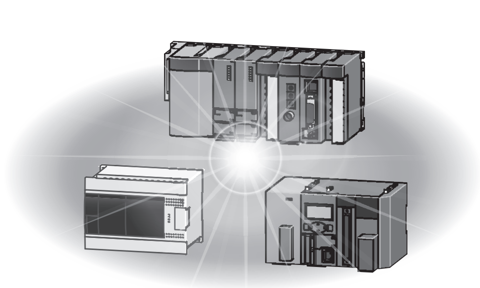
MITSUBISHI ELECTRIC

Programmable Controller

MELSEC Q series MELSEC L series

MELSEC-Q/L Structured Programming Manual (Special Instructions)



SAFETY PRECAUTIONS

(Read these precautions before using this product.)

Before using MELSEC-Q or -L series programmable controllers, please read the manuals included with each product and the relevant manuals introduced in those manuals carefully, and pay full attention to safety to handle the product correctly.

Make sure that the end users read the manuals included with each product, and keep the manuals in a safe place for future reference.

CONDITIONS OF USE FOR THE PRODUCT

(1) Mitsubishi programmable controller ("the PRODUCT") shall be used in conditions;

i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and

ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI'S USER, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above, restrictions Mitsubishi may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTs are required. For details, please contact the Mitsubishi representative in your region.

INTRODUCTION

Thank you for purchasing the Mitsubishi MELSEC-Q or -L series programmable controllers.
Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the programming specifications to handle the product correctly.
When applying the program examples introduced in this manual to an actual system, ensure the applicability and confirm that it will not cause system control problems.

CONTENTS

MANUALS

RELEVANT MANUALS

The manuals related to this product are listed below. Order each manual as needed, referring to the following lists.

■Structured programming

Manual name Manual number (model code)	Description
MELSEC-Q/L/F Structured Programming Manual (Fundamentals) <SH-080782ENG>	Methods and languages for structured programming
MELSEC-Q/L Structured Programming Manual (Common Instructions) <SH-080783ENG>	Specifications and functions of common instructions, such as sequence instructions, basic instructions, and application instructions, that can be used in structured programs
MELSEC-Q/L Structured Programming Manual (Application Functions) <SH-080784ENG>	Specifications and functions of application functions that can be used in structured programs

■Operation of GX Works2

Manual name Manual number (model code)	Description
GX Works2 Version 1 Operating Manual (Common) <SH-080779ENG>	System configuration, parameter settings, and online operations of GX Works2, which are common to Simple projects and Structured projects
GX Works2 Version 1 Operating Manual (Structured Project) <SH-080781ENG>	Operations, such as programming and monitoring in Structured projects, of GX Works2
GX Works2 Beginner's Manual (Structured Project) <SH-080788ENG>	Basic operations, such as programming, editing, and monitoring in Structured projects, of GX Works2. This manual is intended for first-time users of GX Works2.

■Detailed specifications of instructions

- Analog instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	Analog-Digital Converter Module User's Manual <SH-080055>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q64AD, Q68ADV, and Q68ADI
●	—	Channel Isolated High Resolution Analog-Digital Converter Module / Channel Isolated High Resolution Analog-Digital Converter Module (With Signal Conditioning Function) User's Manual <SH-080277>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q64AD-GH and Q62AD-DGH
●	—	Channel Isolated Analog-Digital Converter Module/Channel Isolated Analog-Digital Converter Module (With Signal Conditioning Function) User's Manual <SH-080647ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q68AD-G and Q66AD-DG
●	—	MELSEC-Q High Speed Analog-Digital Converter Module User's Manual <SH-080987ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q64ADH
●	—	MELSEC-Q High Speed Digital-Analog Converter Module User's Manual <SH-081101ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q64DAH
●	—	Digital-Analog Converter Module User's Manual <SH-080054>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q62DAN, Q64DAN, Q68DAVN, and Q68DAIN
●	—	Channel Isolated Digital-Analog Converter Module User's Manual <SH-080281E>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q62DA-FG
●	—	Channel Isolated Digital-Analog Converter Module User's Manual <SH-080648ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q66DA-G
●	—	RTD Input Module Channel Isolated RTD Input Module User's Manual <SH-080142>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q64RD and Q64RD-G

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	Thermocouple Input Module Channel Isolated Thermocouple/Micro Voltage Input Module User's Manual <SH-080141>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q64TD and Q64TDV-GH
●	—	Channel Isolated Thermocouple Input Module User's Manual <SH-080795ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q68TD-G-H01/Q68TD-G-H02
●	—	Channel Isolated RTD Input Module User's Manual <SH-080722ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q68RD3-G
●	—	Load Cell Input Module User's Manual <SH-080821ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q61LD
●	—	MELSEC-Q Current Transformer Input Module User's Manual <SH-081033ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the Q68CT
—	●	MELSEC-L Analog-Digital Converter Module User's Manual <SH-080899ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the analog-digital converter module
—	●	MELSEC-L Dual Channel Isolated High Resolution Analog-Digital Converter Module User's Manual <SH-081103ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the L60AD4-2GH
—	●	MELSEC-L Digital-Analog Converter Module User's Manual <SH-080900ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the digital-analog converter module
—	●	MELSEC-L Analog Input/Output Module User's Manual <SH-081167ENG>	System configuration, specifications, settings, and troubleshooting of the analog input/output module

• Positioning instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	Type QD75P/QD75D Positioning Module User's Manual (Details) <SH-080058>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the QD75P1N/QD75P2N/QD75P4N/QD75D1N/QD75D2N/QD75D4N/QD75P1/QD75P2/QD75P4/QD75D1/QD75D2/QD75D4
●	—	Type QD75M Positioning Module User's Manual (Details) <IB-0300062>	System configuration, performance specifications, functions, handling, procedures before operation, and troubleshooting of the QD75M1/QD75M2/QD75M4
●	—	Type QD75MH Positioning Module User's Manual (Details) <IB-0300117>	System configuration, performance specifications, functions, handling, procedures before operation, and troubleshooting of the QD75MH1/QD75MH2/QD75MH4
—	●	MELSEC-L LD75P/LD75D Positioning Module User's Manual <SH-080911ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the LD75P1/LD75P2/LD75P4/LD75D1/LD75D2/LD75D4

• Serial communication instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	Q Corresponding Serial Communication Module User's Manual (Basic) <SH-080006>	The overview for use of the module, applicable system configuration, specifications, procedures before operation, fundamental data communication with external devices, maintenance, inspection, and troubleshooting
—	●	MELSEC-L Serial Communication Module User's Manual (Basic) <SH-080894ENG>	The overview for use of the module, applicable system configuration, specifications, procedures before operation, fundamental data communication with external devices, maintenance, inspection, and troubleshooting
●	●	MELSEC-Q/L Serial Communication Module User's Manual (Application) <SH-080007>	The specifications and usage of special functions of the module, settings for special functions, and data communication with external devices

• Network dedicated instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	MELSEC-Q CC-Link System Master/Local Module User's Manual <SH-080394E>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the QJ61BT11N
—	●	MELSEC-L CC-Link System Master/Local Module User's Manual <SH-080895ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the built-in CC-Link and CC-Link system master/local modules
●	—	CC-Link IE Controller Network Reference Manual <SH-080668ENG>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the CC-Link IE Controller Network

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual <SH-080917ENG>	The specifications, procedures before operation, system configuration, installation, settings, functions, programming, and troubleshooting of the CC-Link IE Field Network and the CC-Link IE Field Network master/local module
—	●	MELSEC-L CC-Link IE Field Network Master/Local Module User's Manual <SH-080972ENG>	The specifications, procedures before operation, system configuration, installation, settings, functions, programming, and troubleshooting of the CC-Link IE Field Network and the CC-Link IE Field Network master/local module
●	—	Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network) <SH-080049>	The specifications, settings and procedures before operation, parameter setting, programming, and troubleshooting of the MELSECNET/H PLC-to-PLC network system
●	—	Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network) <SH-080124>	System configuration, performance specifications, and programming of the MELSECNET/H network system (remote I/O network)
●	—	Q Corresponding Ethernet Interface Module User's Manual (Basic) <SH-080009>	The specifications of the Ethernet module, data communication procedure with external devices, line connection (open/close), fixed buffer communication, random access buffer communication, and troubleshooting
—	●	MELSEC-L Ethernet Interface Module User's Manual (Basic) <SH-081105ENG>	The specifications of the Ethernet module, data communication procedure with external devices, line connection (open/close), fixed buffer communication, random access buffer communication, and troubleshooting
●	●	MELSEC-Q/L Ethernet Interface Module User's Manual (Application) <SH-080010>	The e-mail function of the Ethernet module, programmable controller CPU status monitoring, communication function using the MELSECNET/H or MELSECNET/10 as a relay station, communication with data link instructions, and the use of file transfer (FTP server) function

- PID control instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	●	MELSEC-Q/L/QnA Programming Manual (PID Control Instructions) <SH-080040>	The dedicated instructions for PID control

- Socket communication function instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	—	QnUCPU User's Manual (Communication via Built-in Ethernet Port) <SH-080811ENG>	Functions for the communication via built-in Ethernet port of the CPU module
—	●	MELSEC-L CPU Module User's Manual (Built-In Ethernet Function) <SH-080891ENG>	The built-in Ethernet function of the CPU module

- Built-in I/O function instruction

Series		Manual name Manual number (model code)	Description
Q	L		
—	●	MELSEC-L CPU Module User's Manual (Built-In I/O Function) <SH-080892ENG>	The general-purpose I/O function, interrupt input function, pulse catch function, positioning function, and high-speed counter function of the CPU module

- Data logging function instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	●	QnUDVCPU/LCPU User's Manual (Data Logging Function) <SH-080893ENG>	Specifications of the data logging function, and operating method of the LCPU logging configuration tool

- SFC control instruction

Series		Manual name Manual number (model code)	Description
Q	L		
●	●	MELSEC-Q/L/QnA Programming Manual (SFC) <SH-080041>	The programming methods required to create SFC program, specifications and functions

TERMS

This manual uses the generic terms and abbreviations listed in the following table to discuss the software packages and programmable controller CPUs. Corresponding module models are also listed if needed.

Term	Description
Application function	A generic term for the functions, such as functions and function blocks, defined in IEC61131-3. (The functions are executed with a set of common instructions in a programmable controller.)
Basic model QCPU	A generic term for the Q00JCPU, Q00CPU, and Q01CPU
Built-in Ethernet port LCPU	A generic term for the L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT
Built-in Ethernet port QCPU	A generic term for the Q03UDVCPU, Q03UDECPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU
CC-Link IE	A generic term for CC-Link IE Controller Network system and CC-Link IE Field Network system
Common instruction	A generic term for the sequence instructions, basic instructions, application instructions, data link instructions, multiple CPU dedicated instructions, multiple CPU high-speed transmission dedicated instructions, and redundant system instructions
CPU module	A generic term for QCPU (Q mode) and LCPU
GX Works2	Product name of the software package for the MELSEC programmable controllers
High Performance model QCPU	A generic term for the Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
High-speed Universal model QCPU	A generic term for the Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU, and Q26UDVCPU
IEC61131-3	The abbreviation for the IEC 61131-3 international standard
LCPU	A generic term for the L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, and L26CPU-PBT
MELSECNET/H	The abbreviation for MELSECNET/H network system
Personal computer	A generic term for personal computer on which Windows® operates
Process CPU	A generic term for the Q02PHCPU, Q06PHCPU, Q12PHCPU, and Q25PHCPU
QCPU (Q mode)	A generic term for the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU.
Redundant CPU	A generic term for the Q12PRHCPU and Q25PRHCPU
Special instruction	A generic term for the module dedicated instructions, PID control instructions, socket communication function instructions, built-in I/O function instructions, and data logging function instructions
Universal model Process CPU	A generic term for the Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, and Q26UDPVCPU
Universal model QCPU	A generic term for the Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU

1 OVERVIEW

1.1 Purpose of This Manual

This manual explains the instructions for the network module, intelligent function module, PID control, socket communication function, built-in I/O function, and data logging function among common instructions and special instructions necessary for creating programs using the structured programming technique. Manuals for reference are listed in the following table according to their purpose.

For information such as the contents and number of each manual, refer to the list of 'Related manuals'.

☞ Page 6 RELEVANT MANUALS

Operation of GX Works2

Purpose		Overview	Details
Installation	Learning the operating environment and installation method	—	📖GX Works2 Installation Instructions
	Learning a USB driver installation method	—	📖 GX Works2 Version 1 Operating Manual (Common)
Operation of GX Works2	Learning all functions of GX Works2	📖 GX Works2 Version 1 Operating Manual (Common)	—
	Learning the project types and available languages in GX Works2		—
	Learning the basic operations and operating procedures when creating a simple project for the first time	—	📖 GX Works2 Beginner's Manual (Simple Project)
	Learning the basic operations and operating procedures when creating a structured project for the first time	—	📖 GX Works2 Beginner's Manual (Structured Project)
	Learning the operations of available functions regardless of project type.	—	📖 GX Works2 Version 1 Operating Manual (Common)
	Learning the functions and operation methods for programming	📖 GX Works2 Version 1 Operating Manual (Common)	📖 GX Works2 Version 1 Operating Manual (Simple Project) 📖 GX Works2 Version 1 Operating Manual (Structured Project)
	Learning data setting methods for intelligent function module	—	📖 GX Works2 Version 1 Operating Manual (Intelligent Function Module)

Operations in each programming language

For details of instructions used in each programming language, refer to the following.

☞ Page 11 Details of instructions in each programming language

Purpose		Overview	Details
Simple Project	Ladder	📖 GX Works2 Beginner's Manual (Simple Project)	📖 GX Works2 Version 1 Operating Manual (Simple Project)
	SFC	📖 GX Works2 Beginner's Manual (Simple Project)*1	
	ST	📖 GX Works2 Beginner's Manual (Structured Project)	📖 GX Works2 Version 1 Operating Manual (Structured Project)
Structured Project	Ladder	📖 GX Works2 Beginner's Manual (Simple Project)	📖 GX Works2 Version 1 Operating Manual (Simple Project)
	SFC	📖 GX Works2 Beginner's Manual (Simple Project)*1	
	Structured ladder/FBD	📖 GX Works2 Beginner's Manual (Structured Project)	📖 GX Works2 Version 1 Operating Manual (Structured Project)
	ST		

*1 MELSAP3 and FX series SFC only

Details of instructions in each programming language

Purpose		Overview	Details
All languages	Learning details of programmable controller CPU error codes, special relays, and special registers	—	Use's Manual (Hardware Design, Maintenance and Inspection) for the CPU module used
Using ladder diagram	Learning the types and details of common instructions	—	MELSEC-Q/L Programming Manual (Common Instruction)
	Learning the types and details of instructions for intelligent function modules	—	Manual for the intelligent function module used
	Learning the types and details of instructions for network modules	—	Manual for the network module used
	Learning the types and details of instructions for the PID control function	—	MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)
	Learning the types and details of the process control instructions	—	MELSEC-Q Programming/Structured Programming Manual (Process Control Instructions)
Using SFC language	Learning details of specifications, functions, and instructions of SFC (MELSAP3)	—	MELSEC-Q/L/QnA Programming Manual (SFC)
Using structured ladder/FBD or structured text language	Learning the fundamentals for creating a structured program	—	MELSEC-Q/L/F Structured Programming Manual (Fundamentals)
	Learning the types and details of common instructions	—	MELSEC-Q/L Structured Programming Manual (Common Instructions)
	Learning the types and details of instructions for intelligent function modules	MELSEC-Q/L Structured Programming Manual (Special Instructions)	Manual for the intelligent function module used
	Learning the types and details of instructions for network modules		Manual for the network module used
	Learning the types and details of instructions for the PID control function		MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)
	Learning the types and details of application functions	—	MELSEC-Q/L Structured Programming Manual (Application Functions)
	Learning the types and details of the process control instructions	—	MELSEC-Q Programming/Structured Programming Manual (Process Control Instructions)

1.2 Explanation Content in This Manual

This manual explains the programming methods and data used for control of the following modules and PID control using structured programming technique.

Function/module for explaining an instruction		Processing performed by the instruction	Reference
Analog module		• Switches the mode. (Offset/gain setting mode or normal mode) • Reads the user range setting offset/gain value. • Restores the user range setting offset/gain value.	Page 33 Analog Instruction
Positioning module		• Restores the absolute position of the specified axis. • Starts positioning of the specified axis. • Executes teaching of the specified axis. • Writes parameters/positioning data and block start data to a flash ROM. • Initializes setting data.	Page 39 Positioning Instruction
Serial communication module		• Sends and receives data to and from an external device. • Registers and reads user frames.	Page 51 Serial Communication Instruction
CC-Link system master/local module		• Reads and writes data from and to an intelligent device station on the CC-Link system. • Reads and writes data from and to the auto-refresh buffer memory at the master station. • Sets the network parameters.	Page 98 Network Dedicated Instruction
CC-Link IE network module MELSECNET/H network module Ethernet interface module		• Sends and receives data to and from an external device. • Reads and writes data from and to another station on the CC-Link IE or MELSECNET/H network system. • Reads and clears error information. • Sends and receives e-mails.	
PID control instruction		• Sets PID control data and performs PID operation for inexact differential and exact differential. • Stops and starts operation of the specified loop. • Changes the parameter of the specified loop.	Page 227 PID CONTROL INSTRUCTION
Socket communication function		• Opens/closes a connection. • Reads receive data. • Changes the receive mode.	Page 252 SOCKET COMMUNICATION FUNCTION INSTRUCTION
Built-in I/O function	Positioning function	• Starts positioning of the specified axis. • Starts OPR of the specified axis. • Starts JOG operation of the specified axis. • Restores the absolute position of the specified axis. • Stops the operating axis. • Changes the speed and the target position of the specified axis.	Page 272 BUILT-IN I/O FUNCTION INSTRUCTION
	Counter function	• Updates the current value of the specified CH. • Sets a ring counter lower limit value and a ring counter upper limit value. • Sets a preset value/latch counter value/sampling counter value. • Sets the coincidence output No. n point. • Measures the frequency/rotation speed. • Stores the measured pulse value. • Outputs the PWM wave form.	
Data logging function		• Generates a trigger on the data logging of the specified data logging configuration number. • Resets the LOGTRG instruction of the specified data logging configuration number.	Page 299 DATA LOGGING FUNCTION INSTRUCTION
SFC control		• Reads comment of an active step in the specified SFC block. • Reads comment of transition condition associated with an active step in the specified SFC block.	Page 301 SFC CONTROL INSTRUCTION

Point

• Precautions on using instructions

For details of the specifications, functions, and operating timing of each instruction, refer to the related manuals of each module.

☞ MANUALS

1.3 Modules and Versions Applicable to Instructions

This section describes the modules and versions applicable to the instructions explained in this manual. For details of applicable versions, refer to each instruction in Chapter 5

Function/module for explaining an instruction		Applicable version/serial number
Analog module	Q64AD, Q68ADV, Q68ADI, Q64AD-GH, Q62AD-DGH, Q68AD-G, Q66AD-DG, Q64ADH, Q64DAH, Q62DAN, Q64DAN, Q68DAVN, Q68DAIN, Q62DA, Q64DA, Q68DAV, Q68DAI, Q62DA-FG, Q66DA-G, Q64RD, Q64RD-G, Q64TD, Q64TDV-GH, Q68TD-G-H01, Q68TD-G-H02, Q68RD3-G, Q61LD, Q68CT, L60AD4, L60AD4-2GH, L60DA4, L60AD2DA2, L60ADVL8, L60ADIL8, L60DAVL8, L60DAIL8	Applicable to all versions
Positioning module	QD75P1N, QD75P2N, QD75P4N, QD75D1N, QD75D2N, QD75D4N, QD75P1, QD75P2, QD75P4, QD75D1, QD75D2, QD75D4, QD75M1, QD75M2, QD75M4, QD75MH1, QD75MH2, QD75MH4, LD75P1, LD75P2, LD75P4, LD75D1, LD75D2, LD75D4	Applicable to all versions
Serial communication module	QJ71C24N, QJ71C24N-R2, QJ71C24N-R4, QJ71C24, QJ71C24-R2, LJ71C24, LJ71C24-R2	The modules that can use the UINI instruction are limited. ☞ Page 89 Mode switching
CC-Link system master/local module	QJ61BT11N, LJ61BT11	Applicable to all versions
	QJ61BT11	The modules that can use the RLPASET instruction are limited. The instruction is applicable to the module of which the function version is B and the first five digits of the serial number are '03042' or higher. ☞ Page 118 Network parameter setting
CC-Link IE Controller Network module	QJ71GP21-SX, QJ71GP21S-SX	Applicable to all versions
CC-Link IE Field Network module	QJ71GF11-T2, LJ71GF11-T2	Applicable to all versions
MELSECNET/H network module	QJ71LP21, QJ71LP21-25, QJ71LP21S-25, QJ71LP21G, QJ71BR11, QJ72LP25-25, QJ72LP25G, QJ72BR15	Applicable to all versions
Ethernet interface module	QJ71E71-100, QJ71E71-B5, QJ71E71-B2, LJ71E71-100	Applicable to all versions
CPU module supporting the PID control instruction	Q00JCPU, Q00UJCPU, Q00CPU, Q00UCPU, Q01CPU, Q01UCPU, Q02CPU, Q02HCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06HCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q12HCPU, Q12PRHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q25HCPU, Q25PRHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU, L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	The modules that can use the instruction are limited. ☞ Page 227 PID Control Instruction (Inexact Differential),Page 240 PID Control Instruction (Exact Differential)
Built-in Ethernet port QCPU, Built-in Ethernet port LCPU (Built-in Ethernet function)	Q03UDVCPU, Q03UDECPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	The modules that can use the socket communication function instruction are limited when using the Built-in Ethernet port QCPU. The instruction is applicable to the module of which the function version is B and the first five digits of the serial number are '11012' or higher. The instruction is applicable to all versions when using the Built-in Ethernet port LCPU.
LCPU (Built-in I/O function)	L02SCPU, L02SCPU-P, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	Applicable to all versions
Data logging function	Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU, L02CPU, L02CPU-P, L06CPU, L06CPU-P, L26CPU, L26CPU-P, L26CPU-BT, L26CPU-PBT	Applicable to all versions

Function/module for explaining an instruction		Applicable version/serial number
CPU module supporting the SFC control instruction	Q02CPU, Q02HCPU, Q02PHCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06HCPU, Q06PHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q12HCPU, Q12PHCPU, Q12PRHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q25HCPU, Q25PHCPU, Q25PRHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU	The modules that can use the instruction are limited. ☞ Page 301 SFC Step Comment Read

Point

- How to check the applicable version or serial number

Intelligent function modules: User's Manual or Reference Manual for the module listed in 'Manuals'

CPU modules supporting PID control: User's Manual (Function Explanation, Program Fundamentals) of the CPU module to be used

Built-in Ethernet port QCPU: QnUCPU User's Manual (Communication via Built-in Ethernet Port)

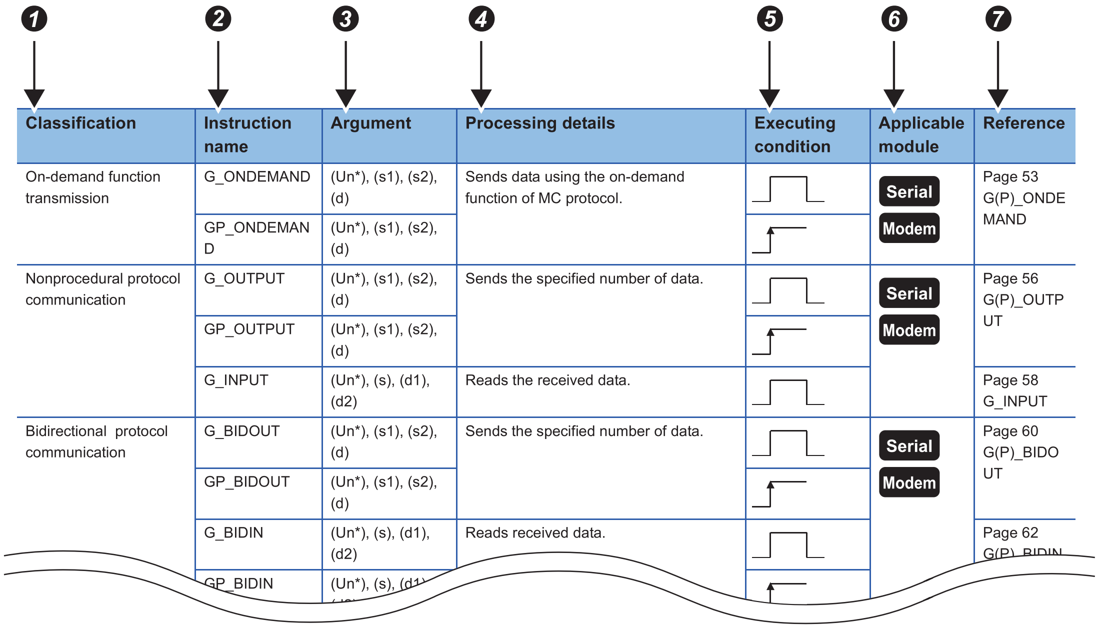
- Manual for reference

☞ MANUALS

2 INSTRUCTION TABLES

2.1 How to Read Instruction Tables

The instruction tables found from Page 16 Module Dedicated Instruction to Page 28 SFC Control Instruction have been made according to the following format:



Description

❶Classifies instructions by application.

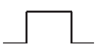
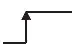
❷Indicates the instructions used in a program.

❸Indicates the arguments of the instruction.

Symbol	Name	Description
(s), (s1)	Source	Stores data before operation.
(d), (d1)	Destination	Indicates the destination of data after operation.
n, n1	—	Specifies the number of devices and the number of transfers.
(Jn*)	—	Specifies the network number.
(Un*)	—	Specifies the start I/O number of a module.

❹Indicates the processing details of each instruction.

❺Details of executing condition of each instruction are as follows:

Symbol	Executing condition
	Indicates an 'executed while ON' type instruction that is executed only while the precondition is ON. When the precondition is OFF, the instruction is not executed and does not perform processing.
	Indicates an 'executed once at ON' type instruction that is executed only at the rising pulse (OFF → ON) of the precondition of the instruction.

❻Indicates the execution target module of each instruction. For details of the icons, refer to Chapter Page 31 HOW TO READ INSTRUCTIONS.

❼Indicates the references on which the instructions are explained.

2.2 Module Dedicated Instruction

Analog instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Mode switching	G_OFFGAN	(Un*), (s)	• Moves to the offset/gain setting mode. • Moves to the normal mode.		Page 33 G(P)_OFFGAN
	GP_OFFGAN				
Setting value reading	G_OGLOAD	(Un*), (s), (d)	• Reads the user range settings offset/gain value to the programmable controller CPU.		Page 35 G(P)_OGLOAD
	GP_OGLOAD				
Setting value restoration	G_OGSTOR	(Un*), (s), (d)	• Restores the user range settings offset/gain value stored in the programmable controller CPU.		Page 37 G(P)_OGSTOR
	GP_OGSTOR				

Positioning instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Absolute position restoration	Z_ABRST1	(Un*), (s), (d)	Restores the absolute position of the specified axis.		Page 39 Z_ABRST1, Z_ABRST2, Z_ABRST3, Z_ABRST4
	Z_ABRST2	(Un*), (s), (d)			
	Z_ABRST3	(Un*), (s), (d)			
	Z_ABRST4	(Un*), (s), (d)			
Positioning start	ZP_PSTRT1	(Un*), (s), (d)	Starts positioning of the specified axis.		Page 43 ZP_PSTRT1, ZP_PSTRT2, ZP_PSTRT3, ZP_PSTRT4
	ZP_PSTRT2	(Un*), (s), (d)			
	ZP_PSTRT3	(Un*), (s), (d)			
	ZP_PSTRT4	(Un*), (s), (d)			
Teaching	ZP_TEACH1	(Un*), (s), (d)	Performs teaching for the specified axis.		Page 45 ZP_TEACH1, ZP_TEACH2, ZP_TEACH3, ZP_TEACH4
	ZP_TEACH2	(Un*), (s), (d)			
	ZP_TEACH3	(Un*), (s), (d)			
	ZP_TEACH4	(Un*), (s), (d)			
Writing to flash ROM	ZP_PFWRT	(Un*), (s), (d)	Writes the positioning module parameters, positioning data, and block start data to the flash ROM.		Page 47 ZP_PFWRT
Setting data initialization	ZP_PINIT	(Un*), (s), (d)	Initializes the positioning module setting data.		Page 49 ZP_PINIT

Serial communication instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Applicable module	Reference
On-demand function transmission	G_ONDEMAND	(Un*), (s1), (s2), (d)	Sends data using the on-demand function of MC protocol.		Serial Modem	Page 51 G(P)_ONDEMAND
	GP_ONDEMAND	(Un*), (s1), (s2), (d)				
Nonprocedural protocol communication	G_OUTPUT	(Un*), (s1), (s2), (d)	Sends the specified number of data.		Serial Modem	Page 54 G(P)_OUTPUT
	GP_OUTPUT	(Un*), (s1), (s2), (d)				
	G_INPUT	(Un*), (s), (d1), (d2)	Reads the received data.			Page 57 G_INPUT
Bidirectional protocol communication	G_BIDOUT	(Un*), (s1), (s2), (d)	Sends the specified number of data.		Serial Modem	Page 59 G(P)_BIDOUT
	GP_BIDOUT	(Un*), (s1), (s2), (d)				
	G_BIDIN	(Un*), (s), (d1), (d2)	Reads received data.			Page 61 G(P)_BIDIN
	GP_BIDIN	(Un*), (s), (d1), (d2)				
Communication status check	G_SPBUSY	(Un*), (d)	Reads the data transmission/reception status using the instruction.		Serial Modem	Page 63 G(P)_SPBUSY
	GP_SPBUSY	(Un*), (d)				
Receive data clear	ZP_CSET	(Un*), (s1), (s2), (d1), (d2)	Clears receive data without stopping transmission using the nonprocedural protocol.		Serial Modem	Page 64 ZP_CSET
Data transmission/ reception	Z_BUFRCVS	(Un*), (s), (d)	Receives data with an interrupt program using the nonprocedural protocol or bidirectional protocol.		Serial Modem	Page 66 Z_BUFRCVS
	G_PRR	(Un*), (s), (d)	Sends data by user frame according to the specification in user frame specification area for transmission using the nonprocedural protocol.			Page 68 G(P)_PRR
	GP_PRR	(Un*), (s), (d)				
Initial setting	ZP_CSET	(Un*), (s1), (s2), (d1), (d2)	Sets the unit (word/byte) of the number of the data to be sent or received.		Serial Modem	Page 72 ZP_CSET
Programmable controller CPU monitor	ZP_CSET	(Un*), (s1), (s2), (d1), (d2)	Registers and cancels the programmable controller CPU monitoring for using the programmable controller CPU monitoring function.		Serial Modem	Page 75 ZP_CSET
Flash ROM user frame registration/reading	G_PUTE	(Un*), (s1), (s2), (d)	Registers a user frames to the flash ROM.		Serial Modem	Page 83 G(P)_PUTE
	GP_PUTE	(Un*), (s1), (s2), (d)				
	G_GETE	(Un*), (s1), (s2), (d)	Reads a user frames from the flash ROM.			Page 86 G(P)_GETE
	GP_GETE	(Un*), (s1), (s2), (d)				
Mode switching	ZP_UINI	(Un*), (s), (d)	Switches the mode, transmission specification, and host station number.		Serial	Page 89 ZP_UINI
Pre-defined protocol communication	G_CPRTCL	(Un*), n1, n2, (s), (d)	Executes the protocols and functional protocols written to the flash ROM.		Serial	Page 95 G(P)_CPRTCL
	GP_CPRTCL	(Un*), n1, n2, (s), (d)				

Network dedicated instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Applicable module	Reference
Reading from the buffer memory of an intelligent device station	J_RIRD	(Jn*), (s), (d1), (d2)	Reads data for the specified number of points from the buffer memory or device of the specified station.		CC IE C	Page 98 J(P)_RIRD, G(P)_RIRD
	JP_RIRD	(Jn*), (s), (d1), (d2)			CC IE C CC IE F	
	G_RIRD	(Un*), (s), (d1), (d2)			CC-Link CC IE C	
	GP_RIRD	(Un*), (s), (d1), (d2)			CC-Link CC IE C CC IE F	
Writing to the buffer memory of an intelligent device station	J_RIWT	(Jn*), (s1), (s2), (d)	Writes data for the specified number of points to the buffer memory or device of the specified station.		CC IE C	Page 102 J(P)_RIWT, G(P)_RIWT
	JP_RIWT	(Jn*), (s1), (s2), (d)			CC IE C CC IE F	
	G_RIWT	(Un*), (s1), (s2), (d)			CC-Link CC IE C	
	GP_RIWT	(Un*), (s1), (s2), (d)			CC-Link CC IE C CC IE F	
Reading from the buffer memory of an intelligent device station (with handshake)	G_RIRCV	(Un*), (s1), (s2), (d1), (d2)	Automatically performs handshaking with the specified station and reads data from the buffer memory of the specified station. This instruction is applicable with a module having a handshake signal, such as the AJ65BT-R2(N).		CC-Link	Page 106 G(P)_RIRCV
	GP_RIRCV	(Un*), (s1), (s2), (d1), (d2)				
Writing to the buffer memory of an intelligent device station (with handshake)	G_RISEND	(Un*), (s1), (s2), (d1), (d2)	Automatically performs handshaking with the specified station and writes data to the buffer memory of the specified station. This instruction is applicable with a module having a handshake signal, such as the AJ65BT-R2(N).		CC-Link	Page 110 G(P)_RISEN D
	GP_RISEND	(Un*), (s1), (s2), (d1), (d2)				
Reading from the auto-refresh buffer memory of the master station	G_RIFR	(Un*), n1, n2, n3, (d)	Reads data from the auto-refresh buffer memory of the specified station. This instruction is applicable with a module having an auto-refresh buffer, such as the AJ65BT-R2.		CC-Link	Page 114 G(P)_RIFR
	GP_RIFR	(Un*), n1, n2, n3, (d)				
Writing to the auto-refresh buffer memory of the master station	G_RITO	(Un*), n1, n2, n3, (d)	Writes data to the auto-refresh buffer memory of the specified station. This instruction is applicable with a module having an auto-refresh buffer, such as the AJ65BT-R2.		CC-Link	Page 116 G(P)_RITO
	GP_RITO	(Un*), n1, n2, n3, (d)				
Network parameter setting	G_RLPASET	(Un*), (s1), (s2), (s3), (s4), (s5), (d)	Sets network parameter to the master station and starts up the data link.		CC-Link	Page 118 G(P)_RLPAS ET
	GP_RLPASET	(Un*), (s1), (s2), (s3), (s4), (s5), (d)				

Classification	Instruction name	Argument	Processing details	Executing condition	Applicable module	Reference
Device data read/write	J_READ	(Jn*), (s1), (s2), (d1), (d2)	Reads data from a word device of another station.		CC IE C CC IE F NET/H Ether	Page 125 J(P)_READ, G(P)_READ
	JP_READ	(Jn*), (s1), (s2), (d1), (d2)				
	G_READ	(Un*), (s1), (s2), (d1), (d2)				
	GP_READ	(Un*), (s1), (s2), (d1), (d2)				
	J_SREAD	(Jn*), (s1), (s2), (d1), (d2), (d3)	Reads data from a device of another station (with completion device).			Page 130 J(P)_SREAD , G(P)_SREA D
	JP_SREAD	(Jn*), (s1), (s2), (d1), (d2), (d3)				
	G_SREAD	(Un*), (s1), (s2), (d1), (d2), (d3)				
	GP_SREAD	(Un*), (s1), (s2), (d1), (d2), (d3)				
	J_WRITE	(Jn*), (s1), (s2), (s3), (d1)	Writes data to a device of another station.			Page 134 J(P)_WRITE, G(P)_WRITE
	JP_WRITE	(Jn*), (s1), (s2), (s3), (d1)				
	G_WRITE	(Un*), (s1), (s2), (s3), (d1)				
	GP_WRITE	(Un*), (s1), (s2), (s3), (d1)				
	J_SWRITE	(Jn*), (s1), (s2), (d1), (d2), (d3)	Writes data to a device of another station (with completion device).			Page 140 J(P)_SWRIT E, G(P)_SWRIT E
	JP_SWRITE	(Jn*), (s1), (s2), (d1), (d2), (d3)				
	G_SWRITE	(Un*), (s1), (s2), (d1), (d2), (d3)				
	GP_SWRITE	(Un*), (s1), (s2), (d1), (d2), (d3)				
Message (user-specified data) communication	J_SEND	(Jn*), (s1), (s2), (d)	Sends data to another station.		CC IE C CC IE F NET/H Ether	Page 144 J(P)_SEND, G(P)_SEND
	JP_SEND	(Jn*), (s1), (s2), (d)				
	G_SEND	(Un*), (s1), (s2), (d)				
	GP_SEND	(Un*), (s1), (s2), (d)				
	J_RECV	(Jn*), (s), (d1), (d2)	Reads received data from another station (for main program).			Page 150 J(P)_RECV, G(P)_RECV
	JP_RECV	(Jn*), (s), (d1), (d2)				
	G_RECV	(Un*), (s), (d1), (d2)				
	GP_RECV	(Un*), (s), (d1), (d2)				
	Z_RECVS	(Un*), (s1), (s2), (d)	Reads received data from another station (for interrupt program)			Page 154 Z_RECVS

