF bits in Wd.

Precautions

The complement of Wd will be calculated every cycle if the undifferentiated form of COM(29) is used. Use the differentiated form (@COM(29)) or combine COM(29) with DIFU(13) or DIFD(14) to calculate the complement just once.

Example

	15															00
Original	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1

⬇

	15															00
Complement	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

Flags

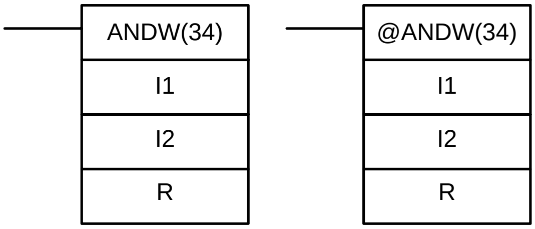
ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

N: ON when the leftmost bit of R is 1.

2-15-2 LOGICAL AND – ANDW(34)

Ladder Symbols



Operand Data Areas

I1: Input 1

IR, SR, AR, DM, EM, TIM/CNT, LR, #

I2: Input 2

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word

IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, ANDW(34) is not executed. When the execution condition is ON, ANDW(34) logically AND's the contents of I1 and I2 bit-by-bit and places the result in R.

Example

	15															00
I1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1

	15															00
I2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

↓

	15															00
R	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1

Flags

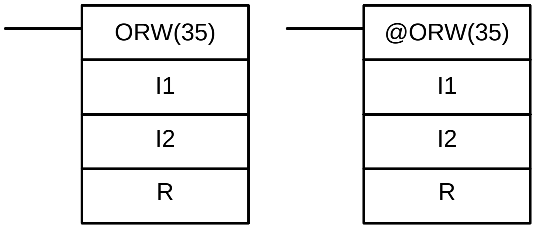
ER: Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

N: ON when the leftmost bit of R is 1.

2-15-3 LOGICAL OR – ORW(35)

Ladder Symbols



Operand Data Areas

I1: Input 1

IR, SR, AR, DM, EM, TIM/CNT, LR, #

I2: Input 2

IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word

IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, ORW(35) is not executed. When the execution condition is ON, ORW(35) logically OR's the contents of I1 and I2 bit-by-bit and places the result in R.

Example

	15															00
I1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1

	15															00
I2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

⬇

	15															00
R	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1

Flags

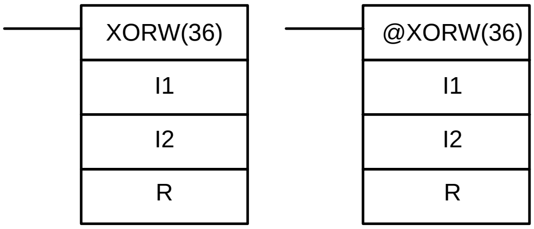
ER: Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

N: ON when the leftmost bit of R is 1.

2-15-4 EXCLUSIVE OR – XORW(36)

Ladder Symbols



Operand Data Areas

I1: Input 1
IR, SR, AR, DM, EM, TIM/CNT, LR, #

I2: Input 2
IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for R.

Description

When the execution condition is OFF, XORW(36) is not executed. When the execution condition is ON, XORW(36) exclusively OR's the contents of I1 and I2 bit-by-bit and places the result in R.

Example

	15															00
I1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1

	15															00
I2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

⬇

	15															00
R	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0

Flags

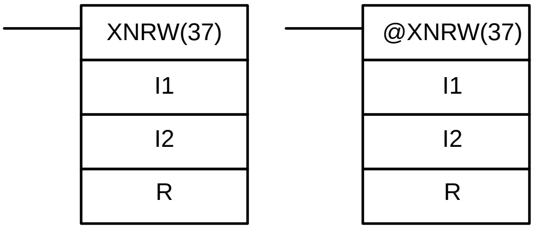
ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the result is 0.

N: ON when the leftmost bit of R is 1.

2-15-5 EXCLUSIVE NOR – XNRW(37)

Ladder Symbols



Operand Data Areas

I1: Input 1
IR, SR, AR, DM, EM, TIM/CNT, LR, #

I2: Input 2
IR, SR, AR, DM, EM, TIM/CNT, LR, #

R: Result word
IR, SR, AR, DM, EM, LR

Limitations DM 6144 to DM 6655 cannot be used for R.

Description When the execution condition is OFF, XNRW(37) is not executed. When the execution condition is ON, XNRW(37) exclusively NOR's the contents of I1 and I2 bit-by-bit and places the result in R.

	15															00
I1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1

	15															00
I2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

⬇

	15															00
R	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

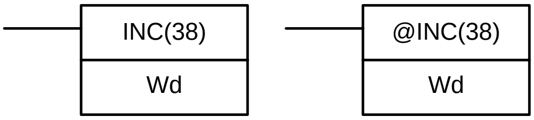
EQ: ON when the result is 0.

N: ON when the leftmost bit of R is 1.

2-16 Increment/Decrement Instructions

2-16-1 BCD INCREMENT – INC(38)

Ladder Symbols



Operand Data Areas

Wd: Increment word (BCD)
IR, SR, AR, DM, EM, LR

Limitations DM 6144 to DM 6655 cannot be used for Wd.

Description

When the execution condition is OFF, INC(38) is not executed. When the execution condition is ON, INC(38) increments Wd, without affecting Carry (CY).

Precautions

The content of Wd will be incremented every cycle if the undifferentiated form of INC(38) is used. Use the differentiated form (@INC(38)) or combine INC(38) with DIFU(13) or DIFD(14) to increment Wd just once.

Flags

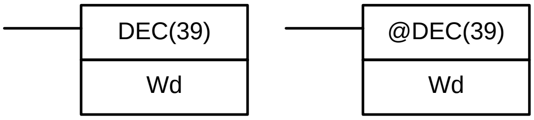
ER: Wd is not BCD

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

EQ: ON when the incremented result is 0.

2-16-2 BCD DECREMENT – DEC(39)

Ladder Symbols



Operand Data Areas

Wd: Decrement word (BCD)
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for Wd.

Description

When the execution condition is OFF, DEC(39) is not executed. When the execution condition is ON, DEC(39) decrements Wd, without affecting CY. DEC(39) works the same way as INC(38) except that it decrements the value instead of incrementing it.

Precautions

The content of Wd will be decremented every cycle if the undifferentiated form of DEC(39) is used. Use the differentiated form (@DEC(39)) or combine DEC(39) with DIFU(13) or DIFD(14) to decrement Wd just once.

Flags

ER: Wd is not BCD.

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

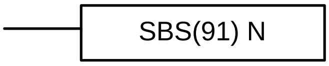
EQ: ON when the decremented result is 0.

2-17 Subroutine Instructions

Subroutines break large control tasks into smaller ones and enable you to reuse a given set of instructions. When the main program calls a subroutine, control is transferred to the subroutine and the subroutine instructions are executed. The instructions within a subroutine are written in the same way as main program code. When all the subroutine instructions have been executed, control returns to the main program to the point just after the point from which the subroutine was entered (unless otherwise specified in the subroutine).

2-17-1 SUBROUTINE ENTER – SBS(91)

Ladder Symbol



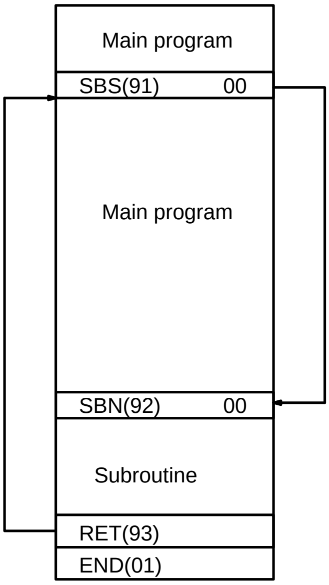
Definer Data Areas

N: Subroutine number
000 to 049 or 100 to 199

⚠ **Caution** Subroutine numbers 100 to 199 are used for dummy subroutines to start external interrupt tasks in the CPU Unit when executing MCRO (99). Do not use subroutine numbers 100 to 199 for normal subroutines.

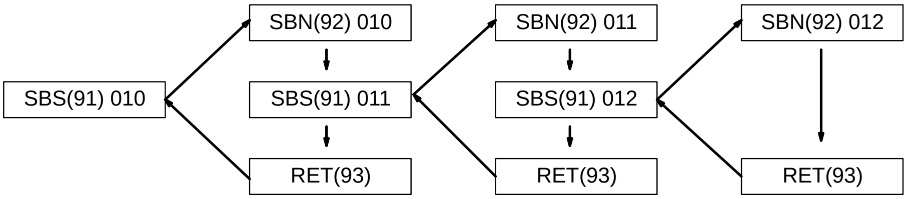
Description

A subroutine can be executed by placing SBS(91) in the main program at the point where the subroutine is desired. The subroutine number used in SBS(91) indicates the desired subroutine. When SBS(91) is executed (i.e., when the execution condition for it is ON), the instructions between the SBN(92) with the same subroutine number and the first RET(93) after it are executed before execution returns to the instruction following the SBS(91) that made the call.

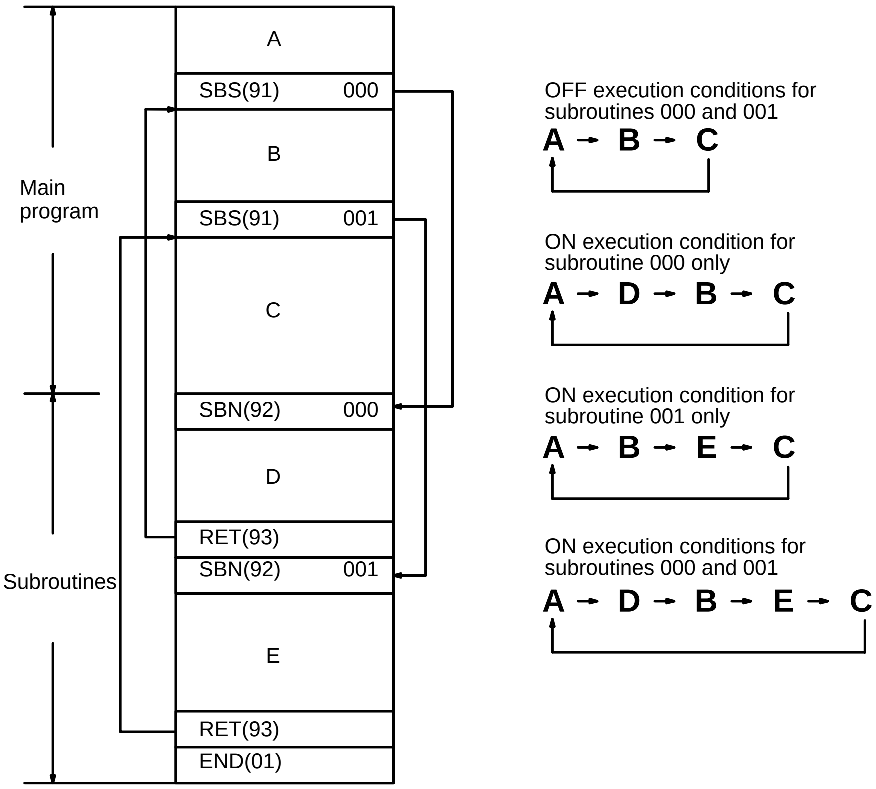


SBS(91) may be used as many times as desired in the program, i.e., the same subroutine may be called from different places in the program).

SBS(91) may also be placed into a subroutine to shift program execution from one subroutine to another, i.e., subroutines may be nested. When the second subroutine has been completed (i.e., RET(93) has been reached), program execution returns to the original subroutine which is then completed before returning to the main program. Nesting is possible to up to sixteen levels. A subroutine cannot call itself (e.g., SBS(91) 000 cannot be programmed within the subroutine defined with SBN(92) 000). The following diagram illustrates two levels of nesting.



The following diagram illustrates program execution flow for various execution conditions for two SBS(91).


A
SBS(91) 000
B
SBS(91) 001
C
SBN(92) 000
D
RET(93)
SBN(92) 001
E
RET(93)
END(01)
Main program
Subroutines
OFF execution conditions for subroutines 000 and 001
A → B → C
ON execution condition for subroutine 000 only
A → D → B → C
ON execution condition for subroutine 001 only
A → B → E → C
ON execution conditions for subroutines 000 and 001
A → D → B → E → C

Flags **ER:** A subroutine does not exist for the specified subroutine number.

A subroutine has called itself.

An active subroutine has been called.

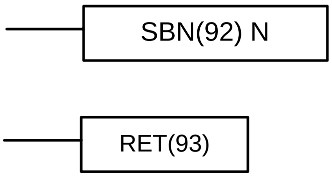
An illegal subroutine number has been used (i.e., 050 to 099 or 200 or higher).

Note Subroutine numbers 100 to 199 are used for dummy subroutines to start external interrupt tasks in the CPU Unit when executing MCRO (99). Do not use subroutine numbers 100 to 199 for normal subroutines.

⚠ **Caution** SBS(91) will not be executed and the subroutine will not be called when ER is ON.

2-17-2 SUBROUTINE DEFINE and RETURN – SBN(92)/RET(93)

Ladder Symbols



Definer Data Areas

N: Subroutine number
000 to 049 or 100 to 199

Limitations Each subroutine number can be used in SBN(92) only once.

Description SBN(92) is used to mark the beginning of a subroutine program; RET(93) is used to mark the end. Each subroutine is identified with a subroutine number, N, that is programmed as a definer for SBN(92). This same subroutine number is used in any SBS(91) that calls the subroutine (see 2-17-1 SUBROUTINE ENTER – SBS(91)). No subroutine number is required with RET(93).

All subroutines must be programmed at the end of the main program. When one or more subroutines have been programmed, the main program will be executed up to the first SBN(92) before returning to address 00000 for the next cycle. Subroutines will not be executed unless called by SBS(91).

END(01) must be placed at the end of the last subroutine program, i.e., after the last RET(93). It is not required at any other point in the program.

Precautions

If SBN(92) is mistakenly placed in the main program, it will inhibit program execution past that point, i.e., program execution will return to the beginning when SBN(92) is encountered.

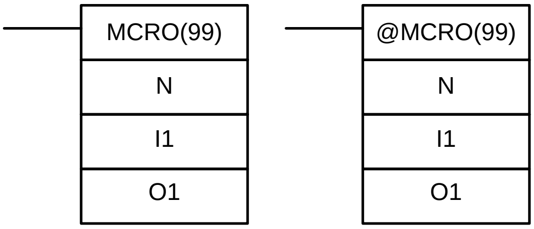
If either DIFU(13) or DIFU(14) is placed within a subroutine, the operand bit will not be turned OFF until the next time the subroutine is executed, i.e., the operand bit may stay ON longer than one cycle.

Flags

There are no flags directly affected by these instructions.

2-17-3 MACRO – MCRO(99)

Ladder Symbols



Operand Data Areas

N: Subroutine number
000 to 049 or 100 to 199
I1: First input word (for N=000 to 049)
IR, SR, AR, DM, EM, TIM/CNT, LR
O1: First output word (for N=000 to 049)
IR, SR, AR, DM, EM, LR

Limitations

DM 6144 to DM 6655 cannot be used for O1.

Description

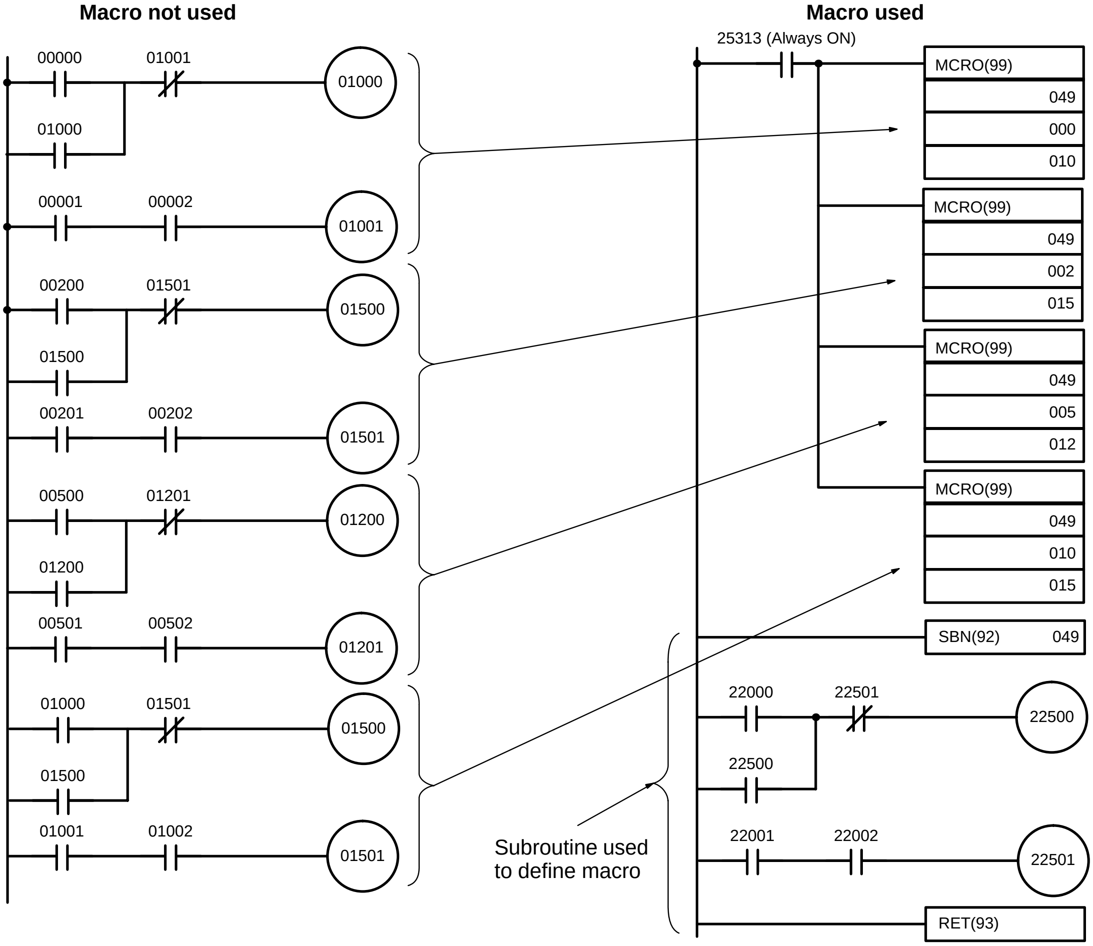
MCRO(99) has two different functions: A normal macro function and external interrupt execution.

Normal Macro Function (N = 000 to 049)
MCRO(99) allows a single subroutine to replace several subroutines that have identical structure but different operands. There are 5 input words, SR 220 to SR 224, and 5 output words, SR 225 to SR 229, allocated to MCRO(99). These 10 words are used in the subroutine and take their contents from I1 to I1+4 and O1 to O1+4 when the subroutine is executed.

When the execution condition is OFF, MCRO(99) is not executed. When the execution condition is ON, MCRO(99) copies the contents of I1 to I1+4 to SR 220 to SR 224, copies the contents of O1 to O1+4 to SR 225 to SR 229, and then calls and executes the subroutine specified in N. When the subroutine is completed, the contents of SR 225 through SR 229 are then transferred back to O1 to O1+4 before MCRO(99) is completed.

The macro function allows a single subroutine (programming pattern) to be used by simply changing the I/O words. A number of similar program sections can be managed with just one subroutine, thereby greatly reducing the number of steps in the program and making the program easier to understand.

When a macro is used, the program can be simplified as shown below.

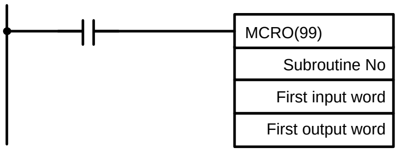


External Interrupt Task Execution (N = 100 to 199)

MCRO(99) can also be used to execute an external interrupt task in the CPU Unit. To do this, set N to 100 + the external interrupt task number and set I1 and O1 to 000.

Using Normal Macros

To use a macro, call a subroutine by means of the MACRO instruction, MCRO(99), as shown below, instead of SBS(91) (SUBROUTINE ENTRY).



When MCRO(99) is executed, operation will proceed as follows:

1, 2, 3...

1. The contents of the five consecutive words beginning with the first input word will be transferred to SR 220 through SR 224.
2. The specified subroutine will be executed until RET(93) (Subroutine Return) is executed.
3. The contents of SR 225 through SR 229 will be transferred to the five consecutive words beginning with the first output word.
4. MCRO(99) will then be finished.

When MCRO(99) is executed, the same instruction pattern can be used as needed simply by changing the first input word and the first output word.

The following restrictions apply when the macro function is used.

- The only words that can be used for each execution of the macro are the five consecutive words beginning with the first input word number (for input) and the five consecutive words beginning with the first output word (for output).
- The specified inputs and outputs must correctly correspond to the words used in the subroutine.
- Even when the direct output method is used for outputs,subroutine results will be actually reflected in the specified output words only when the subroutine has been completed (step 3 above).

Note SR 220 to SR 224 and SR 225 to SR 229 can be used as work bits when MCRO(99) is not used.

The first input word and the first output word can be specified not only with I/O bits, but also with other bits (such as work bits) or with DM words.

Subroutines called by MCRO(99) are defined by SBN(92) and RET(93), just as are ordinary subroutines.

Executing External Interrupt Tasks

MCRO(99) can also be used to execute external interrupt tasks 0 to 99 in the CPU Unit. To do this, set N to 100 + the external interrupt task number and set I1 and O1 to 000. The interrupt will be written to the CPU Unit interface area and then the Equals Flag will be turned ON.

The subroutine program to be executed for the interrupt task must be programmed in the CPU Unit in advance.

Note
1. Always program a dummy subroutine program for the specified interrupt number. If a dummy subroutine is not programmed, a program error will occur attempting to transfer the program from the Programming Device, preventing the program from being transferred.
2. CPU Unit interrupt task 001 is the power interruption task. Interrupt tasks 002 and 003 are then scheduled interrupt tasks. If these interrupt tasks are specified for executing using MCRO(99), they will be executed along with any other external interrupt tasks assigned the same numbers. The following precautions must be observed in doing this.

When external interrupt task 001 is executed using MCRO(99), set the PC Setup in the CPU Unit to disable the power interruption task so that both the external interrupt from the Customizable Counter Unit and the power interrupt in the CPU will not be processed.

When external interrupt task 002 or 003 is executed using MCRO(99), program the CPU Unit so that both the scheduled interrupt and the external interrupt from the Customizable Counter Unit are not processed at the same time.

Refer to the *CS1 Series Programmable Controller Programming Manual* for details on interrupt tasks.

Flags

ER: A subroutine does not exist for the specified subroutine number.

An operand has exceeded a data area boundary.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

A subroutine has called itself.

An active subroutine has been called.

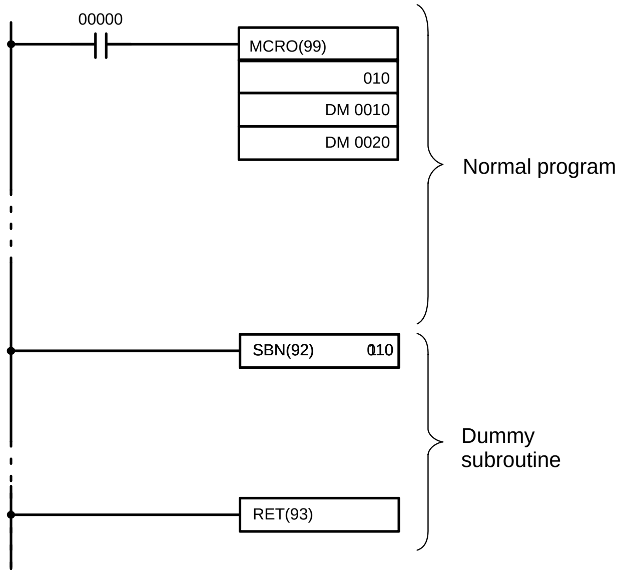
EQ: ON when N is between 100 and 199 to designate an external interrupt task and the CPU Unit has been notified of the external interrupt task number.

Examples

Normal Macro Function

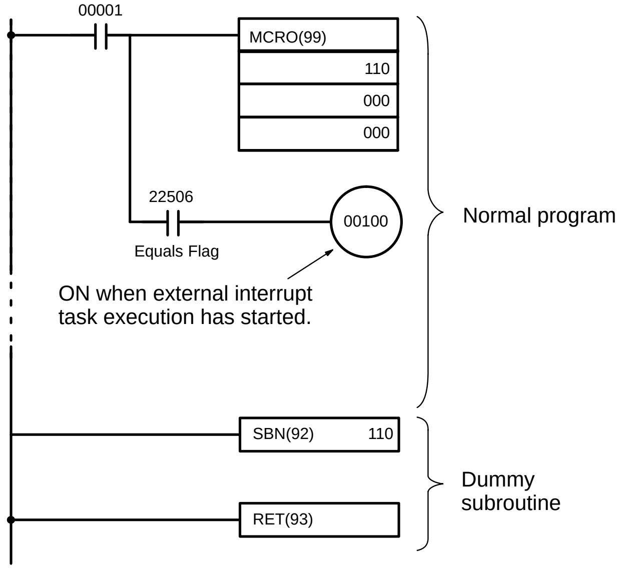
1, 2, 3...

1. In the following program section, the subroutine will be executed when IR 00000 turns ON.
2. The contents of DM 0010 to DM 0014 will be transferred to SR 220 to SR 224.
3. The specified subroutine will be executed.
4. The contents of SR 225 through SR 229 will be transferred to DM 0020 to DM 0024



External Interrupt Task Execution

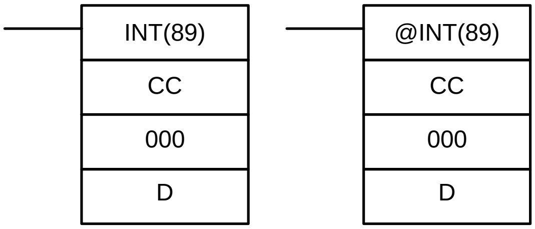
When IR 00001 turns ON in the following example, execution of external interrupt 10 will start in the CPU Unit. IR 00100 will be turned ON when execution has started.



2-18 Interrupt Control Instructions

2-18-1 INTERRUPT CONTROL – INT(89)

Ladder Symbols



Operand Data Areas

CC: Control code

# (000 to 003, 100, or 200)

000: No function

# (000)

D: Control data

IR, SR, AR, DM, EM, TIM/CNT, LR, #

Limitations DM 6644 to DM 6655 cannot be used for D when CC=002.

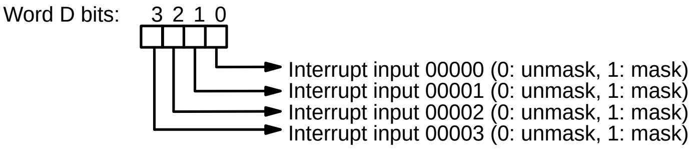
Description When the execution condition is OFF, INT(89) is not executed. When the execution condition is ON, INT(89) is used to control interrupts and performs one of the six functions shown in the following table depending on the value of CC.

INT(89) function	CC
Mask/unmask input interrupts	000
Clear input interrupts	001
Read current mask status	002
Renew counter SV	003
Mask all interrupts	100
Unmask all interrupts	200

Mask/Unmask I/O Interrupts (CC=000)

This function is used to mask and unmask I/O interrupt inputs 00000 to 00003. Masked inputs are recorded, but ignored. When an input is masked, the interrupt program for it will be run as soon as the bit is unmasked (unless it is cleared beforehand by executing INT(89) with CC=001).