Classification	Instruction name	Argument	Processing details	Executing condition	Applicable module	Reference
Transient request to another station	J_REQ	(Jn*), (s1), (s2), (d1), (d2)	Executes remote RUN/STOP for another station. Reads/writes clock data from another station.		CC IE C	Page 157 J(P)_REQ, G(P)_REQ
	JP_REQ	(Jn*), (s1), (s2), (d1), (d2)			CC IE C CC IE F NET/H Ether	
	G_REQ	(Un*), (s1), (s2), (d1), (d2)			CC IE C	
	GP_REQ	(Un*), (s1), (s2), (d1), (d2)			CC IE C CC IE F NET/H Ether	
Read from other station devices	J_ZNRD	(Un*), n1, (s), n2, (d1), (d2)	Reads data from a device of a programmable controller on another station. (In units of words)		CC IE C NET/H Ether	Page 165 J(P)_ZNRD
	JP_ZNRD	(Un*), n1, (s), n2, (d1), (d2)				
Write to other station devices	J_ZNWR	(Un*), n1, (s), n2, (d1), (d2)	Writes data to a device of a programmable controller on another station. (In units of words)			Page 168 J(P)_ZNWR
	JP_ZNWR	(Un*), n1, (s), n2, (d1), (d2)				
Remote RUN	Z_RRUN_J	(Jn*), (s1), (s2), (s3), (s4), (d)	Executes remote RUN for a CPU module on another station.		CC IE C NET/H	Page 171 Z(P)_RRUN_J, Z(P)_RRUN_U
	ZP_RRUN_J	(Jn*), (s1), (s2), (s3), (s4), (d)				
	Z_RRUN_U	(Un*), (s1), (s2), (s3), (s4), (d)				
	ZP_RRUN_U	(Un*), (s1), (s2), (s3), (s4), (d)				
Remote STOP	Z_RSTOP_J	(Jn*), (s1), (s2), (s3), (s4), (d)	Executes remote STOP for a CPU module on another station.		CC IE C NET/H	Page 174 Z(P)_RSTOP_J, Z(P)_RSTOP_U
	ZP_RSTOP_J	(Jn*), (s1), (s2), (s3), (s4), (d)				
	Z_RSTOP_U	(Un*), (s1), (s2), (s3), (s4), (d)				
	ZP_RSTOP_U	(Un*), (s1), (s2), (s3), (s4), (d)				
Reading clock data from another station	Z_RTMRD_J	(Jn*), (s1), (s2), (s3), (d1), (d2)	Reads clock data from a CPU module on another station.		CC IE C NET/H	Page 177 Z(P)_RTMRD_J, Z(P)_RTMRD_U
	ZP_RTMRD_J	(Jn*), (s1), (s2), (s3), (d1), (d2)				
	Z_RTMRD_U	(Un*), (s1), (s2), (s3), (d1), (d2)				
	ZP_RTMRD_U	(Un*), (s1), (s2), (s3), (d1), (d2)				
Writing clock data to another station	Z_RTMWR_J	(Jn*), (s1), (s2), (s3), (s4), (d)	Writes clock data to a CPU module on another station.		CC IE C NET/H	Page 179 Z(P)_RTMWR_J, Z(P)_RTMWR_U
	ZP_RTMWR_J	(Jn*), (s1), (s2), (s3), (s4), (d)				
	Z_RTMWR_U	(Un*), (s1), (s2), (s3), (s4), (d)				
	ZP_RTMWR_U	(Un*), (s1), (s2), (s3), (s4), (d)				

Classification	Instruction name	Argument	Processing details	Executing condition	Applicable module	Reference
Reading from buffer memory of intelligent function module on remote I/O station	Z_REMFR	(Jn*), n1, n2, n3, n4, n5, (d1), (d2)	Reads data from the buffer memory of an intelligent function module on the remote I/O station.		NET/H	Page 182 Z(P)_REMFR
	ZP_REMFR	(Jn*), n1, n2, n3, n4, n5, (d1), (d2)			CC IE F NET/H	
Writing to buffer memory of intelligent function module on remote I/O station	Z_REMTO	(Jn*), n1, n2, n3, n4, n5, (d1), (d2)	Writes data to the buffer memory of an intelligent function module on the remote I/O station.		NET/H	Page 185 Z(P)_REMTO
	ZP_REMTO	(Jn*), n1, n2, n3, n4, n5, (d1), (d2)			CC IE F NET/H	
Setting parameter	G_CCPASET	(Un*), (s1), (s2), (s3), (s4), (d)	Set parameters for master/local modules (master station).		CC IE F	Page 188 G(P)_CCPASET
	GP_CCPASET	(Un*), (s1), (s2), (s3), (s4), (d)				
Connection opening or closing	ZP_OPEN	(Un*), (s1), (s2), (d)	Opens a connection.		Ether	Page 194 ZP_OPEN
	ZP_CLOSE	(Un*), (s1), (s2), (d)	Closes a connection.			Page 198 ZP_CLOSE
Fixed buffer communication	ZP_BUFRCV	(Un*), (s1), (s2), (d1), (d2)	Reads received data. (for main program).		Ether	Page 201 ZP_BUFRCV
	Z_BUFRCVS	(Un*), (s), (d)	Reads received data. (for interrupt program)			Page 204 Z_BUFRCVS
	ZP_BUFSND	(Un*), (s1), (s2), (s3), (d)	Sends data.			Page 206 ZP_BUFSND
Reading or clearing error information	ZP_ERRCLR	(Un*), (s), (d)	Clears error information.		Ether	Page 209 ZP_ERRCLR
	ZP_ERRRD	(Un*), (s), (d)	Reads error information.			Page 212 ZP_ERRRD
Re-initialization/station number setting/changing switch setting	Z_UINI	(Un*), (s), (d)	• Executes re-initialization. • Sets the host station number. • Changes the switch setting.		CC IE C	Page 214 Z(P)_UINI
	ZP_UINI	(Un*), (s), (d)			CC IE C Ether	
E-mail communication	ZP_MRECV	(Un*), (s), (d1), (d2)	Reads received e-mail.		Ether	Page 218 ZP_MRECV
	ZP_MSEND	(Un*), (s1), (s2), (d)	Sends an e-mail.			Page 221 ZP_MSEND

2.3 PID Control Instruction

PID control instruction (inexact differential)

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Data setting	S_PIDINIT	(s)	Sets data to be used for PID operation.		Page 227 S(P)_PIDINIT
	SP_PIDINIT	(s)			
PID operation	S_PIDCONT	(s)	Performs PID operation based on the set value (SV) and process value (PV).		Page 232 S(P)_PIDCONT
	SP_PIDCONT	(s)			
PID operation stop	S_PIDSTOP	n	Stops the PID operation for the specified loop number.		Page 236 S_PIDSTOP, S_PIDRUN
	SP_PIDSTOP	n			
PID operation start	S_PIDRUN	n	Starts the PID operation for the specified loop number.		
	SP_PIDRUN	n			
Operation parameter change	S_PIDPRMW	n, (s)	Changes operation parameter of the specified loop number.		Page 237 S(P)_PIDPRMW
	SP_PIDPRMW	n, (s)			

PID control instruction (exact differential)

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Data setting	PIDINIT	(s)	Sets data to be used for PID operation.		Page 240 PIDINIT(P)
	PIDINITP	(s)			
PID operation	PIDCONT	(s)	Performs PID operation based on the set value (SV) and process value (PV).		Page 244 PIDCONT(P)
	PIDCONTP	(s)			
PID operation stop	PIDSTOP	n	Stops the PID operation for the specified loop number.		Page 248 PIDSTOP, PIDRUN
	PIDSTOPP	n			
PID operation start	PIDRUN	n	Starts the PID operation for the specified loop number.		
	PIDRUNP	n			
Operation parameter change	PIDPRMW	n, (s)	Changes operation parameter of the specified loop number.		Page 249 PIDPRMW(P)
	PIDPRMWP	n, (s)			

2.4 Socket Communication Function Instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Opening/closing connection	SP_SOCOPEN	(Un*), (s1), (s2), (d)	Establishes a connection.		Page 252 SP_SOCOPEN
	SP_SOCCLOSE	(Un*), (s1), (s2), (d)	Shuts a connection off.		Page 255 SP_SOCCLOSE
Reading receive data	SP_SOCRCV	(Un*), (s1), (s2), (d1), (d2)	Reads receive data. (Reading at the end process)		Page 257 SP_SOCRCV
	S_SOCRCVS	(Un*), (s), (d)	Reads receive data. (Reading at the instruction execution)		Page 259 S_SOCRCVS
Sending data	SP_SOCSND	(Un*), (s1), (s2), (s3), (d)	Sends data.		Page 261 SP_SOCSND
Reading connection information	SP_SOCCINF	(Un*), (s1), (s2), (d)	Reads connection information.		Page 264 SP_SOCCINF
Changing destination	SP_SOCCSET	(Un*), (s1), (s2)	Changes a destination of a UDP/IP connection.		Page 266 SP_SOCCSET
Changing receive mode	SP_SOCRMODE	(Un*), (s1), (s2)	Changes the receive mode of a connection.		Page 268 SP_SOCRMODE
Reads data from the receive data area.	S_SOCRDATA	(Un*), (s1), (s2), n, (d)	Reads data from the receive data area.		Page 270 S(P)_SOCRDATA
	SP_SOCRDATA				

2.5 Built-in I/O Function Instruction

Positioning function dedicated instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Positioning start	IPPSTRT1	n	Specifies a data number to be executed from "Positioning Data" No. 1 to No. 10 which are previously set in GX Works2, and starts the positioning.		Page 272 IPPSTRT1, IPPSTRT2
	IPPSTRT1P	n			
	IPPSTRT2	n			
	IPPSTRT2P	n			
	IPDSTRT1	(s)	Regardless of "Positioning Data" No. 1 to No. 10 which are previously set in GX Works2, starts the positioning using the data stored in the devices starting from the one specified for control data.		Page 273 IPDSTRT1, IPDSTRT2
	IPDSTRT1P	(s)			
	IPDSTRT2	(s)			
	IPDSTRT2P	(s)			
	IPSIMUL	n1, n2	Starts the positioning of the axis 1 "Positioning Data" number and the axis 2 "Positioning Data" number simultaneously.		Page 275 IPSIMUL(P)
	IPSIMULP	n1, n2			
OPR start	IPOPR1	(s)	Specifies a method and starts the OPR of the specified axis.		Page 276 IPOPR1, IPOPR2
	IPOPR1P	(s)			
	IPOPR2	(s)			
	IPOPR2P	(s)			
JOG start	IPJOG1	(s1), (s2)	Starts the JOG operation of the specified axis.		Page 278 IPJOG1, IPJOG2
	IPJOG2	(s1), (s2)			
Absolute position restoration	IPABRST1	(s), (d)	Executes the absolute position restoration of the specified axis.		Page 280 IPABRST1, IPABRST2
	IPABRST2	(s), (d)			
Stop	IPSTOP1	—	Stops the axis in operation.		Page 282 IPSTOP1, IPSTOP2
	IPSTOP2	—			
Speed change	IPSPCHG1	(s)	Changes the speed of the specified axis.		Page 283 IPSPCHG1, IPSPCHG2
	IPSPCHG1P	(s)			
	IPSPCHG2	(s)			
	IPSPCHG2P	(s)			
Target position change	IPTPCHG1	(s)	Changes the target position of the specified axis.		Page 285 IPTPCHG1, IPTPCHG2
	IPTPCHG1P	(s)			
	IPTPCHG2	(s)			
	IPTPCHG2P	(s)			

Counter function dedicated instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Current value read	ICCNTRD1	—	Stores the most recent value for the current value of the specified CH.		Page 287 ICCNTRD1, ICCNTRD2
	ICCNTRD1P	—			
	ICCNTRD2	—			
	ICCNTRD2P	—			
Ring counter upper/lower limit value write	ICRNGWR1	(s1), (s2)	Sets a ring counter lower limit value and upper limit value of the specified CH.		Page 288 ICRNGWR1, ICRNGWR2
	ICRNGWR1P	(s1), (s2)			
	ICRNGWR2	(s1), (s2)			
	ICRNGWR2P	(s1), (s2)			
Preset value write	ICPREWR1	(s)	Sets a preset value of the specified CH.		Page 290 ICPREWR1, ICPREWR2
	ICPREWR1P	(s)			
	ICPREWR2	(s)			
	ICPREWR2P	(s)			
Latch counter value read	ICLTHRD1	n, (d)	Stores a latch counter value of the specified CH.		Page 291 ICLTHRD1, ICLTHRD2
	ICLTHRD1P	n, (d)			
	ICLTHRD2	n, (d)			
	ICLTHRD2P	n, (d)			
Sampling counter value read	ICSMPRD1	(d)	Stores a sampling counter value of the specified CH.		Page 292 ICSMPRD1, ICSMPRD2
	ICSMPRD1P	(d)			
	ICSMPRD2	(d)			
	ICSMPRD2P	(d)			
Coincidence output point write	ICCOVWR1	n, (s)	Sets a coincidence output No. n point of the specified CH.		Page 293 ICCOVWR1, ICCOVWR2
	ICCOVWR1P	n, (s)			
	ICCOVWR2	n, (s)			
	ICCOVWR2P	n, (s)			
Frequency measurement	ICFCNT1	(d)	Measures the frequency of the specified CH.		Page 294 ICFCNT1, ICFCNT2
	ICFCNT2	(d)			
Rotation speed measurement	ICRCNT1	(d)	Measures the rotation speed of the specified CH.		Page 295 ICRCNT1, ICRCNT2
	ICRCNT2	(d)			

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Pulse measurement read	ICPLSRD1	(d)	Stores the measured pulse value of the specified CH.		Page 296 ICPLSRD1, ICPLSRD2
	ICPLSRD1P	(d)			
	ICPLSRD2	(d)			
	ICPLSRD2P	(d)			
PWM output	ICPWM1	(s1), (s2)	Outputs the PWM waveform of the specified CH.		Page 297 ICPWM1, ICPWM2
	ICPWM2	(s1), (s2)			

2.6 Data Logging Function Instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
Trigger logging set/reset	LOGTRG	n	Generates the trigger conditions in a trigger logging. Stores the data sampling results to the data logging file for the number of times specified in the trigger logging configuration of the programming tool.		Page 299 LOGTRG Instruction, LOGTRGR Instruction
	LOGTRGR	n	Resets the trigger conditions		

2.7 SFC Control Instruction

Classification	Instruction name	Argument	Processing details	Executing condition	Reference
SFC step comment read	S_SFCSCOMR	n1, n2, n3, (d1), (d2)	Reads comment of an active step in the specified SFC block by the specified number.		Page 301 S(P)_SFCSCOMR
	SP_SFCSCOMR	n1, n2, n3, (d1), (d2)			
SFC transition condition comment read	S_SFCTCOMR	n1, n2, n3, (d1), (d2)	Reads comment of transition condition associated with an active step in the specified SFC block by the specified number.		Page 303 S(P)_SFCTCOMR
	SP_SFCTCOMR	n1, n2, n3, (d1), (d2)			

3 CONFIGURATION OF INSTRUCTIONS

Instructions available in the CPU module can be divided into an instruction name and an argument.
The application of an instruction name and an argument are as follows:

- Instruction name

Indicates the function of the instruction.

- Argument

Indicates the I/O data used in the instruction.
Arguments are classified into I/O number, source data, destination data, number of devices, executing condition, and execution result.

3

I/O number

I/O number is data that set a module in which the instruction is to be executed.
Set the I/O number by start I/O number or a network number of the module depending on the instruction.

■Setting the start I/O number (Un) of the module

Set the higher two digits when expressing the start I/O number in three digits for the module in which the instruction is to be executed. Set the start I/O number in a numeric value or character string according to the data type available with the instruction.

- Setting the start I/O number in word (unsigned)/16-bit string or word (signed) data type

Set the start I/O number of the module for 'n' of 'Un'.

Ex.

For the module whose start I/O number is 020H: 02

- Setting the start I/O number in string data type

Set the start I/O number in the format of "Un" (n: start I/O number of the module).

Ex.

For the module whose start I/O number is 020H: "02"

■Network number (Jn) setting

Set the network number of the network module/Ethernet module in which the instruction is to be executed. Set a network number indicated below, in word (unsigned)/16-bit string or word (signed) data type, for 'n' of 'Jn'.

- 1 to 239: Network number
- 254: Network specified in "Valid module during other station access" on the GX Works2 network parameter screen

Ex.

When the network number is 1: 1

Source (s)

A source is data used in an operation.

The following source types are available depending on the device specified in an instruction:

Type	Description
Constant	Specifies a numeric value used in an operation. Constants are set during programming so that they cannot be changed while the program is being executed. Perform index modification when using them as variable data.
Bit device and word device	Specifies the device in which the data used in the operation are stored. Data must be stored to the specified device before executing the operation. By changing the data to be stored to the specified device while a program is being executed, the data used in the instruction can be changed.

The instructions explained in this manual use special data. Refer to the explanation for each instruction and use data correctly.

Destination (d)

Data after the operation are stored to a destination.

Set a device in which data are to be stored to a destination.

The instructions explained in this manual use special data. Refer to the explanation for each instruction and use data correctly.

Number of devices and number of transfers (n)

Data such as a channel number, loop number, read data length, and logging setting number are set to the number of devices and number of transfers (n).

Executing condition (EN)

An input variable EN inputs an executing condition of an instruction.

Execution result (ENO)

An output variable ENO outputs an execution result.

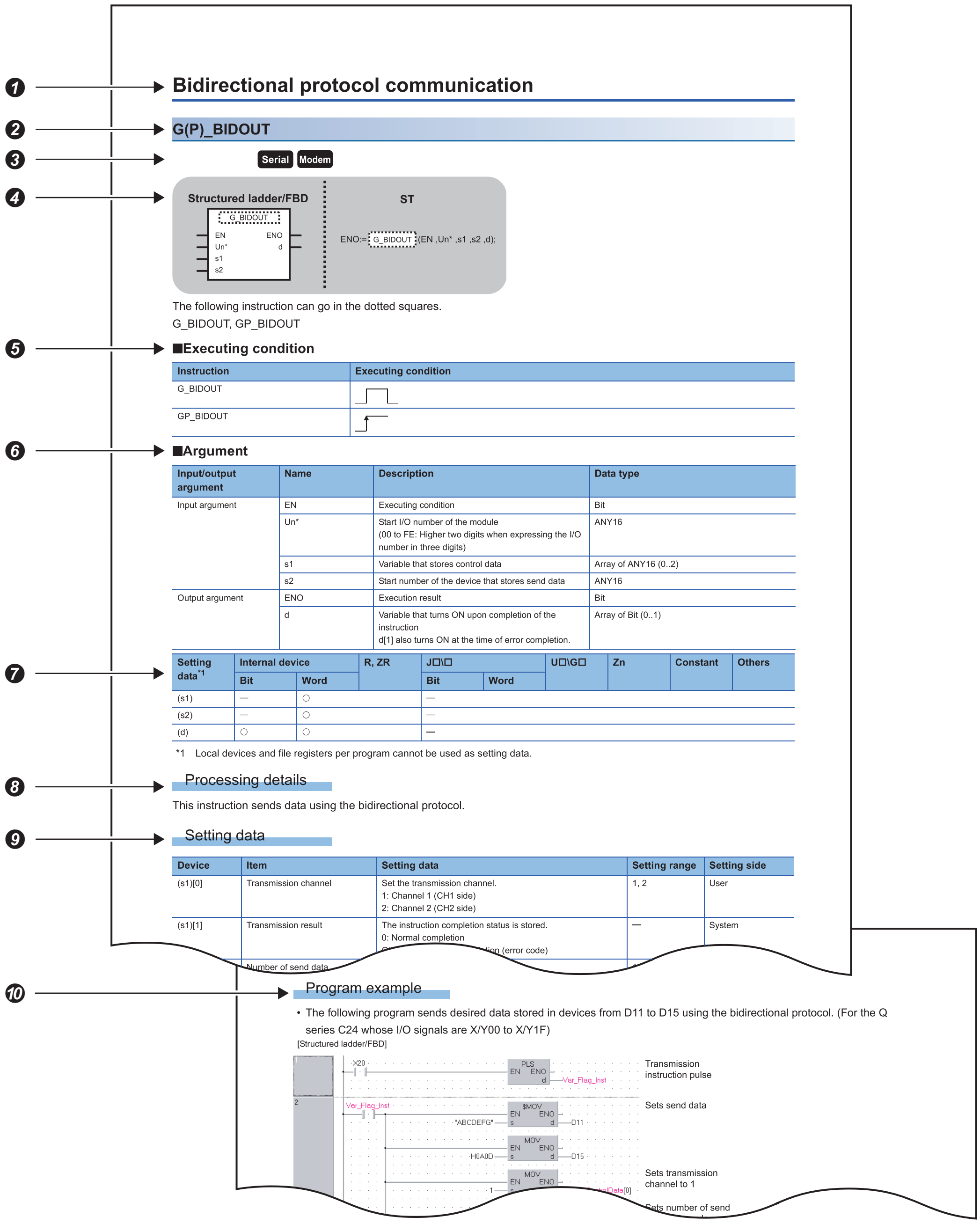
Point

For details of the configuration of instructions for labels and structures, refer to the following.

📖 MELSEC-Q/L/F Structured Programming Manual (Fundamentals)

4 HOW TO READ INSTRUCTIONS

Chapter 5 provides detailed explanation on each instruction in the layout as shown below.



❶ ❷ ❸ ❹ ❺ ❻ ❼ ❽ ❾ ❿

Bidirectional protocol communication

G(P)_BIDOUT

Serial Modem

Structured ladder/FBD | ST

ENO:= G_BIDOUT (EN ,Un* ,s1 ,s2 ,d);

The following instruction can go in the dotted squares.
G_BIDOUT, GP_BIDOUT

■Executing condition

Instruction	Executing condition
G_BIDOUT	
GP_BIDOUT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 (0..2)
	s2	Start number of the device that stores send data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of Bit (0..1)

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data using the bidirectional protocol.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Transmission channel	Set the transmission channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s1)[1]	Transmission result	The instruction completion status is stored. 0: Normal completion ...ion (error code)	—	System
	Number of send data			

Program example

- The following program sends desired data stored in devices from D11 to D15 using the bidirectional protocol. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]

❶Indicates an outline of an instruction.

❷Indicates an instruction to be explained.

❸Indicates the instruction execution target module. If one instruction is to be executed in two or more modules, applicable modules are indicated using icons.

Module	Icon	Module	Icon
Serial communication	Serial	Built-in Ethernet port QCPU	QnUDE(H)
Modem interface	Modem	High-speed Universal model QCPU, Universal model Process CPU	QnUDV
CC-Link	CC-Link	LCPU	LCPU
CC-Link IE Controller Network	CC IE C	Universal model QCPU	Universal
CC-Link IE Field Network	CC IE F	High Performance model QCPU	High performance
MELSECNET/H	NET/H	Process CPU	Process
Ethernet	Ether	Redundant CPU	Redundant

❹Written formats in the structured ladder/FBD and structured text language.

❺Indicates the instruction name and executing condition of the instruction.

Executing condition	Non-conditional execution	Executed while ON	Executed once at ON	Executed while OFF	Executed once at OFF
Symbols on the corresponding page	No symbol				

❻Indicates the names of input and output arguments, and the data type of each argument. For details of each data type, refer to the following.

MELSEC-Q/L/F Structured Programming Manual (Fundamentals)

❼Devices that can be used in the instruction are marked with ○.

The following table shows applicable classification for usable devices.

Device classification	Internal device (system, user)		File register R, ZR	Link direct device J□\□*4*6		Intelligent function module U□\G□	Index register Zn	Constant*5	Others *5
	Bit	Word		Bit	Word				
Usable device *1	X, Y, M, L, SM, F, B, SB, FX, FY*2*2	T*3, ST*3, C*3, D, W, SD, SW, FD*2, @□	R, ZR	J□\X J□\Y J□\B J□\SB	J□\W J□\SW	U□\□G	Z	K, H, E, $,	P, I, J, U, DX, DY, N, BL, TR, BL\S, V

*1 For description of each device, refer to the User's Manual (Function Explanation, Program Fundamentals) of the CPU module currently being used.

*2 FX and FY can be used in bit data only, and FD can be used in word data only in the PID control instruction.

*3 T, ST, and C can be used in word data only (cannot be used in bit data).

*4 These devices can be used in CC-Link IE, MELSECNET/H, and MELSECNET/10.

*5 The Constant and Others columns describe settable devices.

*6 Link direct devices (J□\□) cannot be used for LCPU.

❽Indicates the processing performed by the instruction.

❾Indicates data such as control data, send data or receive data, that are used for an input argument or output argument in an instruction.

The setting side indicates the following:

- User : Data set by user before dedicated instruction execution
- System : Data stored by the programmable controller CPU after dedicated instruction execution. The setting does not need to be set by the user. If the setting is set by the user, data cannot be read normally.

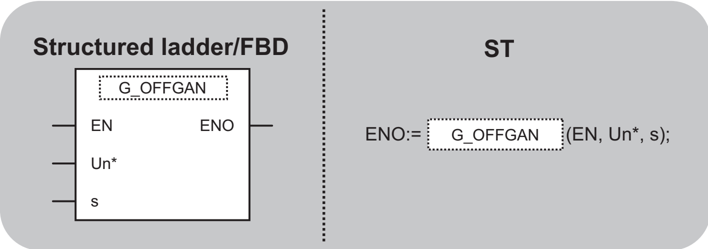
❿Indicates the program examples of structured ladder/FBD/ST.

5 MODULE DEDICATED INSTRUCTION

5.1 Analog Instruction

OFFGAN instruction

G(P)_OFFGAN



The following instruction can go in the dotted squares.
G_OFFGAN, GP_OFFGAN

■Executing condition

Instruction	Executing condition
G_OFFGAN	
GP_OFFGAN	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Mode switching 0: To normal mode 1: To offset/gain setting mode	ANY16
Output argument	ENO	Execution result	Bit

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

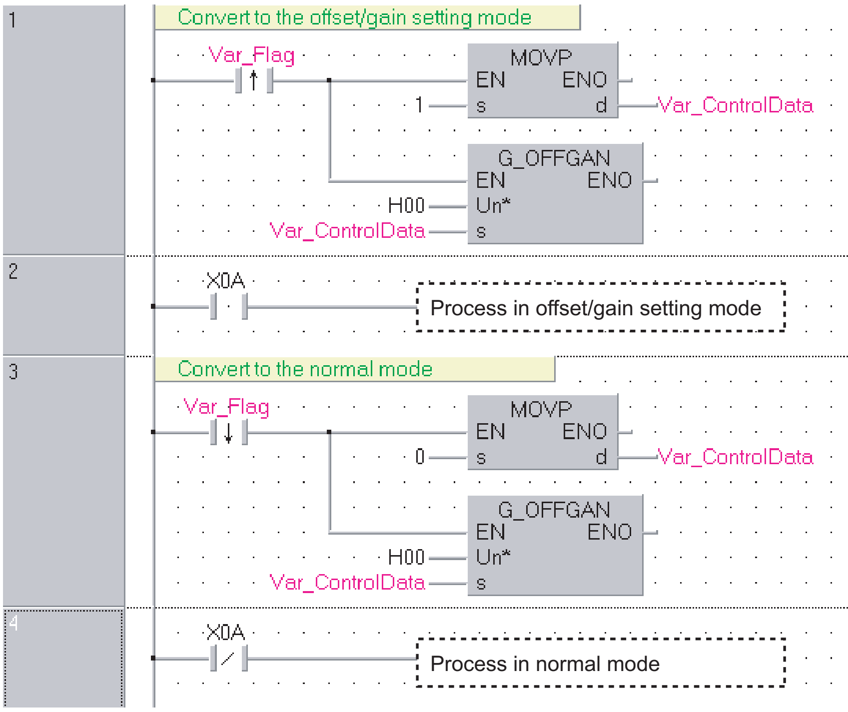
Processing details

This instruction converts the mode of analog modules. (normal mode to offset/gain setting mode, offset/gain setting mode to normal mode)

Program example

- The following program converts the mode of the A/D converter module mounted on the I/O numbers from X/Y00 to X/Y0F to the offset/gain setting mode when Var_Flag turns ON, and gets it back to the normal mode when Var_Flag turns OFF.

[Structured ladder/FBD]



Sets mode when Var_Flag turns ON

Converts mode

Sets mode when Var_Flag turns OFF

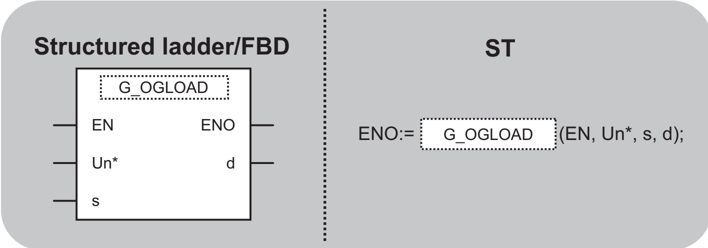
Converts mode

[ST]

```
(* Convert to the offset/gain setting mode *)
IF(Var_Flag=TRUE)THEN (* Var_Flag ON *)
    MOVP(TRUE,1,Var_ControlData); (* Sets mode *)
    G_OFFGAN(TRUE,H00,Var_ControlData); (* Converts mode *)
END_IF;
IF(X0A=TRUE)THEN
 (* Process in offset/gain setting mode *)
END_IF;

(* Convert to the normal mode *)
IF(Var_Flag=FALSE)THEN (* Var_Flag OFF *)
    MOVP(TRUE,0,Var_ControlData); (* Sets mode *)
    G_OFFGAN(TRUE,H00,Var_ControlData); (* Converts mode *)
END_IF;
IF(X0A=FALSE)THEN
 (* Process in normal mode *)
END_IF;
```

Setting value reading

G(P)_OGLOAD



The following instruction can go in the dotted squares.
G_OGLOAD, GP_OGLOAD

■Executing condition

Instruction	Executing condition
G_OGLOAD	
GP_OGLOAD	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..35]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○			—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads the user range settings offset/gain values of analog modules to the CPU.

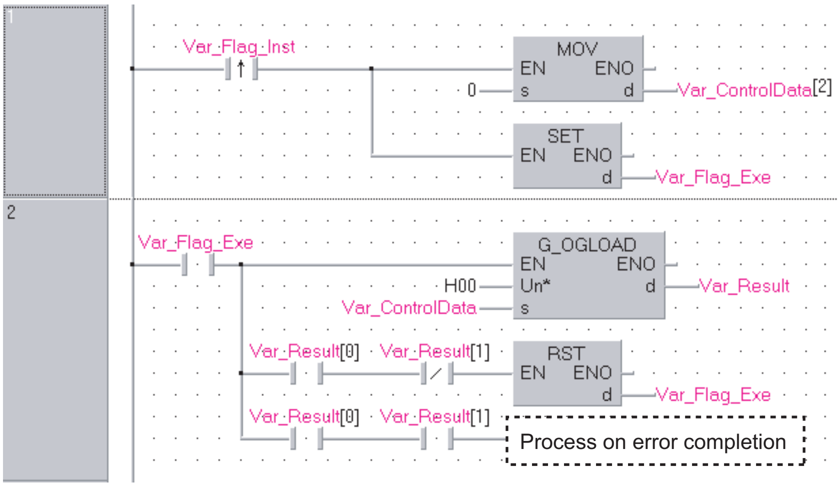
Setting data

For the control data, refer to the manual for each module used.

Program example

- The following program reads out the offset/gain value of the A/D converter module mounted on the I/O numbers from X/Y00 to X/Y0F when the flag turns ON.

[Structured ladder/FBD]



Specifies voltage when instruction flag turns ON

Turns execution flag ON

Performs readout

Normal completion
Turns execution flag OFF

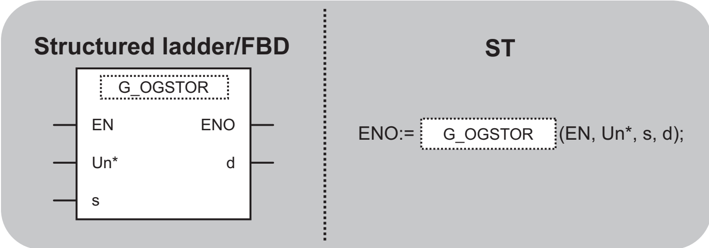
Error completion

[ST]

```
IF(Var_Flag_Inst=TRUE)THEN (* Instruction flag ON *)
 MOV(TRUE,0,Var_ControlData[2]);   (* Specifies voltage *)
    SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;

IF(Var_Flag_Exe=TRUE)THEN (* Execution flag ON *)
    G_OGLOAD(TRUE, H00, Var_ControlData, Var_Result); (* Performs readout *)
    IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
      IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
      ELSE (* Error completion *)
 (* Process on error completion *)
      END_IF;
    END_IF;
END_IF;
```

Setting value restoration

G(P)_OGSTOR



The following instruction can go in the dotted squares.
G_OGSTOR, GP_OGSTOR

■Executing condition

Instruction	Executing condition
G_OGSTOR	
GP_OGSTOR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..35]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○			—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction restores the user range settings offset/gain values stored in the programmable controller CPU to the analog modules.

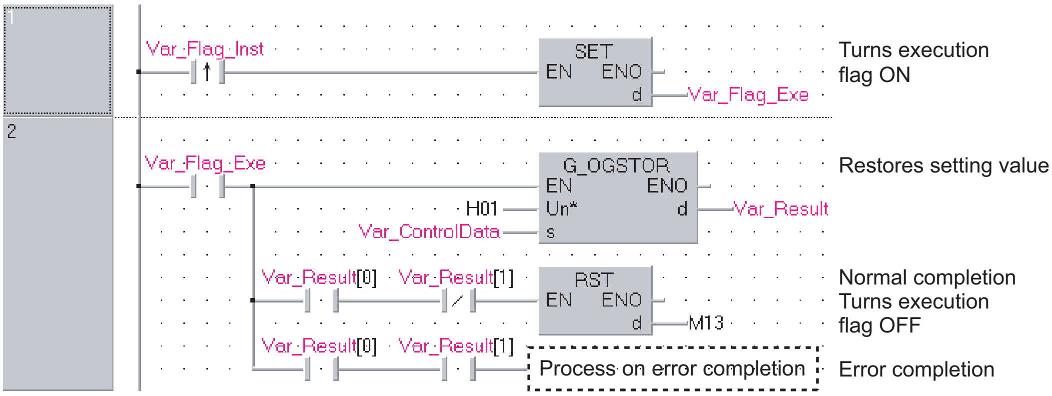
Setting data

For the control data, refer to the manual for each module used.

Program example

- The following program restores the offset/gain setting value to the A/D converter module mounted on the I/O numbers from X/Y10 to X/Y1F when the flag turns ON.