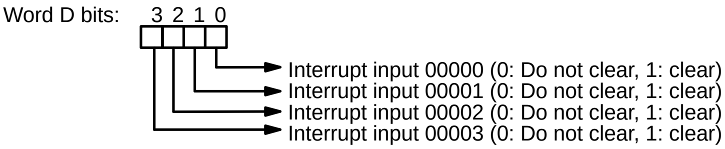
Set the corresponding bit in D to 0 or 1 to unmask or mask an I/O interrupt input. Bits 00 to 03 correspond to 00000 to 00003. Bits 04 to 15 should be set to 0.


Word D bits:
3 2 1 0
Interrupt input 00000 (0: unmask, 1: mask)
Interrupt input 00001 (0: unmask, 1: mask)
Interrupt input 00002 (0: unmask, 1: mask)
Interrupt input 00003 (0: unmask, 1: mask)

Clear I/O Interrupts (CC=001)

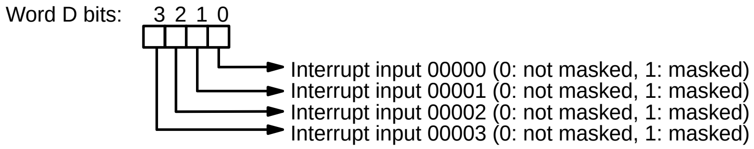
This function is used to clear I/O interrupt inputs 00000 to 00003. Since interrupt inputs are recorded, masked interrupts will be serviced after the mask is removed unless they are cleared first.

Set the corresponding bit in D to 1 to clear an I/O interrupt input. Bits 00 to 03 correspond to 00000 to 00003. Bits 04 to 15 should be set to 0.


Word D bits:
3 2 1 0
Interrupt input 00000 (0: Do not clear, 1: clear)
Interrupt input 00001 (0: Do not clear, 1: clear)
Interrupt input 00002 (0: Do not clear, 1: clear)
Interrupt input 00003 (0: Do not clear, 1: clear)

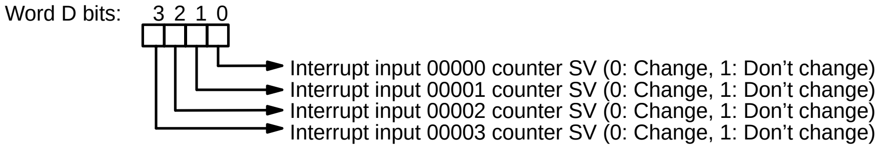
Read Current Mask Status (CC=002)

This function is used to write the current mask status for I/O interrupt inputs 00000 to 00003 to word D. The corresponding bit will be ON if the input is masked. (Bits 00 to 03 correspond to 00000 to 00003.)


Word D bits:
3 2 1 0
Interrupt input 00000 (0: not masked, 1: masked)
Interrupt input 00001 (0: not masked, 1: masked)
Interrupt input 00002 (0: not masked, 1: masked)
Interrupt input 00003 (0: not masked, 1: masked)

Renew Counter SV (CC=003)

This function is used to renew the counter SV for I/O interrupt inputs 00000 to 00003 to word D. Set the corresponding bit in D to 1 in order to renew the input's counter SV. (Bits 00 to 03 correspond to 00000 to 00003.)


Word D bits:
3 2 1 0
Interrupt input 00000 counter SV (0: Change, 1: Don't change)
Interrupt input 00001 counter SV (0: Change, 1: Don't change)
Interrupt input 00002 counter SV (0: Change, 1: Don't change)
Interrupt input 00003 counter SV (0: Change, 1: Don't change)

Mask/Unmasking All Interrupts (CC=100/200)

This function is used to mask or unmask all interrupt processing. Masked inputs are recorded, but ignored. The control data, D, is not used for this function. Set D to #0000.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

CC=100 or 200 while an interrupt program was being executed.

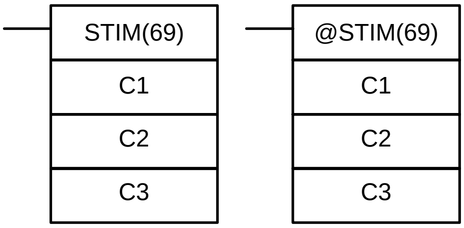
CC=100 when all inputs were already masked.

CC=200 when all inputs were already unmasked.

CC and/or D are not within specified values.

2-18-2 INTERVAL TIMER – STIM(69)

Ladder Symbols



Operand Data Areas

C1: Control data #1

000 to 003, 006, 010 to 012

C2: Control data #2

C1=000 to 003:
IR, SR, AR, DM, EM, TIM/CNT, LR, #
C1=006:
IR, SR, AR, DM, EM, TIM/CNT, LR
C1=010: 000
C1=011, 012: 000 or 001

C3: Control data #3

C1=000, 003:
IR, SR, AR, DM, EM, TIM/CNT, LR, #
C1=006:
IR, SR, AR, DM, EM, TIM/CNT, LR
C1=010: 000;
C1=001, 002, 011, 012: 000 to 003

Limitations

Pulse output functions are supported by the CS1W-HCP22 only. (Interval timer functions are supported by all Customizable Counter Units.)

Note 1. The pulse output mode must be set in the Unit Setup Area to one-shot pulse outputs to enable one-shot pulse outputs or to output pulse counter timing to enable output pulse counter timing. The Error Flag (SR 25503) will turn ON if the wrong mode is set.

2. The following settings cannot be made from the CX-Programmer. To make these settings, transfer the program to the Customizable Counter Unit and then use the Programming Console to adjust the final settings.

   C1 = 011 or 012
   C2 = 001
   C3 = 001 to 003

Description

STIM(69) is used both to control interval timers and to control pulse output ports. The value of C1 determines the overall function of STIM(69).

000: One-shot interrupt timer start
001: One-shot pulse output 1 (CS1W-HCP22 only)
002: One-shot pulse output 2 (CS1W-HCP22 only)
003: Scheduled interrupt timer start
006: Timer PV read
010: Timer stop
011: Start/stop output pulse counter 1 timer (CS1W-HCP22 only)
012: Start/stop output pulse counter 2 timer (CS1W-HCP22 only)

One-shot Interrupt Timer (C1 = 000) and Scheduled Interrupt Timer (C1 = 003)

Set C1=000 to start the interval timer to activate a one-shot interrupt. Set C1=003 to start scheduled interrupts using the interval timer.

C2 specifies the timer's SV and can be a constant or the first of two words containing the SV. The settings are slightly different depending on the method used.

If C2 is a constant, it specifies the initial value of the decrementing counter (BCD, 0005 to 0100). The decrementing time interval is 0.1 ms, i.e., the set value is from 0.5 to 10.0 ms.

If C2 is a word address, C2 specifies the initial value of the decrementing counter (BCD, 0001 to 9999), and C2+1 specifies the decrementing time interval (BCD, 0005 to 0100) in units of 0.1 ms. The decrementing time interval can thus be 0.5 to 10.0 ms.

C3 specifies subroutine number 0000 to 0049 BCD.

Note The time required from interval timer startup to time-up is as follows:
(the content of C2) × (the content of C2+1) × 0.1 ms = 0.5 to 99,990 ms

One-shot Pulse Outputs 1 and 2 (C1 = 001 or 002)

If C1 = 001 or 002, a one-shot output will be produced. C2 will specify the ON time between 0001 and 9999 BCD, with the time unit specified in C3 as follows:

000:	0.1 ms
001:	0.01 ms
002:	0.1 ms
003:	1 ms

Note

1. Once started, pulse output will be performed for the specified time. The differentiated version of STIM(69) should thus be used for one-shot pulse outputs.
2. STIM(69) will be ignored if it is executed for one-shot pulse output when a previous one-shot pulse output has not been completed.
3. The Pulse Output Flag (AR 1817 or AR 1815) will turn ON during pulse output.

Timer PV Read (C1 = 006)

If C1 = 006, the present value of the interval timer for one-shot output or scheduled interrupts is read.

C2 specifies the first of two destination words that will receive the timer's PV. C2 will receive the number of times the decrementing counter has been decremented in hexadecimal and C2+1 will receive the decrementing time interval (BCD in 0.1 ms units).

C3 specifies the destination word that will receive the time which has elapsed since the last time the timer was decremented (BCD in 0.1 ms units).

Note The time that has elapsed since the timer was started is computed as follows: (Content of C2) × (Content of C2 + 1) + (Content of C3) × 0.1 ms

Timer Stop (C1 = 010)

If C1 = 010, the interval timer for one-shot output or scheduled interrupts will be stopped.

C2 and C3 have no function and should both be set to 000.

Note The time stop designation does not work for one-shot pulse output or output pulse counter timing.

Output Pulse Counter 1 or 2 Timer (C1 = 011 or 012)

If C1 is 011 or 012, timing by counting the number of output pulses will be started or stopped.

C2 is set to 000 to start timing and to 001 to stop timing.

C3 specified the time unit as follows:

000:	0.1 ms
001:	0.01 ms
002:	0.1 ms
003:	1 ms

Note

1. Once started, output pulse counter timing will be performed until it is stopped (by setting C2 to 001). The differentiated version of STIM(69) should thus be used for output pulse counter timing.
2. Timing will be restarted if STIM(69) is executed to start output pulse counter timing when output pulse counter timing has already been started.

Flags

ER: A control data setting is not within range, e.g., C1 is not 001 to 003, 006, or 010 to 012.

The Unit is not set in the correct pulse output mode.

C1 = 001, 002, 011, or 012 was specified for the CS1W-HIO01 or CS1W-HCA22.

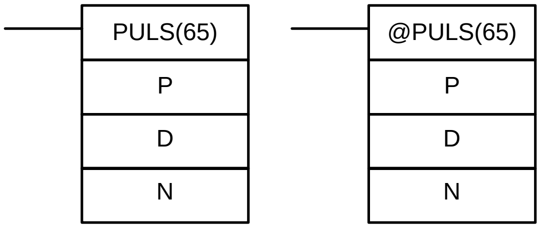
Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

A data area boundary has been exceeded.

2-19 High-Speed Counter/Pulse Output Instructions

2-19-1 SET PULSES – PULS(65)

Ladder Symbols



Operand Data Areas

P: Port specifier
001, 002

D: Pulse type designation
000 to 002

N: First word containing pulse parameters
IR, SR, AR, DM, EM, LR

Limitations

Pulse output functions are supported by the CS1W-HCP22 only. PULS(65) will be processed as a NOP if programmed in any other Customizable Counter Unit.

N to N+3 must be in the same data area.

DM 6143 to DM 6655 cannot be used for N.

Description

PULS(65) is used either for independent mode positioning or for electronic cam mode pulse outputs. The operation performed by PULS(65) is determined by the mode set in the Unit Setup Area.

Note
1. The following output modes can be set in the Unit Setup Area: Relative pulse, absolute linear pulse, absolute ring pulse, or electronic cam (absolute positioning pulse) mode.
2. PULS(65) can be used to independently and simultaneously output pulses on both ports.
3. As a rule, use the differentiated version (@PULS(65)) of the instruction. It is not necessary to keep the instruction execution condition ON to complete the specified output.

Independent Mode Positioning

For independent mode positioning, PULS(65) is used to set parameters for pulse outputs that are started later in the program using SPED(64) or ACC(—).

To use independent mode positioning, set the pulse output mode (DM 6613 and DM 6614) in the Unit Setup Area to one of the following modes: Relative pulse output, linear absolute pulse output, or ring absolute pulse output.

Electronic Cam Mode

In electronic cam mode, PULS(65) is used both to set the number and frequency of output pulses and to actually output pulses according to the settings.

To use PULS (65) for the electronic cam mode, set the pulse output mode (DM 6613 and DM 6614) in the Unit Setup Area to the electronic cam mode.

Port Specifier (P)

The port specifier indicates the pulse output location. (For independent mode positioning, the parameters set in D and N will apply to the next SPED(64) or ACC(—) instruction in which the same port output location is specified.)

Pulse output location	P
Pulse output 1	001
Pulse output 2	002

Pulse Type Designation (D)

D specifies the type of pulses that are output as follows:

Pulse output location	P
Relative	000
Absolute (linear or ring)	001
Electronic cam mode (absolute positioning)	002

Note The type of pulses set here must agree with the setting of the pulse output mode (DM 6613 and DM 6614) in the Unit Setup Area.

Number of Pulses (N and N+1)

N and N+1 specify the number of pulses for relative pulse output or the absolute target position for absolute pulse or electronic cam mode output in 8 digit hexadecimal.

	Leftmost 4 digits	Rightmost 4 digits
Number of pulses:	N+1	N

The setting range depends on the mode as follows:

Mode	P
Relative pulse output	0000 0000 to FFFF FFFF
Linear absolute pulse output	8000 0000 to 7FFF FFFF
Ring absolute pulse output	0000 0000 to ring set value
Electronic cam mode (absolute positioning)	8000 0000 to 7FFF FFFF

Pulse Output Frequency (N+2 and N+3)

N+2 and N+3 specify the pulse output frequency for electronic cam mode output in 8-digit BCD. This setting is used only when D = 002. It is ignored for other settings.

	Leftmost 4 digits	Rightmost 4 digits
Number of pulses:	N+1	N

The setting range is 0000 0001 to 0020 000 in hertz (1 Hz to 200 kHz).

Note The setting range may be restricted by the clock frequency. Refer to the *Customizable Counter Unit Operation Manual* for details and be sure that the frequency setting is actually possible. If the supported setting range is exceeded, the Error Flag (SR 25503) will turn ON and the instruction will not be executed. If a frequency less than the lowest supported frequency is specified, pulses will be output at the lowest supported frequency. (If 0000 0000 is set, PULS(65) will be treated as a NOP and the current status of any previous PULS(65) instructions currently being executed will not change.)

Execution

The number of pulses output is calculated from the specification of the number of output pulses as follows:

Relative pulse output: Specified number of pulses
Absolute pulse output: | Current position – Specified number of pulses |

Independent Mode Positioning

The number of pulses set to be output will be used even if SPED(64) is used to change the pulse frequency during operation. (The number of pulses cannot be changed during operation.)