[Structured ladder/FBD]



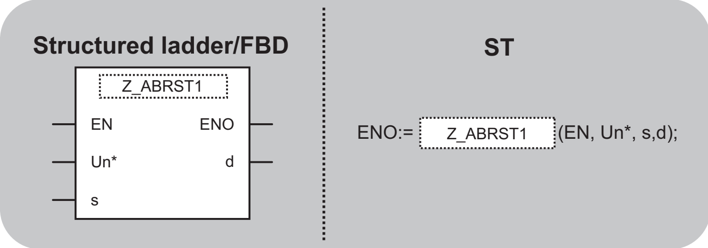
[ST]

```
IF(Var_Flag_Inst=TRUE)THEN (* Instruction flag ON *)
    SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;

IF(Var_Flag_Exe=TRUE)THEN (* Execution flag ON *)
    G_OGSTOR(TRUE, H01, Var_ControlData, Var_Result); (* Restores setting value *)

    IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
      IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
      ELSE (* Error completion *)
(* Process on error completion *)
      END_IF;
    END_IF;
END_IF;
```

5.2 Positioning Instruction

Absolute position restoration

Z_ABRST1, Z_ABRST2, Z_ABRST3, Z_ABRST4



The following instruction can go in the dotted squares.
Z_ABRST1, Z_ABRST2, Z_ABRST3, Z_ABRST4

■Executing condition

Instruction	Executing condition
Z_ABRST1, Z_ABRST2, Z_ABRST3, Z_ABRST4	_⌐¬_

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..7]
Output argument	ENO	Execution result (TRUE: Normal, FALSE: Error)	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○	—	—					

Processing details

This instruction restores the absolute position of the specified axis. (Refer to the following)

- Z_ABRST1: Axis 1
- Z_ABRST2: Axis 2
- Z_ABRST3: Axis 3
- Z_ABRST4: Axis 4

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. • 0: Normal completion • Other than 0: Error completion (error code)	—	System
(s)[2]	Receive signal from servo amplifier	Write the following signal status read from the servo amplifier to the input module. • b0: ABS data bit0 • b1: ABS data bit1 • b2: Send data READY flag	b0: 0/1 b1: 0/1 b2: 0/1	User
(s)[3]	Send signal to servo amplifier	The ON/OFF status of the following data, that are calculated by the dedicated instructions on the basis of "receive signal from servo amplifier" and output to the amplifier, are stored. • b0: Servo ON • b1: ABS transfer mode • b2: ABS request flag	—	System
(s)[4]	Status	Communication status with the servo amplifier • 0: Communication completed(Set by the user at the start of communication) • Other than 0: During communication (Stored by the system.)	0	User/System
(s)[5] ⋮ (s)[7]	System area	—	—	—

Program example

The following program restores the absolute position of the axis 1.

The devices from X47 to X49 and from Y50 to Y52 are used for the communication with the servo amplifier.

X47: ABS data bit0

X48: ABS data bit1

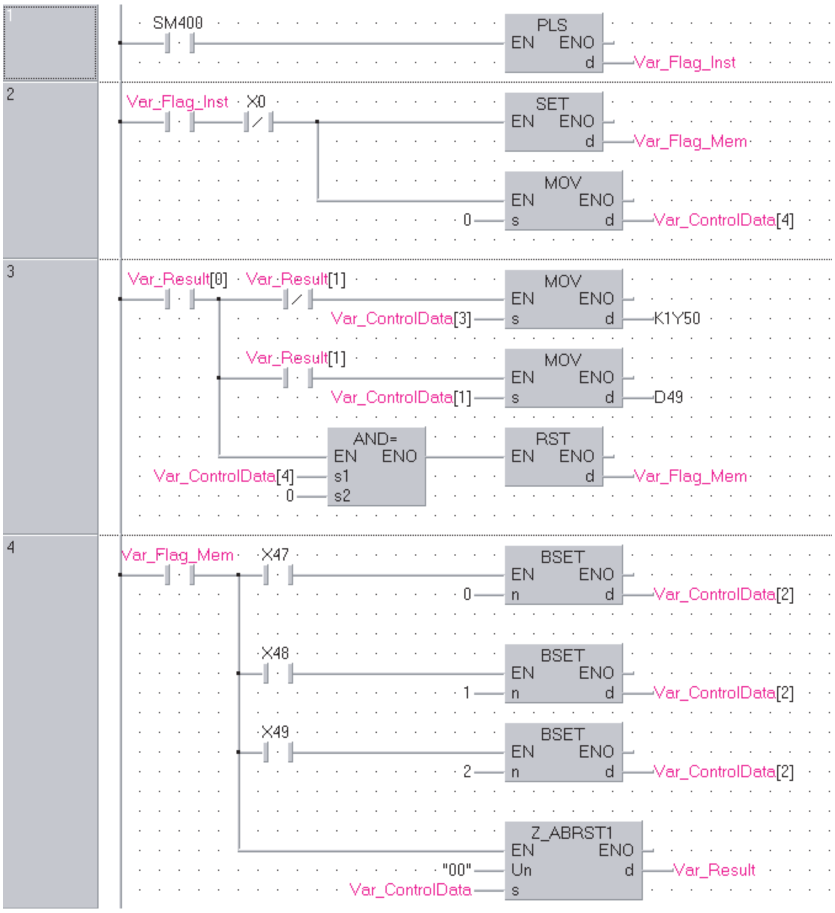
X49: Send data READY flag

Y50: Servo ON

Y51: ABS transfer mode

Y52: ABS request flag

[Structured ladder/FBD]



Absolute position restoration pulse

Turns absolute position restoration memory ON

Clears completion status

Turns the servo ON with the data to be sent to the servo amplifier

Sets completion status to error code

Turns absolute position restoration memory OFF

Sets ABS data
Sets ABS data in data b0 received from the servo

Sets ABS data in data b1

Sets send data ready flag in data b2

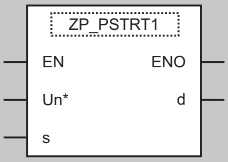
Restores absolute position

[ST]
```
PLS(SM400, Var_Flag_Inst); (* Absolute position restoration pulse *)

IF((Var_Flag_Inst=TRUE) & (X0=FALSE))THEN
    SET(TRUE, Var_Flag_Mem); (* Turns absolute position restoration memory ON *)
    MOV(TRUE, 0, Var_ControlData[4]); (* Clears completion status *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        MOV(TRUE, Var_ControlData[3], K1Y50); (* Turns the servo ON with the data to be sent to the servo amplifier *)
    ELSE (* Error completion *)
        MOV(TRUE, Var_ControlData[4], Var_ErrorCode); (* Sets completion status to error code *)
    END_IF;

    IF(Var_ControlData[4]=0)THEN
        RST(TRUE, Var_Flag_Mem); (* Turns absolute position restoration memory OFF *)
    END_IF;
END_IF;

IF(Var_Flag_Mem=TRUE)THEN (* absolute position restoration memory ON *)
    (* Sets ABS data *)
    BSET(X47, 0, Var_ControlData[2]); (* Sets ABS data in data b0 received from the servo *)
    BSET(X48, 1, Var_ControlData[2]); (* Sets ABS data in data b1 received from the servo *)
    BSET(X49, 2, Var_ControlData[2]); (* Sets send data ready flag in data b2 received from the servo *)

    Z_ABRST1(TRUE, "00", Var_ControlData, Var_Result); (* Restores absolute position *)
END_IF;
```

Positioning start

ZP_PSTRT1, ZP_PSTRT2, ZP_PSTRT3, ZP_PSTRT4

Structured ladder/FBD



ST

ENO:= ZP_PSTRT1 (EN, Un*, s,d);

The following instruction can go in the dotted squares.
ZP_PSTRT1, ZP_PSTRT2, ZP_PSTRT3, ZP_PSTRT4

■Executing condition

Instruction	Executing condition
ZP_PSTRT1, ZP_PSTRT2, ZP_PSTRT3, ZP_PSTRT4	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..2]
Output argument	ENO	Execution result (TRUE: Normal, FALSE: Error)	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○	—	—					

Processing details

This instruction starts positioning of the specified axis. (Refer to the following.)

- ZP_PSTRT1: Axis 1
- ZP_PSTRT2: Axis 2
- ZP_PSTRT3: Axis 3
- ZP_PSTRT4: Axis 4

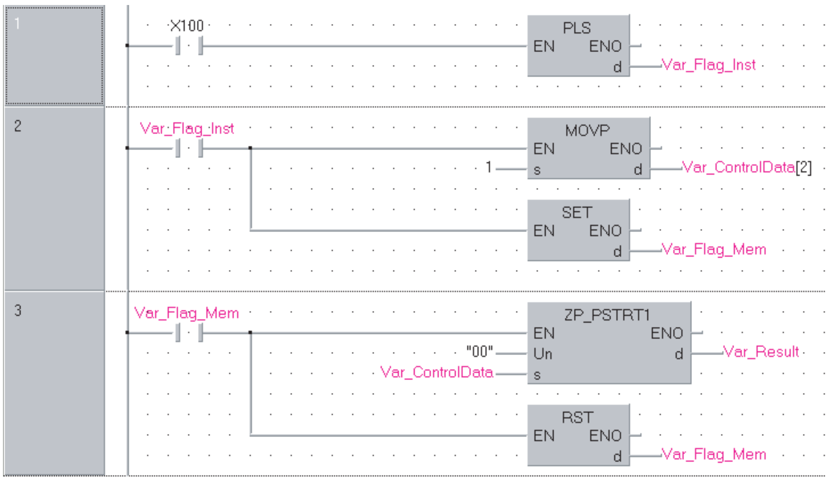
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. • 0: Normal completion • Other than 0: Error completion (error code)	—	System
(s)[2]	Start No.	Specify the following data number to be started by the PSTRT instruction. • 1 to 600: Positioning data number • 7000 to 7004: Block start • 9001: Machine OPR • 9002: Fast OPR • 9003: Current value change • 9004: Multiple axes concurrent start	1 to 600, 7000 to 7004, 9001 to 9004	User

Program example

- The following program executes the positioning start of the positioning data number 1 when X100 turns ON.

[Structured ladder/FBD]



Positioning start pulse

Sets start signal number 1

Turns positioning start instruction memory ON

Performs positioning start

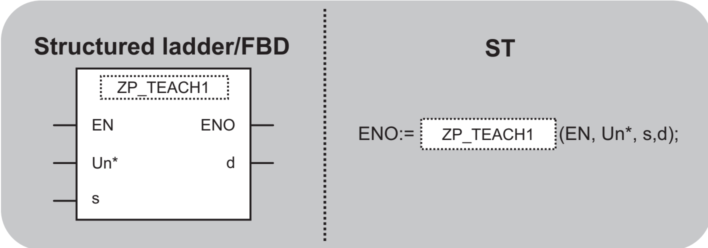
Turns positioning start instruction memory OFF

[ST]

```
PLS(X100, Var_Flag_Inst); (* Positioning start pulse *)

IF(Var_Flag_Inst=TRUE)THEN
    MOVP(TRUE, 1, Var_ControlData[2]); (* Sets start signal number 1 *)
    SET(TRUE, Var_Flag_Mem); (* Turns positioning start instruction memory ON *)
END_IF;

IF(Var_Flag_Mem=TRUE)THEN (* Positioning start instruction memory ON *)
    ZP_PSTRT1(TRUE, "00", Var_ControlData, Var_Result); (* Performs positioning start *)
    RST(TRUE, Var_Flag_Mem); (* Turns positioning start instruction memory OFF *)
END_IF;
```

Teaching

ZP_TEACH1, ZP_TEACH2, ZP_TEACH3, ZP_TEACH4



The following instruction can go in the dotted squares.
ZP_TEACH1, ZP_TEACH2, ZP_TEACH3, ZP_TEACH4

■Executing condition

Instruction	Executing condition
ZP_TEACH1, ZP_TEACH2, ZP_TEACH3, ZP_TEACH4	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module 00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..3]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○	—	—					

Processing details

This instruction performs teaching for the specified axis. (Refer to the following)

- ZP_TEACH1: Axis 1
- ZP_TEACH2: Axis 2
- ZP_TEACH3: Axis 3
- ZP_TEACH4: Axis 4

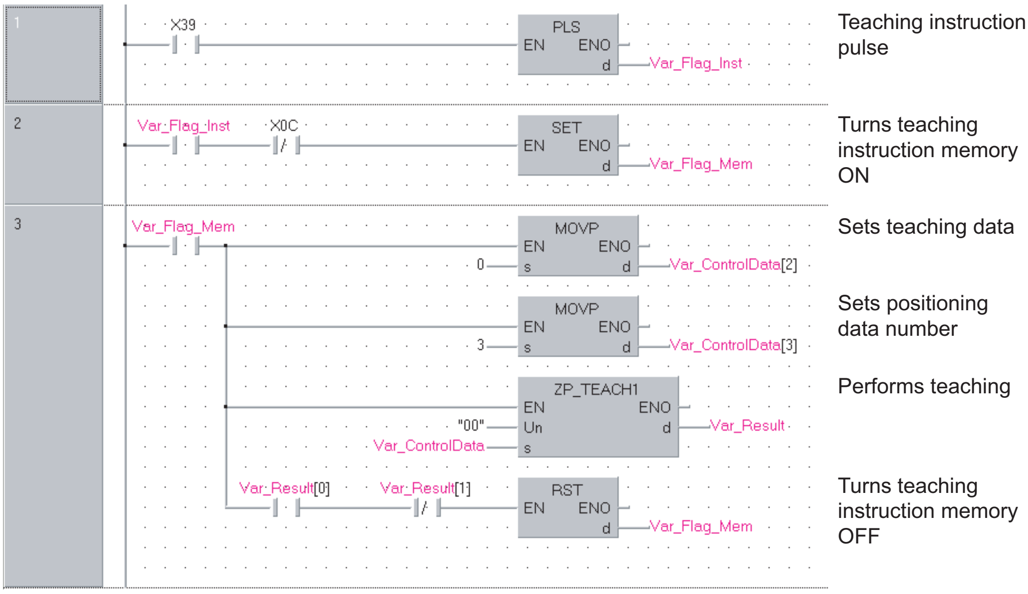
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. • 0: Normal completion • Other than 0: Error completion (error code)	—	System
(s)[2]	Teaching data selection	Set the address (positioning address/circular address) to which the current feed value is written. 0: Write the current feed value to the positioning address 1: Write the current feed value to the circular address	0, 1	User
(s)[3]	Positioning data No.	Set the positioning data number for which teaching is performed.	1 to 600	User

Program example

- The following program performs teaching for the positioning data number 3 of the axis 1 when X39 turns ON.

[Structured ladder/FBD]



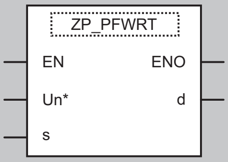
[ST]

```
PLS(X39, Var_Flag_Inst); (* Teaching instruction pulse *)

IF((Var_Flag_Inst=TRUE)&(X0C=FALSE))THEN
    SET(TRUE, Var_Flag_Mem); (* Turns teaching instruction memory ON *)
END_IF;

IF(Var_Flag_Mem=TRUE)THEN (* Teaching instruction memory ON *)
    MOVP(TRUE, H0, Var_ControlData[2]); (* Sets teaching data *)
    MOVP(TRUE, K3, Var_ControlData[3]); (* Sets positioning data number *)

    ZP_TEACH1(TRUE, "00", Var_ControlData, Var_Result); (* Performs teaching *)

    IF((Var_Result[0]=TRUE)&(Var_Result[1]=FALSE))THEN
       RST(TRUE, Var_Flag_Mem); (* Turns teaching instruction memory OFF *)
    END_IF;
END_IF;
```

PFWRT instruction

ZP_PFWRT

Structured ladder/FBD



ST

ENO:= ZP_PFWRT (EN, Un*, s,d);

The following instruction can go in the dotted squares.
ZP_PFWRT

■Executing condition

Instruction	Executing condition
ZP_PFWRT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..1]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○	—	—					

Processing details

This instruction writes the positioning module parameters, positioning data, and block start data to the flash ROM.

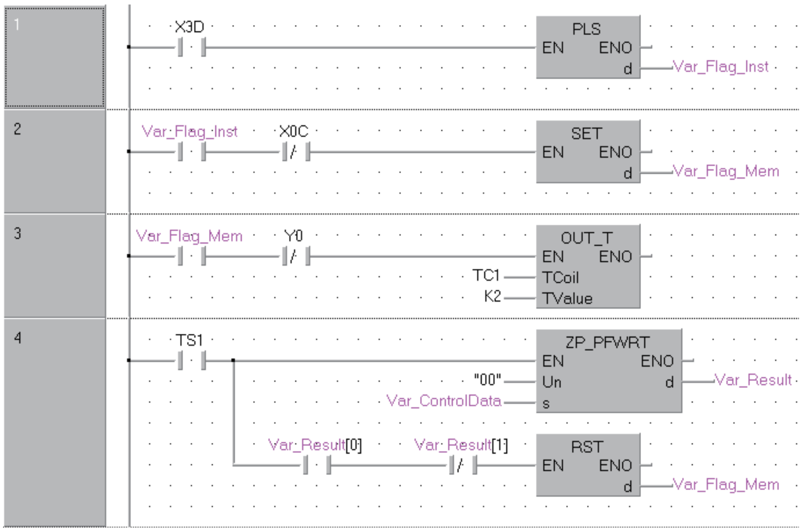
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. • 0: Normal completion • Other than 0: Error completion (error code)	—	System

Program example

- The following program writes the parameters, positioning data, and block start data stored in buffer memory to the flash ROM when X3D turns ON.

[Structured ladder/FBD]



Write to flash ROM instruction pulse

Turns write to flash ROM instruction memory ON

Waits output of programmable controller ready for the positioning module

Writes data to flash ROM

Turns write to flash ROM instruction memory OFF

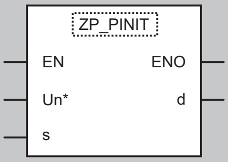
[ST]

```
PLS(X3D, Var_Flag_Inst); (* Write to flash ROM instruction pulse *)

IF((Var_Flag_Inst=TRUE)&(X0C=FALSE))THEN
    SET(TRUE, Var_Flag_Mem); (* Turns write to flash ROM instruction memory ON *)
END_IF;

IF((Var_Flag_Mem=TRUE)&(Y0=FALSE))THEN
    OUT_T(TRUE, TC1, 2); (* Waits output of programmable controller ready for the positioning module *)
END_IF;

IF(TS1=TRUE)THEN (* Write to flash ROM instruction memory ON *)
    ZP_PFWRT(TRUE, "00", Var_ControlData, Var_Result); (* Writes data to flash ROM *)

    IF((Var_Result[0]=TRUE)&(Var_Result[1]=FALSE))THEN
      RST(TRUE, Var_Flag_Mem); (* Turns write to flash ROM instruction memory OFF *)
    END_IF;
END_IF;
```

Setting data initialization

ZP_PINIT

Structured ladder/FBD



ST

ENO:= ZP_PINIT (EN, Un*, s,d);

The following instruction can go in the dotted squares.
ZP_PINIT

■Executing condition

Instruction	Executing condition
ZP_PINIT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..1]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○	—	—					

Processing details

This instruction initializes the positioning module setting data.

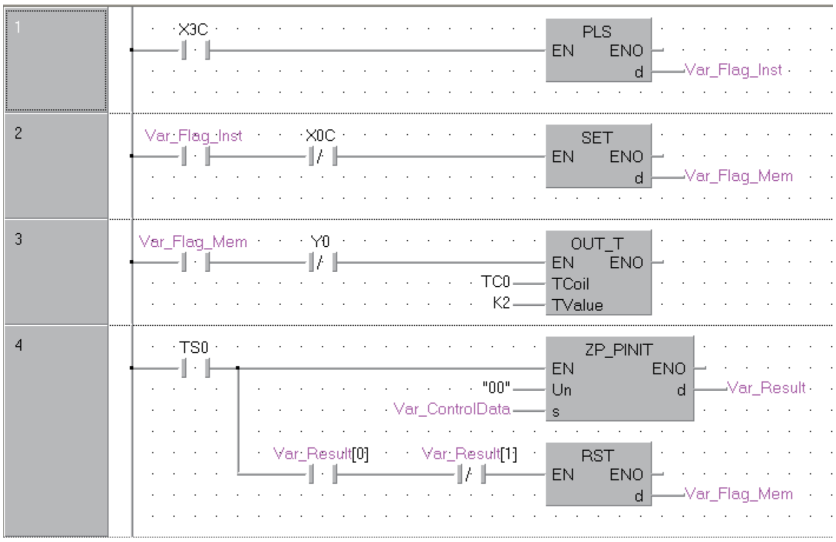
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. • 0: Normal completion • Other than 0: Error completion (error code)	—	System

Program example

• The following program initializes the parameters of buffer memory and those of flash ROM when X3C turns ON.

[Structured ladder/FBD]



Parameter initialization instruction pulse

Turns parameter initialization instruction memory ON

Waits output of programmable controller ready for the positioning module

Performs initialization of parameters

Turns parameter initialization instruction memory OFF

[ST]

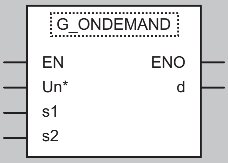
```
PLS(X3C, Var_Flag_Inst); (* Parameter initialization instruction pulse *)

IF((Var_Flag_Inst=TRUE)&(X0C=FALSE))THEN
    SET(TRUE, Var_Flag_Mem); (* Turns parameter initialization instruction memory ON *)
END_IF;

IF((Var_Flag_Mem=TRUE)&(Y0=FALSE))THEN
    OUT_T(TRUE, TC0, 2); (* Waits output of programmable controller ready for the positioning module *)
END_IF;

IF(TS0=TRUE)THEN (* Parameter initialization instruction memory ON *)
    ZP_PINIT(TRUE, "00", Var_ControlData, Var_Result); (* Performs initialization of parameters *)

    IF((Var_Result[0]=TRUE)&(Var_Result[1]=FALSE))THEN
        RST(TRUE, Var_Flag_Mem); (* Turns parameter initialization instruction memory OFF *)
    END_IF;
END_IF;
```

5.3 Serial Communication Instruction

On-demand function transmission

G(P)_ONDEMAND

Serial **Modem**

Structured ladder/FBD	ST
	ENO:=G_ONDEMAND(EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.
G_ONDEMAND, GP_ONDEMAND

■Executing condition

Instruction	Executing condition
G_ONDEMAND	
GP_ONDEMAND	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..2]
	s2	Start number of the device that stores send data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data using the on-demand function of MC protocol.

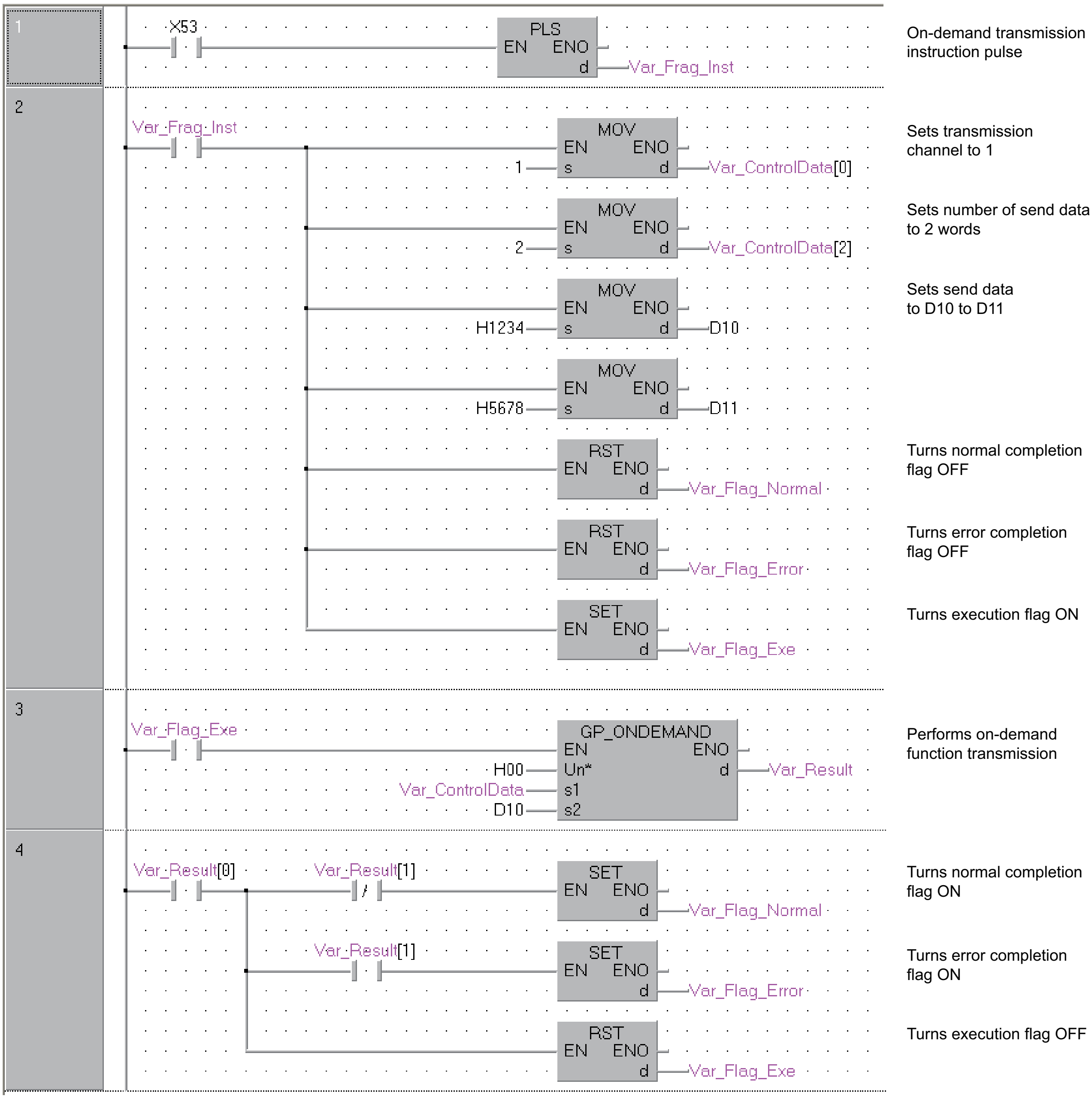
Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Transmission channel	Set the transmission channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s1)[1]	Transmission result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Number of send data	Set the number of send data.	1 or more	User

Program example

- The following program sends data of devices from D10 to D11 using the on-demand function. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



[ST]

```
PLS(X53, Var_Flag_Inst); (* On-demand transmission instruction pulse *)
IF(Var_Flag_Inst=TRUE)THEN (* Instruction flag ON *)
    MOV(TRUE, 1, Var_ControlData[0]); (* Sets transmission channel to 1 *)
    MOV(TRUE, 2, Var_ControlData[2]); (* Sets number of send data to 2 words *)
    MOV(TRUE, H1234, D10); (* Sets send data to D10 to D11 *)
    MOV(TRUE, H5678, D11);
    RST(TRUE, Var_Flag_Normal); (* Turns normal completion flag OFF *)
    RST(TRUE, Var_Flag_Error); (* Turns error completion flag OFF *)
    SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;
IF(Var_Flag_Exe=TRUE)THEN (* Execution flag ON *)
    GP_ONDEMAND(TRUE, H0, Var_ControlData, D10, Var_Result); (* Performs on-demand function transmission *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
      SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
    RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

Point

- The communication status can be checked by the SPBUSY instruction.

☞ Page 63 Communication status check

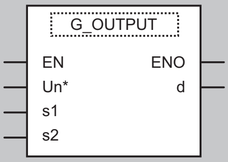
- Specify the capacity of the send data (stored in devices from D10 to D11 in the program example above) and the number of send data within the user-defined buffer memory range assigned for the on-demand function.

Nonprocedural protocol communication

G(P)_OUTPUT

Serial **Modem**

Structured ladder/FBD



ST

ENO:= G_OUTPUT (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.
G_OUTPUT, GP_OUTPUT

■Executing condition

Instruction	Executing condition
G_OUTPUT	
GP_OUTPUT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..2]
	s2	Start number of the device that stores send data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data in the message format specified by the user using the nonprocedural protocol.

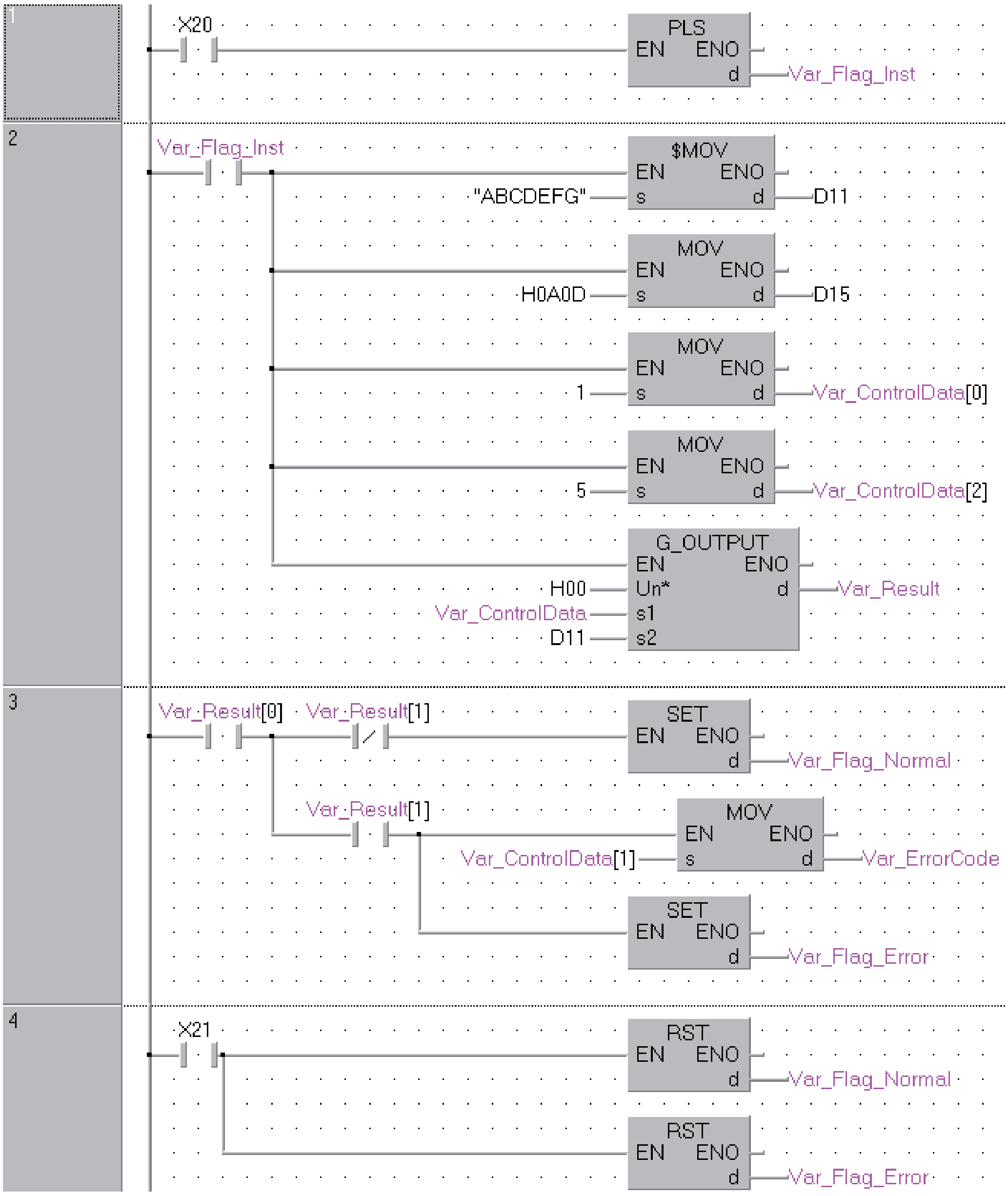
Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Transmission channel	Set the transmission channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s1)[1]	Transmission result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Number of send data	Set the number of send data.	1 or more	User

Program example

- The following program sends data of devices from D11 to D15 using the nonprocedural protocol. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



Transmission instruction pulse

Sets send data

Sets completion code

Sets transmission channel to 1

Sets number of send data to 5 words

Sends data

Turns normal completion flag ON

Stores error code

Turns error completion flag ON

Turns normal completion flag OFF

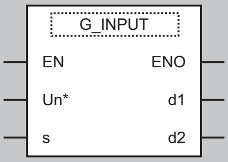
Turns error completion flag OFF

[ST]

```
PLS(X20, Var_Flag_Inst); (* Transmission instruction pulse*)

IF (Var_Flag_Inst=TRUE) THEN
    MOV(TRUE, H4241, D11); (* Sets send data *)
    MOV(TRUE, H4443, D12);
    MOV(TRUE, H4645, D13);
    MOV(TRUE, H0047, D14);
    MOV(TRUE, H0AD, D15);
    MOV(TRUE, 1, Var_ControlData[0]); (* Sets transmission channel to 1 *)
    MOV(TRUE, 5, Var_ControlData[2]); (* Sets number of send data to 5 words *)
    G_OUTPUT(TRUE, H0, Var_ControlData, D11, Var_Result); (* Sends data *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode);(* Stores error code *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;

IF (X21=TRUE) THEN
        RST( TRUE, Var_Flag_Normal ); (* Turns normal completion flag OFF *)
        RST( TRUE, Var_Flag_Error ); (* Turns error completion flag OFF *)
END_IF;
```

G_INPUT

Serial **Modem**

Structured ladder/FBD	ST
	ENO:= G_INPUT (EN, Un*, s, d1, d2);

The following instruction can go in the dotted squares.
G_INPUT

■Executing condition

Instruction	Executing condition
G_INPUT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..3]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction receives data in the message format specified by the user using the nonprocedural protocol.

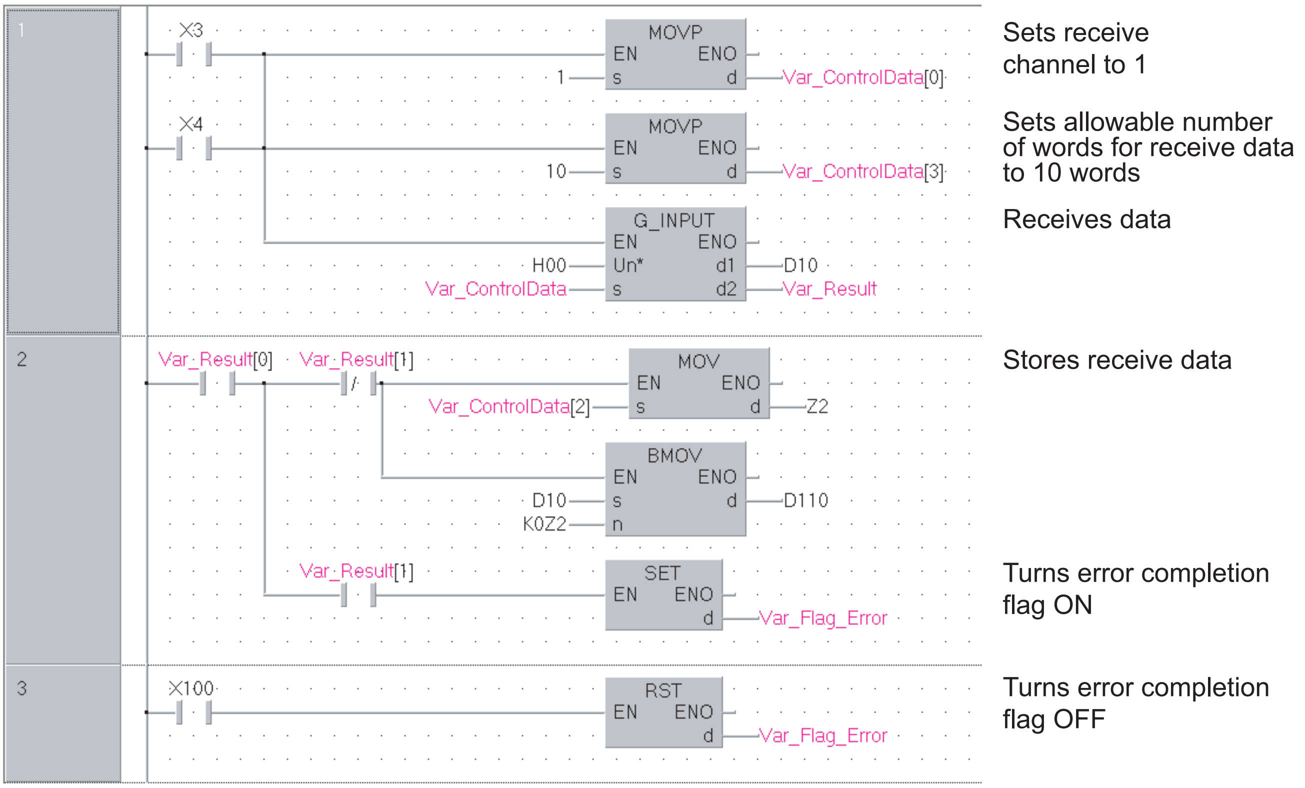
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Reception channel	Set the reception channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s)[1]	Reception result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Number of receive data	The number of receive data are stored.	0 or more	System
(s)[3]	Allowable number of words for receive data	Set the allowable number of words for receive data to be stored in (d1).	1 or more	User

Program example

• The following program stores data which are received using the nonprocedural protocol in the devices starting from D10. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



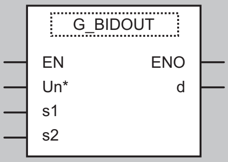
[ST]

```
IF((X3=TRUE) OR (X4=TRUE))THEN
    MOVP(TRUE, 1, Var_ControlData[0]); (* Sets receive channel to 1 *)
    MOVP(TRUE, 10, Var_ControlData[3]); (* Sets allowable number of words for receive data to 10 words *)
    G_INPUT(TRUE, H0, Var_ControlData, D10, Var_Result); (* Receives data *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        MOV(TRUE, Var_ControlData[2], Z2);
        BMOV(TRUE, D10, K0Z2, D110); (* Stores receive data *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;

IF(X100=TRUE)THEN
    RST(TRUE, Var_Flag_Error); (* Turns error completion flag OFF *)
END_IF;
```

Bidirectional protocol communication

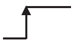
G(P)_BIDOUT

Serial **Modem**

Structured ladder/FBD	ST
	ENO:= G_BIDOUT (EN ,Un* ,s1 ,s2 ,d);

The following instruction can go in the dotted squares.
G_BIDOUT, GP_BIDOUT

■Executing condition

Instruction	Executing condition
G_BIDOUT	
GP_BIDOUT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 (0..2)
	s2	Start number of the device that stores send data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of Bit (0..1)

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data using the bidirectional protocol.

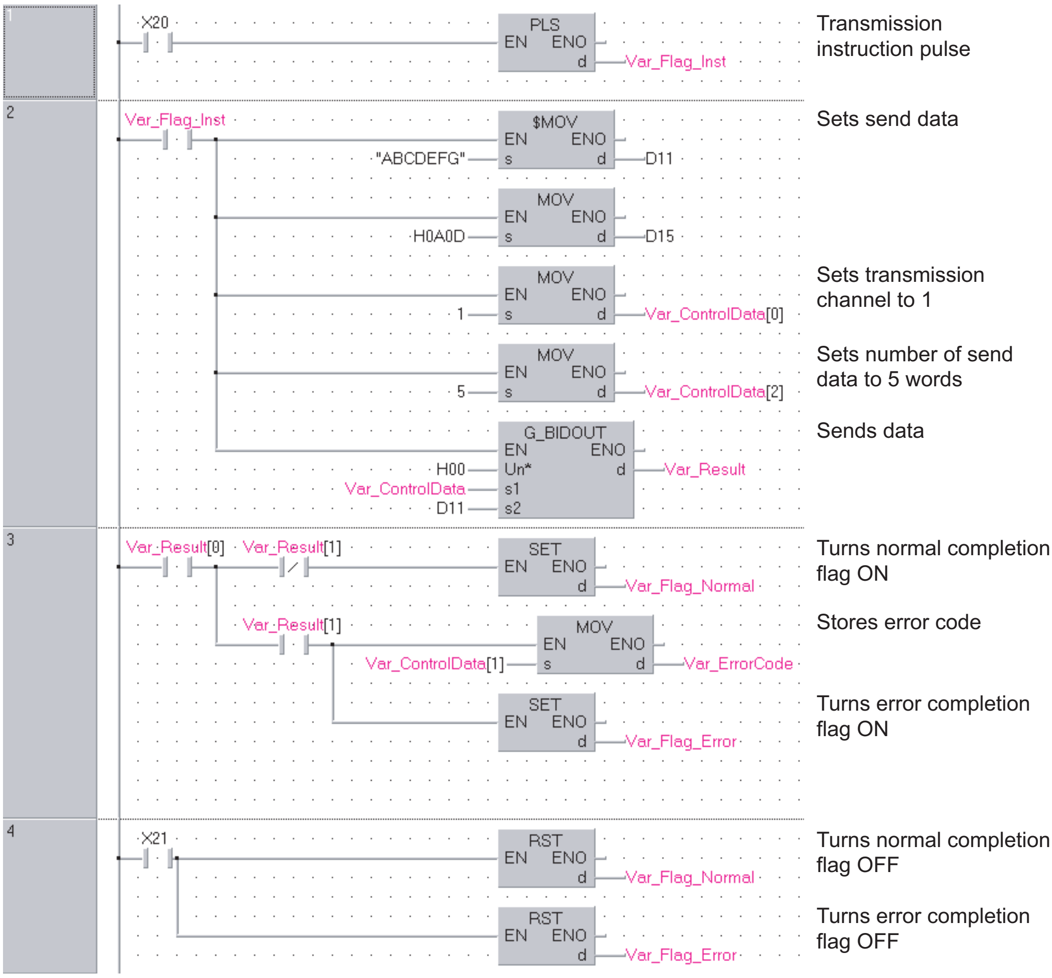
Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Transmission channel	Set the transmission channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s1)[1]	Transmission result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Number of send data	Set the number of send data.	1 or more	User

Program example

- The following program sends desired data stored in devices from D11 to D15 using the bidirectional protocol. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]

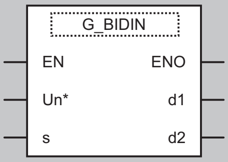


[ST]

```
PLS(X20, Var_Flag_Inst); (* Transmission instruction pulse *)

IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE, H4241, D11); (* Sets send data *)
    MOV(TRUE, H4443, D12);
    MOV(TRUE, H4645, D13);
    MOV(TRUE, H0047, D14);
    MOV(TRUE, H0AD, D15);
    MOV(TRUE, 1, Var_ControlData[0]); (* Sets transmission channel to 1 *)
    MOV(TRUE, 5, Var_ControlData[2]); (* Sets allowable number of words for send data to 5 words *)
    G_BIDOUT(TRUE, H0, Var_ControlData, D11, Var_Result); (* Sends data *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;

IF(X21=TRUE)THEN
    RST(TRUE, Var_Flag_Normal); (* Turns normal completion flag OFF *)
    RST(TRUE, Var_Flag_Error); (* Turns error completion flag OFF *)
END_IF;
```

G(P)_BIDIN

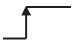
Serial **Modem**

Structured ladder/FBD	ST
	ENO:= G_BIDIN (EN, Un*, s, d1, d2);

The following instruction can go in the dotted squares.

G_BIDIN, GP_BIDIN

■Executing condition

Instruction	Executing condition
G_BIDIN	
GP_BIDIN	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..3]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction receives data using the bidirectional protocol.

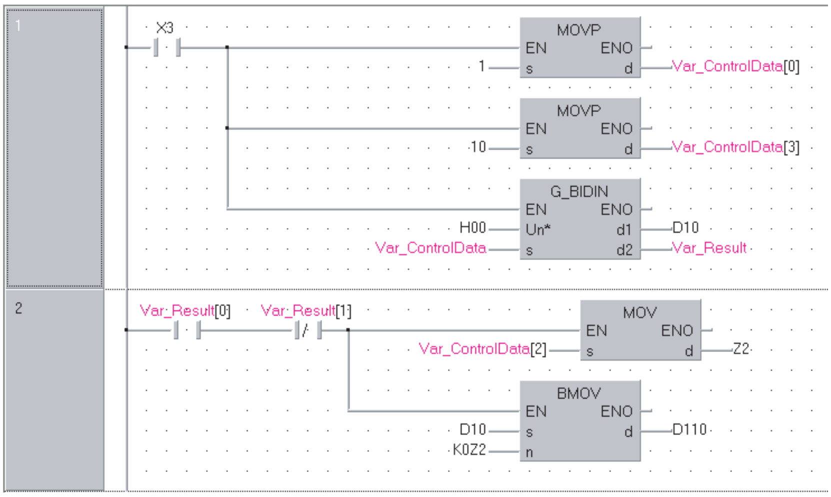
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Reception channel	Set the reception channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s)[1]	Reception result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Number of receive data	The number of received data are stored.	1 or more	System
(s)[3]	Allowable number of words for receive data	Set the allowable number of words for receive data to be stored in (d1).	1 or more	User

Program example

- The following program receives data using the bidirectional protocol and stores the data in the devices starting from D10. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



Sets receive channel to 1

Sets allowable number of words for receive data to 10 words

Receives data

Stores receive data

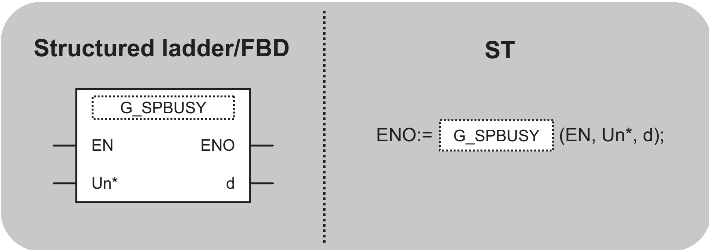
[ST]

```
IF(X3=TRUE)THEN
    MOVP(TRUE, 1, Var_ControlData[0]); (* Sets receive channel to 1 *)
    MOVP(TRUE, 10, Var_ControlData[3]); (* Sets allowable number of words for receive data to 10 *)
    G_BIDIN(TRUE, H00, Var_ControlData, D10, Var_Result); (* Receives data *)
END_IF;

IF((Var_Result[0]=TRUE)&(Var_Result[1]=FALSE))THEN
        BMOV(TRUE, D10, Var_ControlData[2], D110); (* Stores receive data *)
END_IF;
```

Communication status check

G(P)_SPBUSY

Serial **Modem**



The following instruction can go in the dotted squares.
G_SPBUSY, GP_SPBUSY

■Executing condition

Instruction	Executing condition
G_SPBUSY	
GP_SPBUSY	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that stores read communication status	ANY32

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(d)	○	○		—					

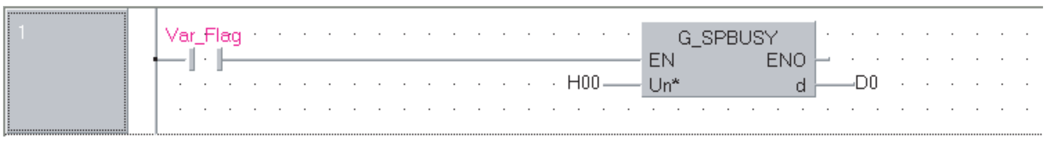
Processing details

This instruction reads the data transmission/reception status.

Program example

- The following program reads out the communication status of the target module. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



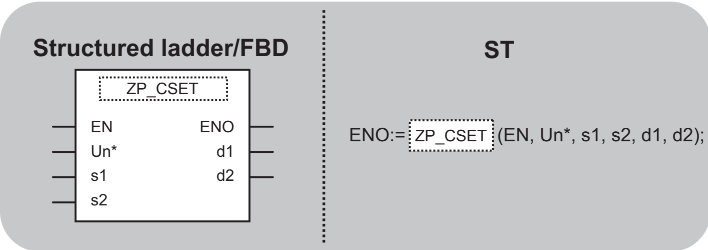
Reads communication status

[ST]
G_SPBUSY(Var_Flag, H00, D0); (* Reads communication status *)

Receive data clear

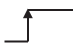
ZP_CSET

Serial **Modem**



The following instruction can go in the dotted squares.
ZP_CSET

■Executing condition

Instruction	Executing condition
ZP_CSET	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Channel number that requests receive data clear 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..111]
Output argument	ENO	Execution result	Bit
	d1	Dummy	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				—	—
(d1)	—	○		—				—	—
(d2)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

Clears receive data without stopping transmission using the nonprocedural protocol.

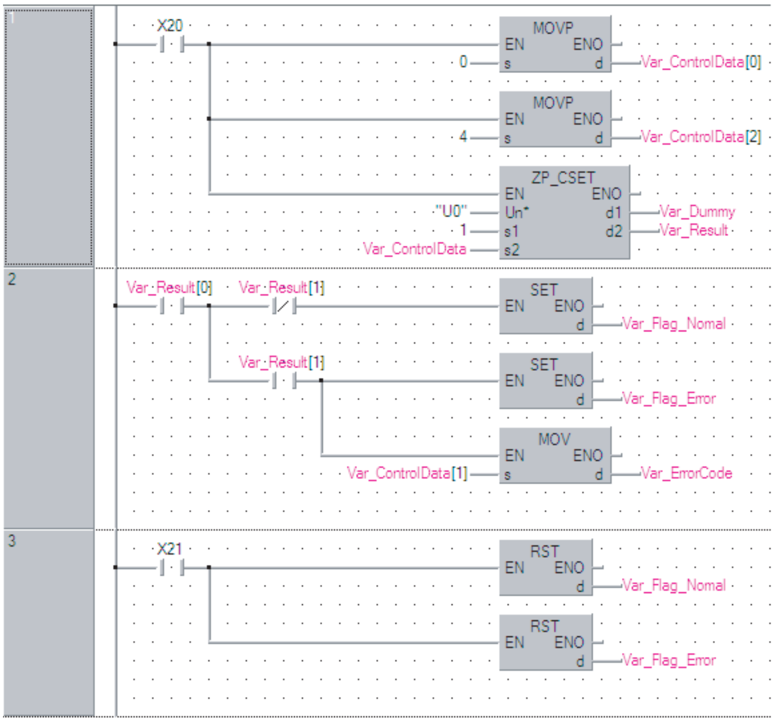
Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	Execution type	Specify '0'.	0	User
(s2)[1]	Completion status	The instruction completion status is stored. • 0: Normal completion • Other than 0: Error completion (error code)	—	System
(s2)[2]	Request type	Specify the request. 4: Receive data clear request	4	User
(s2)[3] ⋮ (s2)[111]	For system	—	—	System

Program example

- The following program clears the receive data in the Q series C24 side. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



Clears execution type to 0

Sets request type

Clears data

Turns normal completion flag ON

Turns error completion flag ON

Stores error code

Turns normal completion flag OFF

Turns error completion flag OFF

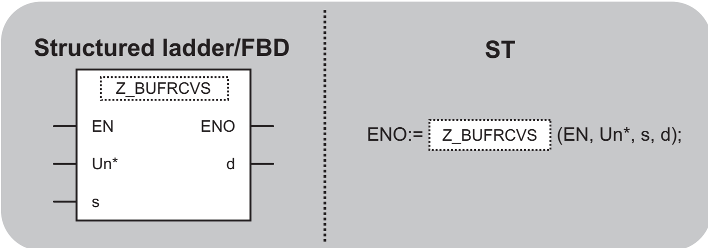
[ST]

```
IF(X20=TRUE)THEN
    MOVP(TRUE, 0, Var_ControlData[0]); (* Clears execution type to 0 *)
    MOVP(TRUE, 4, Var_ControlData[2]); (* Sets request type *)
    ZP_CSET(TRUE, "U0", 1, Var_ControlData, Var_Dummy, Var_Result); (* Clears data *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;

IF(X21=TRUE)THEN
        RST(TRUE, Var_Flag_Normal); (* Turns normal completion flag OFF *)
        RST(TRUE, Var_Flag_Error); (* Turns error completion flag OFF *)
END_IF;
```

BUFRCVS instruction

Z_BUFRCVS

Serial **Modem**



The following instruction can go in the dotted squares.
Z_BUFRCVS

■Executing condition

Instruction	Executing condition
Z_BUFRCVS	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Reception channel number 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	ANY16
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores read data * Receive data are read from the receive area of buffer memory.	ANY16

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s)	—	○		—				○	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction receives data with an interrupt program during communication using the nonprocedural protocol or bidirectional protocol.

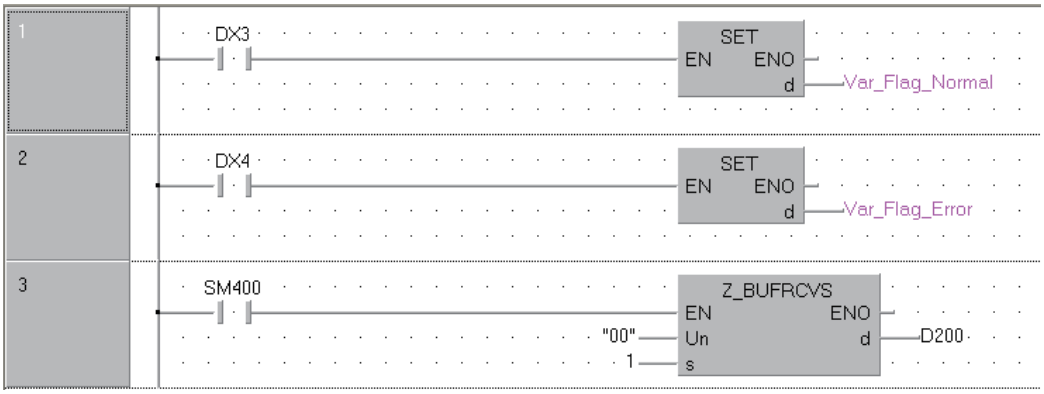
Setting data

Device	Item	Setting data	Setting range	Setting side
(d)+0	Receive data length	The number of data read from the number of receive data storage area is stored.	0 or more	System
(d)+1 ⋮ (d)+n	Receive data	Data read from the receive data storage area are stored in ascending address order.	—	System

Program example

- The following program receives data with an interrupt program.

[Structured ladder/FBD]



Turns normal completion flag ON

Turns error completion flag ON

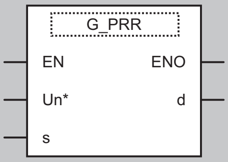
Executes interrupt receive program

[ST]

```
(* Set the normal/error confirmation flag for the main program *)
(* The main program resets flags *)
SET(DX3, Var_Flag_Normal); (* Turns normal completion flag ON *)
SET(DX4, Var_Flag_Error); (* Turns error completion flag ON *)
(* Receives data from CH1 and stores the data in devices starting from D200 *)
Z_BUFRCVS(SM400, "00", 1, D200); (* Executes interrupt receive program *)
```

G(P)_PRR

Serial **Modem**

Structured ladder/FBD	ST
	ENO:= G_PRR (EN, Un*, s, d);

The following instruction can go in the dotted squares.
G_PRR, GP_PRR

■Executing condition

Instruction	Executing condition
G_PRR	
GP_PRR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..4]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data by user frame according to the specification in user frame specification area for transmission during communication using the nonprocedural protocol.

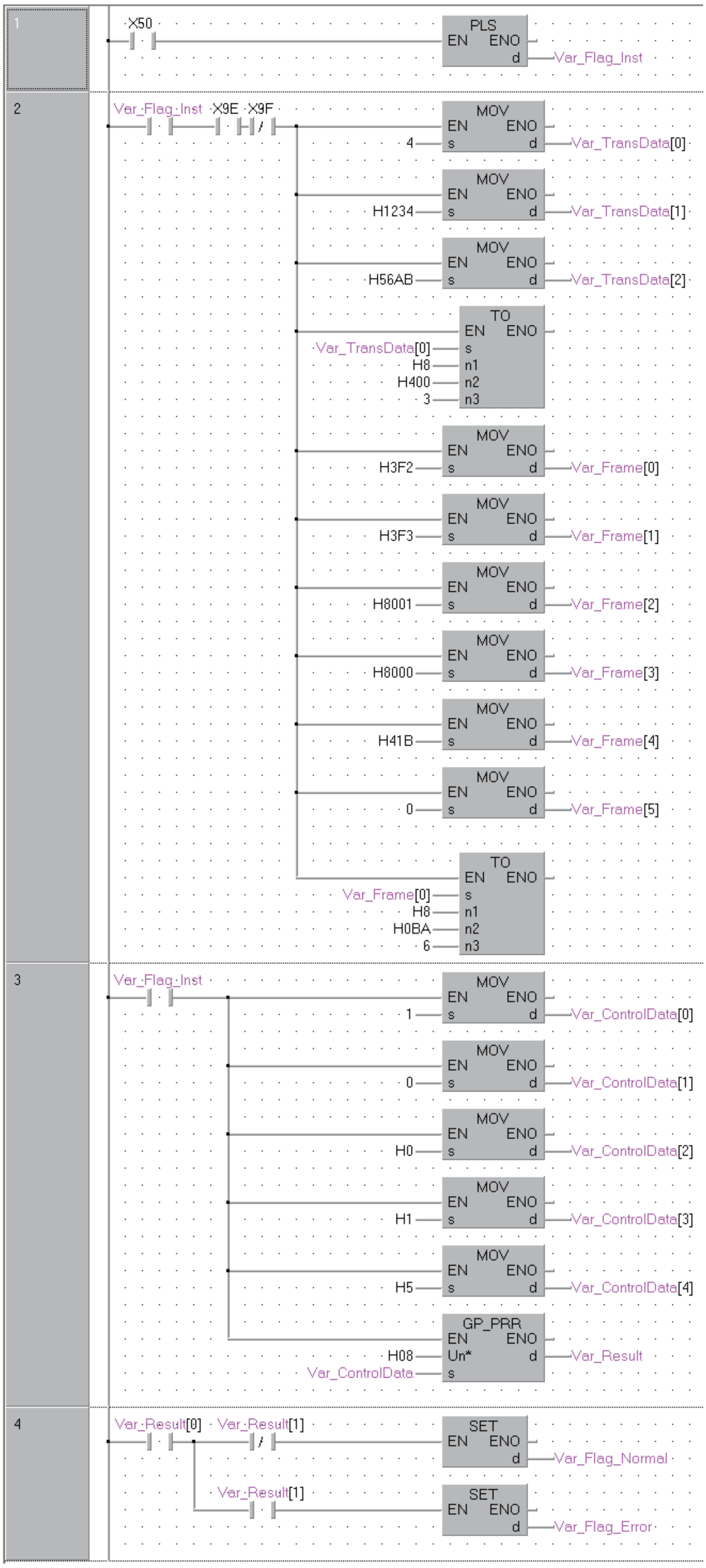
Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Transmission channel	Set the transmission channel. 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(s)[1]	Transmission result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	CR/LF addition specification	Specify whether to add CR/LF codes to the send data. 0: CR/LF not added 1: CR/LF added	0, 1	User
(s)[3]	Transmission pointer	Specify the position in the user frame specification area for transmission from where the frame number data are to be sent.	1 to 100	User
(s)[4]	Number of send data	Set the number of user frames to be sent.	1 to 100	User

Program example

- The following program sends desired data and the user frames from number 1 to number 5 which are registered in the transmission frame setting. (For the Q series C24 whose I/O signals are X/Y80 to X/Y9F)

[Structured ladder/FBD]



Transmission instruction pulse

Sets number of send data

Sets desired send data

Sets send data to buffer memory

User frame 0

User frame 1

User frame 2

User frame 3

User frame 4

User frame 5

Sets user frames to buffer memory

Sets transmission channel to 1

Clears transmission result

Sets CR/LF to 'CR/LF not added'

Sets transmission pointer

Sets number of send data

Performs user frame transmission

Turns normal completion flag ON

Turns error completion flag ON

[ST]

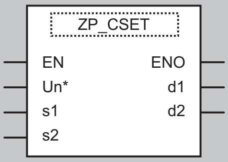
```
PLS(X50, Var_Flag_Inst); (* Transmission instruction pulse *)

IF((Var_Flag_Inst=TRUE) & (X9E=TRUE) & (X9F=FALSE))THEN
    MOV(TRUE, 4, Var_TransData[0]); (* Sets number of send data *)
    MOV(TRUE, H1234, Var_TransData[1]); (* Sets desired send data *)
    MOV(TRUE, H56AB, Var_TransData[2]);
    TO(TRUE, Var_TransData[0], H8, H400, 3); (* Sets send data to buffer memory *)

    MOV(TRUE, H3F2, Var_Frame[0]); (* Sets user frame 0 *)
    MOV(TRUE, H3F3, Var_Frame[1]); (* Sets user frame 1 *)
    MOV(TRUE, H8001,Var_Frame[2]); (* Sets user frame 2 *)
    MOV(TRUE, H8000,Var_Frame[3]); (* Sets user frame 3 *)
    MOV(TRUE, H41B, Var_Frame[4]); (* Sets user frame 4 *)
    MOV(TRUE, 0, Var_Frame[5]); (* Sets user frame 5 *)
    TO(TRUE, Var_Frame[0], H8, H0BA, 6); (* Sets user frames to buffer memory *)
END_IF;

IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE, 1, Var_ControlData[0]); (* Sets transmission channel to 1 *)
    MOV(TRUE, 0, Var_ControlData[1]); (* Clears transmission result *)
    MOV(TRUE, H0, Var_ControlData[2]); (* Sets CR/LF to 'CR/LF not added' *)
    MOV(TRUE, H1, Var_ControlData[3]); (* Sets transmission pointer *)
    MOV(TRUE, H5, Var_ControlData[4]); (* Sets number of send data *)
    GP_PRR(TRUE, H08, Var_ControlData, Var_Result); * Performs user frame transmission *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
       SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
       SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

5

Initial setting

ZP_CSET

Serial **Modem**

Structured ladder/FBD	ST
	ENO:=ZP_CSET(EN, Un*, s1, s2, d1, d2);

The following instruction can go in the dotted squares.
ZP_CSET

■Executing condition

Instruction	Executing condition
ZP_CSET	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Reception channel number 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..111]
Output argument	ENO	Execution result	Bit
	d1	Dummy	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				—	—
(d1)	—	○		—				—	—
(d2)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction changes the setting values for sending/receiving data using communication protocols.

Setting data

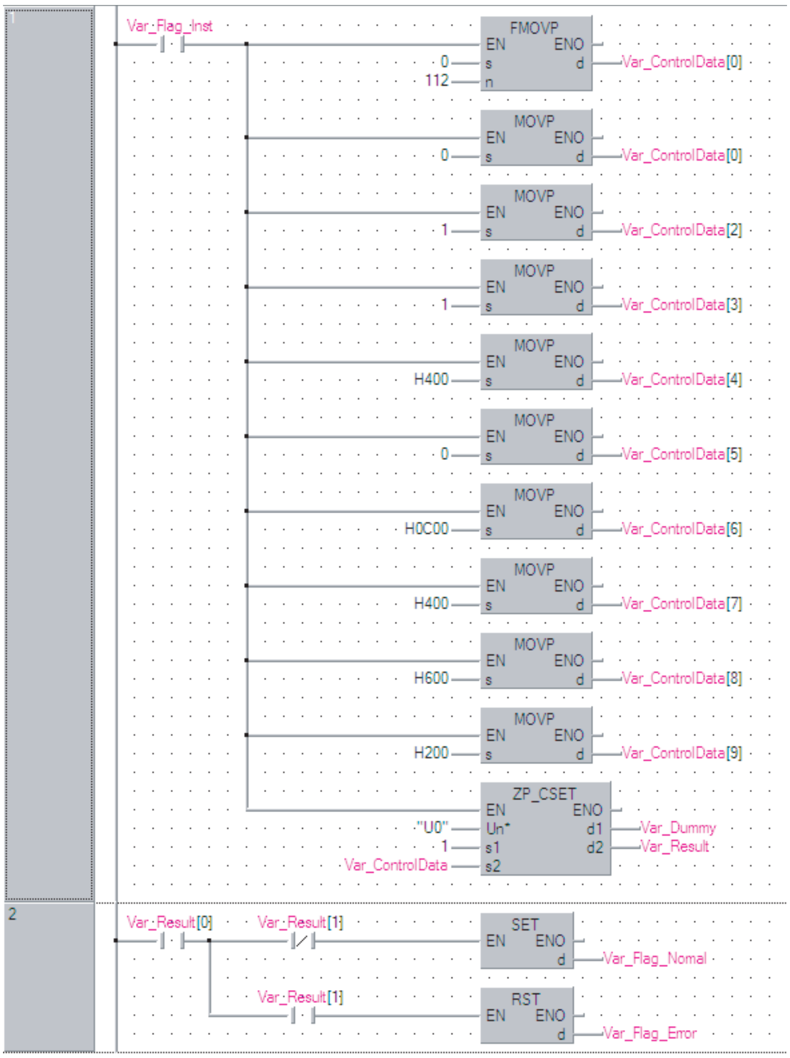
Device	Item	Setting data	Setting range	Setting side
(s2)[0]	Execution type	Specify '0'.	0	User
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2]	Request type	Specify the request. 1: Change of unit (word/byte) and buffer memory assignment	1	User
(s2)[3]	Word/byte unit specification	Specify the unit of the number of send/receive data. 0: Current setting value 1: In units of words 2: In units of bits	0, 1, 2	User
(s2)[4]	Buffer memory start address for on-demand function	Specify the start address of the buffer memory used by the on-demand function 0H: Current setting value is used. 400H to 1AFFH, 2600H to 3FFFH: Start address	0H, 400H to 1AFFH, 2600H to 3FFFH	User
(s2)[5]	Buffer memory size for on-demand function	Specify the size (the number of words) of the buffer memory to be used by the on-demand function. 0H: Current setting value is used. 1H to 1A00H: Size	0H, 1H to 1A00H	User
(s2)[6]	Send area start address	Specify the start address of the send area used for the nonprocedural/bidirectional protocol. 0H: Current setting value is used. 400H to 1AFFH, 2600H to 3FFFH: Start address	0H, 400H to 1AFFH, 2600H to 3FFFH	User
(s2)[7]	Send area size	Specify the size (the number of words) of the send area used by the nonprocedural/bidirectional protocol. 0H: Current setting value is used. 1H to 1A00H: Size * The start area of the send area (1 word) is used for the number of send data specification area.	0H, 1H to 1A00H	User
(s2)[8]	Receive area start address	Specify the start address of the receive area used for the nonprocedural/bidirectional protocol. 0H: Current setting value is used. 400H to 1AFFH, 2600H to 3FFFH: Start address	0H, 400H to 1AFFH, 2600H to 3FFFH	User
(s2)[9]	Receive area size	Specify the size (the number of words) of the receive area used for the nonprocedural/bidirectional protocol. 0H: Current setting value is used. 1H to 1A00H: Size * The start area of the receive area (1 word) is used for the number of receive data storage area.	0H, 1H to 1A00H	User
(s2)[10] ⋮ (s2)[111]	For system	—	—	System

Program example

The following program changes the send buffer area of the CH1 side interface. (For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

- Sets send buffer to C00H to FFFH.
- Sets receive buffer to 600H to 7FFH.

[Structured ladder/FBD]