When performing linear absolute pulse output and the target position is the same as the current position, PULS(65) will not be executed and the target position will not be set. In this case, the Equals Flag (SR 25506) will remain OFF.

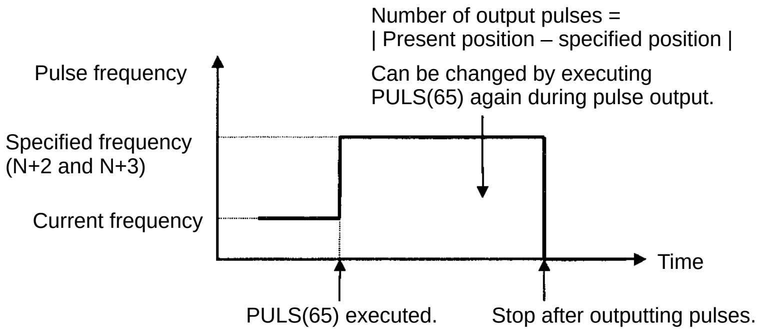
Electronic Cam Mode (Absolute Positioning)

The direction of movement will be as follows for electronic cam mode positioning:

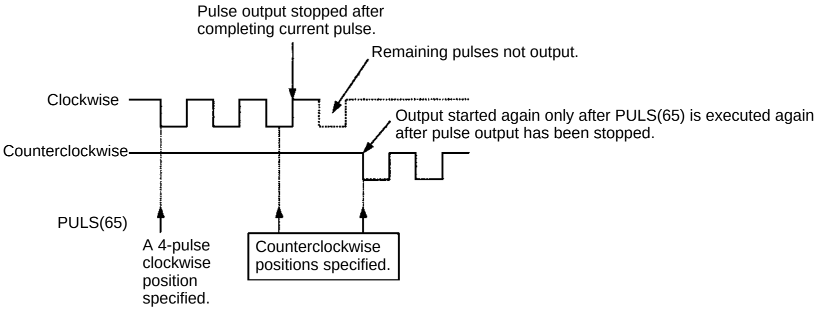
Present position < Designated position: Clockwise
Present position > Designated position: Counterclockwise
Present position = Designated position: No movement

Pulse output will be stopped immediately for electronic cam mode positioning: 1) when pulse output is stopped using INI(61) (C1 = 003), 2) the specified number of pulses have been output, or 3) the Customizable Counter Unit mode is changed to PROGRAM mode.

The target position and output frequency will be updated if PULS(65) is executed for electronic cam mode positioning before a previous execution has been completed.



Note 1. If the direction of pulse output is reversed, pulse output will be ended after outputting the current pulse. The pulse waveform will not be cut off in the middle, but any remaining pulses will not be output. It is thus not possible to automatically reverse direction, and output in the reverse direction will start only when the pulse output specification is given a second time. Also, even a second specification will not be effective if it is given before pulse output has been stopped. Allow for this in programming, keeping in mind that time may be required to stop pulse output for low output frequencies.


Pulse output stopped after
completing current pulse.
Remaining pulses not output.
Clockwise
Output started again only after PULS(65) is executed again
after pulse output has been stopped.
Counterclockwise
PULS(65)
A 4-pulse
clockwise
position
specified.
Counterclockwise
positions specified.

2. If a previous position is reached when PULS(65) is being executed, pulse output will be stopped and PULS(65) execution will not be completed. In this case, the Equals Flag will remain OFF. Re-execute the instruction.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

A data area boundary was exceeded.

There is an error in the operand settings, i.e., P is not 001 or 002 and M is not between 000 and 002.

The Unit is set for one-shot pulse output or output pulse counter timing.

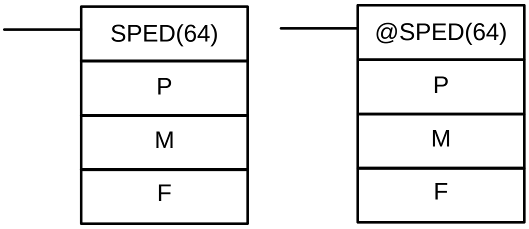
PULS(65) was executed in an interrupt subroutine while an instruction that controls pulse output was being executed in the main program.

EQ: **D = 002**
Pulse output has been executed for PULS(65).

D = 000 or 001
Target position has been set for PULS(65).

2-19-2 SPEED OUTPUT– SPED(64)

Ladder Symbols



Operand Data Areas

P: Port specifier
001, 002

M: Output mode
000 to 003

F: Pulse frequency/Analog amount
IR, SR, AR, DM, EM, LR, #

Limitations

DM 6144 to DM 6655 cannot be used for F.

This instruction is not supported by the CS1W-HIO01 and will be treated as a NOP if execution is attempted.

Description

SPED(64) can be used with the functions listed in the following table.

Unit	Function
CS1W-HCP22	Pulse output (specifying the pulse output frequency and starting output)
CS1W-HCA22	Analog output (specifying the analog output amount and starting output)

For the CS1W-HCP22, SPED(64) is used to set the output pulse frequency and start pulse output in which the frequency will be changed in steps. Either independent positioning mode or continuous speed control mode is possible. For independent positioning mode, the number of pulses is actually set using PULS(65).

Note To use SPED(64) for the CS1W-HCP22, set the pulse output mode (DM 6613 and DM 6614) in the Unit Setup Area to one of the following modes: Relative pulse output, linear absolute pulse output, or ring absolute pulse output.

For the CS1W-HCA22, SPED(64) is used to set the analog output amount and start analog output.

Note To use SPED(64) for the CS1W-HCA22, set the analog output mode (DM 6630) in the Unit Setup Area for refreshing by instruction execution.

Note

1. SPED(64) can be used to independently and simultaneously output pulses or analog amounts on both ports.
2. As a rule, use the differentiated version (@SPED(64)) of the instruction. It is not necessary to keep the instruction execution condition ON to complete the specified output.

Port Specifier (P)

The port specifier specifies the port or output bit where the pulses will be output.

P	Pulse output location
001	Pulse/analog output 1
002	Pulse/analog output 2

Output Mode (M)

The value of M determines the output mode for the CS1W-HCP22.

M	Output mode
000	Continuous mode, clockwise
001	Continuous mode, counterclockwise
002	Independent mode, clockwise
003	Independent mode, counterclockwise

Note M must always be 000 for the CS1W-HCA22.

Pulse Frequency/Analog Amount (F)

The value of F sets the pulse frequency or the analog output amount.

Pulse Frequency (CS1W-HCP22)
The range depends on whether a word address or a constant is designated for F.

F	Unit	Possible values of F
Word address	1 Hz	8-digit BCD 0000 0000 (Stops output.) or 0000 0001 to 0020 0000 (1 Hz to 200 kHz)
Constant	10 Hz	4-digit BCD: 0000 (Stops output.) or 0001 to 9999 (10 Hz to 99,990 Hz)

Note The setting range may be restricted by the clock frequency. Refer to the *Customizable Counter Unit Operation Manual* for details and be sure that the frequency setting is actually possible. If the supported setting range is exceeded, the Error Flag (SR 25503) will turn ON and the instruction will not be executed. If a frequency less than the lowest supported frequency is specified, pulses will be output at the lowest supported frequency.
The source clock is divided by an integer dividing ratio to create the output pulse frequency. This means that the actual output frequency may vary from the specified frequency. Refer to the *Customizable Counter Unit Operation Manual* for precautions on using the pulse output functions.
F cannot be set to values higher than #5001 from the CX-Programmer. To make these settings, transfer the program to the Customizable Counter Unit and then use the Programming Console to adjust the final settings.

Analog Output Amount (CS1W-HCA22)
Set the analog output amount according to the analog range.

Range	Possible values of F
–10 to 10 V	4-digit BCD: EC78 to 1388 (–5,000 to 5,000 decimal, resolution: 10,000, equivalent to 0% to 100%, i.e., –10 to 10 V)
0 to 10 V, 0 to 5 V, or 1 to 5 V	4-digit hexadecimal: 0000 to 0FA0 (0 to 4,000 decimal, resolution: 4,000, equivalent to 0% to 100%, i.e., 0 to 10 V, 0 to 5 V, or 1 to 5 V)

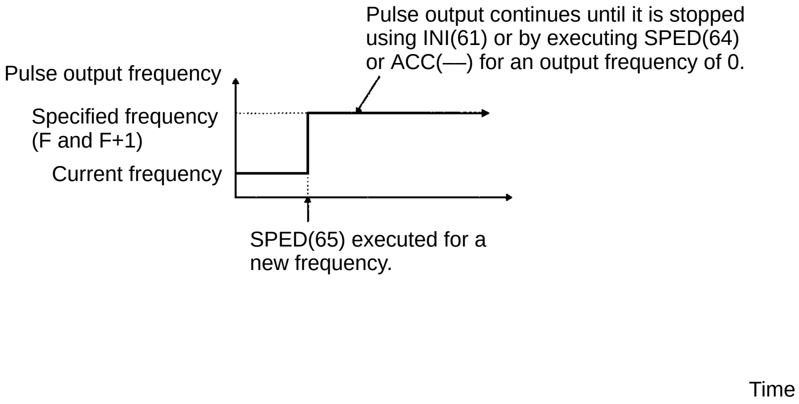
Execution

Pulse Output (CS1W-HCP22)

There are two modes that can be used for pulse output: Continuous and independent positioning. Continuous mode is used to output pulses indefinitely, i.e., until they are stopped by the program. Independent positioning is used to output a specified number of pulses.

Speed Control (Continuous) Mode (M = 000 or 001)
When SPED(64) is executed, the output frequency will be changed stepwise from the current frequency to the specified frequency. Output will be continued

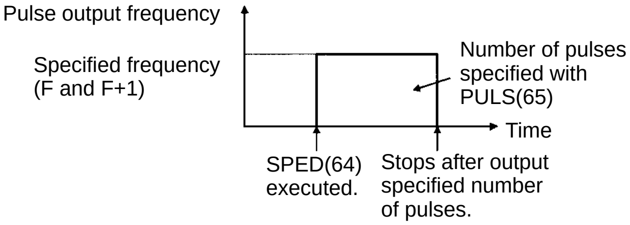
until pulse output is stopped using INI(61) or by executing SPED(64) or ACC(—) for an output frequency of 0.



Independent Positioning Mode (M = 002 or 003)

When SPED(64) is executed, the output is stepped to the specified output frequency, the specified number of output pulses are output, and then pulse output is stopped. The number of pulses must be specified in advance using PULS(65).

PULS(65) must be executed to specify the number of pulses before each execution of SPED(64). If the number of pulses has not been specified each time, SPED(64) will not be executed.



Pulse output will continue until one of the following occurs:

- The number of pulses specified by the PULS(65) instruction is reached in independent positioning mode. (Execute PULS(65) before SPED(64) when specifying independent mode.)
- The INI(61) instruction is executed with C=003.
- SPED(64) is executed again with the output frequency, F, set to 0.
- The Customizable Counter Unit mode is changed to PROGRAM mode

SPED(64) can be executed during independent positioning to change the output frequency. The number of output pulses that was previously specified will be output correctly even if the output frequency is changed. The frequency cannot be changed, however, unless the direction is the same and unless the independent positioning mode is specified.

SPED(64) may not be executed if pulse output is already being controlled by another instruction (e.g., ACC(—) or PLS2(—). If this occurs, the Error Flag (SR 25503) will turn ON. Refer to the *Customizable Counter Unit Operation Manual* for details on conditions for execution during pulse output.

If the relationship between the present position and the target position does not agree with the direction for absolute positioning, the direction designation will be used. In linear absolute positioning mode, this means that the target position will not be reached within the range 8000 0000 to 7FFF FFFF. Pulse output, however, will continue without an overflow or underflow error occurring, and the position will be reached once the range has been exceeded.

In independent mode, the number of pulses that have already been output to ports 1 and 2 are contained in AR 14 and AR 15 for port 1 and in AR 16 and AR 17 for port 2.

	Leftmost 4 digits	Rightmost 4 digits
Port 1 pulse output PV:	AR 15	AR 14
Port 2 pulse output PV:	AR 17	AR 16

Analog Output (CS1W-HCA22)

There is no continuous or independent mode for analog output. The specified analog output will be maintained until 1) SPED(64) or ACC(—) is used to change the output value, or 2) The analog output hold function is set to not hold the previous value and PROGRAM mode is entered or the Analog Output Conversion Enable Bit (AR 1600 or AR 1601) is turned OFF.

The analog output value can be changed using ACC(—). The output value will not be changed, however, if the previous target value has not yet been reached. The output value will also not be changed if SPED(64) is executed for the same port where SPED(64) is already being executed. If either of these occur, the Error Flag (SR 25503) will turn ON.

Flags

ER: Operand settings were out of range. (P was not 001 or 002 or F was output of range.)

A data area boundary was exceeded.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

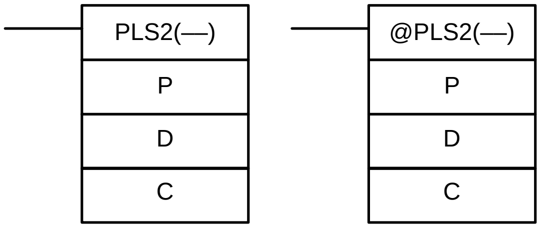
The range supported by the specified clock was exceeded for the CS1W-HCP22.

SPED(64) was executed during pulse output under conditions that do not allow the output frequency or analog output value to be changed (e.g., target value or frequency not yet reached for ACC(—)).

SPED(64) was executed in an interrupt subroutine while the analog or pulse output was being controlled by another instruction in the main program.

2-19-3 PULSE OUTPUT – PLS2(––)

Ladder Symbols



Operand Data Areas

P: Communications port
001 or 002

D: Direction specifier
000 or 001

C: First control word
IR, SR, AR, DM, EM, LR

Limitations

P must be 001 or 002 and D must be 000 or 001.