Clears D0 to D111 to 0

Sets execution type

Sets request type

Sets word/byte unit to word

Sets on-demand start address

Sets on-demand buffer size

Sets send buffer start address

Sets send buffer size

Sets receive buffer start address

Sets receive buffer size

Performs initialization

Turns normal completion flag ON

Turns error completion flag ON

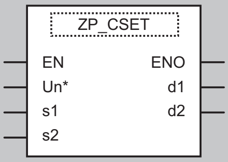
[ST]

```
IF(Var_Flag_Inst=TRUE)THEN
    FMOVP(TRUE,0,112, Var_ControlData[0]); (* Resets D0 to D111 to 0 *)
    MOVP(TRUE, 0, Var_ControlData[0]); (* Sets execution type *)
    MOVP(TRUE, 1, Var_ControlData[2]); (* Sets request type *)
    MOVP(TRUE, 1, Var_ControlData[3]); (* Sets word/byte unit to word *)
    MOVP(TRUE, H400, Var_ControlData[4]); (* Sets on-demand start address *)
    MOVP(TRUE, 0, Var_ControlData[5]); (* Sets on-demand buffer size *)
    MOVP(TRUE,H0C00, Var_ControlData[6]); (* Sets send buffer start address *)
    MOVP(TRUE, H400, Var_ControlData[7]); (* Sets send buffer size *)
    MOVP(TRUE, H600, Var_ControlData[8]); (* Sets receive buffer start address *)
    MOVP(TRUE, H200, Var_ControlData[9]); (* Sets receive buffer size *)
    ZP_CSET(TRUE, "U0", 1, Var_ControlData, Var_Dummy, Var_Result); (* Performs initialization *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
      SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

CSET instruction (programmable controller CPU monitor)

ZP_CSET

Serial **Modem**

Structured ladder/FBD	ST
	ENO:=ZP_CSET(EN, Un*, s1, s2, d1, d2);

The following instruction can go in the dotted squares.

ZP_CSET

■Executing condition

Instruction	Executing condition
ZP_CSET	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Reception channel number 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..111]
Output argument	ENO	Execution result	Bit
	d1	Dummy	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				—	—
(d1)	—	○		—				—	—
(d2)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction registers and cancels the programmable controller CPU monitoring.

Setting data

■Registering the programmable controller CPU monitoring

Device	Item		Setting data	Setting range	Setting side
(s2)[0]	Execution type		Specify '0'.	0	User
(s2)[1]	Completion status		The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2]	Request type		Specify the request. 2: Registration of programmable controller CPU monitoring	2	User
(s2)[3]	Cycle time unit		Specify the unit of cycle time. 0: 100ms 1: Second 2: Minute	0 to 2	User
(s2)[4]	Cycle time		Specify the cycle time. 1H to FFFFH: Cycle time	1H to FFFFH	User
(s2)[5]	Programmable controller CPU monitoring function		Specify the monitoring function. 1: Constant cycle transmission 2: Condition agreement transmission	1,2	User
(s2)[6]	Programmable controller CPU monitoring transmission method		Specify the transmission method. 0: Data transmission (device data, CPU error information) 1: Notification	0, 1	User
(s2)[7]	Constant cycle transmission	User frame output start pointer	Specify the start pointer of the table to which the user frame number for constant cycle transmission is set. 0: No specification (at condition agreement transmission and notification) 1 to 100: Start pointer	0, 1 to 100	User
(s2)[8]		Number of user frame transmissions	Specify the number of user frame transmissions (outputs) for constant cycle transmission. 0: No specification (at condition agreement transmission and notification) 1 to 100: Number of transmissions	0, 1 to 100	User
(s2)[9]		Modem connection data No.	Specify the data number for modem function connection when making notification in constant cycle transmission. 0: No specification (at data transmission and condition agreement transmission) BB8H to BD5H: Connection data number (flash ROM) 8001H to 801FH: Connection data number (buffer memory)	0, BB8H to BD5H, 8001H to 801FH	User
(s2)[10]	Number of registered word blocks		Specify the number of blocks of the word device to be monitored.	0 to 10	User
(s2)[11]	Number of registered bit blocks		Specify the number of blocks of the bit device to be monitored.	0 to 10	User
(s2)[12]	Programmable controller CPU error monitoring (programmable controller CPU status monitoring)		Specify whether to also execute programmable controller CPU error monitoring. 0: Not monitored 1: Monitored	0, 1	User

<table>
<tr><th>Device</th><th colspan="3">Item</th><th>Setting data</th><th>Setting range</th><th>Setting side</th></tr>
<tr><td>(s2)[13]</td><td rowspan="9">Programmable controller CPU monitoring setting

1st
* 1st block</td><td colspan="2">Device code</td><td>Specify the code of the device to be monitored.
0: No device monitored
Other than 0: Device code</td><td>90H to CCH
(Device code)</td><td>User</td></tr>
<tr><td>(s2)[14]</td><td colspan="2" rowspan="2">Monitoring start device</td><td rowspan="2">Specify the start number of the monitoring device in this block.</td><td rowspan="2">0 or more</td><td rowspan="2">User</td></tr>
<tr><td>(s2)[15]</td></tr>
<tr><td>(s2)[16]</td><td colspan="2">Number of registered points</td><td>Specify the number of registered points (read points) of this block.
0: No device monitored
1 or more: Number of registered points
* For a bit device, specify the number of points in units of words.</td><td>0, 1 or more</td><td>User</td></tr>
<tr><td>(s2)[17]</td><td rowspan="5">Condition agreement transmission</td><td>Monitoring condition</td><td>Specify the monitoring condition of this block.
0: No specification (at constant cycle transmission)
1 or more: Monitoring condition</td><td>0 to 65535</td><td>User</td></tr>
<tr><td>(s2)[18]</td><td>Monitoring condition value</td><td>Specify the monitoring condition value for this block.
0 or more: Monitoring condition
* Specify '0' at constant cycle transmission.</td><td>0 to 000AH,
0101H to
010AH</td><td>User</td></tr>
<tr><td>(s2)[19]</td><td>User frame output start pointer</td><td>Specify the start pointer of the table to which the user frame number for condition agreement transmission for this block is set.
0: No specification (at constant cycle transmission and notification)
1 to 100: Start pointer</td><td>0, 1 to 100</td><td>User</td></tr>
<tr><td>(s2)[20]</td><td>Number of user frame transmissions</td><td>Specify the number of user frame transmissions (outputs) for condition agreement transmission for this block.
0: No specification (at constant cycle transmission and notification)
1 to 100: Number of transmissions</td><td>0, 1 to 100</td><td>User</td></tr>
<tr><td>(s2)[21]</td><td>Modem connection data No.</td><td>Specify the data number for modem function connection when making notification in condition agreement transmission for this block.
0: No specification (at data transmission and constant cycle transmission)
BB8H to BD5H: Connection data number (flash ROM)
8001H to 801FH: Connection data number (buffer memory)</td><td>0,
BB8H to
BD5H,
8001H to
801FH</td><td>User</td></tr>
<tr><td>(s2)[22]
⋮
(s2)[102]</td><td colspan="3">Programmable controller CPU monitoring setting

2nd to 10th
* 2nd to 10th block</td><td>The same item arrangement as the first programmable controller CPU monitoring setting item.</td><td>—</td><td>User</td></tr>
</table>

Device	Item			Setting data	Setting range	Setting side
(s2)[103]	CPU status monitoring setting * Error monitoring 11th * 11th block	Condition agreement transmission	Fixed value	Specify a fixed value to monitor the CPU status.	1	User
(s2)[104]					0	
(s2)[105]					0	
(s2)[106]					1	
(s2)[107]					5	
(s2)[108]					1	
(s2)[109]			User frame output start pointer	Specify the start pointer of the to which the user frame number for condition agreement transmission for this block is set. 0: No specification (at constant cycle transmission and notification) 1 to 100: Start pointer	0, 1 to 100	User
(s2)[110]			Number of user frame transmissions	Specify the number of user frame transmissions (outputs) for condition agreement transmission for this block. 0: No specification (at constant cycle transmission and notification) 1 to 100: Number of transmissions	0, 1 to 100	User
(s2)[111]			Modem connection data No.	Specify the data number for modem function connection when making notification in condition agreement transmission for this block. 0: No specification (at data transmission and constant cycle transmission) BB8H to BD5H: Connection data number (flash ROM) 8001H to 801FH: Connection data number (buffer memory)	0, BB8H to BD5H, 8001H to 801FH	User

■Canceling the programmable controller CPU monitoring

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	Execution type	Specify '0H'.	0	User
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2]	Request type	Specify the request. 3: Cancel of the programmable controller CPU monitoring	3	User
(s2)[3] ⋮ (s2)[111]	For system	—	—	System

Program example

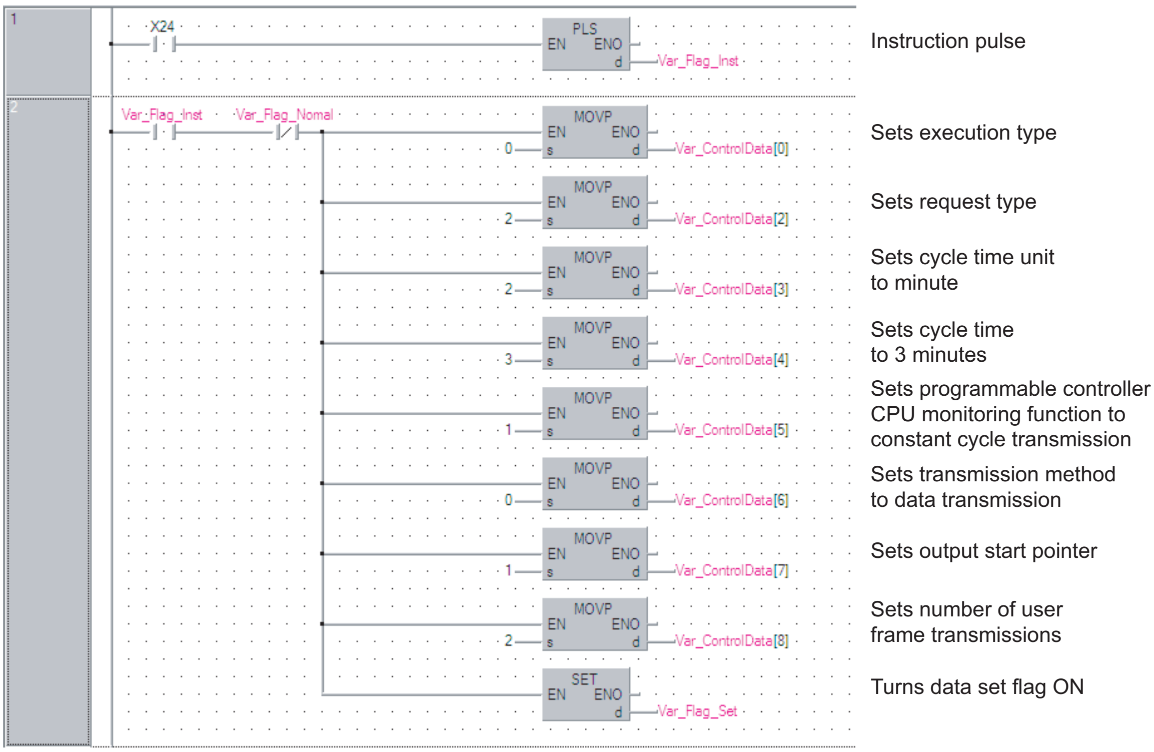
- Program to register the programmable controller CPU monitoring

The following program registers the programmable controller CPU monitoring and sends the monitoring result from the CH1 side interface.

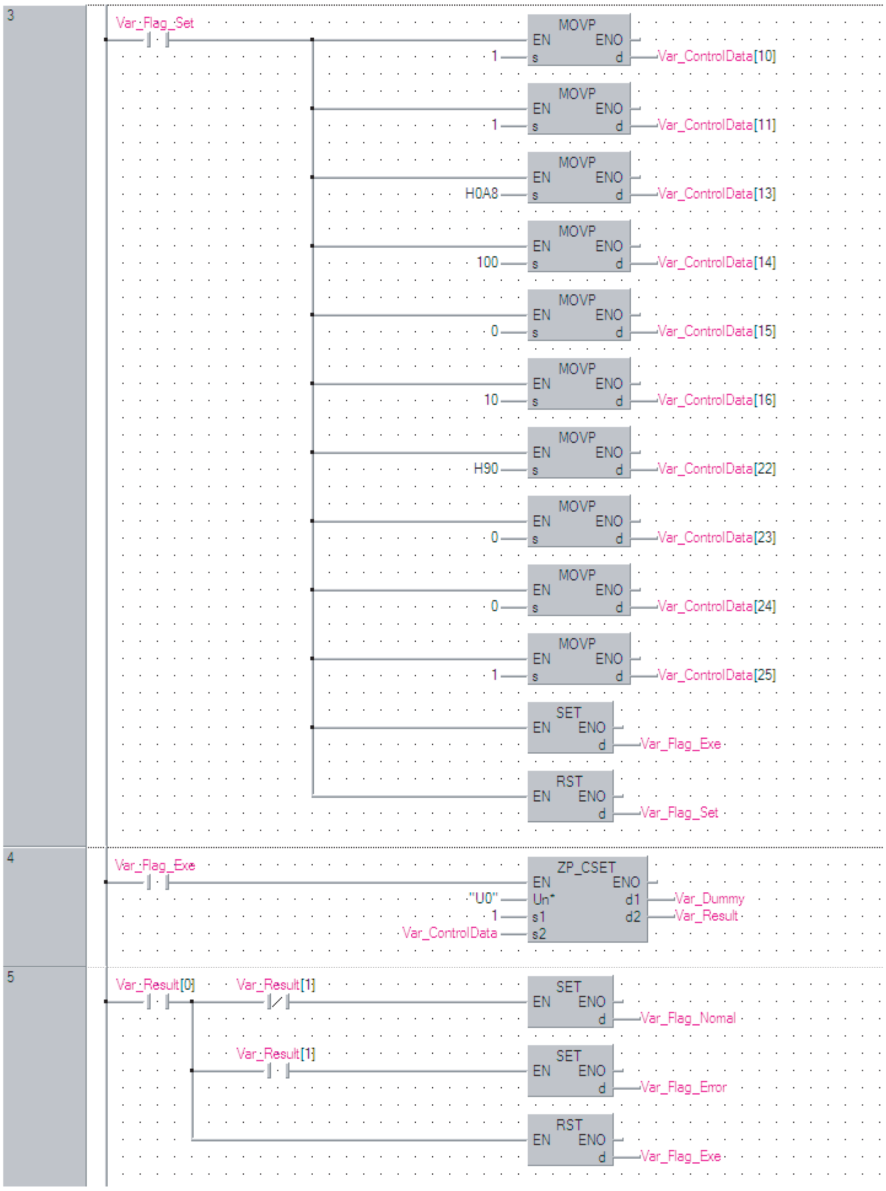
The following setting is to send content of devices from M0 to M15 and devices from D100 to D109 to the external device through the constant cycle transmission. (Cycle time: 3 minutes)

(For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



5



Sets number of registered word blocks

Sets number of registered bit blocks

Sets device code

Sets start number of monitoring device

Sets number of registered points

Sets device code

Sets start number of monitoring device

Sets number of registered points (For bit device, specify it in units of words)

Turns execution flag ON

Turns data set flag OFF

Registers programmable controller CPU monitoring

Turns normal completion flag ON

Turns error completion flag ON

Turns execution flag OFF

[ST]

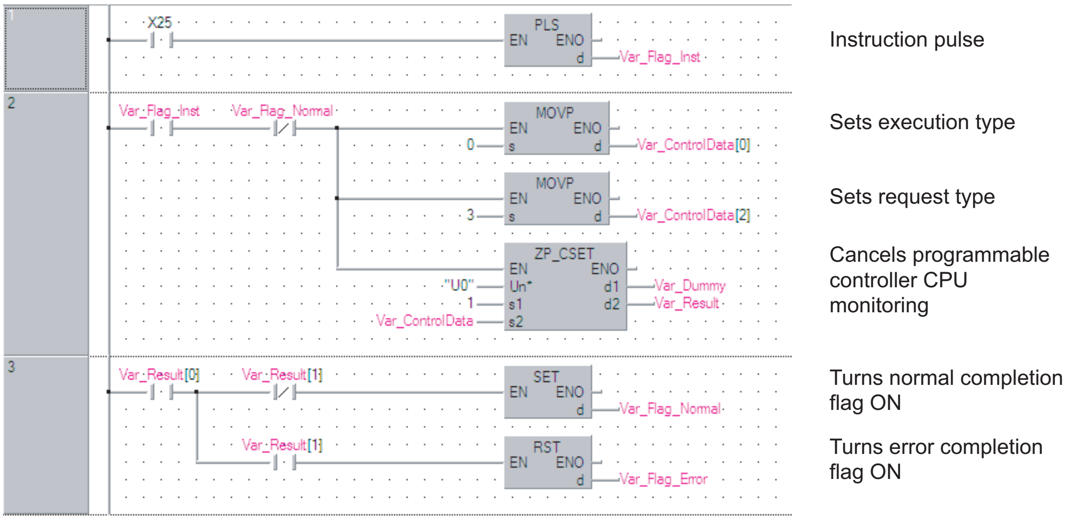
```
PLS(X24, Var_Flag_Inst); (* Instruction pulse *)

IF((Var_Flag_Inst=TRUE) & (Var_Flag_Normal=FALSE))THEN
    MOV(TRUE, 0, Var_ControlData[0]); (* Sets execution type *)
    MOV(TRUE, 2, Var_ControlData[2]); (* Sets request type *)
    MOV(TRUE, 2, Var_ControlData[3]); (* Sets cycle time unit to minute *)
    MOV(TRUE, 3, Var_ControlData[4]); (* Sets cycle time to 3 minutes *)
    MOV(TRUE, 1, Var_ControlData[5]); (* Sets programmable controller CPU monitoring function to constant cycle transmission. *)
    MOV(TRUE, 0, Var_ControlData[6]); (* Sets transmission method to data transmission *)
    MOV(TRUE, 1, Var_ControlData[7]); (* Sets output start pointer *)
    MOV(TRUE, 2, Var_ControlData[8]); (* Sets number of user frame transmissions *)
    SET(TRUE, Var_Flag_Set); (* Turns data set flag ON *)
END_IF;

IF(Var_Flag_Set=TRUE)THEN
    MOV(TRUE, 1, Var_ControlData[10]); (* Sets number of registered word blocks *)
    MOV(TRUE, 1, Var_ControlData[11]); (* Sets number of registered bit blocks *)
    (* Sets the 1st block of the CPU monitoring to D100 to D109 *)
    MOV(TRUE, H0A8, Var_ControlData[13]); (* Sets device code *)
    MOV(TRUE, 100, Var_ControlData[14]);(* Sets start number of monitoring device *)
    MOV(TRUE, 0, Var_ControlData[15]);
    MOV(TRUE, 10, Var_ControlData[16]); (* Sets number of registered points *)
    (* Sets the 2nd block of the CPU monitoring to M0 to M15 *)
    MOV(TRUE, H90, Var_ControlData[22]); (* Sets device code *)
    MOV(TRUE, 0, Var_ControlData[23]);(* Sets start number of monitoring device *)
    MOV(TRUE, 0, Var_ControlData[24]);
    MOV(TRUE, 1, Var_ControlData[25]); (* Sets number of registered points. (For bit device, specify it in units of words.) *)
    SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
    RST(TRUE, Var_Flag_Set); (* Turns data set flag OFF *)
END_IF;

IF(Var_Flag_Exe=TRUE)THEN
    ZP_CSET(TRUE, "U0", 1, Var_ControlData, Var_Dummy, Var_Result); (* Registers the programmable controller CPU monitoring *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;

    RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

5

- Program to cancel the programmable controller CPU monitoring

The following program cancels the programmable controller CPU monitoring of the CH1 side interface.
(For the Q series C24 whose I/O signals are X/Y00 to X/Y1F)

[Structured ladder/FBD]



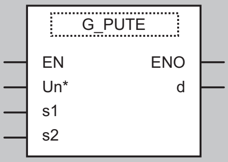
[ST]

```
PLS(X25, Var_Flag_Inst); (* Instruction pulse *)

IF((Var_Flag_Inst=TRUE) & (Var_Flag_Normal=FALSE))THEN
    MOV(TRUE, 0, Var_ControlData[0]); (* Sets execution type *)
    MOV(TRUE, 3, Var_ControlData[2]); (* Sets request type *)
    ZP_CSET(TRUE, "U0", 1, Var_ControlData, Var_Dummy, Var_Result); (* Cancels programmable controller CPU monitoring *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

PUTE instruction

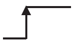
G(P)_PUTE

Serial **Modem**

Structured ladder/FBD	ST
	ENO:= G_PUTE (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.
G_PUTE, GP_PUTE

■Executing condition

Instruction	Executing condition
G_PUTE	
GP_PUTE	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..3]
	s2	Start number of the device that stores read registration data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d1)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction registers a user frame.

Setting data

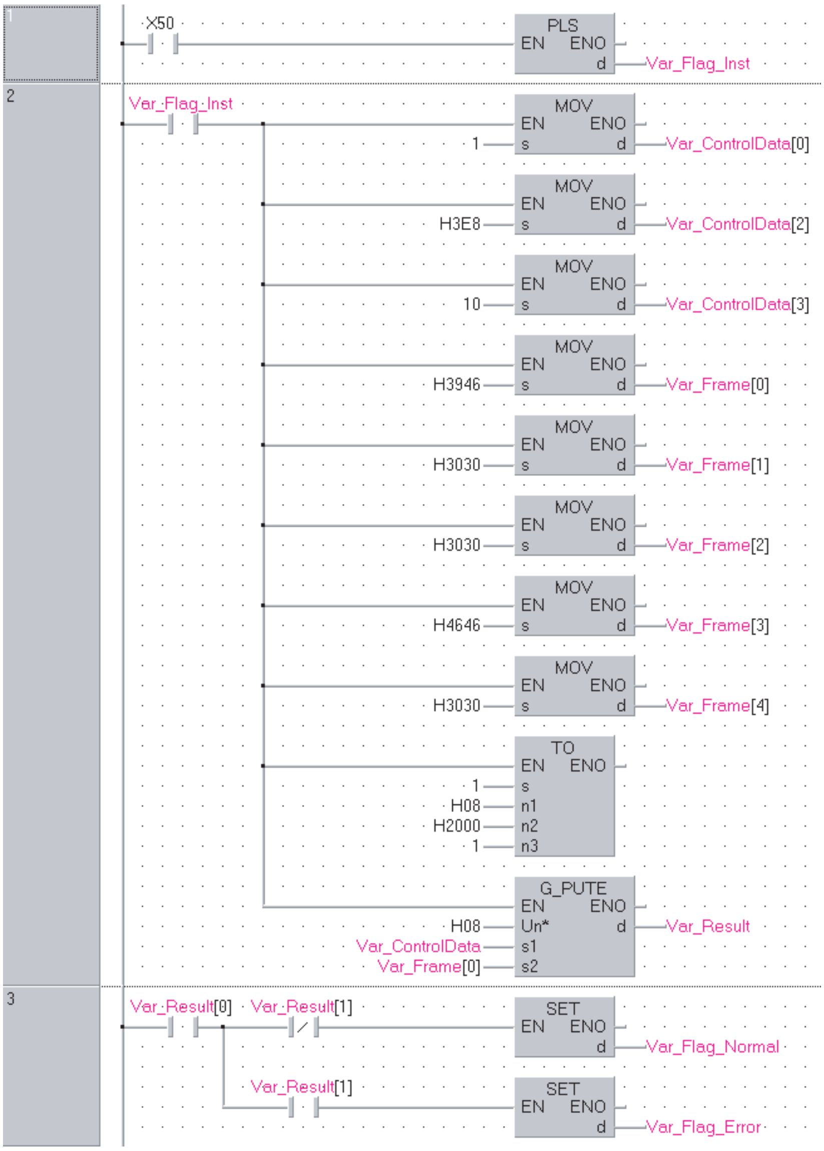
Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Registration/deletion specification	Specify whether to register/delete the user frame of the number specified by (s1)[2]. 1: Registered 3: Deleted	1, 3	User
(s1)[1]	Registration/deletion result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

Device	Item	Setting data	Setting range	Setting side
(s1)[2]	Frame No.	Specify the user frame number.	1000 to 1199	User
(s1)[3]	Number of registered bytes	1 to 80: Number of bytes of the user frame to be registered. * Specify any number in the range from 1 to 80 as a dummy when '3: Deleted' is selected.	1 to 80	User

Program example

- The following program registers a user frame as the registration number 3E8H. (For the Q series C24 whose I/O signals are X/Y80 to X/Y9F)

[Structured ladder/FBD]



Registration request pulse

Sets registration request

Sets user frame number

Sets number of registered bytes

User frame 0

User frame 1

User frame 2

User frame 3

User frame 4

Sets write enable in flash ROM side

Registers user frame

Turns normal completion flag ON

Turns error completion flag ON

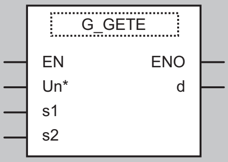
[ST]

```
PLS(X50, Var_Flag_Inst); (* Registration request pulse *)

IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE, 1, Var_ControlData[0]); (* Sets registration request *)
    MOV(TRUE, H3E8, Var_ControlData[2]); (* Sets user frame number *)
    MOV(TRUE, 10, Var_ControlData[3]); (* Sets number of registered bytes *)
    MOV(TRUE, H3946, Var_Frame[0]); (* User frame 0 *)
    MOV(TRUE, H3030, Var_Frame[1]); (* User frame 1 *)
    MOV(TRUE, H3030, Var_Frame[2]); (* User frame 2 *)
    MOV(TRUE, H4646, Var_Frame[3]); (* User frame 3 *)
    MOV(TRUE, H3030, Var_Frame[4]); (* User frame 4 *)
    TO(TRUE, 1, H08, H2000, 1); (* Sets write enable in flash ROM side *)
    G_PUTE(TRUE, H08, Var_ControlData,Var_Frame[0], Var_Result); (* Registers user frame *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

5

G(P)_GETE

Serial **Modem**

Structured ladder/FBD	ST
	ENO:= G_GETE (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.

G_GETE, GP_GETE

■Executing condition

Instruction	Executing condition
G_GETE	
GP_GETE	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..3]
	s2	Start number the device that stores the read registration data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads a user frame.

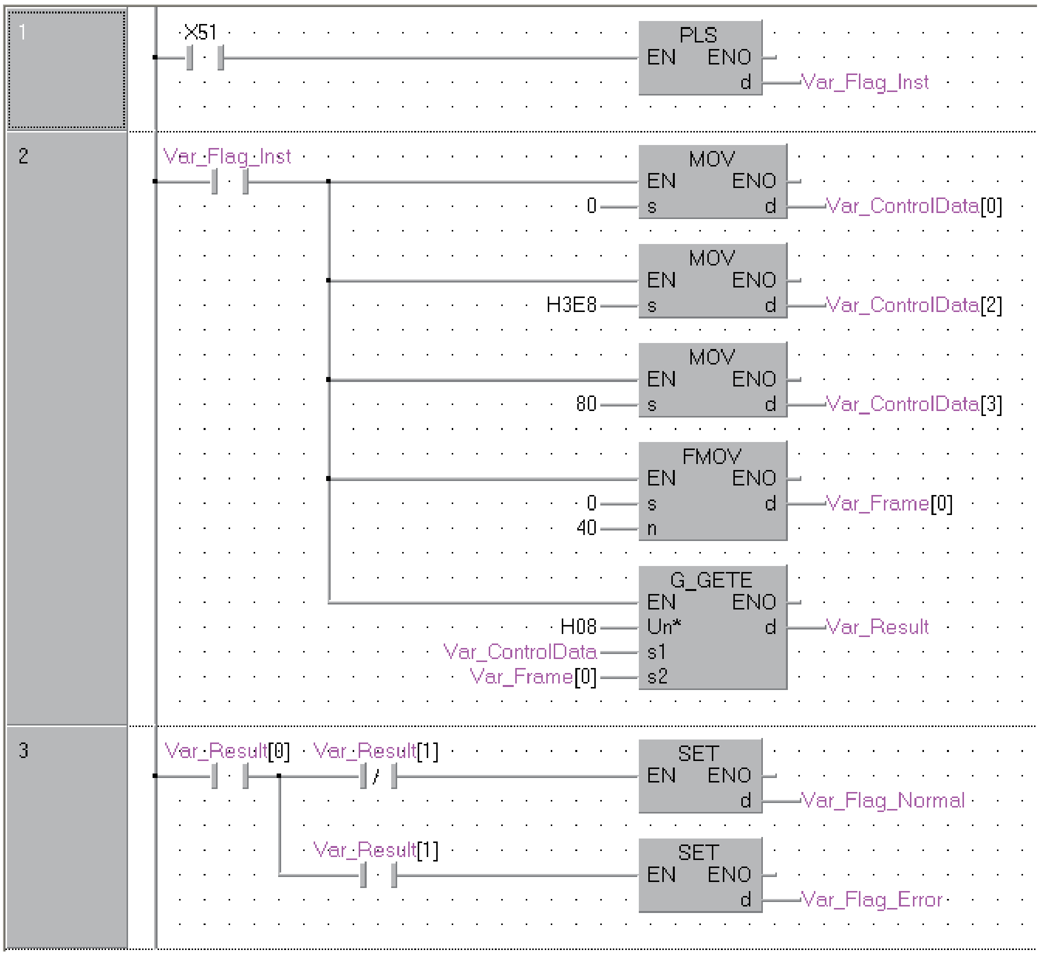
Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Dummy	—	0	—
(s1)[1]	Read result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Frame No. specification	Specify the user frame number.	1000 to 1199	User
(s1)[3]	Allowable number of bytes for read data	Specify the maximum number of bytes for storing the registered data of the read user frame to (s2).	1 to 80	User
	Number of registered bytes	The number of bytes of the registered data for the read user frame is stored.	1 to 80	System

Program example

- The following program reads out the registration data of the user frame number 3E8H. (For the Q series C24 whose I/O signals are X/Y80 to X/Y9F)

[Structured ladder/FBD]



Read request pulse

Sets user frame number

Sets allowable number of bytes for read data

Clears user frame to 0

Reads user frame

Turns normal completion flag ON

Turns error completion flag ON

5

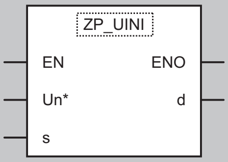
[ST]

```
PLS(X51, Var_Flag_Inst); (* Read request pulse *)

IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE, 0, Var_ControlData[0]);
    MOV(TRUE, H3E8, Var_ControlData[2]); (* Sets user frame number *)
    MOV(TRUE, 80, Var_ControlData[3]); (* Sets allowable number of bytes for read data *)
    FMOV(TRUE, 0, 40, Var_Frame[0]); (* Clears user frame to 0 *)
    G_GETE(TRUE, H08, Var_ControlData, Var_Frame[0], Var_Result); (* Reads user frame *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
      SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

Mode switching

ZP_UINI

Serial

Structured ladder/FBD	ST
	ENO:= ZP_UINI (EN, Un*, s, d);

The following instruction can go in the dotted squares.

ZP_UINI

■Executing condition

Instruction	Executing condition
ZP_UINI	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..9]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

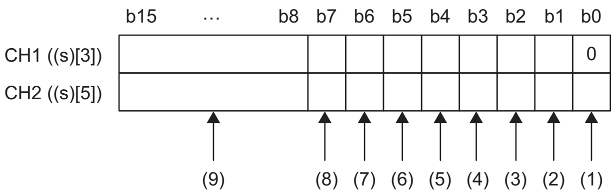
Processing details

This instruction switches the mode, transmission specification, and host station number of the Q series C24.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	For system	Always specify '0'.	0	User
(s)[1]	Execution result	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Execution type	Specify the execution type. 0: Switches the execution type according to the setting in the area starting from (s)[3]. 1: Returns the execution type according to the switch setting on GX Works2.	0, 1	User
(s)[3]	CH1 Transmission specification setting	Set the transmission specifications for CH1. ☞ Page 91 (s)[3] (CH1 Transmission specification setting) and (s)[5] (CH2 Transmission specification setting)*1	0 to 0FFEH	
(s)[4]	CH1 Communication protocol setting	Set the communication protocol for CH1. ☞ Page 92 (s)[4] (CH1 Communication protocol setting) and (s)[6] (CH2 Communication protocol setting)	0 to 8	
(s)[5]	CH2 Transmission specification setting	Set the transmission specifications for CH2. ☞ Page 91 (s)[3] (CH1 Transmission specification setting) and (s)[5] (CH2 Transmission specification setting)*1	0 to 0FFFH	
(s)[6]	CH2 Communication protocol setting	Set the communication protocol for CH2. ☞ Page 92 (s)[4] (CH1 Communication protocol setting) and (s)[6] (CH2 Communication protocol setting)	0 to 7	
(s)[7]	Station No. setting	Set the host station number.	0 to 31	
(s)[8] ⋮ (s)[12]	For system	Always specify '0'.	0	

■(s)[3] (CH1 Transmission specification setting) and (s)[5] (CH2 Transmission specification setting)*1



No.	Bit	Description	Setting value	Remarks
(1)	b0	Operation setting	OFF(0): Independence ON (1): Link	Always set the CH1 side ((s)[3]) to 0.
(2)	b1	Data bit	OFF(0): 7 ON(1): 8	Parity bit is not included.
(3)	b2	Parity bit	OFF(0): Without ON (1): With	Vertical parity
(4)	b3	Even/Odd parity	OFF(0): Odd ON (1): Even	Valid only when parity bit is set to 'With'.
(5)	b4	Stop bit	OFF(0): 1 ON(1): 2	—
(6)	b5	Sumcheck code	OFF(0): Without ON (1): With	—
(7)	b6	Write during RUN	OFF(0): Inhibited ON (1): Allowed	—
(8)	b7	Setting change	OFF(0): Inhibited ON (1): Allowed	—
(9)	b15 to b8	Communication speed	0FH: 50bps 00H: 300bps 01H: 600bps 02H: 1200bps 03H: 2400bps 04H: 4800bps 05H: 9600bps 06H: 14400bps 07H: 19200bps 08H: 28800bps 09H: 38400bps 0AH: 57600bps 0BH: 115200bps 0CH: 230400bps	• 230400bps is selectable only at CH1 side ((s)[3]). (Select 300bps at CH2 side ((s)[5]).) • The sum of communication speeds selected at CH1 side and CH2 side must be within 230400bps.

*1 Specify '0000H' at the CH side for which "MELSOFT connection" is specified in the communication protocol setting.

■(s)[4] (CH1 Communication protocol setting) and (s)[6] (CH2 Communication protocol setting)

Setting No.	Description		Remarks
0H	MELSOFT connection		Specify '0000H' for the transmission specification setting.
1H	MC protocol	Format 1	—
2H		Format 2	—
3H		Format 3	—
4H		Format 4	—
5H		Format 5	—
6H	Nonprocedural protocol		—
7H	Bidirectional protocol		—
8H	For link setting		Setting is possible only for CH1 side ((s)[4])
9H	Pre-defined protocol		Pre-defined protocol communication

Precautions

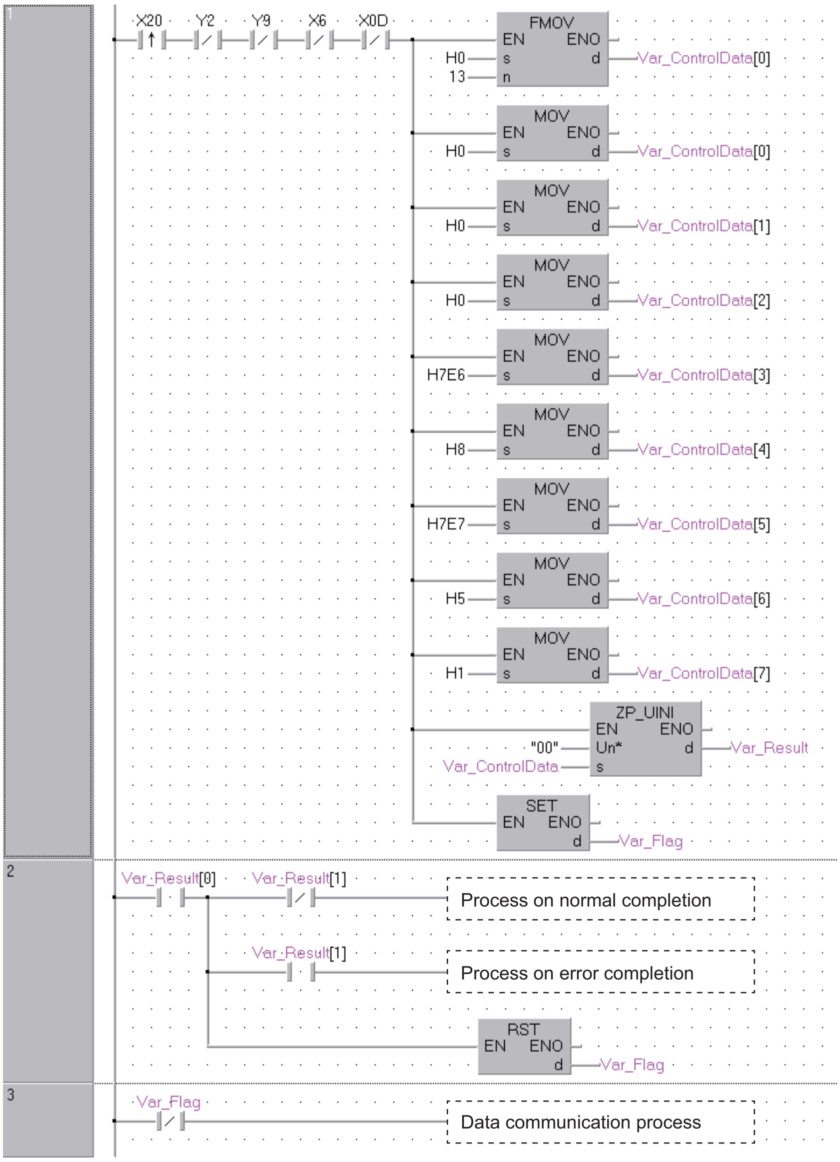
The UINI instruction is applicable to the QJ71C24N (-R2/R4) of which the function version is B and the first five digits of the serial number are '06062' or higher.

Program example

- The following program changes settings of the Q series C24 mounted on the I/O numbers X/Y00 to X/Y1F as follows when X20 turns ON.

Device	Bit		Description			Setting value
	Position	Specified value				
(s)[3]	b0	OFF	CH1 Transmission specification setting	Operation setting	Independence	07E6H
	b1	ON		Data bit	8	
	b2	ON		Parity bit	With	
	b3	OFF		Even/Odd parity	Odd	
	b4	OFF		Stop bit	1	
	b5	ON		Sumcheck code	With	
	b6	ON		Write during RUN	Allowed	
	b7	ON		Setting change	Allowed	
	b15 to b8	—		Communication speed	19200bps	
(s)[4]	—		CH1 Communication protocol setting		Link setting	0008H
(s)[5]	b0	ON	CH2 Transmission specification setting	Operation setting	Link	07E7H
	b1	ON		Data bit	8	
	b2	ON		Parity bit	With	
	b3	OFF		Even/Odd parity	Odd	
	b4	OFF		Stop bit	1	
	b5	ON		Sumcheck code	With	
	b6	ON		Write during RUN	Allowed	
	b7	ON		Setting change	Allowed	
	b15 to b8	—		Communication speed	19200bps	
(s)[6]	—		CH2 Communication protocol setting		MC protocol Format 5	0005H
(s)[7]	—		Station No. setting		1	0001H

[Structured ladder/FBD]



UINI instruction command

Always sets 0

Clears control data to 0

Sets execution type

Sets CH1 transmission specification

Sets CH1 communication protocol

Sets CH2 transmission specification

Sets CH2 communication protocol

Sets host station number

Switches mode

Turns interlock signal for communication stop ON*1

Normal completion

Error completion

Turns interlock signal for communication stop OFF*1

*1 Create a program so that the data communication process does not run while the interlock signal for communication stop is ON.

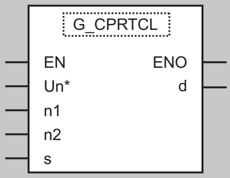
```
[ST]
IF(LDP(TRUE,X20) (* UINI instruction command *)
&(Y2=FALSE) (* CH1 mode switching request *)
    &(Y9=FALSE) (* CH2 mode switching request *)
    &(X6=FALSE) (* CH1 mode switching *)
    &(X0D=FALSE))THEN (* CH2 mode switching *)
        (* Runs if there is no mode switching *)
    FMOV(TRUE, H0, 13, Var_ControlData[0]); (* Clears control data to 0 *)
    MOV(TRUE, H0, Var_ControlData[0]); (* Always sets 0 *)
    MOV(TRUE, H0, Var_ControlData[1]); (* Clears execution result to 0 *)
    MOV(TRUE, H0, Var_ControlData[2]); (* Sets execution type *)
    MOV(TRUE,H7E6,Var_ControlData[3]); (* Sets CH1 transmission specification *)
    MOV(TRUE,H8,Var_ControlData[4]); (* Sets CH1 communication protocol *)
    MOV(TRUE, H7E7, Var_ControlData[5]); (* Sets CH2 transmission specification *)
    MOV(TRUE, H5, Var_ControlData[6]); (* Sets CH2 communication protocol *)
    MOV(TRUE, H1, Var_ControlData[7]); (* Sets host station number *)
    ZP_UINI(TRUE, "00", Var_ControlData, Var_Result); (* Switches mode *)
    SET(TRUE, Var_Flag ); (* Turns interlock signal for communication stop ON *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
    RST(TRUE, Var_Flag); (* Turns interlock signal for communication stop OFF *)*1
END_IF;