C to C+7 must be in the same data area.

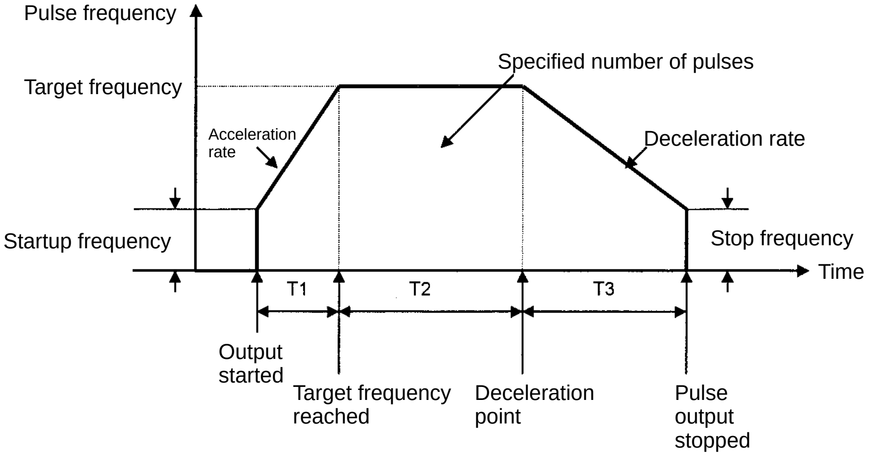
PLS2(—) is supported by the CS1W-HCP22. It will be treated as a NOP if executed for any other Customizable Counter Unit.

Note

1. PLS2(—) will not operate if pulses are already being output from the specified port.
2. To use PLS2(—), set the pulse output mode (DM 6613 and DM 6614) in the Unit Setup Area to relative pulse output or linear absolute pulse output. PLS2(—) will not be executed if the wrong mode is set.
3. As a rule, use the differentiated version (@PLS2(—)) of the instruction. It is not necessary to keep the instruction execution condition ON to complete the specified output.

Description

PLS2(—) is used to output a specified number of CW or CCW pulses from port 1 or 2 in a trapezoid. The pulse output starts at the specified startup frequency, accelerates to the target frequency at a specified acceleration rate, decelerates at the specified deceleration rate, and stops at approximately the same frequency as the startup frequency



The following equations show how to calculate the approximate acceleration time T_1, running time T_2, and deceleration time T_3. All times are in seconds.

$$T_1 \cong 0.002 \times \frac{\text{Target frequency} - \text{Startup frequency}}{\text{Acceration rate}}$$

$$T_2 \cong \frac{\text{Number of pulses} - ((\text{Target frequency} + \text{Startup frequency}) \times (T_1 + T_3)) \div 2}{\text{Target frequency}}$$

$$T_3 \cong 0.002 \times \frac{\text{Target frequency} - \text{Startup frequency}}{\text{Deceleration rate}}$$

Operand Settings

P specifies the port where the pulses will be output. Pulse output 1 is used when P = 001, and pulse output 2 is sued when P = 002.

D specifies whether the output signal is clockwise (CW) or counter-clockwise (CCW). The output is CW when D = 000 and CCW when D = 001.

The content of C to C+7 control the pulse output as shown in the following table.

Words	Contents	Range
C and C+1	Number of pulses	Relative pulse output: 0000 0000 to FFFF FFFF Hex Linear absolute pulse output: 8000 0000 to 7FFF FFFF Hex
C+2 and C+3	Target frequency	0000 0001 to 0020 0000 BCD In hertz (1 Hz to 200 kHz)
C+4 and C+5	Startup frequency	0000 0001 to 0020 0000 BCD In hertz (1 Hz to 200 kHz)
C+6	Acceleration rate	0001 to 2000 BCD Acceleration per 2 ms in hertz (1 Hz to 2 kHz)
C+7	Deceleration rate	0001 to 2000 BCD Acceleration per 2 ms in hertz (1 Hz to 2 kHz)

Note

1. The setting range may be restricted by the clock frequency. Refer to the *Customizable Counter Unit Operation Manual* for details and be sure that the frequency setting is actually possible. If the supported setting range is exceeded, the Error Flag (SR 25503) will turn ON and the instruction will not be executed. If a frequency less than the lowest supported frequency is specified, pulses will be output at the lowest supported frequency.
2. The source clock is divided by an integer dividing ratio to create the output pulse frequency. This means that the actual output frequency may vary from the specified frequency. Refer to the *Customizable Counter Unit Operation Manual* for precautions on using the pulse output functions, including calculation methods for actual output frequencies.
3. If the startup frequency is set to 0 or to a value less than the minimum supported output frequency, the minimum output frequency will be used.

Operation

The following operation is performed for PLS2(—).

1, 2, 3...

1. Pulse output is started at the specified startup frequency.
2. The output frequency is increased at the specified acceleration rate until the specified target frequency is reached.
3. The target frequency is maintained until the deceleration point is reached. The deceleration point is calculated from the remaining number of pulses and the deceleration rate.
4. From the deceleration point, pulse output is decelerated at the specified rate approximately every 2 ms until the stop frequency is reached. The stop frequency is calculated to be as close to but not less than the startup frequency given the target frequency and deceleration rate.

The number of pulses output is calculated from the specification of the number of output pulses as follows:

Relative pulse output: Specified number of pulses
Absolute pulse output: | Current position – Specified number of pulses |

The output pulses may not form a trapezoid, e.g., if the number of pulses is insufficient to reach the target frequency. If the target frequency is not reached, the PLS2 Target Frequency Not Reached Flag (AR 1802 or AR 1810) will turn ON. It will turn ON: 1) when the instruction is executed if the total number of pulses is less than the pulses required for deceleration, and 2) if the deceleration point is reached during acceleration.

If the deceleration point is reach during acceleration, PLS2(—) will still output the correct number of pulses, but the pulses remaining at the end of deceleration will be output at the stop frequency. (Pulses will remain after deceleration because deceleration calculations are calculated from the target frequency.)

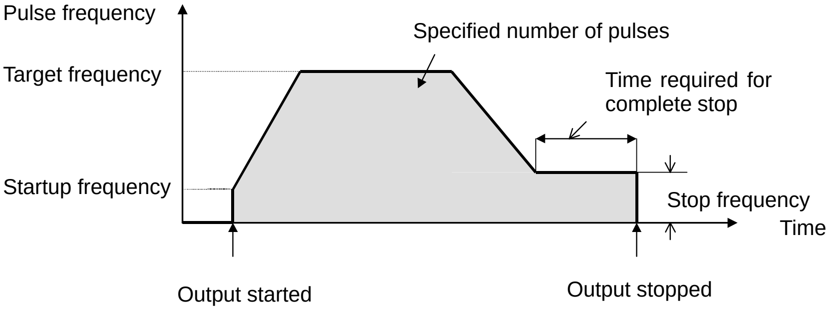
Note

1. When using PLS2(—) in the linear absolute pulse output mode, confirm the present position and set the direction accordingly. Pulses will not be output if

the relationship between the present position and target position does not agree with the specified direction. If this occurs, the Error Flag (SR25503) will turn ON.

2. PLS2(—) cannot be executed for the same port if a previous execution is still in progress.
3. Pulse output will be completed: 1) when pulse output is stopped using INI(61) (C1 = 003), 2) the specified number of pulses have been output, or 3) the Customizable Counter Unit mode is changed to PROGRAM mode.

⚠ Caution Depending on the control data that is set for PLS2(—), there may be pulses remaining when the stop frequency is reached. These pulses will be output at the stop frequency to ensure that the specified number of pulses is output correctly but this will cause an increase in the time required to output all the pulses.



Correct the system by adjusting the acceleration rate, the deceleration rate, the startup speed, or the target speed.

Flags

ER: Indirectly addressed EM/DM word is non-existent. (Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

A data area boundary was exceeded.

There is an error in the operand settings, e.g., P was not 001 or 002 or M was not 000 or 001.

The Unit Setup Area is set to ring absolute pulse output, electronic cam output, one-shot pulse output, or pulse counter timing.

The target frequency, acceleration rate, or deceleration rate is not suitable (e.g., Target frequency < Startup frequency).

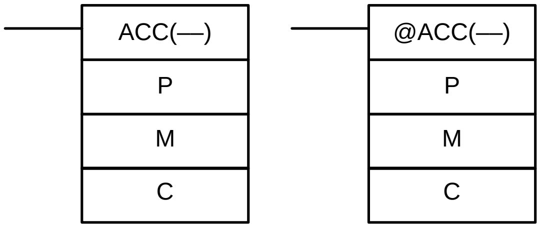
The relationship between the present position and the target position does not agree with the direction setting.

Pulses are already being output for the specified port.

PLS2(—) was executed in an interrupt subroutine while another instruction was controlling pulse output for the same port in the main program.

2-19-4 ACCELERATION CONTROL – ACC(––)

Ladder Symbols



Operand Data Areas

P: Communications port

001 or 002

M: Mode specifier

000 to 007

C: First control word

IR, SR, AR, DM, EM, LR

Limitations

P must be 001 or 002 and M must be 000 to 007.

ACC(—) is not supported by the CS1W-HIO01, and it will be treated as a NOP if executed.

Note

1. To use ACC(—) with the CS1W-HCP22, set the pulse output mode (DM 6613 and DM 6614) in the Unit Setup Area to relative pulse output, linear absolute pulse output, or ring absolute pulse output. ACC(—) will not be executed if the wrong mode is set.
2. To use ACC(—) with the CS1W-HCA22, set the analog output mode (DM 6630) in the Unit Setup Area to refreshing via instruction execution. ACC(—) will not be executed if the wrong mode is set.
3. As a rule, use the differentiated version (@ACC(—)) of the instruction. It is not necessary to keep the instruction execution condition ON to complete the specified output.

Description

ACC(—) can be used for the functions listed in the following table.

Unit	Function
CS1W-HCP22	Pulse Output A specified pulse output frequency can be output using the specified rate of acceleration or deceleration for each port. Either independent positioning or continuous speed control be performed. For independent positioning, ACC(—) is used together with PULS(65).
CS1W-HCA22	Analog Output A sloped analog output value can be output using a specified rate of change.

Operands

<u>CS1W-HCP22: Pulse Output</u>

P specifies the port as follows:

001: Pulse output 1
002: Pulse output 2

M specifies the mode as follows:

000: Clockwise, acceleration, continuous output
001: Counterclockwise, acceleration, continuous output
002: Clockwise, deceleration, continuous output
003: Counterclockwise, deceleration, continuous output
004: Clockwise, acceleration, independent positioning output
005: Counterclockwise, acceleration, independent positioning output
006: Clockwise, deceleration, independent positioning output
007: Counterclockwise, deceleration, independent positioning output

Note M cannot be set to 004 to 007 from the CX-Programmer. To set these values, set dummy values from the CX-Programming, download the program to the Unit, and then correct the settings with a Programming Console.

C is the first of three control words. C contains the acceleration rate as a 4-digit BCD value. Set C to between 0001 and 2000 for an acceleration of 1 Hz to 2 kHz per 2 ms in increments of 1 Hz.

Set C+1 and C+2 to the target frequency as an 8-digit BCD value in increments of 1 Hz. C+1 contains the rightmost 4 digits and C+2 contains the leftmost 4 digits. The setting must be between 0000 0000 and 0020 0000 for a frequency of 0 Hz to 200 kHz.

Note

1. The setting range may be restricted by the clock frequency. Refer to the *Customizable Counter Unit Operation Manual* for details and be sure that the frequency setting is actually possible. If the supported setting range is exceeded, the Error Flag (SR 25503) will turn ON and the instruction will not be executed. If a frequency less than the lowest supported frequency is specified, pulses will be output at the lowest supported frequency.
2. The source clock is divided by an integer dividing ratio to create the output pulse frequency. This means that the actual output frequency may vary from the specified frequency. Refer to the *Customizable Counter Unit Operation Manual* for precautions on using the pulse output functions, including calculation methods for actual output frequencies.
3. If the startup frequency is set to 0 or to a value less than the minimum supported output frequency, the minimum output frequency will be used.
4. ACC(—) can be used to independently and simultaneously output pulses or analog amounts on both ports.

CS1W-HCA22: Analog Output

P specifies the port as follows:

001: Analog output 1
002: Analog output 2

M is always 000.

C is the first of two control words. C contains the rate of change as a 4-digit hexadecimal value. Set C as shown in the following table.

Range	Setting	Value
–10 to 10 V	0000 to 2AF8 Hex (0 to 11,000 decimal)	0% to 110% (0 to 22 V)
0 to 10 V, 0 to 5 V or 1 to 5 V	0000 to 1130 Hex (0 to 4,400 decimal)	0% to 110% (0 to 11 V, 0 to 5.5 V, or 0 to 4.4 V)

Set C+1 to the target analog output value as a 4-digit hexadecimal value as shown in the following table.

Range	Setting	Value
–10 to 10 V	EC78 to 1388 Hex (–5,000 to 5,000 decimal, resolution: 10,000)	0% to 110% (–10 to 10 V)
0 to 10 V, 0 to 5 V or 1 to 5 V	0000 to 0FA0 Hex (0 to 4,000 decimal, resolution: 4000)	0% to 110% (0 to 10 V, 0 to 5 V, or 0 to 5 V)

Execution

The following operation is performed for ACC(—).

CS1W-HCP22: Pulse Output

There are two modes that can be used for pulse output: Continuous speed control and independent positioning. Continuous speed control mode is used to output pulses indefinitely, i.e., until they are stopped by the program. Independent positioning is used to output a specified number of pulses.

Pulse output will continue until one of the following occurs:

- The number of pulses specified by the PULS(65) instruction is reached in independent positioning mode. (Execute PULS(65) before ACC(—) when specifying independent positioning mode.)