(* Do not perform the data communication process during interlock signal for communication stop ON *)
IF(Var_Flag=FALSE)*1 THEN
(* Data communication process *)
END_IF;
```

*1 Create a program so that the data communication process does not run while the interlock signal for communication stop is ON.

Pre-defined protocol communication

G(P)_CPRTCL

Serial

Structured ladder/FBD	ST
	ENO:= G_CPRTCL (EN, Un*, n1, n2, s, d);

The following instruction can go in the dotted squares.
G_CPRTCL, GP_CPRTCL

■Executing condition

Instruction	Executing condition
G_CPRTCL	
GP_CPRTCL	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	n1	Channel to communicate with other devices 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	ANY16
	n2	Number of consecutive protocol executions (1 to 8)	ANY16
	s	Start number of the device in which control data are stored	Array of ANY16 [0..17]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○		—				○	—
n2	—	○		—				○	—
(s)	—	○		—				—	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction executes the protocols and functional protocols written to the flash ROM by pre-defined protocol support function.

Setting data

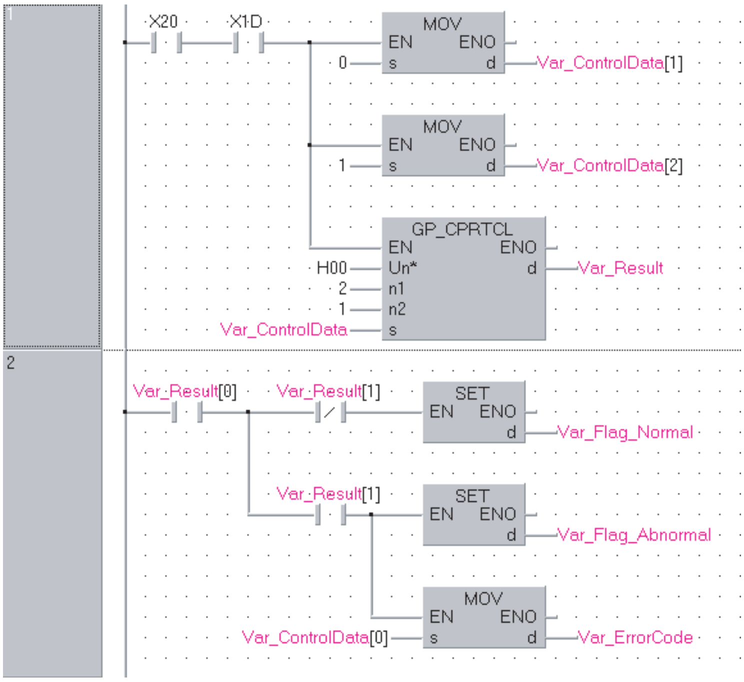
Device	Item	Setting data	Setting range	Setting side
(s)[0]	Completion status	The instruction completion status is stored. When executing multiple protocols, the execution result of the protocol executed at last is stored.*1[*1] 0: Normal completion Other than 0: Error completion (error code)[*2]	—	System
(s)[1]	Number of executions	The number of executions is stored. Protocols with errors are included in the count. When settings of the setting data and control data contain an error, "0" is stored.	1 to 8	System
(s)[2] ⋮ (s)[9]	Execution protocol number designation	Set the first protocol number or functional protocol number to be executed. ⋮ Set the 8th protocol number or functional protocol number to be executed.	1 to 128, 201 to 207	User
(s)[10] ⋮ (s)[17]	Verification match receive packet number	When the communication type of the first protocol executed is "Receive only" or "Send & receive", the matched receive packet number is stored. "0" is stored with the following condition. • When the communication type is "Send only" • If the error occurs to the first protocol executed • When the functional protocol is executed ⋮ When the communication type of the 8th protocol executed is "Receive only" or "Send & receive", the matched receive packet number is stored. "0" is stored with the following condition. • When the communication type is "Send only" • If the error occurs to the 8th protocol executed • When the number of the executed protocols is less than 8 • When the functional protocol is executed	0, 1 to 16	System

*1 When executing multiple protocols, if an error occurs to the nth protocol, the protocols after the nth protocol are not executed.
*2 For details of the error code at the error completion, refer to the following.
Q Corresponding Serial Communication Module User's Manual (Basic)
MELSEC-L Serial Communication Module User's Manual (Basic)

Program example

- This instruction executes the protocol specified in Var_ControlData[2] when X20 turns ON.

[Structured ladder/FBD]



Store the set value to the send data storage area

Designate protocol number 1

Execute the protocol Communicate in CH2

Normal completion flag ON

Abnormal completion flag ON

Store the error code

[ST]

```
IF((X20=TRUE) & (X1D=TRUE))THEN
    MOV(TRUE, 0, Var_ControlData[1]); (* Store the set value to the send data storage area *)
    MOV(TRUE, 1, Var_ControlData[2]); (* Designate protocol number 1 *)
    GP_CPRTCL(TRUE, H00, 2, 1, Var_ControlData, Var_Result); (* Execute the protocol Communicate in CH2 *)
END_IF;

IF(Var_Result[0]=TRUE)THEN
    IF(Var_Result[1]=FALSE)THEN
        SET(TRUE, Var_Flag_Normal); (* Normal completion flag ON *)
    ELSE
        SET(TRUE, Var_Flag_Abnormal); (* Abnormal completion flag ON *)
        MOV(TRUE, Var_ControlData[0], Var_ErrorCode); (* Store the error code *)
    END_IF;
END_IF;
```

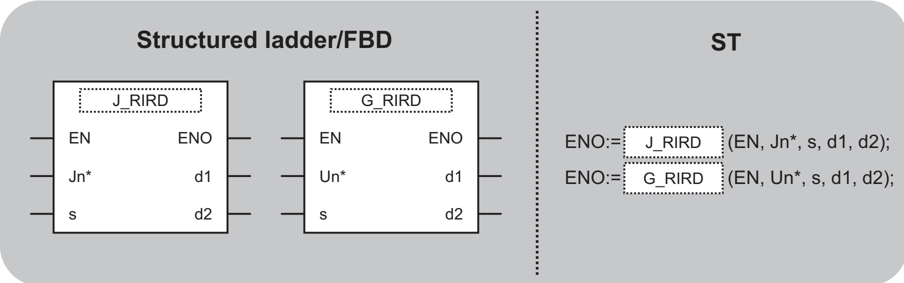
5

5.4 Network Dedicated Instruction

Reading from the buffer memory of an intelligent device station

J(P)_RIRD, G(P)_RIRD

CC-Link | CC IE C | CC IE F



The following instruction can go in the dotted squares.

J_RIRD, JP_RIRD, G_RIRD, GP_RIRD

■Executing condition

Instruction	Executing condition
J_RIRD G_RIRD	
JP_RIRD GP_RIRD	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..4]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

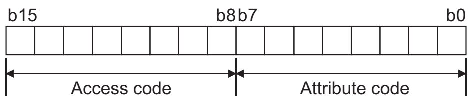
Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads data for the specified number of points from the buffer memory of the CC-Link module or the device of the programmable controller CPU module on the specified station.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code) For error codes when target station is anything other than master/local module, refer to the manual of the target station.	—	System
(s)[1]	Target station No.	Specify the station number of the target station.	0 to 64*1 0 to 120	User
(s)[2]	Access code, Attribute code	Specify the access code and attribute code of the device to be read. 	Page 99 Buffer memory of the CC-Link module,Page 100 Device memory of the programmable controller CPU module	
(s)[3]	Buffer memory address or device No.	Specify the start address of the buffer memory or the start number of the device.	Within the device range*2	
(s)[4]	Number of read points	Specify the number of data to be read (in units of words).	1 to 32*3 1 to 480*4	

*1 For G(P)_RIRD, the setting range shall be 0 to 64.

*2 For details, refer to the manual for the local station or the intelligent device station from which data are read.
When the random access buffer is specified, specify the start address of the random access buffer as 0.

*3 The value indicates the maximum number of data to be read.
Specify the value within the buffer memory capacity of the local station or the intelligent device station, or the receive buffer area setting range set by a parameter.

*4 When reading device data from the programmable controller CPU other than the QCPU (Q mode), QCPU (A mode) or QnACPU/AnUCPU, the setting range shall be 1 to 32 words.

■Buffer memory of the CC-Link module

Buffer memory		Access code	Attribute code
Buffer in an intelligent device station		00H	04H
Buffer in a master or local station	Random access buffer	20H	
	Remote input	21H	
	Remote output	22H	
	Remote register	24H	
	Link special relay	63H	
	Link special register	64H	

■Device memory of the programmable controller CPU module

Device*1	Name	Device type		Unit	Access code*2	Attribute code*2
		Bit	Word			
Input relay	X	○	—	Hexadecimal	01H	05H
Output relay	Y	○	—	Hexadecimal	02H	
Internal relay	M	○	—	Decimal	03H	
Latch relay	L	○	—	Decimal	83H	
Link relay	B	○	—	Hexadecimal	23H	
Timer (contact)	T	○	—	Decimal	09H	
Timer (coil)	T	○	—	Decimal	0AH	
Timer (current value)	T	—	○	Decimal	0CH	
Retentive timer (contact)	ST	○	—	Decimal	89H	
Retentive timer (coil)	ST	○	—	Decimal	8AH	
Retentive timer (current value)	ST	—	○	Decimal	8CH	
Counter (contact)	C	○	—	Decimal	11H	
Counter (coil)	C	○	—	Decimal	12H	
Counter (current value)	C	—	○	Decimal	14H	
Data register*3	D	—	○	Decimal	04H	
Link register*3	W	—	○	Hexadecimal	24H	
File register	R	—	○	Decimal	84H	
Link special relay	SB	○	—	Hexadecimal	63H	
Link special register	SW	—	○	Hexadecimal	64H	
Special relay	SM	○	—	Decimal	43H	
Special register	SD	—	○	Decimal	44H	

*1 Devices other than those listed above cannot be accessed.
When accessing a bit device, specify it with 0 or a multiple of 16.

*2 For access code/attribute code when target station is anything other than master/local module, refer to the manual of the target station.

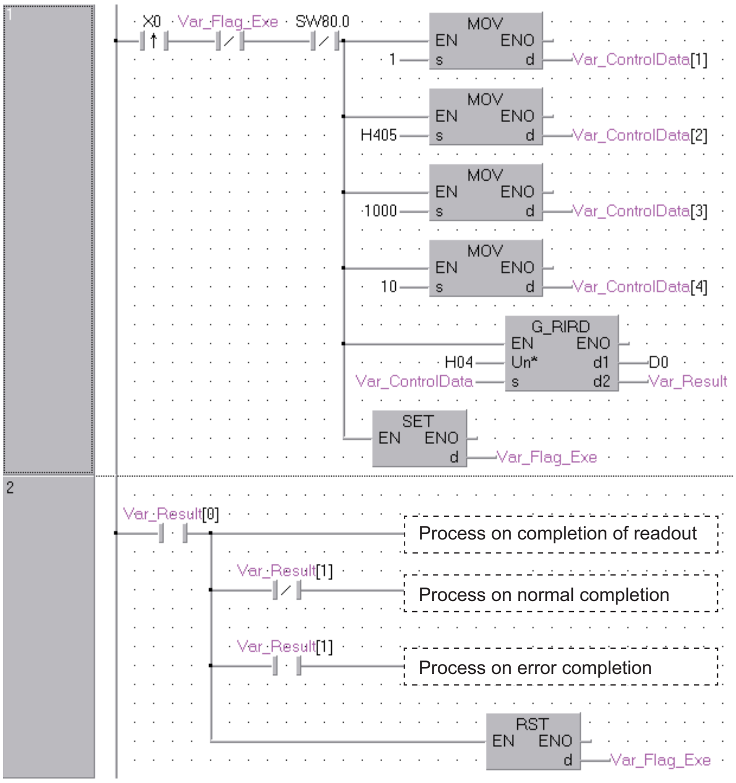
*3 D65536 and the following devices of extended data registers as well as W10000 and the following devices of extended link registers cannot be specified.

Program example

- The following program reads out 10-word data, which start from D1000 of the number 1 local station connected to the master module mounted on the I/O numbers from X/Y40 to X/Y5F, and stores the data in the devices starting from D0 when X0 turns ON.

(When the refresh device of the link special register (SW) is set to SW0.)

[Structured ladder/FBD]



Sets station number

Sets access code and attribute code

Sets device number

Sets number of read points

Performs readout

Turns execution flag ON

Turns execution flag OFF

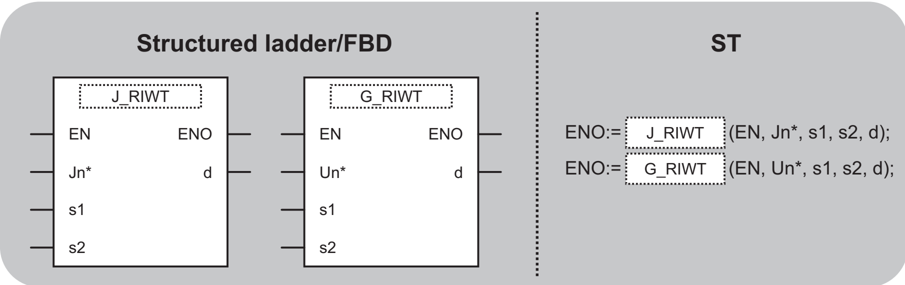
[ST]

```
IF((X0=TRUE)
    &(Var_Flag_Exe=FALSE) (* Execution flag *)
    &(SW80.0=FALSE))THEN (* Data link status of station number 1 *)
      MOV(TRUE,1, Var_ControlData[1]); (* Sets station number *)
      MOV(TRUE,H0405, Var_ControlData[2]); (* Sets access code and attribute code *)
      MOV(TRUE, 1000, Var_ControlData[3]); (* Sets device number *)
      MOV(TRUE, 10, Var_ControlData[4]); (* Sets number of read points *)
      G_RIRD(TRUE, H04, Var_ControlData, D0, Var_Result); (* Performs readout *)
      SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)

(* Process on completion of readout *)

      IF(Var_Result[1]=FALSE)THEN (* Normal completion *)

(* Process on normal completion *)

      ELSE (* Error completion *)

(* Process on error completion *)

      END_IF;

      RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

Writing to the buffer memory of an intelligent device station

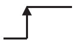
J(P)_RIWT, G(P)_RIWT

CC-Link CC IE C CC IE F



The following instruction can go in the dotted squares.
J_RIWT, JP_RIWT, G_RIWT, GP_RIWT

■Executing condition

Instruction	Executing condition
J_RIWT G_RIWT	
JP_RIWT GP_RIWT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..4]
	s2	Start number of the device that stores write data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

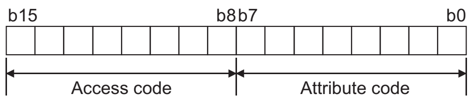
Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction writes the data for the specified number of points to the buffer memory of the CC-Link module or the device of the programmable controller CPU module on the specified station.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code) For error codes when target station is anything other than master/local module, refer to the manual of the target station.	—	System
(s1)[1]	Target station No.	Specify the station number of the target station	0 to 64*1 0 to 120	User
(s1)[2]	Access code Attribute code	Specify the access code and attribute code of the device to be read.  	☞ Page 103 Buffer memory of the CC-Link module,Page 104 Device memory of the programmable controller CPU module	
(s1)[3]	Buffer memory address or device No.	Specify the start address of the buffer memory or the start number of the device.	Within the device range*2	
(s1)[4]	Number of write points	Specify the number of data to be written (in units of words).	1 to 10*3 1 to 480*4	

*1 For G(P)_RIWT, the setting range shall be 0 to 64.

*2 For details, refer to the manual for the local station or the intelligent device station to which data are written.
When the random access buffer is specified, specify the start address of the random access buffer as 0.

*3 When writing device data to the programmable controller CPU other than the QCPU (Q mode), QCPU (A mode) or QnACPU/AnUCPU, the setting range shall be 1 to 10 words.

*4 The value indicates the maximum number of data to be written.
Specify the value within the buffer memory capacity of the local station or the intelligent device station, or the send buffer area setting range set by a parameter.

■Buffer memory of the CC-Link module

Buffer memory category		Access code	Attribute code
Buffer memory		00H	04H
Buffer in a master or local station	Random access buffer	20H	
	Remote input	21H	
	Remote output	22H	
	Remote register	24H	
	Link special relay	63H	
	Link special register	64H	

■Device memory of the programmable controller CPU module

Device*1	Name	Device type		Unit	Access code*2	Attribute code*2
		Bit	Word			
Input relay	X	○	—	Hexadecimal	01H	05H
Output relay	Y	○	—	Hexadecimal	02H	
Internal relay	M	○	—	Decimal	03H	
Latch relay	L	○	—	Decimal	83H	
Link relay	B	○	—	Hexadecimal	23H	
Timer (contact)	T	○	—	Decimal	09H	
Timer (coil)	T	○	—	Decimal	0AH	
Timer (current value)	T	—	○	Decimal	0CH	
Retentive timer (contact)	ST	○	—	Decimal	89H	
Retentive timer (coil)	ST	○	—	Decimal	8AH	
Retentive timer (current value)	ST	—	○	Decimal	8CH	
Counter (contact)	C	○	—	Decimal	11H	
Counter (coil)	C	○	—	Decimal	12H	
Counter (current value)	C	—	○	Decimal	14H	
Data register*3	D	—	○	Decimal	04H	
Link register*3	W	—	○	Hexadecimal	24H	
File register	R	—	○	Decimal	84H	
Link special relay	SB	○	—	Hexadecimal	63H	
Link special register	SW	—	○	Hexadecimal	64H	
Special relay	SM	○	—	Decimal	43H	
Special register	SD	—	○	Decimal	44H	

*1 Devices other than those listed above cannot be accessed.
When accessing a bit device, specify it with 0 or a multiple of 16.

*2 For access code/attribute code when target station is anything other than master/local module, refer to the manual of the target station.

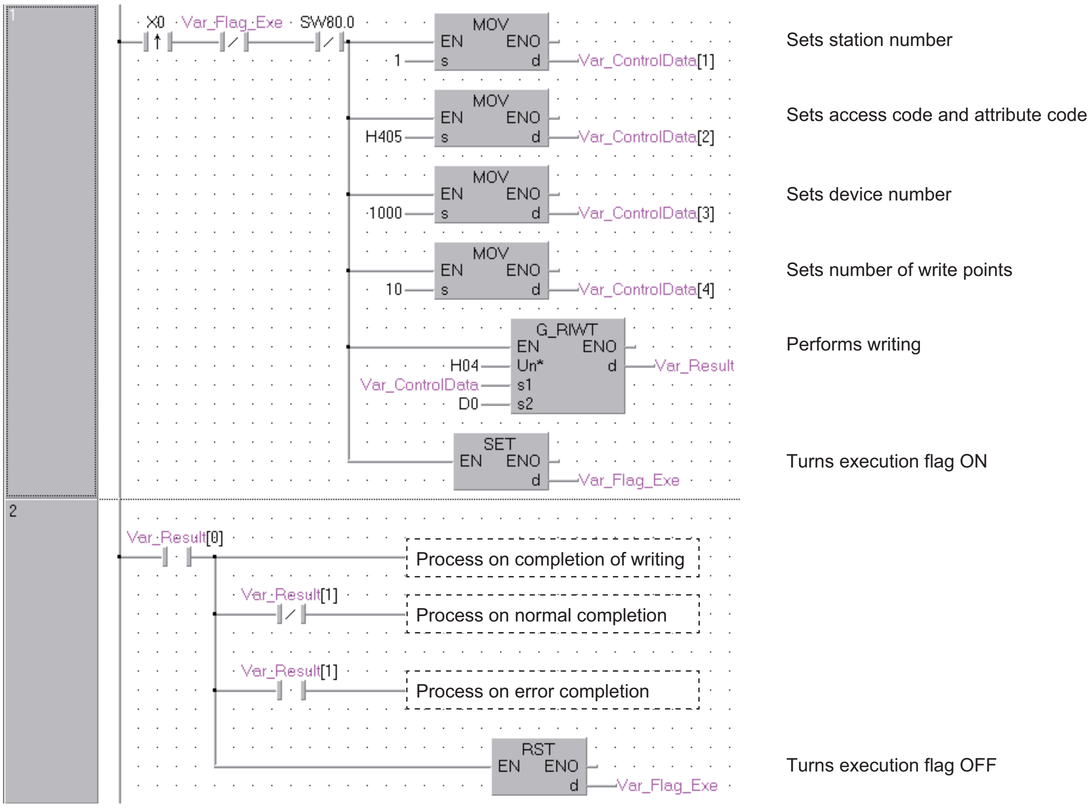
*3 D65536 and the following devices of extended data registers as well as W10000 and the following devices of extended link registers cannot be specified.

Program example

- The following program stores 10-word data, which are stored in the devices starting from D0, to the devices starting from D1000 of the number 1 local station connected to the master module mounted on the I/O numbers from X/Y40 to X/Y5F when X0 turns ON.

(When the refresh device of the link special register (SW) is set to SW0.)

[Structured ladder/FBD]



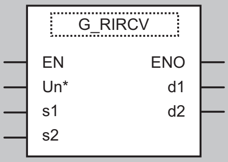
[ST]

```
IF((X0=TRUE)
    &(Var_Flag_Exe=FALSE) (* Execution flag *)
    &(SW80.0=FALSE))THEN (* Data link status of station number 1 *)
      MOV(TRUE, 1, Var_ControlData[1]); (* Sets station number *)
      MOV(TRUE, H0405, Var_ControlData[2]); (* Sets access code and attribute code *)
      MOV(TRUE, 1000, Var_ControlData[3]); (* Sets device number *)
      MOV(TRUE, 10, Var_ControlData[4]); (* Sets number of read points *)
      G_RIWT(TRUE, H04, Var_ControlData, D0, Var_Result); (* Performs writing *)
      SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of writing *)
      IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
      ELSE (* Error completion *)
(* Process on error completion *)
      END_IF;

      RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

RIRCV instruction

G(P)_RIRCV

CC-Link

Structured ladder/FBD	ST
	ENO:= G_RIRCV (EN, Un*, s1, s2, d1, d2);

The following instruction can go in the dotted squares.
G_RIRCV, GP_RIRCV

■Executing condition

Instruction	Executing condition
G_RIRCV	
GP_RIRCV	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..4]
	s2	Variable that stores interlock signal	Array of ANY16 [0..2]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction automatically performs handshaking with an intelligent device station and reads data from the buffer memory of the specified intelligent device station.
This instruction is applicable with a module having a handshake signal, such as the AJ65BT-R2(N).

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[1]	Station No.	Specify the station number of the intelligent device station.	0 to 64	User
(s1)[2]	Access code, Attribute code	Set '0004H'.	0004H	User
(s1)[3]	Buffer memory address	Specify the start address of the buffer memory.	*1	User
(s1)[4]	Number of read points	Specify the number of data to be read (in units of words).	1 to 480*2	User

*1 For details, refer to the manual for the intelligent device station from which data are read.

*2 The value indicates the maximum number of data to be read.
Specify the value within the buffer memory capacity of the intelligent device station or the receive buffer area setting range set by a parameter.

■Interlock signal storage device

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	b15 … b8 b7 … b0 [0][RY]	RY: Request device	0 to 127	User
		Set the high-order 8 bits to 0.	0	User
(s2)[1]	b15 … b8 b7 … b0 [RWr*1][RX]	RX: Completion device	0004H	User
		RWr: Error code storage device Set FFH when no error code storage device exists.	0 to 15, FFH	User
(s2)[2]	b15 … b0 [Completion mode]	0: Completes with the content of one device (RXn). 1: Completes with the content of two devices (RXn, RXn + 1). (RXn + 1 turns ON upon abnormal completion of the instruction.)	0/1	User

*1 The same error code as that for the completion status of control data are stored in the error code storage device.

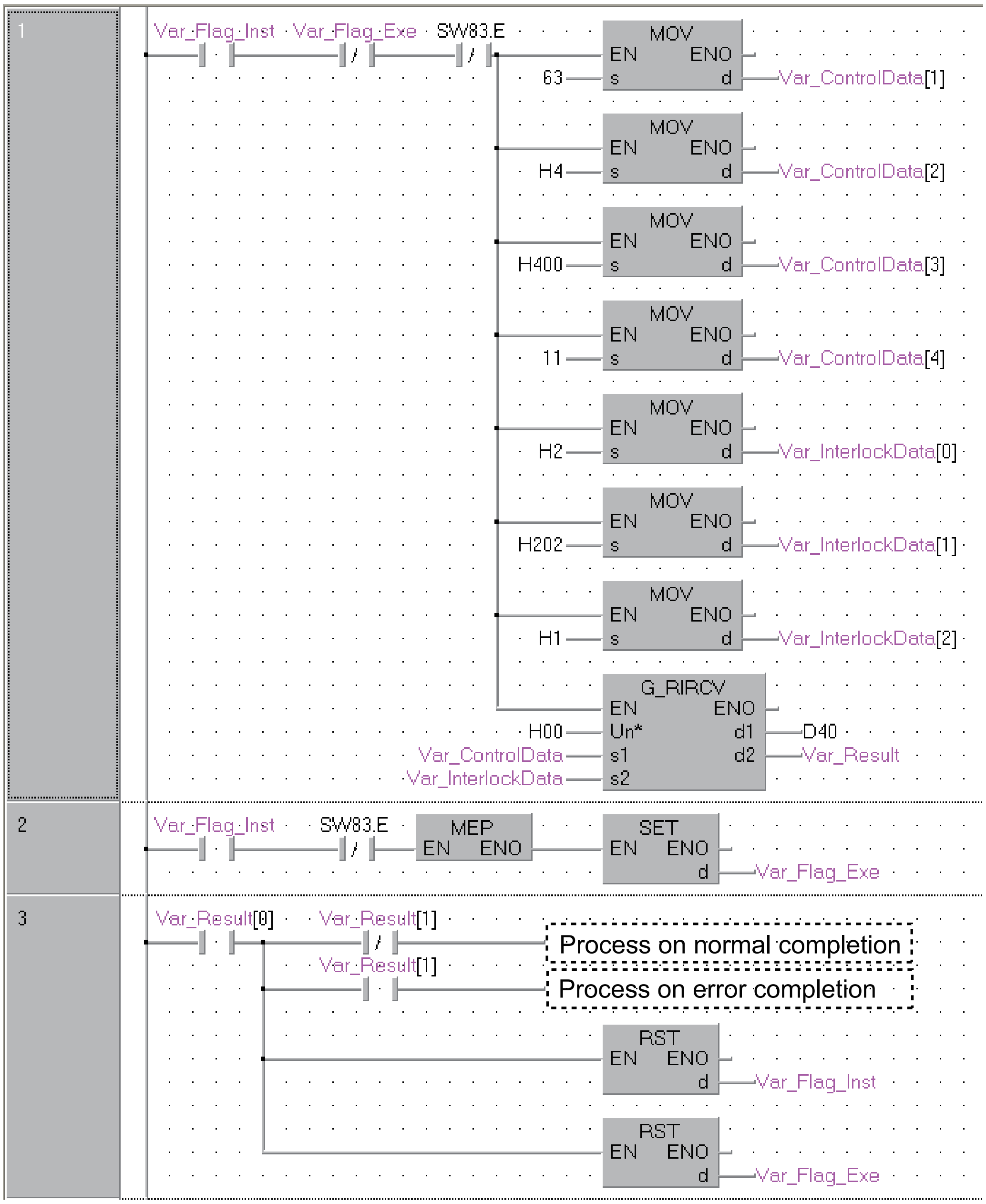
5

Program example

The following program reads 11-word data, which are stored in buffer memory starting from the buffer memory address 400H of the number 63 intelligent device station (AJ65BT-R2(N)) connected to the master module mounted on the I/O numbers X/Y00 to X/Y1F, and stores the data in the devices starting from D40.

The interlock signal storage is set to request device: RY2, completion device: RX2, error code storage device: RWr2, and completion mode: 1.
(When the refresh device of the link special register (SW) is set to SW0.)

[Structured ladder/FBD]



Sets station number

Sets access code and attribute code

Sets buffer memory address

Sets number of read points

Sets request device

Sets completion device and error code storage area

Sets completion mode

Performs readout

Turns execution flag ON

Turns read request OFF

Turns execution flag OFF

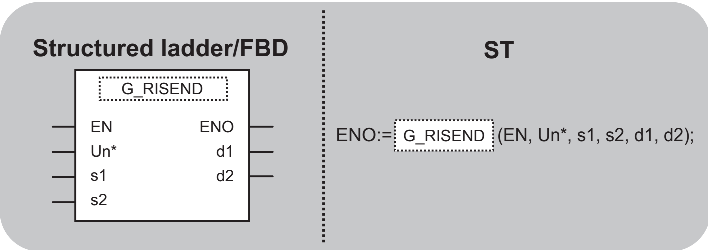
```
[ST]
IF((Var_Flag_Inst=TRUE) (* Read request ON *)
    &(Var_Flag_Exe=FALSE) (* Execution flag *)
    &(SW83.E=FALSE))THEN (* Data link status of station number 63 *)
        (* Sets control data *)
        MOV(TRUE, 63, Var_ControlData[1]); (* Sets station number *)
        MOV(TRUE,H4, Var_ControlData[2]); (* Sets access code and attribute code *)
        MOV(TRUE, H400, Var_ControlData[3]); (* Sets buffer memory address *)
        MOV(TRUE, 11, Var_ControlData[4]); (* Sets number of read points *)

        (* Sets interlock signal storage device *)
        MOV(TRUE, H2, Var_InterlockData[0]); (* Sets request device *)
        MOV(TRUE, H202, Var_InterlockData[1]); (* Sets completion device and error code storage area *)
        MOV(TRUE, H1, Var_InterlockData[2]); (* Sets completion mode *)

        G_RIRCV(TRUE, H00, Var_ControlData, Var_InterlockData,D40, Var_Result); (* Performs readout *)
END_IF;

IF(MEP((Var_Flag_Inst=TRUE) & (SW83.E=FALSE)))THEN (* Read request is ON and data link status of station number 63 is OFF (rising pulse) *)
        SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
        IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
        ELSE (* Error completion *)
(* Process on error completion *)
        END_IF;

        RST(TRUE, Var_Flag_Inst); (* Turns read request OFF *)
        RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

5

RISEND instruction

G(P)_RISEND

CC-Link



The following instruction can go in the dotted squares.
G_RISEND, GP_RISEND

■Executing condition

Instruction	Executing condition
G_RISEND	
GP_RISEND	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..4]
	s2	Variable that stores interlock signal	Array of ANY16 [0..2]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores write data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction automatically performs handshaking with an intelligent device station and writes data to the buffer memory of the specified intelligent device station.
This instruction is applicable with a module having a handshake signal, such as the AJ65BT-R2(N).

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[1]	Station No.	Specify the station number of the intelligent device station.	0 to 64	User
(s1)[2]	Access code, Attribute code	Set '0004H'.	0004H	User
(s1)[3]	Buffer memory address	Specify the start address of the buffer memory.	*1	User
(s1)[4]	Number of write points	Specify the number of data to be written (in units of words).	1 to 480*2	User

*1 For details, refer to the manual for the intelligent device station to which data are written.
*2 The value indicates the maximum number of data to be written.
Specify the value within the buffer memory capacity of the intelligent device station or the receive buffer area setting range set by a parameter.

■Interlock signal storage device

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	b15 … b8 b7 … b0 0 RY	RY: Request device	0 to 127	User
		Set the high-order 8 bits to 0.	0	User
(s2)[1]	b15 … b8 b7 … b0 RWr*1 RX	RX: Completion device	0 to 127	User
		RWr: Error code storage device Set FFH when no error code storage device exists.	0 to 15, FFH	User
(s2)[2]	b15 … b0 Completion mode	0: Completes with the content of one device (RXn). 1: Completes with the content of two devices (RXn, RXn + 1). (RXn + 1 turns ON upon abnormal completion of the instruction.)	0/1	User

*1 The same error code as that for the completion status of control data are stored in the error code storage device.

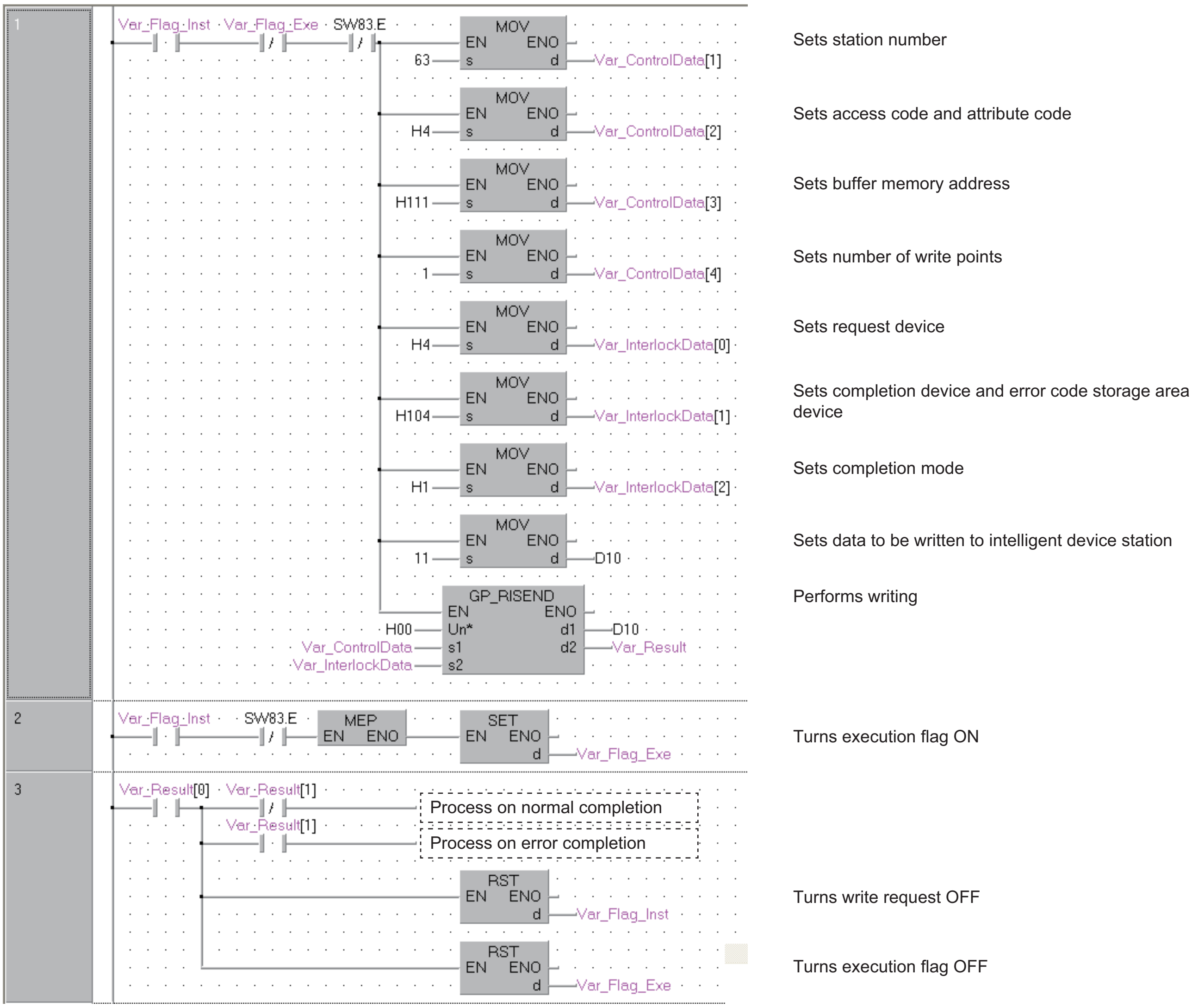
Program example

The following program writes 1-word data of D10 to the buffer memory address 111H of the number 63 intelligent device station (AJ65BT-R2(N)) which is connected to the master module mounted on the I/O numbers from X/Y00 to X/Y1F.

The interlock signal storage settings are set to request device: RY4, completion device: RX4, error code storage device: RWr1, and completion mode: 1.

(When the refresh device of the link special register (SW) is set to SW0.)

[Structured ladder/FBD]



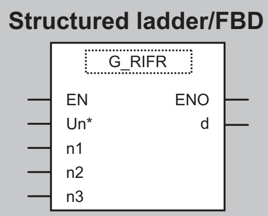
[ST]
```
IF((Var_Flag_Inst=TRUE) (* Write request ON *)
   &(Var_Flag_Exe=FALSE) (* Execution flag *)
   &(SW83.E=FALSE))THEN (* Data link status of station number 63 *)
      (* Sets control data *)
      MOV(TRUE, 63, Var_ControlData[1]); (* Sets station number *)
      MOV(TRUE, H4, Var_ControlData[2]); (* Sets access code and attribute code *)
      MOV(TRUE, H111, Var_ControlData[3]); (* Sets buffer memory address *)
      MOV(TRUE, 1, Var_ControlData[4]); (* Sets number of write points *)

      (* Sets interlock signal storage device *)
      MOV(TRUE, H4, Var_InterlockData[0]); (* Sets request device *)
      MOV(TRUE, H104, Var_InterlockData[1]); (* Sets completion device and error code storage area device *)
      MOV(TRUE, H1, Var_InterlockData[2]); (* Sets completion mode *)
      (* Sets data to be written to intelligent device station *)
      MOV(TRUE, 11, D10);

      GP_RISEND(TRUE, H00, Var_ControlData, Var_InterlockData,D10, Var_Result); (* Performs writing *)
END_IF;
IF(MEP((Var_Flag_Inst=TRUE) & (SW83.E=FALSE)))THEN (* Write request is ON and data link status of station number 63 is OFF (rising pulse) *)
      SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
      IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
 (* Process on normal completion *)
      ELSE (* Error completion *)
 (* Process on error completion *)
      END_IF;

      RST(TRUE, Var_Flag_Inst); (* Turns write request OFF *)
      RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

5

Reading from the auto-refresh buffer memory of the master station

G(P)_RIFR

CC-Link



ST

ENO:= G_RIFR (EN, Un*, n1, n2, n3, d);

The following instruction can go in the dotted squares.

G_RIFR, GP_RIFR

■Executing condition

Instruction	Executing condition
G_RIFR	
GP_RIFR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	n1	Intelligent device station number (1 to 64) Random access buffer specification (FFH)	ANY16
	n2	Offset value of specified intelligent device auto-refresh buffer or random access buffer of the master station	ANY16
	n3	Number of read points (0 to 4096) No processing is performed with setting '0'.	ANY16
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores read data	ANY16

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○		—				○	—
n2	○	○		—				○	—
n3	○	○		—				○	—
(d)	—	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

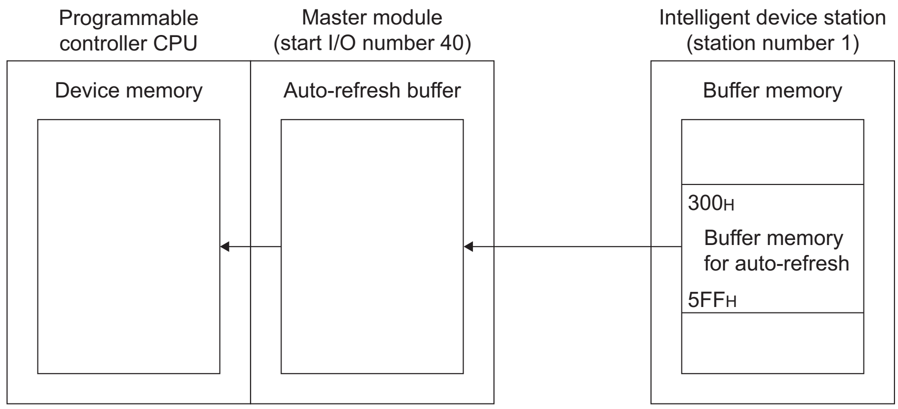
This instruction reads data from the auto-refresh buffer of the specified station.

The instruction is applicable with a module having an auto-refresh buffer, such as the AJ65BT-R2(N).

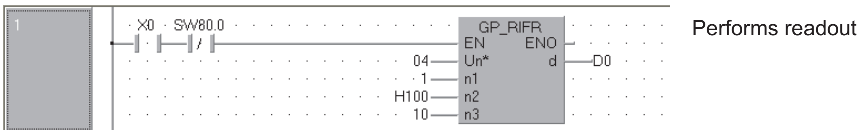
Program example

- The following program reads out 10-word data from buffer memory starting from the offset value 100 of the auto-refresh buffer of the master module (400H in the intelligent device station) and stores the data in the devices starting from D0 when X0 turns ON.

(When the refresh device of the link special register (SW) is set to SW0.)



[Structured ladder/FBD]



[ST]

```
IF((X0=TRUE) & (SW80.0=FALSE))THEN
    GP_RIFR(TRUE, H04, 1, H100, 10, D0); (* Performs readout *)
END_IF;
```

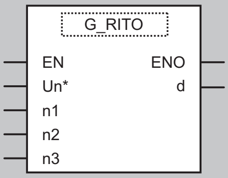
5

Writing to the auto-refresh buffer memory of the master station

G(P)_RITO

CC-Link

Structured ladder/FBD



ST

ENO:= G_RITO (EN, Un*, n1, n2, n3, d);

The following instruction can go in the dotted squares.
G_RITO, GP_RITO

■Executing condition

Instruction	Executing condition
G_RITO	
GP_RITO	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	n1	Intelligent device station number (1 to 64) Random access buffer specification (FFH)	ANY16
	n2	Offset value of specified intelligent device auto-refresh buffer or random access buffer of the master station	ANY16
	n3	Number of write points	ANY16
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores write data	ANY16

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○	○		—				○	—
n2	○	○		—				○	—
n3	○	○		—				○	—
(d)	—	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

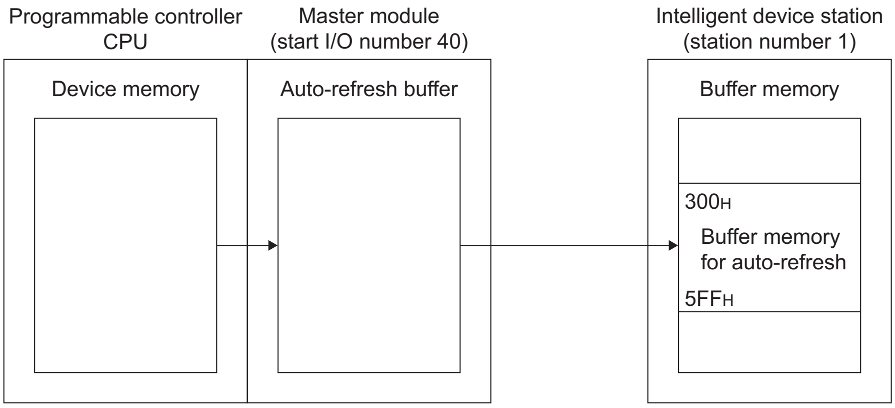
Processing details

This instruction writes the data to the auto-refresh buffer of the specified station.
The instruction is applicable with a module having an auto-refresh buffer, such as the AJ65BT-R2(N).

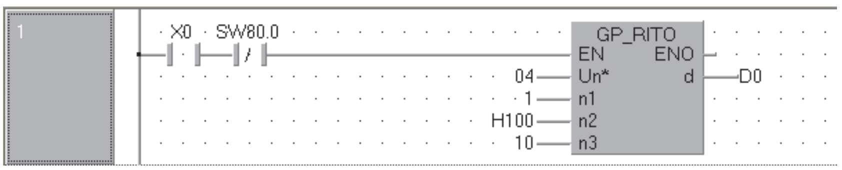
Program example

- The following program write 10-word data which are stored in the devices starting from D0 into buffer memory starting the offset value 100 of the auto-refresh buffer of the master module (400H in the intelligent device station) when X0 turns ON.

(When the refresh device of the link special register (SW) is set to SW0.)



[Structured ladder/FBD]



[ST]

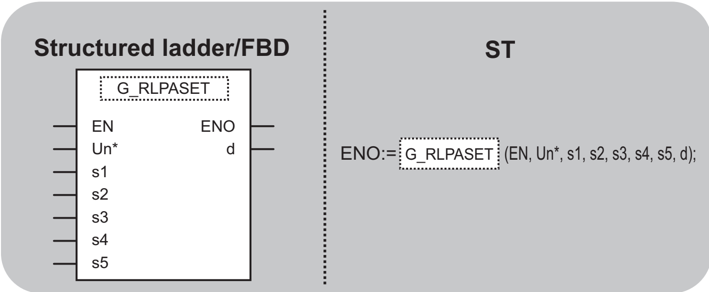
```
IF((X0=TRUE) & (SW80.0=FALSE))THEN
    GP_RITO(TRUE, H04, 1, H100, 10, D0); (* Performs writing *)
END_IF;
```

5

Network parameter setting

G(P)_RLPASET

CC-Link



The following instruction can go in the dotted squares.

G_RLPASET, GP_RLPASET

■Executing condition

Instruction	Executing condition
G_RLPASET	
GP_RLPASET	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module 00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..7]
	s2	Variable that stores slave station setting data	Array of ANY16 [0..63]
	s3	Variable that stores reserved station specification data	Array of ANY16 [0..3]
	s4	Variable that stores error invalid station specification data	Array of ANY16 [0..3]
	s5	Variable that stores send/receive and auto-refresh buffer assignment data	Array of ANY16 [0..77]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

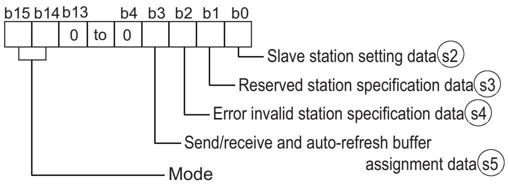
Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(s3)	—	○		—					
(s4)	—	○		—					
(s5)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

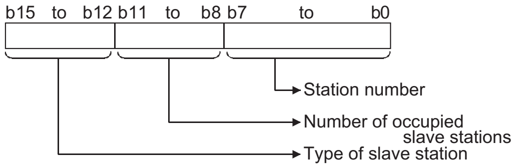
This instruction sets the network parameters to the master station and starts up the data link.

Setting data

Device	Item	Setting data	Setting range*2	Setting side
(s1)[0]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[1]	Setting flag	Specify the validity of each setting data from (s2) to (s5). 0: Invalid*1 1: Valid  00: Remote net (Ver. 1 mode) 01: Remote net (Additional mode) 10: Remote net (Ver. 2 mode) 11: Cannot be used	—	User
(s1)[2]	Number of connected modules	Set the number of connected slave stations.	1 to 64	User
(s1)[3]	Number of retries	Set the number of retries to be performed to a communication error station.	1 to 7	User
(s1)[4]	Number of automatic return modules	Set the number of slave stations that can be returned in one link scan.	1 to 10	User
(s1)[5]	Operation specification when CPU is down	Specify the data link status when a master station programmable controller CPU error occurs. 0: Stop 1: Continue	0, 1	User
(s1)[6]	Scan mode specification	Specify the link scan mode for sequence scan. 0: Asynchronous 1: Synchronous	0, 1	User
(s1)[7]	Delay time specification	Set '0' for the delay time.	0	User

*1 For the setting data for which invalid is specified, default parameter is applied.
*2 Setting a value outside the setting range results in error completion of the instruction.

■Slave station setting data

<table>
<tr><th>Device</th><th>Item</th><th>Setting data</th><th>Setting range</th><th>Setting side</th></tr>
<tr><td rowspan="4">(s2)[0]
⋮
(s2)[63]</td><td rowspan="4">Setting for 1 to 64 modules[*1]</td><td>Set the slave station type, the number of occupied slave stations, and the station number as shown below.


Default parameter setting is '0101H to 0140H (station number: 1 to 64, number of occupied slave stations: 1, type of slave station: Ver.1 compatible remote I/O station)'</td><td>—</td><td rowspan="4">User</td></tr>
<tr><td>Setting of station number
1 to 64 (BIN setting)</td><td>1 to 40H</td></tr>
<tr><td>Setting of the number of occupied slave stations
The setting value for the number of occupied slave stations is described below.
1H: 1 station
2H: 2 stations
3H: 3 stations
4H: 4 stations</td><td>1 to 4H</td></tr>
<tr><td>Setting of slave station type[*2]
The setting value for the slave station type is described below.
0H: Ver.1 compatible remote I/O station
1H: Ver.1 compatible remote device station
2H: Ver.1 compatible intelligent device station
5H: Ver.2 compatible single remote device station
6H: Ver.2 compatible single intelligent device station
8H: Ver.2 compatible double remote device station
9H: Ver.2 compatible double intelligent device station
BH: Ver.2 compatible quadruple remote device station
CH: Ver.2 compatible quadruple intelligent device station
EH: Ver.2 compatible octuple remote device station
FH: Ver.2 compatible octuple intelligent device station</td><td>0 to FH</td></tr>
</table>

*1 Set the same number which was set for Number of connected modules in the control data.
*2 Setting a value outside the setting range in the setting of slave station type results in error completion of the instruction.

■Reserved station specification data

<table>
<tr><th>Device</th><th>Item</th><th>Setting data</th><th>Setting range</th><th>Setting side</th></tr>
<tr><td>(s3)[0]
⋮
(s3)[3]</td><td>Specification for 1 to 64 stations[*1]</td><td>Specify the reserved station.[*2]
0: Not specified
1: Specified
<table>
<tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>to</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>(s3)[0]</td><td>16</td><td>15</td><td>14</td><td>13</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td></tr>
<tr><td>(s3)[1]</td><td>32</td><td>31</td><td>30</td><td>29</td><td>to</td><td>20</td><td>19</td><td>18</td><td>17</td></tr>
<tr><td>(s3)[2]</td><td>48</td><td>47</td><td>46</td><td>45</td><td>to</td><td>36</td><td>35</td><td>34</td><td>33</td></tr>
<tr><td>(s3)[3]</td><td>64</td><td>63</td><td>62</td><td>61</td><td>to</td><td>52</td><td>51</td><td>50</td><td>49</td></tr>
</table>
1 to 64 in the table indicates a station number.
Default parameter setting is '0: Not specified' for all stations.</td><td>—</td><td>User</td></tr>
</table>

*1 Set the parameter up to the largest station number set in the slave station setting data.
*2 Set the parameter only to the start station number of the module for the remote station/local station/intelligent device station that occupies two or more stations.

■Error invalid station specification data

<table>
<tr><th>Device</th><th>Item</th><th>Setting data</th><th>Setting range</th><th>Setting side</th></tr>
<tr><td>(s4)[0]
:
(s4)[3]</td><td>Specification for 1 to 64 stations*1</td><td>Specify the error invalid station.*2
0: Not specified
1: Specified
<table>
<tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>to</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>(s4)[0]</td><td>16</td><td>15</td><td>14</td><td>13</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td></tr>
<tr><td>(s4)[1]</td><td>32</td><td>31</td><td>30</td><td>29</td><td>to</td><td>20</td><td>19</td><td>18</td><td>17</td></tr>
<tr><td>(s4)[2]</td><td>48</td><td>47</td><td>46</td><td>45</td><td>to</td><td>36</td><td>35</td><td>34</td><td>33</td></tr>
<tr><td>(s4)[3]</td><td>64</td><td>63</td><td>62</td><td>61</td><td>to</td><td>52</td><td>51</td><td>50</td><td>49</td></tr>
</table>
1 to 64 in the table indicates a station number.
Default parameter setting is '0: Not specified' for all stations.</td><td>—</td><td>User</td></tr>
</table>

*1 Set the parameter up to the largest station number set in the slave station setting data.

*2 Set the parameter only to the start station number of the module for the remote station/local station/intelligent device station that occupies two or more stations.
Reserved station specification has a priority when an error invalid station and reserved station are specified for the same station.

■Send/receive and auto-refresh buffer assignment data

<table>
<tr><th>Device</th><th>Item</th><th>Setting data</th><th>Setting range</th><th>Setting side</th></tr>
<tr><td>(s5)[0]
:
(s5)[77]</td><td>Specification for 1 to 26 modules*1</td><td>Specify the buffer memory size assignment at transient transmission for local stations and intelligent device stations.
<table>
<tr><td>(s5)[0]</td><td>Send buffer size</td><td rowspan="3">Setting for the 1st module</td></tr>
<tr><td>(s5)[1]</td><td>Receive buffer size</td></tr>
<tr><td>(s5)[2]</td><td>Auto-refresh buffer size</td></tr>
<tr><td></td><td>⋮</td><td></td></tr>
<tr><td>(s5)[75]</td><td>Send buffer size</td><td rowspan="3">Setting for the 26th module</td></tr>
<tr><td>(s5)[76]</td><td>Receive buffer size</td></tr>
<tr><td>(s5)[77]</td><td>Auto-refresh buffer size</td></tr>
</table>
Default parameter setting is 'send buffer size: 40H, receive buffer size: 40H, auto-refresh buffer size: 80H'.</td><td>Send/receive buffer*2
: 0H (no setting) 40H to 1000H
0 (word) (no setting)
64 to 4096 (words)

Auto-refresh buffer*3
: 0H (no setting) 80H to 1000H
0 (word) (no setting)
128 to 4096 (words)</td><td>User</td></tr>
</table>

*1 Set the assignment data, in ascending order, for the stations set for a local station or intelligent device station in the slave station setting data.

*2 Keep the total of the send/receive buffer size within 1000H (4096 (words)).
Specify the size added seven words to the size of send/receive data as the send/receive buffer size.
Setting a value outside the setting range results in error completion of the instruction.

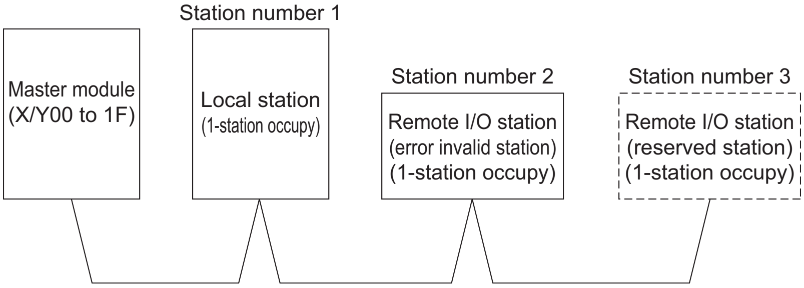
*3 Keep the total of the auto-refresh buffer size within 1000H (4096 (words)).
Specify the necessary auto-refresh buffer size for each intelligent device station.
Setting a value outside the setting range results in error completion of the instruction.

Precautions

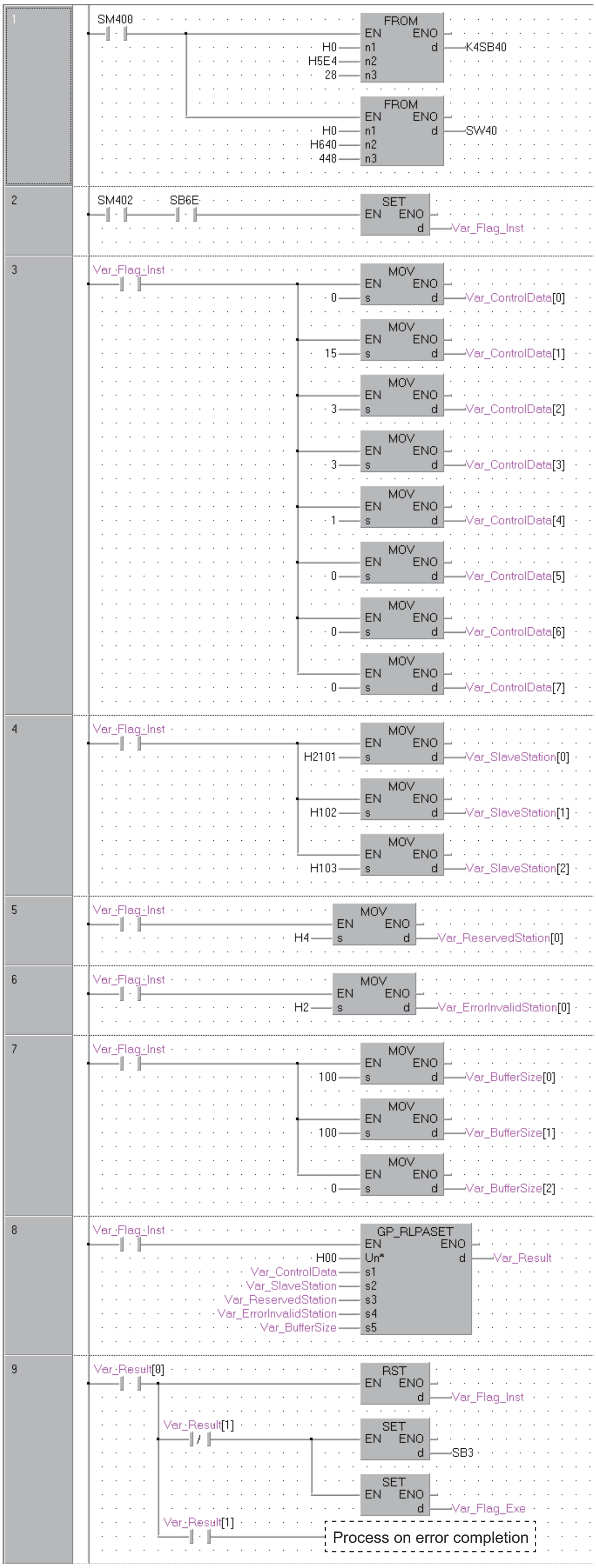
The RLPASET instruction is applicable to the QJ61BT11 of which the function version is B and the first five digits of the serial number are '03042' or higher.
The QJ61BT11N and LJ61BT11 is compatible with the RLPASET instruction.

Program example

The following program sets the network parameter to the master module mounted on the I/O number X/Y00 to X/Y1F, and starts up the data link.



[Structured ladder/FBD]



Reads SB0040 to SB01FF

Reads SW0040 to SW01FF

Parameter setting command

Clears completion status

Sets all of setting flags to Valid

Sets number of connected modules

Sets number of retries

Sets number of automatic return modules

Sets operation specification when CPU is down to stop

Sets scan mode specification to asynchronous

Set delay time specification

First module: local station, 1-station occupy, station number 1

Second module: Remote I/O station, 1-station occupy, station number 2

Third module: Remote I/O station, 1-station occupy, station number 3

Reserved station specification: station number 3

Error invalid station specification: station number 2

First module: local station, send buffer 100 words

Receive buffer 100 words

Auto-refresh buffer 0 word

Performs parameter setting and data link start

Turns parameter setting command OFF

Refresh command

Control program start command

5

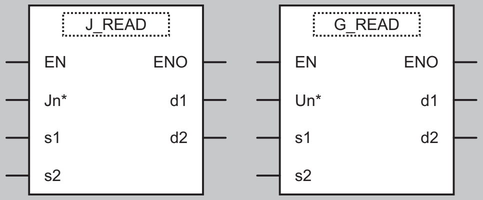
```
[ST]
FROM(TRUE, H0, H5E4, 28, K4SB40); (* Reads SB0040 to SB01FF *)
FROM(TRUE, H0, H640, 448, SW40); (* Reads SW0040 to SW01FF*)
IF((SM402=TRUE) & (SB6E=TRUE))THEN
    SET(TRUE, Var_Flag_Inst); (* Parameter setting command *)
END_IF;
IF(Var_Flag_Inst=TRUE)THEN (* Parameter setting command ON *)
    MOV(TRUE, 0, Var_ControlData[0]); (* Clear completion status *)
    MOV(TRUE, 15, Var_ControlData[1]); (* Sets all of setting flags to Valid *)
    MOV(TRUE, 3, Var_ControlData[2]); (* Sets number of connected modules *)
    MOV(TRUE, 3, Var_ControlData[3]); (* Sets number of retries *)
    MOV(TRUE, 1, Var_ControlData[4]); (* Sets number of automatic return modules *)
    MOV(TRUE, 0, Var_ControlData[5]); (* Sets operation specification when CPU is down to stop *)
    MOV(TRUE, 0, Var_ControlData[6]); (* Sets scan mode specification to asynchronous *)
    MOV(TRUE, 0, Var_ControlData[7]); (* Set delay time specification *)

    MOV(TRUE, H2101, Var_SlaveStation[0]); (* First module: local station, 1-station occupy, station number 1 *)
    MOV(TRUE, H0102, Var_SlaveStation[1]); (* Second module: Remote I/O station, 1-station occupy, station number 2*)
    MOV(TRUE, H0103, Var_SlaveStation[2]); (* Third module: Remote I/O station, 1-station occupy, station number 3 *)

    MOV(TRUE, H4, Var_ReservedStation[0]); (* Reserved station specification: station number 3 *)

    MOV(TRUE, H2, Var_ErrorInvalidStation[0]); (* Error invalid station specification: station number 2 *)

    MOV(TRUE, 100, Var_BufferSize[0]); (* First module: local module, send buffer 100 words *)
    MOV(TRUE, 100, Var_BufferSize[1]); (* Second module: local station, receive buffer 100 words *)
    MOV(TRUE, 0, Var_BufferSize[2]); (* Third module: local station, auto-refresh buffer 0 words *)

    GP_RLPASET(TRUE, H00, Var_ControlData, Var_SlaveStation,
    Var_ReservedStation, Var_ErrorInvalidStation, Var_BufferSize,
    Var_Result); (* Performs parameter setting *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
       SET(TRUE, SB3); (* Refresh command *)
       SET(TRUE, Var_Flag_Exe); (* Control program start command *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;

    RST(TRUE, Var_Flag_Inst); (* Turns parameter setting command OFF *)
END_IF;
```

READ instruction

J(P)_READ, G(P)_READ

CC IE C **CC IE F** **NET/H** **Ether**

Structured ladder/FBD



ST

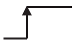
ENO:= J_READ (EN, Jn*, s1, s2, d1, d2);

ENO:= G_READ (EN, Un*, s1, s2, d1, d2);

The following instruction can go in the dotted squares.

J_READ, JP_READ, G_READ, GP_READ

■Executing condition

Instruction	Executing condition
J_READ G_READ	
JP_READ GP_READ	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module 00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Start number of the target station's device from which data are read	ANY
Output argument	ENO	Execution result	Bit
	d1	Start number of the host station's device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	○[*2]	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

*2 Only CC-Link IE Field Network

When the target station is LCPU, Universal model QCPU, or Basic model QCPU, the digit specification of the bit device can be used (example: K4M16).

The digit specification of the bit device can be used when the following conditions are met.

·The device number is a multiple of 16 (10H).

·The digit specification is 4 points (K4).

Processing details

This instruction reads data from a word device of another station.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Error completion type	b15 … b7 … b0 0 | (1) | 0 | 1 Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s1)[11]. 1: Clock data at the time of error completion is set in the area starting from (s1)[11].	0001H 0081H	User
(s1)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Channel used by host station	Specify the channel used by the host station. Setting values are as follows. • Ethernet, MELSECNET/H: 1 to 8 • CC-Link IE Controller Network: 1 to 10 • CC-Link IE Field Network: 1 to 2	1 to 10	User
(s1)[3]	Target station's CPU type	Specify the type of the target station CPU. Setting values are as follows. ■Ethernet • 0000H: Target station CPU/host system CPU (Specified data are the same as '03FFH'.) • 03FFH[*1]: Target station CPU/host system CPU ■MELSECNET/H CC-Link IE • 0000H: Target station CPU/host system CPU (Specified data are the same as '03FFH'.) • 03E0H[*2]: Multi-CPU No. 1/target station CPU (single CPU system) • 03E1H[*2]: Multi-CPU No. 2 • 03E2H[*2]: Multi-CPU No. 3 • 03E3H[*2]: Multi-CPU No. 4 • 03FFH[*1]: Target station CPU/host system CPU	■Ethernet 0000H, 03FFH ■MELSECNET/H, CC-Link IE 0000H, 03E0H to 03E3H, 03FFH	User
(s1)[4]	Target station network No.	Specify the network number of the target station. 1 to 239: Network number 254: Specify this when 254 has been set in Jn.	1 to 239, 254	User
(s1)[5]	Target station No.	Specify the station number of the target station. Setting values are as follows. • MELSECNET/H: 1 to 64 • When the host station is Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to120 • When the host station is anything other than Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to 64 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120	1 to 125	User
(s1)[6]	—	Reserved	0	User
(s1)[7]	Number of resends	• For instruction execution Specify the number of instruction resends when the instruction is not completed within the monitoring time specified in (s1)[8].	0 to 15	User
		• At instruction completion The number of resends (result) is stored.	—	System

Device	Item	Setting data	Setting range	Setting side
(s1)[8]	Arrival monitoring time	Specify the monitoring time required for the instruction completion. If the instruction is not completed within this time, it is resent by the number of times specified in (s1)[7]. Setting values are as follows. ■Ethernet • 0 to 16383 • 0 to TCP retransmission timer value: Monitoring is performed by the TCP retransmission timer value. • (TCP retransmission timer value + 1) to 16383: Monitoring time (unit: second) ■MELSECNET/H • 0 to 32767 • 0: 10 seconds • 1 to 32767: 1 to 32767 seconds	0 to 32767	User
(s1)[9]	Read data length	Specify the number of read data. Setting values are as follows. • Ethernet, MELSECNET/H, CC-Link IE Field Network: 1 to 960 (word) • CC-Link IE Controller Network: 1 to 8192 (word)	1 to 8192	User
(s1)[10]	—	Reserved	—	User
(s1)[11]	Clock set flag*3	Valid/invalid status of the data in the area starting from (s1)[12] is stored. 0: Invalid 1: Valid	—	System
(s1)[12] ⋮ (s1)[15]	Clock data at the time of error completion*3	Clock data at the time of error completion are stored in BCD format. b15 to b8 / b7 to b0 (s1)[12]: Month (01H to 12H) / Year (00H to 99H) Last two digits (s1)[13]: Hour (00H to 23H) / Day (01H to 31H) (s1)[14]: Second (00H to 59H) / Minute (00H to 59H) (s1)[15]: Year (00H to 99H) First two digits / Day of week (00H to 06H) 00H (Sun.) to 06H (Sat.)	—	System
(s1)[16]	Error-detected network No.*3	Network number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) 1 to 239: Network number	—	System
(s1)[17]	Error-detected station No.*3	Number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120	—	System

*1 Specification is possible when the host station is a network module or Ethernet module of function version D or later.
(Specification is not possible for other modules. An access is always made to the target station CPU.)

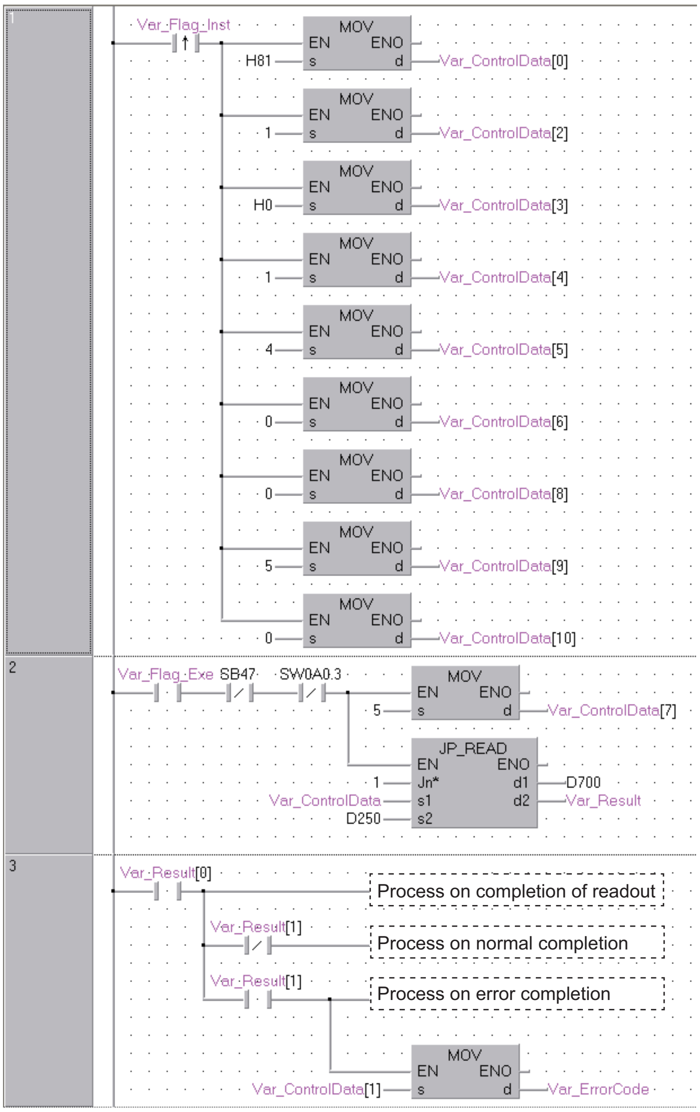
*2 Specification is possible when the versions of the QCPU and the network module on the host station and the target station are as indicated below.
(Specification is not possible for other modules. An access is always made to the target station CPU.)
· Network module: The first five digits of the serial number are '06092' or higher.
· QCPU: The first five digits of the serial number are '06092' or higher.

*3 Data are stored only when 1 is set in bit 7 of Error completion type ((s1)[0]).

Program example

- The following program reads out data from the devices from D250 to D254 in the station number 4 (target station) and stores the data to the devices from D700 to D704 of the station number 1 (host station).

[Structured ladder/FBD]



Sets error completion type

Sets channel used by host station

Sets target station's CPU type

Sets target station network number

Sets target station number

Sets monitoring time

Sets data length by the word

Sets number of resends

Performs readout

Execution finished

Normal completion

Error completion

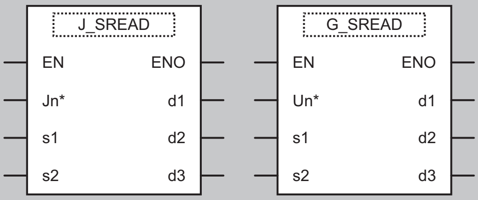
Stores error code

```
[ST]
IF(LDP(TRUE,Var_Flag_Inst))THEN
    MOV(TRUE,H81,Var_ControlData[0]); (* Sets error completion type *)
    MOV(TRUE,1,Var_ControlData[2]); (* Sets channel used by host station *)
    MOV(TRUE,H0,Var_ControlData[3]); (* Sets target station's CPU type *)
    MOV(TRUE,1,Var_ControlData[4]); (* Sets target station network number *)
    MOV(TRUE,4,Var_ControlData[5]); (* Sets target station number *)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets monitoring time *)
    MOV(TRUE,5,Var_ControlData[9]); (* Sets data length by the word *)
    MOV(TRUE,0,Var_ControlData[10]);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.3=FALSE)) THEN
    MOV(TRUE, 5, Var_ControlData[7]); (* Sets number of resends *)
    JP_READ(TRUE,1,Var_ControlData,D250,D700,Var_Result); (* Performs readout *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of readout *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

5

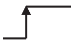
J(P)_SREAD, G(P)_SREAD

CC IE C **CC IE F** **NET/H** **Ether**

Structured ladder/FBD	ST
	ENO:= J_SREAD (EN, Jn*, s1, s2, d1, d2, d3); ENO:= G_SREAD (EN, Un*, s1, s2, d1, d2, d3);

The following instruction can go in the dotted squares.
J_SREAD, JP_SREAD, G_SREAD, GP_SREAD

■Executing condition

Instruction	Executing condition
J_SREAD G_SREAD	
JP_SREAD GP_SREAD	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Start number of the target station's device from which data are read	ANY
Output argument	ENO	Execution result	Bit
	d1	Start number of the host station's device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]
	d3	Variable that turns ON upon completion of the instruction (read notification device)	Bit

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					
(d3)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads data from a word device of another station.

Setting data

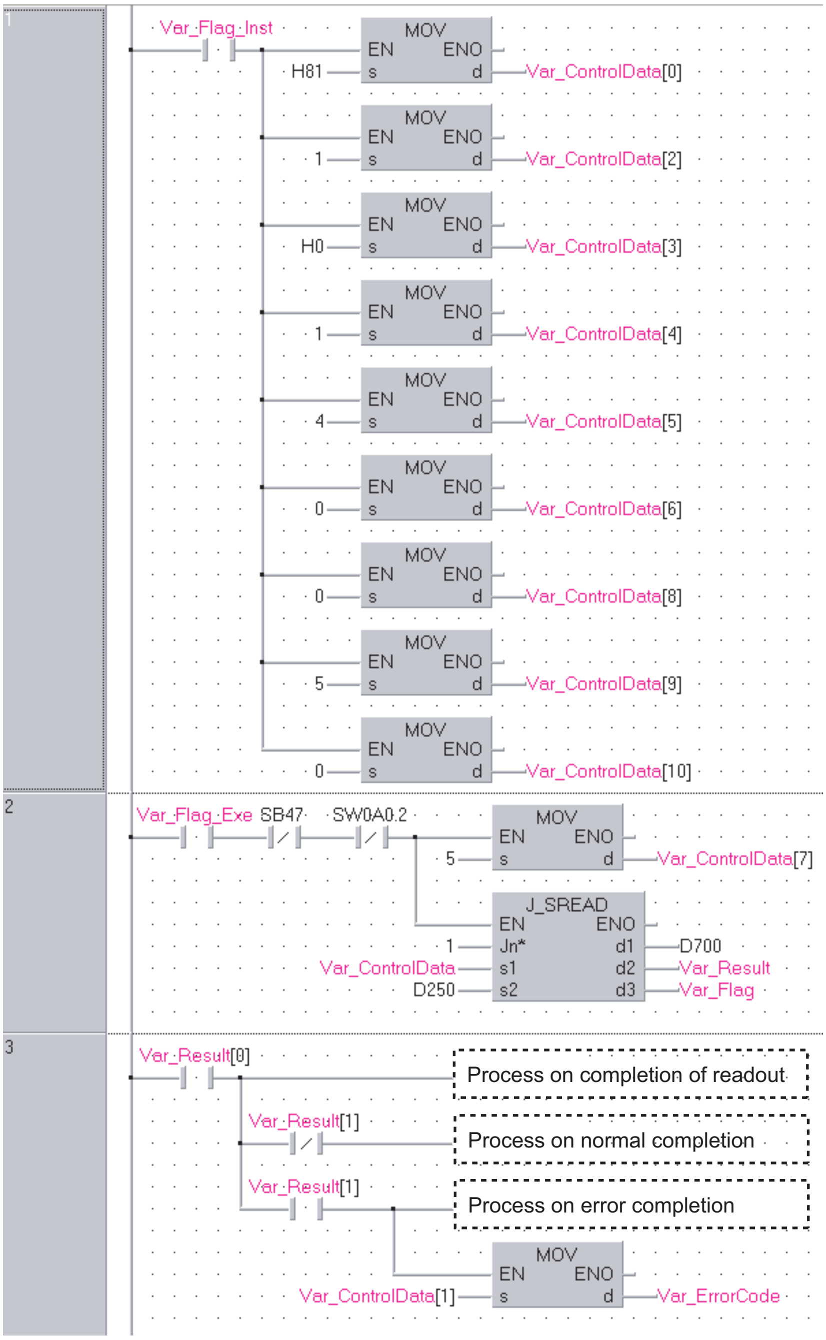
For the control data of the SREAD instruction that reads the word device memory of another station, refer to READ instruction.
The control data of the SREAD instruction are the same as those of the READ instruction.
Accordingly, this section omits the explanation.