- The INI(61) instruction is executed with C=003.
- ACC(—) is executed again with the output frequency set to 0000 0000.
- The Customizable Counter Unit mode is changed to PROGRAM mode

ACC(—) can be executed during independent positioning (for SPED(64) also) to change the output frequency. The number of output pulses that was previously specified will be output correctly even if the output frequency is changed. The frequency cannot be changed, however, unless the target speed is higher than the current frequency for decelerations or lower than the current frequency for accelerations and unless continuous mode operation is already in progress. If these conditions are not met, the Error Flag (SR 25503) will turn ON when execution is attempted.

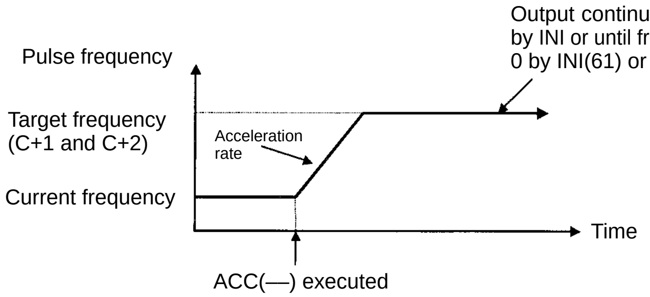
The target frequency may not be reached if the number of pulses is not sufficient to accelerate to it for independent positioning (number of pulses required = time to reach target frequency x target frequency ÷ 2). The target frequency may also not be reached if the number of pulses is not sufficient to decelerate to the target frequency for independent positioning (number of pulses required = time to reach target frequency x (target frequency – initial frequency ÷ 2). If the number of pulses is not sufficient to decelerate to the target frequency for independent positioning when the target frequency is 0, pulse output may be stopped without outputting all pulses (number of pulses required ≐ time to reach target frequency x (target frequency – initial frequency ÷ 2).

If the number of pulses is low and the acceleration or deceleration rate is too high, acceleration or deceleration may not be necessary and operation at a constant speed may be performed.

If the relationship between the present position and the target position does not agree with the direction for independent positioning, the direction designation will be used. In linear absolute positioning mode, this means that the target position will not be reached within the range 8000 0000 to 7FFF FFFF. Pulse output, however, will continue without an overflow or underflow error occurring, and the position will be reached once the range has been exceeded.

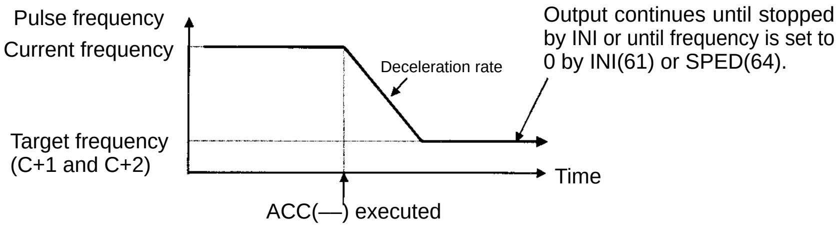
All together there are eight modes that can be set. These are described below.

Modes 000 and 001: Continuous Speed Control with Acceleration
The current output frequency will be increased at the specified acceleration to the specified target frequency. Pulse output will continue after the target frequency has been reached. Pulse output is stopped either by using the INI(61) or by setting the target frequency to 0 using SPED(64) or ACC(—).

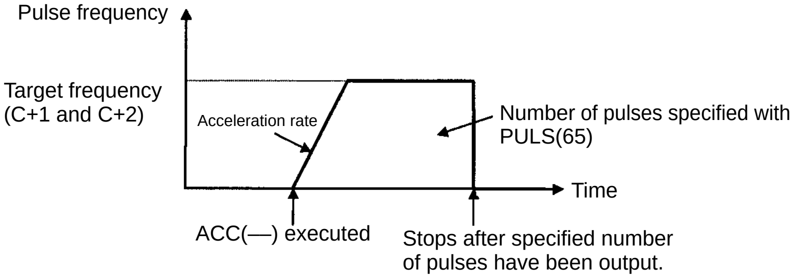


Modes 002 and 003: Continuous Speed Control with Deceleration
The current output frequency will be decreased at the specified deceleration to the specified target frequency. Pulse output will continue after the target fre-

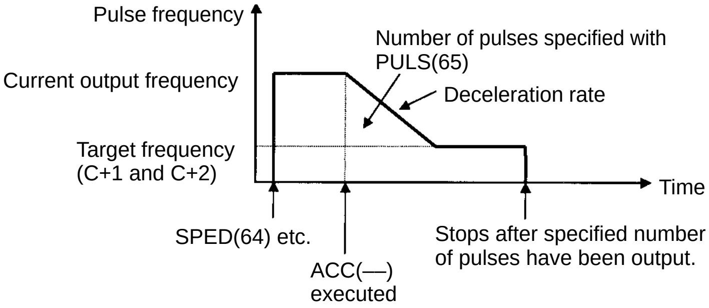
quency has been reached. Pulse output is stopped either by using the INI(61) or by setting the target frequency to 0 using SPED(64) or ACC(—).


Pulse frequency
Current frequency
Deceleration rate
Output continues until stopped by INI or until frequency is set to 0 by INI(61) or SPED(64).
Target frequency (C+1 and C+2)
Time
ACC(—) executed

Modes 004 and 005: Independent Positioning with Acceleration
The output frequency will be increased at the specified acceleration to the specified target frequency. Pulse output will be stopped after the number of pulses specified with PULS(65) have been output. The desired number of pulses must be set in advance with PULS(65) to used these modes.


Pulse frequency
Target frequency (C+1 and C+2)
Acceleration rate
Number of pulses specified with PULS(65)
Time
ACC(—) executed
Stops after specified number of pulses have been output.

Modes 006 and 007: Independent Positioning with Deceleration
The current output frequency (specified by SPED(—) or other instruction) will be decreased at the specified deceleration to the specified target frequency. Pulse output will continue at the target frequency, and be stopped after the number of pulses specified with PULS(65) have been output. The desired number of pulses must be set in advance with PULS(65) to used these modes.


Pulse frequency
Number of pulses specified with PULS(65)
Current output frequency
Deceleration rate
Target frequency (C+1 and C+2)
Time
SPED(64) etc.
Stops after specified number of pulses have been output.
ACC(—) executed

CS1W-HCA22: Analog Output
There is no continuous or independent mode for analog output. The specified analog output will be maintained until 1) SPED(64) or ACC(—) is used to change the output value, or 2) The analog output hold function is set to not hold the previous value and PROGRAM mode is entered or the Analog Output Conversion Enable Bit (AR 1600 or AR 1601) is turned OFF.

The analog output value being output for SPED(64) or ACC(—) can be changed using ACC(—). The output value will not be changed, however, if the previous target value has not yet been reached. If this occurs, the Error Flag (SR 25503) will turn ON.

Flags **ER:** Indirectly addressed EM/DM word is non-existent. (Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

A data area boundary has been crossed.

There is an error in the operand settings.
P does not equal 001 or 002
M does not equal 000 to 007 for the CS1W-HCP22 or 000 for the CS1W-HCA22.

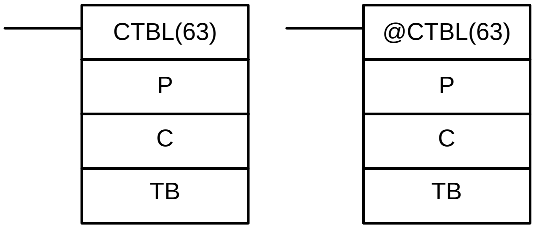
The range supported by the specified clock was exceeded for the CS1W-HCP22.

ACC(––) was executed during pulse output or analog output under conditions that do not allow the output frequency or analog output value to be changed (e.g., target value or frequency not yet reached for ACC(––)).

ACC(––) was executed in an interrupt subroutine while analog or pulse output was being controlled by another instruction in the main program.

2-19-5 REGISTER COMPARISON TABLE – CTBL(63)

Ladder Symbols



Operand Data Areas

P: Port specifier
001 to 004

C: Control data
000 to 002

TB: First comparison table word
IR, SR, AR, DM, EM, LR

Limitations The first and last comparison table words must be in the same data area. (The length of the comparison table varies according to the settings.)

This instruction is not supported by the CS1W-HIO01 and will be treated as a NOP if execution is attempted.

Note As a rule, use the differentiated version (@CTBL(63)) of the instruction when performing target value comparisons. Comparisons will continue until stopped by the program. (For range comparisons, however, only one comparison is made each time CTBL(63) is executed.)

Description CTBL(63) is used to register a comparison table and perform comparisons for a high-speed counter PV, pulse output PV, or pulse counter timer PV. Either target value or range comparisons are possible.

For target comparisons, a subroutine is executed when the PV equals a target position. Comparison is started and continues when CTBL(63) is executed.

For range comparisons, a bit pattern is output internally (AR 11 and AR 13 or AR 21 and AR 23) when the PV is within one of the set ranges. One comparison is made each time CTBL(63) is executed.

The functions of CTBL(63) depends on the Unit, as shown in the following table.

Unit	Function
CS1W-HCP22/ HCA22	Pulse Input Function Used to register a target value comparison table and start comparison, to execute a range comparison, or to just register a target value comparison table (without starting comparison)
CS1W-HCP22	Pulse Output Function Used to register a target value comparison table and start comparison, to execute a range comparison, or to just register a target value comparison table (without starting comparison)

Operands

P specifies the port for which pulses are to be counted as shown in the following table.

P	Port
001	High-speed counter 1
002	High-speed counter 2
003	Pulse output 1
004	Pulse output 2

The function of CTBL(63) is determined by the control data, C, as shown in the following table.

C	CTBL(63) function
000	Registers a target value comparison table and starts comparison.
001	Registers a range comparison table and performs one comparison.
002	Registers a target value comparison table. Start comparison with INI(61).

Note If CTBL(63) is executed with C set to 002 when target value comparison is already in progress, comparison will be stopped. Use CTBL(63) with C set to 000 or use INI(61) to start comparison.

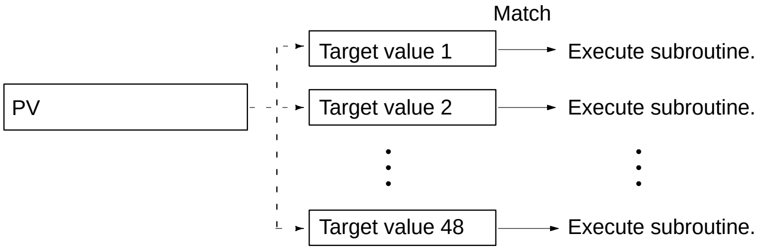
TB is the first word of the comparison table. The structure of the comparison table depends on the type of comparison being performed. Refer to the following sections for details.

Execution

The operation of CTBL(63) depends on whether target value comparison or range comparison has been specified.

Target Value Comparison

Up to 48 target values can be registered. A subroutine number (1 to 48) is registered for each target value. The corresponding subroutine is called and executed when the PV matches a target value. (When interrupt processing is not required, FFFF Hex may be entered for the subroutine number.)

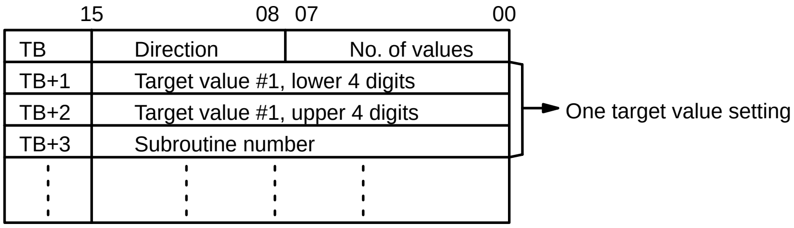


Target value comparisons are performed one item at a time in the comparison table. When the direction is set for incrementing values, comparison will start at the first value in the table that is larger than the PV. When the direction is set for decrementing values, comparison will start at the first value in the table that is smaller than the PV. (If a value larger or smaller than the PV is not found for the specified direction, comparison will start with the first entry in the table.)

When the PV equals the starting value in the table, the interrupt subroutine will be executed and comparison will continue to the next value in the table. When processing has been completed for the last target value in the table, comparison goes to the first value in the table and the process is repeated.

Comparison will continue until it is stopped using INI(61) or until a new comparison table is registered.

The following diagram shows the structure of the target value comparison table.



	15–08	07–00
TB	Direction	No. of values
TB+1	Target value #1, lower 4 digits	
TB+2	Target value #1, upper 4 digits	
TB+3	Subroutine number	
⋮	⋮	⋮

Set the values in the table as follows:

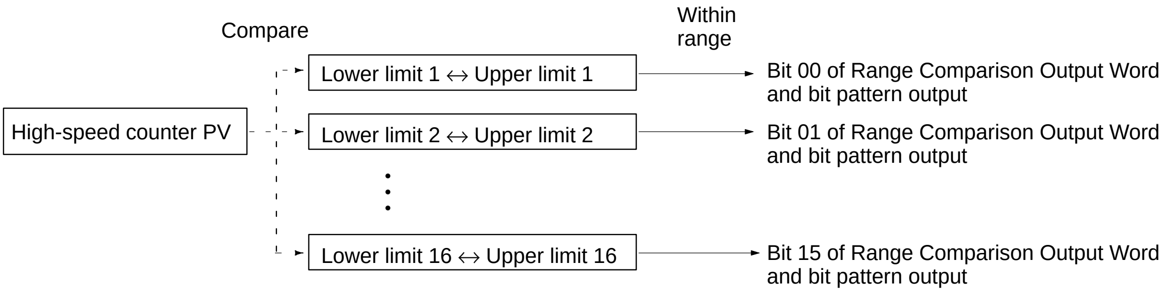
Direction:	00 Hex for incrementing values, F0 Hex for decrementing
No. of values:	01 to 48 BCD
Target values:	8-digit hexadecimal values
Subroutine No.:	0000 to 0049 BCD (The same subroutine can be set more than once.)

Note
1. Set the target values so that interrupts are separated by at least the following interval: Interrupt overhead time + subroutine execution time.
2. Do not change the ring value during comparison in Ring Mode.
3. Do not use a pulse output counter in the following modes; operation will not be correct: Independent positioning, electronic cam, or one-shot pulse output.
4. Counting will be started when the Count Start Bit is turned ON or the pulse output operation is started, but interrupt subroutines will not be executed until comparison is started.
5. Use INI(61) to stop the comparison operation.
6. The registered comparison table is valid until the Customizable Counter Unit is turned OFF or until a different table is registered. Use the differentiated form of CTBL(63) to reduce the scan time.

Range Comparison

A range comparison table contains 16 ranges, each of which is defined by an 8-digit lower limit and an 8-digit upper limit, as well as a bit pattern. Each time CTBL(63) is executed, the PV is compared to each range in the comparison table.

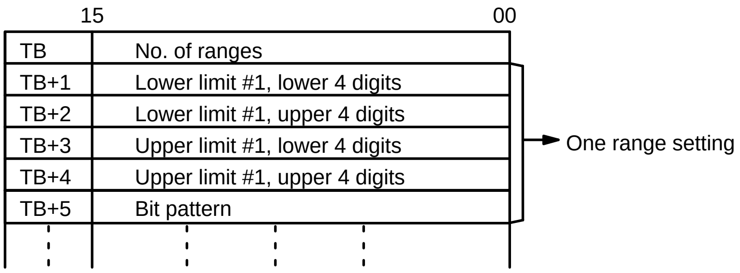
When the PV is within a range in the table, the corresponding bit in the AR Area is turned ON and the registered bit pattern is output to corresponding AR Area word. If the PV is within more than one range, an OR of the bit patterns is output to corresponding AR Area word.


Compare
Within range
High-speed counter PV
Lower limit 1 ↔ Upper limit 1
Lower limit 2 ↔ Upper limit 2
Lower limit 16 ↔ Upper limit 16
Bit 00 of Range Comparison Output Word and bit pattern output
Bit 01 of Range Comparison Output Word and bit pattern output
Bit 15 of Range Comparison Output Word and bit pattern output

The AR Area words corresponding to each range are listed in the following table.

Port	Contents	Word
High-speed counter 1	Range Comparison Output	Corresponding bits of AR 10
	Bit Pattern Output	AR 11
High-speed counter 2	Range Comparison Output	Corresponding bits of AR 12
	Bit Pattern Output	AR 13
Pulse output 1	Range Comparison Output	Corresponding bits of AR 20
	Bit Pattern Output	AR 21
Pulse output 2	Range Comparison Output	Corresponding bits of AR 22
	Bit Pattern Output	AR 23

The following diagram shows the structure of the range comparison table.



TB	No. of ranges
TB+1	Lower limit #1, lower 4 digits
TB+2	Lower limit #1, upper 4 digits
TB+3	Upper limit #1, lower 4 digits
TB+4	Upper limit #1, upper 4 digits
TB+5	Bit pattern

Set the values in the table as follows:

No. of ranges:	0001 to 0016 BCD
Target values:	8-digit hexadecimal values
Bit pattern.:	0000 to FFFF Hex

Note

1. The ranges may overlap.
2. Each upper limit must be greater than or equal to the corresponding lower limit in Linear Counting Mode. If it is not, operation will not be correct and the Error Flag will not turn ON.
3. Do not change the ring value during comparison in Ring Mode.

Flags

ER: The specified port and function are not correct, i.e., P is not 001 to 004 or M is not 000 to 002.

Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

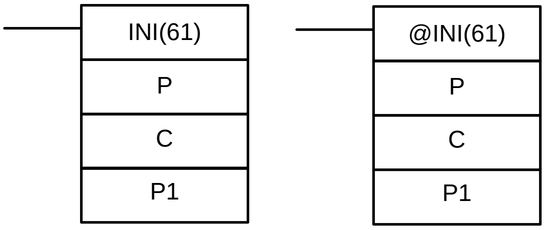
The comparison table exceeds the data area boundary.

There is an error in the comparison table settings. (For target value comparison, the number of values is not 01 to 48, a target value exceeds the ring value for Ring Counter Mode, a subroutine number is not 0000 to 0049 or FFFF Hex, or all subroutine numbers are FFFF Hex. For range comparison, S is not 0001 to 0016.)

CTBL(63) was executed in an interrupt subroutine while another instruction controlling the high-speed counter was being executed in the main program.

2-19-6 MODE CONTROL – INI(61)

Ladder Symbols



Operand Data Areas

P: Port specifier
001 to 004

C: Control data
000 to 003

P1: First PV word
IR, SR, AR, DM, EM, LR

Limitations

P1 must be 000 unless C is 002.

P1 and P1+1 must be in the same data area.

This instruction is not supported by the CS1W-HIO01 and will be treated as a NOP if execution is attempted.

INI(61) cannot be used to change the current value of the input interrupt counter mode.

Description

INI(61) can be used with the functions listed in the following table.

Unit	Function
CS1W-HCP22/ HCA22	Pulse Input Function Used to start and stop target value comparison or to change the current value of the counter.
CS1W-HCP22	Pulse Output Function Used to start and stop target value comparison, to change the current value of the pulse output, or to stop pulse output.

Note If INI(61) is specified for a pulse output port for the CS1W-HCA22, the Error Flag will turn ON.

Operands

P specifies the port to which the operation applies.

P	Port
001	High-speed counter 1
002	High-speed counter 2
003	Pulse output 1
004	Pulse output 2

The function of INI(61) is determined by the control data, C.

C	INI(61) function
000	Starts target value comparison.
001	Stops target value comparison.
002	Changes high-speed counter PV or current pulse output value.
003	Stops pulse output.

P1 and P1+1 contain the new PV when changing the PV.

CTBL(63) Table Comparison

If C is 000 or 001, INI(61) starts or stops comparison of the high-speed counter's PV to the target value comparison table registered with CTBL(63).

Note

1. A target value comparison table must be registered in advance with CTBL(63). If INI(61) is executed without registering a table, the Error Flag will turn ON.
2. INI(61) cannot be used to start range comparison.

3. Comparison will continue until stopped by the program. Use the differentiated form of INI(61) to reduce the scan time.

PV Change

If C is 002, INI(61) changes the high-speed counter's PV or the current pulse output value to the 8-digit value in P1 and P1+1. The leftmost 4 digits are stored in P1+1 and the rightmost 4 digits are stored in P1. A hexadecimal value of F in the most significant digit of PV indicates that the PV is negative.

Note

1. If the PV is changed to the value of a target value in the comparison table, the corresponding subroutine will be executed. This does not apply, however, if the PV is changed to the same value as the current PV even if that value equals a target value.
2. Comparison will continue after the PV is changed until stopped by the program. Use the differentiated form of INI(61) to reduce the scan time.

Pulse Output 1 or 2 (P = 003 or 004)

The following table shows the possible 8-digit hexadecimal values for the PV of the pulse outputs. The pulse output value can be changed with INI(61) only when pulse output is stopped.

Mode	Range
Linear counting	8000 0000 to 7FFF FFFF Hex
Ring counting	0000 0000 Hex to ring set value

Note The counter will be reset to 0 if the pulse output value is changed for relative pulse output, one-shot pulse output, or pulse counting timing, i.e., INI(61) cannot be used to change the PV to a specific value in these modes.

High-speed Counter 1 or 2

The following table shows the possible 8-digit hexadecimal values for the PV of high-speed counters 1 and 2.

Mode	Range
Linear counting	8000 0000 to 7FFF FFFF Hex
Ring counting	0000 0000 Hex to ring set value

Stop Pulse Output

If C is 003, INI(61) stops pulse output. C cannot be set to 003 for a high-speed counter and the Error Flag will turn ON if it is.

Note

1. If pulse output cannot be stopped, e.g., if the output is high, pulse output will be stopped during the next I/O refresh period, i.e., time may be required before pulse output actually stops.
2. INI(61) cannot be used to stop pulse output for one-shot pulse output or pulse counter timing. The Error Flag will turn ON.

Flags

ER: Indirectly addressed EM/DM word is non-existent.
(Content of *EM/*DM word is not BCD, or the EM/DM area boundary has been exceeded.)

P1+1 exceeds the data area boundary.

A new PV was specified for a port that is currently outputting.

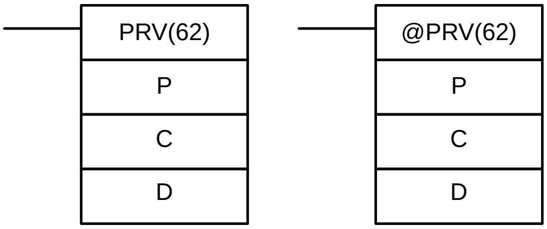
INI(61) was executed to stop pulse output when using a one-shot pulse output or pulse counter timing.

There is an error in the operand settings. (P is not 001 to 004 (001 or 002 for the CS1W-HCA22), C is not 000 to 003 (003 for the CS1W-HCA22), or P1 and P1+1 exceed the ring set value in ring counter mode.

INI(61) was executed in an interrupt subroutine while another instruction controlling the counter was being executed in the main program.

2-19-7 HIGH-SPEED COUNTER PV READ – PRV(62)

Ladder Symbols



Operand Data Areas

P: Port specifier
001 to 004

C: Control data
000 or 001

D: First destination word
IR, SR, AR, DM, EM, LR

Limitations

D and D+1 must be in the same data area.

This instruction is not supported by the CS1W-HIO01 and will be treated as a NOP if execution is attempted.

PRV(62) cannot be used to read the current value of the input interrupt counter mode.

Description

The functions of PRV(62) depends on the Unit, as shown in the following table.

Unit	Function
CS1W-HCP22/ HCA22	Pulse Input Function Used to read the counter PV, read the counter change amount, or read the frequency.
CS1W-HCP22	Pulse Output Function Used to read the pulse output PV, read the pulse counter timing PV, or read the pulse output time for one-shot pulse output.

When the execution condition is OFF, PRV(62) is not executed. When the execution condition is ON, PRV(62) reads data specified by P and C and writes it to D and D+1.

Operands

P specifies the port for which data is to be read.

P	Port
001	High-speed counter 1
002	High-speed counter 2
003	Pulse output 1
004	Pulse output 2

The control data, C, determines which type of data will be accessed.

C	Data
000	High-speed counter PV, pulse output PV, or pulse output counter timer PV
001	Change in counter PV or current frequency

Reading the PV (C=000)

If C is 000, PRV(62) reads the specified PV and writes the 8-digit value in D and D+1. The leftmost 4 digits are stored in D+1 and the rightmost 4 digits are stored in D. A hexadecimal value of F in the most significant digit of PV indicates that the PV is negative.

Mode	Possible values
Linear counter	8000 0000 to 7FFF FFFF Hex
Ring counter	0000 0000 Hex to ring set value
Absolute pulse linear output	8000 0000 to 7FFF FFFF Hex
Absolute pulse ring output	0000 0000 Hex to ring set value
Absolute pulse output	0000 0000 to FFFF FFFF Hex
Output pulse counter timer	0000 0000 to FFFF FFFF Hex
One-shot output time	0000 0000 to 0000 270F Hex

Note Although the PV read by PRV(62) is essentially the same as the value stored in AR 00 though AR 03 or AR 14 through AR 17, the values in the AR Area are refreshed only once a scan, whereas PRV(62) can be used to read the most current value at the time of PRV(62) execution.

Reading the Counter PV Change or Frequency (C=001)

If C is 001, PRV(62) reads the change in the counter PV or the output frequency and writes the 8-digit value in D and D+1. The leftmost 4 digits are stored in D+1 and the rightmost 4 digits are stored in D. A hexadecimal value of F in the most significant digit of PV indicates that the PV is negative.

Set whether to read the change in the counter PV or the frequency in the Unit Setup Area (DM 6606 or DM 6608).

Mode	Possible values
Change in counter PV	0000 0000 to FFFF FFFF Hex
Frequency	0000 0000 to 9999 9999 BCD

Flags

ER: The specified port and function are not compatible (e.g., P = 003 and C = 001).

Indirectly addressed EM/DM word is non-existent.
(Content of ∗EM/∗DM word is not BCD, or the EM/DM area boundary has been exceeded.)

D+1 exceeds the data area boundary.

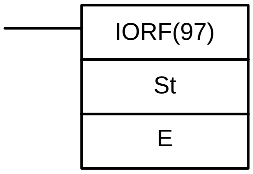
There is an error in the operand settings. (P is not 01 to 004 or C is not 000 or 001.)

PRV(62) was executed in an interrupt subroutine when another instruction controlling the counter was being executed in the main program.

2-20 I/O Instructions

2-20-1 I/O REFRESH – IORF(97)

Ladder Symbol



Operand Data Areas

St: Starting word

IR 000 to IR 001

E: End word

IR 000 to IR 001

Limitations

St must be less than or equal to E.

Description

To refresh I/O words, specify the first (St) and last (E) I/O words to be refreshed. When the execution condition for IORF(97) is ON, all words between St and E will be refreshed. This will be in addition to the normal I/O refresh performed during the Customizable Counter Unit’s cycle.

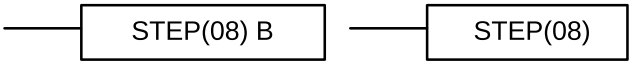
Note This instruction will have no effect on words that are not being used for I/O.

Flags

There are no flags affected by this instruction.

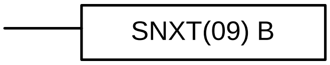
2-21 Step Instructions: STEP DEFINE and STEP START–STEP(08)/SNXT(09)

Ladder Symbols



Definer Data Areas

B: Control bit
IR, AR, LR



B: Control bit
IR, AR, LR

Limitations

All control bits must be in the same word and must be consecutive.

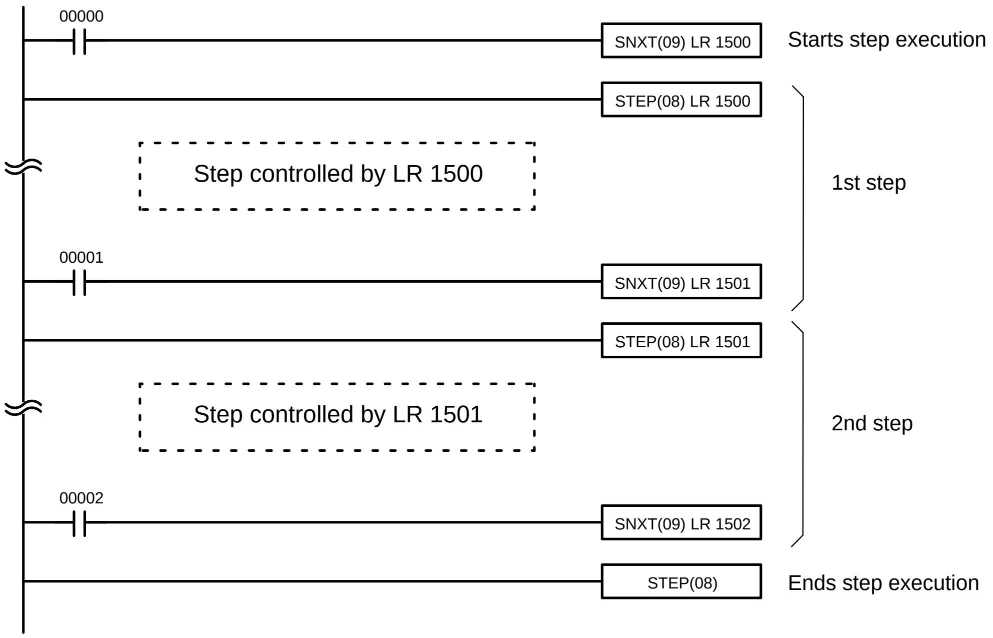
Description

The step instructions STEP(08) and SNXT(09) are used together to set up breakpoints between sections in a large program so that the sections can be executed as units and reset upon completion. A section of program will usually be defined to correspond to an actual process in the application. (Refer to the application examples later in this section.) A step is like a normal programming code, except that certain instructions (i.e., END(01), IL(02)/ILC(03), JMP(04)/JME(05), and SBN(92)) may not be included.

STEP(08) uses a control bit in the IR or HR areas to define the beginning of a section of the program called a step. STEP(08) does not require an execution condition, i.e., its execution is controlled through the control bit. To start execution of the step, SNXT(09) is used with the same control bit as used for STEP(08). If SNXT(09) is executed with an ON execution condition, the step with the same control bit is executed. If the execution condition is OFF, the step is not executed. The SNXT(09) instruction must be written into the program so that it is executed before the program reaches the step it starts. It can be used at different locations before the step to control the step according to two different execution conditions (see example 2, below). Any step in the program that has not been started with SNXT(09) will not be executed.

Once SNXT(09) is used in the program, step execution will continue until STEP(08) is executed without a control bit. STEP(08) without a control bit must be preceded by SNXT(09) with a dummy control bit. The dummy control bit may be any unused IR bit. It cannot be a control bit used in a STEP(08).

Execution of a step is completed either by execution of the next SNXT(09) or by turning OFF the control bit for the step (see example 3 below). When the step is completed, all of the IR bits in the step are turned OFF and all timers in the step are reset to their SVs. Counters, shift registers, and bits used in KEEP(11) maintain status. Two simple steps are shown below.



Address	Instruction	Operands
00000	LD	00000
00001	SNXT(09)	LR 1500
00002	STEP(08)	LR 1500
Step controlled by LR 1500.		
00100	LD	00001
00101	SNXT(09)	LR 1501
00102	STEP(08)	LR 1501
Step controlled by LR 1501.		
00200	LD	00002
00201	SNXT(09)	LR 1502
00202	STEP(08)	---

Steps can be programmed in consecutively. Each step must start with STEP(08) and generally ends with SNXT(09) (see example 3, below, for an exception). When steps are programmed in series, three types of execution are possible: sequential, branching, or parallel. The execution conditions for, and the positioning of, SNXT(09) determine how the steps are executed. The three examples given below demonstrate these three types of step execution.

Precautions

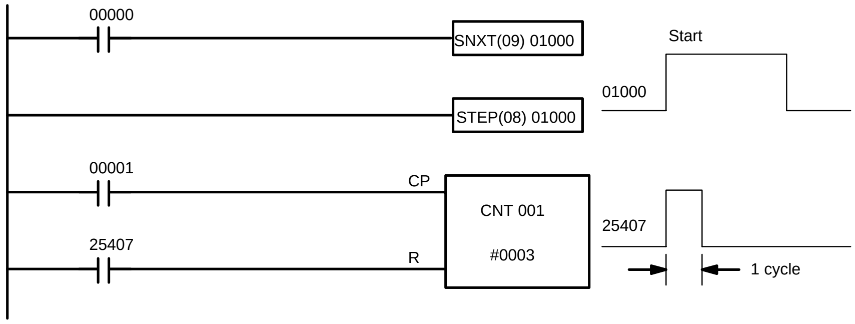
Interlocks, jumps, SBN(92), and END(01) cannot be used within step programs.

Bits used as control bits must not be used anywhere else in the program unless they are being used to control the operation of the step (see example 3, below). All control bits must be in the same word and must be consecutive.

If IR or LR bits are used for control bits, their status will be lost during any power interruption.

Flags

25407: Step Start Flag; turns ON for one cycle when STEP(08) is executed and can be used to reset counters in steps as shown below if necessary.

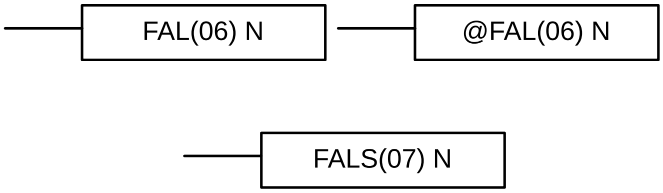


Address	Instruction	Operands
00000	LD	00000
00001	SNXT(09)	01000
00002	STEP(08)	01000
00003	LD	00001

Address	Instruction	Operands
00004	LD	25407
00005	CNT	001
		# 0003

2-22 User Error Instructions: FAILURE ALARM AND RESET – FAL(06) and SEVERE FAILURE ALARM – FALS(07)

Ladder Symbols



Definer Data Areas

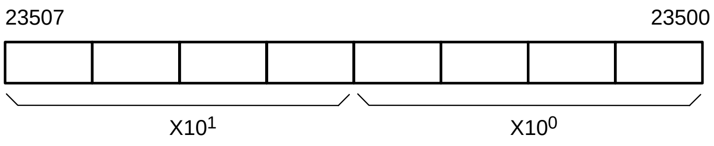
N: FAL number
(00 to 99)

N: FAL number
(01 to 99)

Description

FAL(06) and FALS(07) are provided so that the programmer can output error numbers for use in operation, maintenance, and debugging. When executed with an ON execution condition, either of these instructions will output a FAL number to bits 00 to 07 of SR 235. The FAL number that is output can be between 01 and 99 and is input as the definer for FAL(06) or FALS(07). FAL(06) with a definer of 00 is used to reset this area (see below).

FAL Area



FAL(06) produces a non-fatal error and FALS(07) produces a fatal error. When FAL(06) is executed with an ON execution condition, the ERC indicator on the front of the Customizable Counter Unit will flash, but PC operation will continue. When FALS(07) is executed with an ON execution condition, the ERC indicator will light and PC operation will stop.

The system also generates error codes to the FAL area.

Resetting Errors

FAL error codes will be retained in memory, although only one of these is available in the FAL area. To access the other FAL codes, reset the FAL area by executing FAL(06) 00. Each time FAL(06) 00 is executed, another FAL error will be moved to the FAL area, clearing the one that is already there. FAL error codes are recorded in numerical order.

If the FAL area cannot be cleared, as is generally the case when FALS(07) is executed, first remove the cause of the error and then clear the FAL area through the Programming Console or the CX-Programmer.

Index

A

analog output amount,

analog outputs,

applications, precautions,

AR Area,

B

backup. *See* flash memory

bits, controlling,

C

comparing tables,

comparison,

Continuous Speed Control with Acceleration,

Continuous Speed Control with Deceleration,

counters,
- conditions when reset, ,
- creating extended timers,
- reversible counters,

D

data
- decrementing,
- incrementing,

decrementing,

E

EC Directives,

electronic cam mode,

EM Area,

error codes, programming,

errors, resetting,

expansion instructions,

F

FAL area,

flags
- arithmetic, programming example, ,
- CY
  - clearing,
  - setting,

flash memory, copying,

function codes,

H

high-speed counters,

high-speed counting, reading the PV,

high-speed timer,

I

I/O points, refreshing,

incrementing,

independent mode positioning,

independent positioning,

Independent Positioning with Acceleration,

Input Area,

instruction set,
- ACC(—),
- ADB(50),
- ADBL(—),
- ADD(30),
- ADDL(54),
- AND,
- AND LD,
- AND NOT,
- ANDW(34),
- ASFT(17),
- ASL(25),
- ASR(26),
- AVG(—),
- BCD(24),
- BCDL(59),
- BCMP(68),
- BCNT(67),
- BIN(23),
- BINL(58),
- BSET(71),
- CLC(41),
- CMP(20),
- CMPL(60),
- CNT,
- CNTR(12),
- COLL(81),
- COM(29),
- CPS(—),
- CPSL(—),
- CTBL(63),
- DBS(—),
- DBSL(—),

DEC(39),
DIFD(14),
using in interlocks,
using in jumps,
DIFU(13),
using in interlocks,
using in jumps,
DIST(80),
DIV(33),
DIVL(57),
DVB(53),
END(01),
FAL(06),
FALS(07),
IL(02),
ILC(03),
INC(38),
INI(61),
INT(89),
IORF(97),
JME(05),
JMP(04),
KEEP(11),
LD,
LD NOT,
MAX(—),
MBS(—),
MBSL(—),
MCRO(99),
MIN(—),
MLB(52),
MOV(21),
MOVB(82),
MOVD(83),
MOVL(—),
MUL(32),
MULL(56),
MVN(22),
NEG(—),
NEGL(—),
NOP(00),
OR,
OR LD,
OR NOT,
ORW(35),
OUT,
OUT NOT,
PLS2(—),
PRV(62),
PULS(65),
RET(93),
ROL(27),
ROR(28),
RSET,
SBB(51),
SBBL(—),
SBN(92),
SBS(91),
SCL(66),
SCL2(—),
SCL3(—),
SET,
SFT(10),
SFTR(84),
SLD(74),
SNXT(09),
SPED(64),
SRD(75),
STC(40),
STEP(08),
STIM(69),
SUB(31),
SUBL(55),
TCMP(85),
TIM,
TIMH(15),
TMHH(—),
VCAL(—),
WSFT(16),
XCHG(73),
XFER(70),
XNRW(37),
XORW(36),
ZCP(—),
ZCPL(—),

instructions
comparison instructions,
conversion,
data control instructions,
data movement instructions,
data shift instructions,
high-speed counter instructions,
I/O instructions,
incrementing/decrementing instructions,
instruction set lists,
interrupt control instructions,
list,
logic instructions,
math instructions,
mnemonics,
pulse output instructions,
sequence input instructions,
sequence output instructions,
special instructions,
special math instructions,
subroutine instructions,
subroutines,
table data processing instructions,
timer and counter,

interlocks,

interrupt tasks, executing interrupts in CPU Unit,

interrupts,
control,

J

jump numbers,

jumps,

L

ladder diagram
controlling bit status
using DIFU(13) and DIFD(14),

using KEEP(11),
using SET and RSET,
instructions, controlling bit status, using OUT and OUT NOT,

linear approximation,

linear functions,

LR Area,

M

macro function, subroutines. *See* programming

memory areas,

N

nesting, subroutines,

O

one-shot interrupt,

operating environment, precautions,

Output Area,

output bit, controlling ON/OFF time,

P

precautions,
applications,
general,
operating environment,
safety,

programming
macro function, subroutines,
preparing data in data areas,

pulse frequency,

pulse outputs, , ,
controlling, ,

PV
CNTR(12),
timers and counters,

R

range comparison,

Read/Write portion of DM Area,

S

safety precautions. *See* precautions

speed control, ,

speed outputs,

SR Area,

subroutine number,

subroutines,
executing,
executing in CPU Unit,

SV
CNTR(12),
timers and counters,

T

target value comparison,

TIM/CNT numbers,

Timer/Counter Area ,

timer/counter numbers,

timers,
conditions when reset, , ,
reading PV,

TR Area,

W

Work Area,

Revision History

A manual revision code appears as a suffix to the catalog number on the front cover of the manual.

Cat. No. W384-E1-1

Revision code

The following table outlines the changes made to the manual during each revision. Page numbers refer to the previous version.

Revision code	Date	Revised content
1	January 2001	Original production

OMRON Corporation
Control Devices Division H.Q.
Shiokoji Horikawa, Shimogyo-ku,
Kyoto, 600-8530 Japan
Tel: (81)75-344-7109/Fax: (81)75-344-7149

Regional Headquarters

OMRON EUROPE B.V.
Wegalaan 67-69, NL-2132 JD Hoofddorp
The Netherlands
Tel: (31)2356-81-300/Fax: (31)2356-81-388

OMRON ELECTRONICS LLC
1 East Commerce Drive, Schaumburg, IL 60173
U.S.A.
Tel: (1)847-843-7900/Fax: (1)847-843-8568

OMRON ASIA PACIFIC PTE. LTD.
83 Clemenceau Avenue,
#11-01, UE Square,
Singapore 239920
Tel: (65)6835-3011/Fax: (65)6835-2711

OMRON (CHINA) CO., LTD.
Room 2211, Bank of China Tower,
200 Yin Cheng Zhong Road,
PuDong New Area, Shanghai, 200120 China
Tel: (86)21-5037-2222/Fax: (86)21-5037-2200

OMRON

Authorized Distributor:

Cat. No. W384-E1-1

Note: Specifications subject to change without notice.

Printed in Japan
0101-3M