5

Program example

- The following program example of the SREAD instruction is different from that of the READ instruction by assigning the read notification device (d3) at the end of arguments.

[Structured ladder/FBD]



Sets error completion type

Sets channel used by host station

Sets target station's CPU type

Sets target station network number

Sets target station number

Sets monitoring time

Sets data length by the word

Sets number of resends

Performs readout

Execution finished

Normal completion

Error completion

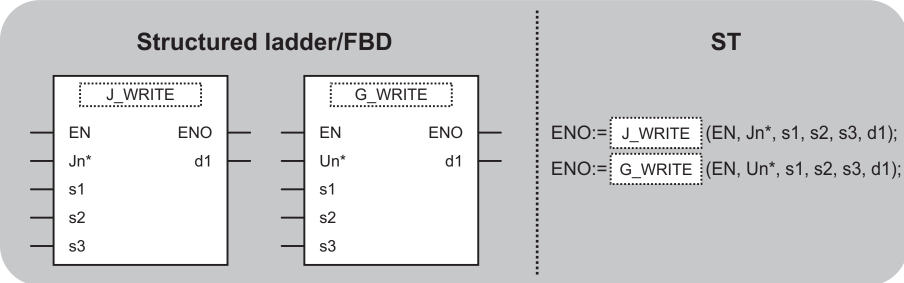
Stores error code

```
[ST]
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H81,Var_ControlData[0]); (* Sets error completion type *)
    MOV(TRUE,1,Var_ControlData[2]); (* Sets channel used by host station *)
    MOV(TRUE,H0,Var_ControlData[3]); (* Sets target station's CPU type *)
    MOV(TRUE,1,Var_ControlData[4]); (* Sets target station network number *)
    MOV(TRUE,4,Var_ControlData[5]); (* Sets target station number*)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets monitoring time *)
    MOV(TRUE,5,Var_ControlData[9]); (* Sets data length by the word *)
    MOV(TRUE,0,Var_ControlData[10]);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.3=FALSE)) THEN
    MOV(TRUE, 5, Var_ControlData[7]); (* Sets number of resends *)
    J_SREAD(TRUE,1,Var_ControlData,D250,D700,Var_Result,Var_Flag); (* Performs readout *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of readout *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

5

J(P)_WRITE, G(P)_WRITE

CC IE C **CC IE F** **NET/H** **Ether**



The following instruction can go in the dotted squares.
J_WRITE, JP_WRITE, G_WRITE, GP_WRITE

■Executing condition

Instruction	Executing condition
J_WRITE G_WRITE	
JP_WRITE GP_WRITE	

■Argument

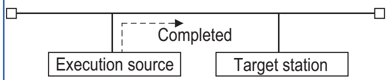
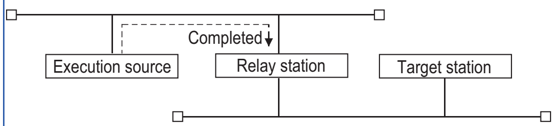
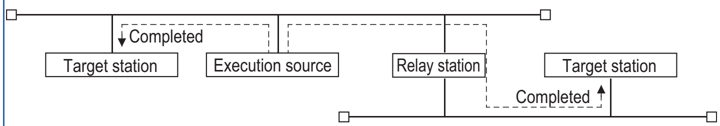
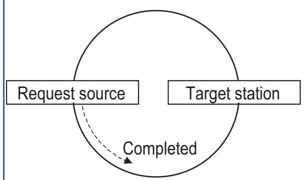
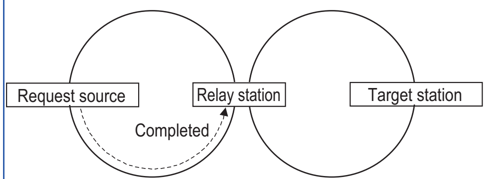
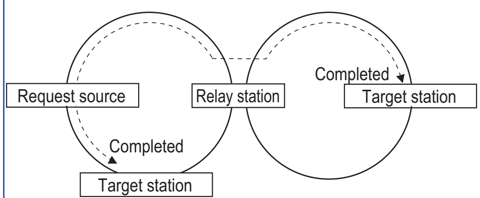
Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Start number of the host station's device that stores write data	ANY16
	s3	Start number of the target station's device to which data are written	ANY
Output argument	ENO	Execution result	Bit
	d1	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(s3)	—	○		—					
(d1)	○	○		—					

Processing details

This instruction writes data to a word device of another station.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Execution/Error completion type	b15 … b7 … b0 0 | (2) | 0 | (1) Execution type (bit 0) ■Ethernet 0: Without arrival confirmation • When the target station is on the same network Completed when data are sent from the host station.  • When the target station is on another network Completed when data reach to a relay station on the same network.  1: With arrival confirmation Completed when data are written to the target station.  ■MELSECNET/HCC-Link IE 0: Without arrival confirmation • When the target station is on the same network Completed when data are sent from the host station.  • When the target station is on another network Completed when data reach to a relay station on the same network.  1: With arrival confirmation Completed when data are written to the target station.  When '0: Without arrival confirmation' is specified, even if writing to the target station is completed abnormally in the following cases, the processing of the instruction in the host station is completed normally. • Communication itself was completed normally, although the sent data were erroneous. • Data could not be written to the target station because instructions from multiple stations were executed to the same station. (An error code (F222H, E006H, E205H, D202H, or D282H) is detected at the target station.)	0000H, 0001H, 0080H, 0081H	User

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Execution/Error completion type	Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s1)[11]. 1: Clock data at the time of error completion is set in the area starting from (s1)[11].	0000H, 0001H, 0080H, 0081H	User
(s1)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Channel used by host station	Specify the channel used by the host station. Setting values are as follows. • Ethernet, MELSECNET/H: 1 to 8 • CC-Link IE Controller Network: 1 to 10 • CC-Link IE Field Network: 1 to 2	1 to 10	User
(s1)[3]	Target station's CPU type	Specify the type of the target station CPU. Setting values are as follows. ■Ethernet • 0000H: Target station CPU/host system CPU (Specified data are the same as '03FFH'.) • 03FFH*1: Target station CPU/host system CPU ■MELSECNET/H CC-Link IE • 0000H: Target station CPU/host system CPU (Specified data are the same as '03FFH'.) • 03E0H*2: Multi-CPU No. 1/target station CPU (single CPU system) • 03E1H*2: Multi-CPU No. 2 • 03E2H*2: Multi-CPU No. 3 • 03E3H*2: Multi-CPU No. 4 • 03FFH*1: Target station CPU/host system CPU	■Ethernet 0000H,03FFH ■MELSECNET/H, CC-Link IE 0000H, 03E0H to 03E3H, 03FFH	User
(s1)[4]	Target station network No.	Specify the network number of the target station. 1 to 239: Network number 254: Specify this when 254 has been set in Jn.	1 to 239, 254	User
(s1)[5]	Target station No.	Specify the station number of the target station. Setting values are as follows. ■Station number specification • MELSECNET/H: 1 to 64 • When the host station is Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to120 • When the host station is anything other than Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to 64 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120 To increase the data reliability when the station number is specified, executing the instruction with setting Execution/Error completion type ((s1)[0]) to '1: With arrival confirmation' is recommended. ■Group specification (target station is anything other than CC-Link IE Field Network) 81H to A0H: All stations in group numbers 1 to 32 (Setting is available when Execution type is set to '0: Without arrival confirmation' in (s1)[0].) Group No.1 · · · 81H Group No.2 · · · 82H to Group No.32 · · · A0H ■All stations specification FFH: All stations of the target network number (Except the host station.) (Setting is available when Execution type is set to '0: Without arrival confirmation' in (s1)[0].) To specify a group or all stations. • Specify '0000H' or '03FFH' for the target station's CPU type ((s1)[3]). • Group specification cannot be set for the station of the CC-Link IE Field Network. • It cannot be confirmed if the data are written to the target station normally. Confirm the device of the target station of the write destination.	1 to 120 125 (7DH) 81H to A0H FFH	User
(s1)[6]	—	(Fixed value)	0	User

Device	Item	Setting data	Setting range	Setting side
(s1)[7]	Number of resends	• For instruction execution Specify the number of instruction resends when the instruction is not completed within the monitoring time specified in (s1)[8]. (Setting is available when Execution type is set to '1: With arrival confirmation' in ((s1)[0].)	0 to 15	User
		• At instruction completion The number of resends (result) is stored. (Setting is available when Execution type is set to '1: With arrival confirmation' in (s1)[0].)	—	System
(s1)[8]	Arrival monitoring time	Specify the monitoring time required for instruction completion. (Setting is available when Execution type is set to '1: With arrival confirmation' in ((s1)[0].) If the instruction is not completed within this time, it is resent by the number of times specified in (s1)[7]. Setting values are as follows. ■Ethernet • 0 to 16383 • 0 to TCP retransmission timer value: Monitoring is performed by the TCP retransmission timer value. • (TCP retransmission timer value + 1) to 16383: Monitoring time (unit: second) ■MELSECNET/H • 0 to 32767 • 0: 10 seconds • 1 to 32767: 1 to 32767 seconds	0 to 32767	User
(s1)[9]	Write data length	Specify the number of write data. Setting values are as follows. • Ethernet, MELSECNET/H, CC-Link IE Field Network: 1 to 960 (word) • CC-Link IE Controller Network: 1 to 8192 (word)	—	User
(s1)[10]	(Reserved)	—	—	—
(s1)[11]	Clock set flag*3	Valid/invalid status of the data in the area starting from (s1)[12] is stored. 0: Invalid 1: Valid	—	System
(s1)[12] ⋮ (s1)[15]	Clock data at the time of error completion*3	Clock data at the time of error completion are stored in BCD format. b15 to b8 / b7 to b0 (s1)[12]: Month (01H to 12H) / Year (00H to 99H) Last two digits (s1)[13]: Hour (00H to 23H) / Day (01H to 31H) (s1)[14]: Second (00H to 59H) / Minute (00H to 59H) (s1)[15]: Year (00H to 99H) First two digits / Day of week (00H to 06H) 00H (Sun.) to 06H (Sat.)	—	System
(s1)[16]	Error-detected network No.*3	Network number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) 1 to 239: Network number	—	System
(s1)[17]	Error-detected station No.*3	Number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120	—	System

*1 Specification is possible when the host station is a network module or Ethernet module of function version D or later. (Specification is not possible for other modules. An access is always made to the target station CPU.)

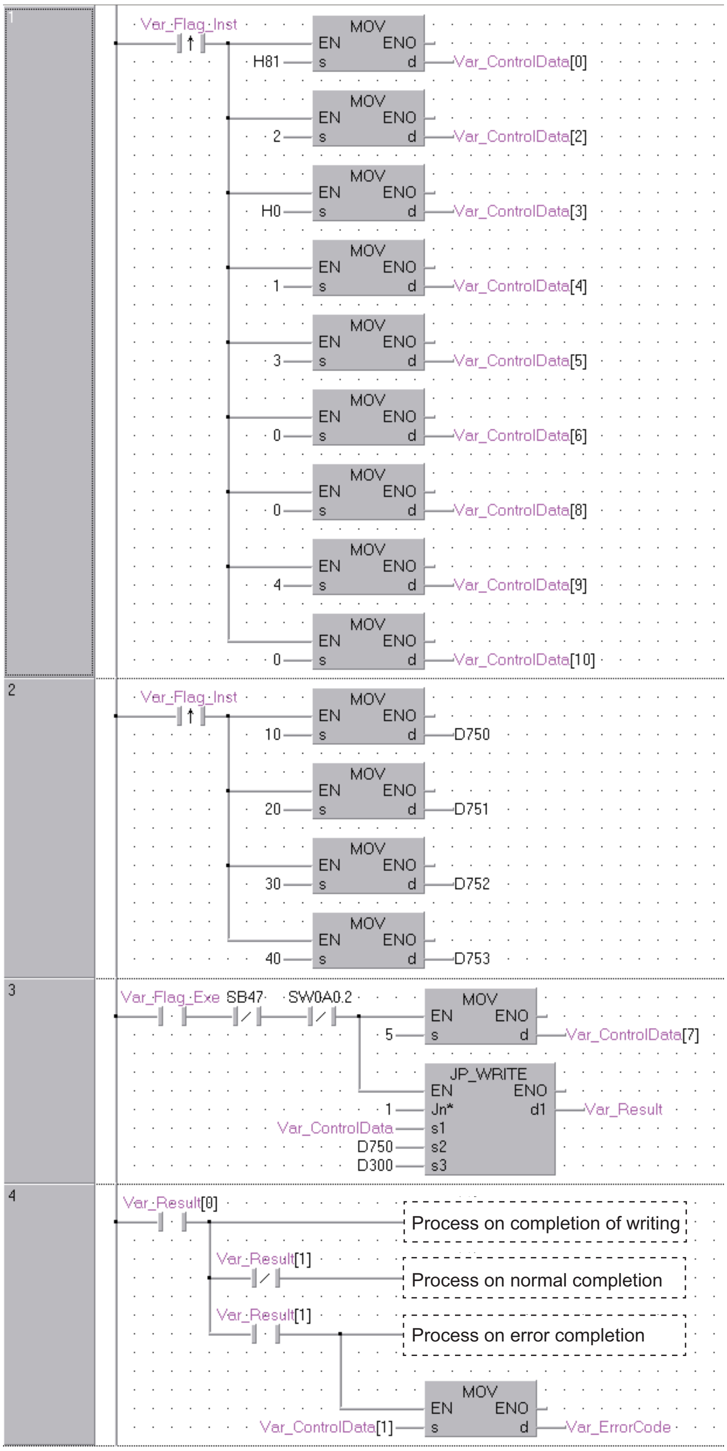
*2 Specification is possible when the versions of the QCPU and the network module on the host station and the target station are as indicated below.
(Specification is not possible for other modules. An access is always made to the target station CPU.)
· Network module: The first five digits of the serial number are '06092' or higher.
· QCPU: The first five digits of the serial number are '06092' or higher.

*3 Data are stored only when 1 is set in bit 7 of Error completion type ((s1)[0]).

Program example

- The following program writes data which are stored in the devices from D750 to D753 of the station number 2 (host station) to the devices from D300 to D303 of the station number 3 (target station).

[Structured ladder/FBD]



Sets execution/error completion type

Sets channel used by host station

Sets target station's CPU type

Sets target station network number

Sets target station number

Sets monitoring time

Sets data length by the word

Sets write data to D750 to D753

Sets number of resends

Performs writing

Execution finished

Normal completion

Error completion

Stores error code

[ST]

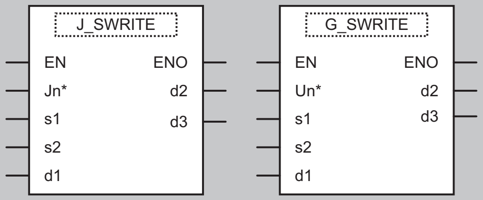
```
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H81,Var_ControlData[0]); (* Sets execution/error completion type *)
    MOV(TRUE,2,Var_ControlData[2]); (* Sets channel used by host station *)
    MOV(TRUE,H0,Var_ControlData[3]); (* Sets target station's CPU type *)
    MOV(TRUE,1,Var_ControlData[4]); (* Sets target station network number *)
    MOV(TRUE,3,Var_ControlData[5]); (* Sets target station number *)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets monitoring time *)
    MOV(TRUE,4,Var_ControlData[9]); (* Sets data length by the word *)
    MOV(TRUE,0,Var_ControlData[10]);
END_IF;
IF(LDP(TRUE,Var_Flag_Inst2)) THEN
    MOV(TRUE,10,D750); (* Sets write data to D750 to D753 *)
    MOV(TRUE,20,D751);
    MOV(TRUE,30,D752);
    MOV(TRUE,40,D753);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.2=FALSE)) THEN
    MOV(TRUE, 5, Var_ControlData[7]); (* Sets number of resends *)
    JP_WRITE(TRUE,1,Var_ControlData,D750,D300,Var_Result); (* Performs writing *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of writing *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

5

J(P)_SWRITE, G(P)_SWRITE

CC IE C | CC IE F | NET/H | Ether

Structured ladder/FBD



ST

ENO:=J_SWRITE(EN, Jn*, s1, s2, d1, d2, d3);
ENO:=G_SWRITE(EN, Un*, s1, s2, d1, d2, d3);

The following instruction can go in the dotted squares.
J_SWRITE, JP_SWRITE, G_SWRITE, GP_SWRITE

■Executing condition

Instruction	Executing condition
J_SWRITE G_SWRITE	
JP_SWRITE GP_SWRITE	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Start number of the host station's device that stores send data	ANY16
	d1	Start number of the target station to which data are written	ANY
Output argument	ENO	Execution result	Bit
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]
	d3	Variable that turns ON upon completion of the instruction (Write notification device)	Bit

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					
(d3)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction writes data to a word device of another station.

Setting data

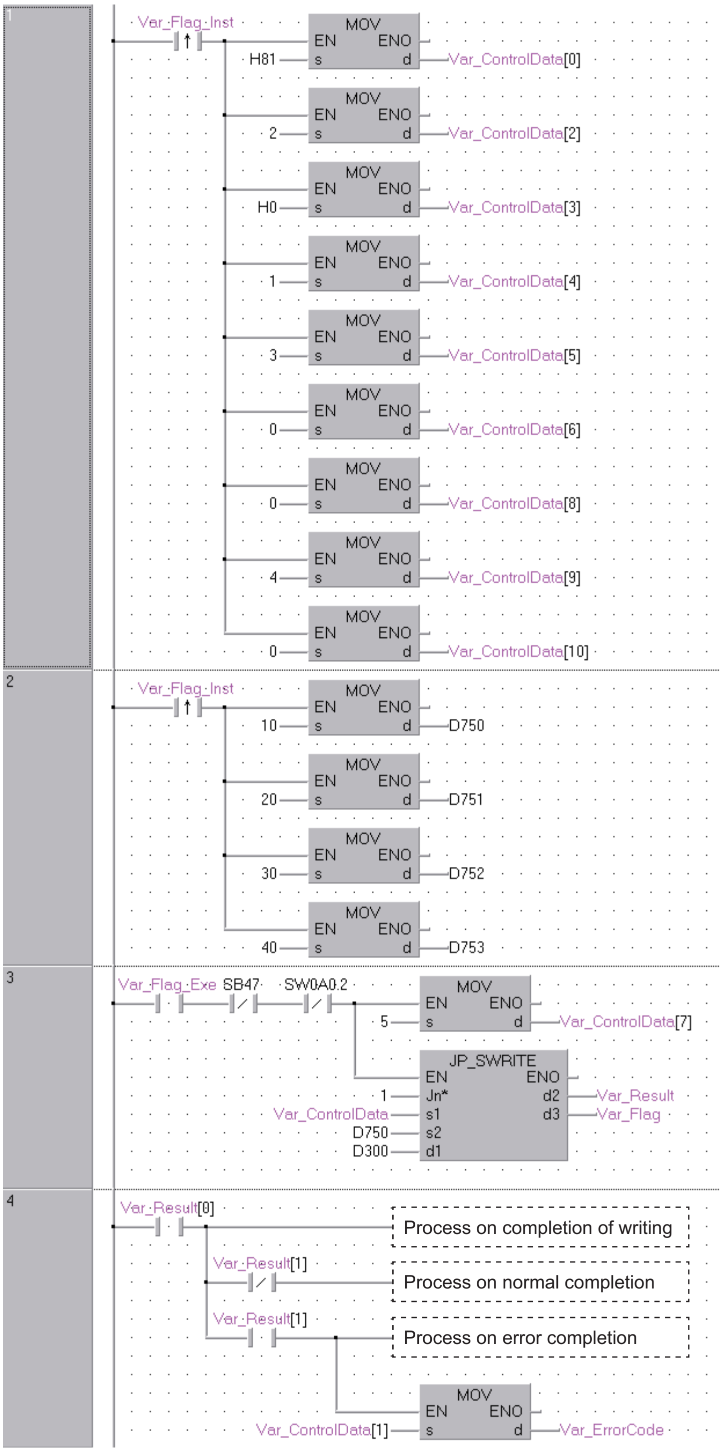
For the control data of the SWRITE instruction that writes data to the word device memory of another station, refer to WRITE instruction.
The control data of the SWRITE instruction are the same as those of the WRITE instruction.
Accordingly, this section omits the explanation.

Program example

- The following program example of the SWRITE instruction is different from that of the WRITE instruction by assigning the write notification device (d3) at the end of arguments.

[Structured ladder/FBD]

(1)Program on the request source (station number 2) of the SWRITE instruction



Sets execution/error completion type

Sets channel used by host station

Sets target station's CPU type

Sets target station network number

Sets target station number

Sets monitoring time

Sets data length by the word

Sets write data to D750 to D753

Sets number of resends

Performs writing

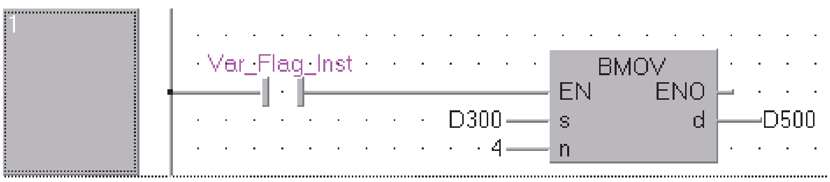
Execution finished

Normal completion

Error completion

Stores error code

(2)Program on the request target (station number 3) of the SWRITE instruction



Stores data of devices from D300 to D303 to devices from D500 to D503

[ST]
(1)Program on the request source (station number 2) of the SWRITE instruction

```
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H81,Var_ControlData[0]); (* Sets execution/error completion type *)
    MOV(TRUE,2,Var_ControlData[2]); (* Sets channel used by host station *)
    MOV(TRUE,H0,Var_ControlData[3]); (* Sets target station's CPU type *)
    MOV(TRUE,1,Var_ControlData[4]); (* Sets target station network number *)
    MOV(TRUE,3,Var_ControlData[5]); (* Sets target station number *)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets monitoring time *)
    MOV(TRUE,4,Var_ControlData[9]); (* Sets data length by the word *)
    MOV(TRUE,0,Var_ControlData[10]);
END_IF;
IF(LDP(TRUE,Var_Flag_Inst2)) THEN
    MOV(TRUE,10,D750); (* Sets write data to D750 to D753 *)
    MOV(TRUE,20,D751);
    MOV(TRUE,30,D752);
    MOV(TRUE,40,D753);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.2=FALSE)) THEN
    MOV(TRUE, 5, Var_ControlData[7]); (* Sets number of resends *)
    JP_SWRITE(TRUE,1,Var_ControlData,D750,D300,Var_Result,Var_Flag); (* Performs writing *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *
```

(* Process on completion of writing *)

```
    IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
```

(* Process on normal completion *)

```
    ELSE (* Error completion *)
```

(* Process on error completion *)

```
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

(2)Program on the request target (station number 3) of the SWRITE instruction

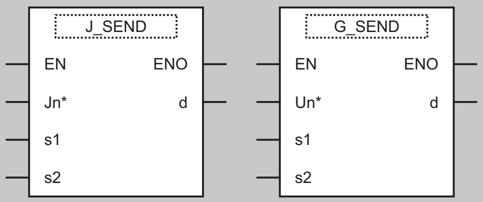
```
IF(Var_Flag=TRUE) THEN
    BMOV(TRUE,D300,4,D500); (* Stores data of devices from D300 to D303 to devices from D500 to D503 *)
END_IF;
```

Message (user-specified data) communication

J(P)_SEND, G(P)_SEND

CC IE C | CC IE F | NET/H | Ether

Structured ladder/FBD



ST

ENO:= J_SEND (EN, Jn*, s1, s2, d);
ENO:= G_SEND (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.
J_SEND, JP_SEND, G_SEND, GP_SEND

■Executing condition

Instruction	Executing condition
J_SEND G_SEND	
JP_SEND GP_SEND	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Start number of the host station's device that stores write data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

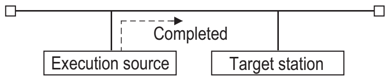
*1 Local devices and file registers per program cannot be used as setting data.

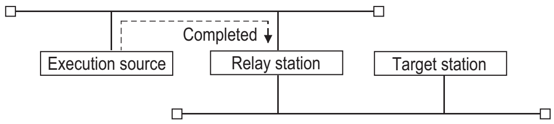
Processing details

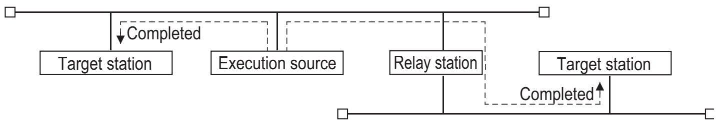
This instruction sends data to another station.

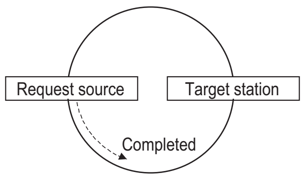
Setting data

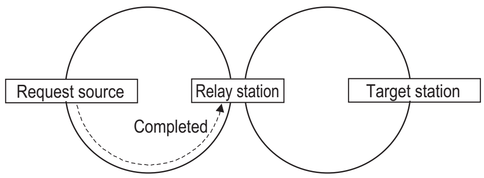
<table>
<tr><th>Device</th><th>Item</th><th>Setting data</th><th>Setting range</th><th>Setting side</th></tr>
<tr><td>(s1)[0]</td><td>Execution/Error completion type</td><td>
b15 … b7 … b0
0 | (2) | 0 | (1)
Execution type (bit 0)

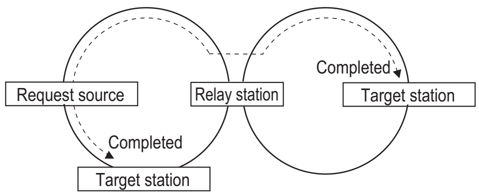
■Ethernet
0: Without arrival confirmation
• When the target station is on the same network
Completed when data are sent from the host station.


• When the target station is on another network
Completed when data reach to a relay station on the same network.


1: With arrival confirmation
Completed when data are stored in the specified channel of the target station.


■MELSECNET/H CC-Link IE
0: Without arrival confirmation
• When the target station is on the same network
Completed when data are sent from the host station.


• When the target station is on another network
Completed when data reach to a relay station on the same network.


1: With arrival confirmation
Completed when data are written to the target station.


When '0: Without arrival confirmation' is specified, even if writing to the target station is completed abnormally in the following cases, the processing of the instruction in the host station is completed normally.
• Communication itself was completed normally, although the sent data were erroneous.
• Data could not be written to the target station because instructions from multiple stations were executed to the same station. (An error code (F222H, E006H, E205H, D202H, or D282H) is detected at the target station.)
</td><td>0000H, 0001H, 0080H, 0081H</td><td>User</td></tr>
</table>

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Execution/Error completion type	Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s1)[11]. 1: Clock data at the time of error completion is set in the area starting from (s1)[11].	0000H, 0001H, 0080H, 0081H	User
(s1)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Channel used by host station	Specify the channel used by the host station. Setting values are as follows. • Ethernet, MELSECNET/H, CC-Link IE Controller Network: 1 to 8 • CC-Link IE Field Network: 1 to 2	1 to 8	User
(s1)[3]	Target station channel	Specify the channel of the target station that stores data.*2[*2] Setting values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE: 1 to 8	1 to 64	User
(s1)[4]	Target station network No.	Specify the network number of the target station. 1 to 239: Network number 254: Specify this when 254 has been set in Jn. (Network specified in 'Valid module during other station access')	1 to 239, 254	User
(s1)[5]	Target station No.	Specify the station number of the target station. Setting values are as follows. ■Station number specification • MELSECNET/H: 1 to 64 • When the host station is Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to120 • When the host station is anything other than Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to 64 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120 To increase the data reliability when the station number is specified, executing the instruction with setting Execution/Error completion type ((s1)[0]) to '1: With arrival confirmation' is recommended. ■Group specification (target station is anything other than CC-Link IE Field Network) 81H to A0H: All stations in group numbers 1 to 32 (Setting is available when Execution type is set to '0: Without arrival confirmation' in (s1)[0].) Group No.1 · · · 81H Group No.2 · · · 82H to Group No.32 · · · A0H ■All stations specification FFH: All stations of the target network number (Except the host station.) (Setting is available when Execution type is set to '0: Without arrival confirmation' in (s1)[0].) To specify a group or all stations. • Specify '0000H' or '03FFH' for the target station's CPU type ((s1)[3]). • Group specification cannot be set for the station of the CC-Link IE Field Network. • It cannot be confirmed if the data are written to the target station normally. Confirm the device of the target station of the write destination.	1 to 120, 125 (7DH)	User
(s1)[6]	—	(Fixed value)	0	User
(s1)[7]	Number of resends	• For instruction execution Specify the number of instruction resends when the instruction is not completed within the monitoring time specified in (s1)[8]. (Setting is available when Execution type is set to '1: With arrival confirmation' in ((s1)[0].)	0 to 15	User
		• At instruction completion The number of resends (result) is stored. (Setting is available when Execution type is set to '1: With arrival confirmation' in (s1)[0].)	—	System

Device	Item	Setting data	Setting range	Setting side
(s1)[8]	Arrival monitoring time	Specify the monitoring time required for instruction completion. (Setting is available when Execution type is set to '1: With arrival confirmation' in ((s1)[0].) If the instruction is not completed within this time, it is resent by the number of times specified in (s1)[7]. Setting values are as follows. ■Ethernet • 0 to 16383 • 0 to TCP retransmission timer value: Monitoring is performed by the TCP retransmission timer value. • (TCP retransmission timer value + 1) to 16383: Monitoring time (unit: second) ■MELSECNET/H CC-Link IE • 0 to 32767 • 0: 10 seconds • 1 to 32767: 1 to 32767 seconds	0 to 32767	User
(s1)[9]	Send data length	Specify the number of send data.	1 to 960	User
(s1)[10]	(Reserved)	—	—	—
(s1)[11]	Clock set flag*1	Valid/invalid status of the data in the area starting from (s1)[12] is stored. 0: Invalid 1: Valid	—	System
(s1)[12] ⋮ (s1)[15]	Clock data at the time of error completion*1	Clock data at the time of error completion are stored in BCD format. b15 to b8 / b7 to b0 (s1)[12]: Month (01H to 12H) / Year (00H to 99H) Last two digits (s1)[13]: Hour (00H to 23H) / Day (01H to 31H) (s1)[14]: Second (00H to 59H) / Minute (00H to 59H) (s1)[15]: Year (00H to 99H) First two digits / Day of week (00H to 06H) 00H (Sun.) to 06H (Sat.)	—	System
(s1)[16]	Error-detected network No.*1	Network number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) 1 to 239: Network number	—	System
(s1)[17]	Error-detected station No.*1	Number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120	—	System

*1 Data are stored only when 1 is set in bit 7 of Error completion type ((s1)[0]).
*2 Logical channel setting is not available for the CC-Link IE network module.

Program example

The following program sends data of the devices from D750 to D753 of the station number 1 (host station) to the channel 5 of the station number 2 (target station).
For the method for reading the data, which are sent by the SEND instruction, from the channel 5 of the station number 2 (target station), refer to the following sections.

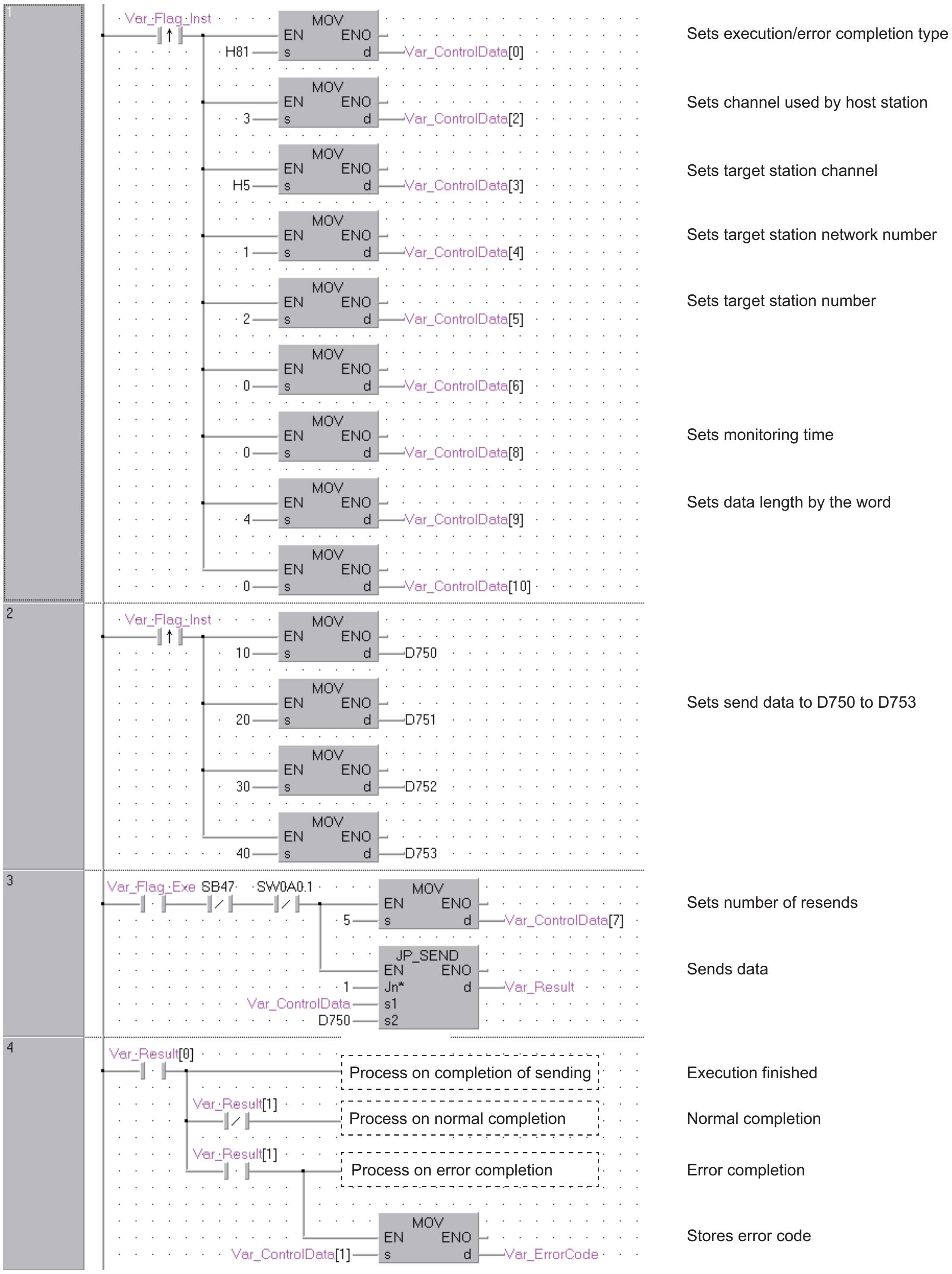
- For reading out data in a main program

☞Page 150 J(P)_RECV, G(P)_RECV

- For reading out data in an interrupt program

☞Page 154 Z_RECVS

[Structured ladder/FBD]



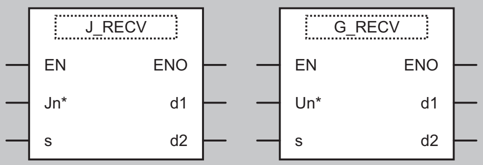
```
[ST]
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H81,Var_ControlData[0]); (* Sets execution/error completion type *)
    MOV(TRUE,3,Var_ControlData[2]); (* Sets channel used by host station *)
    MOV(TRUE,H5,Var_ControlData[3]); (* Sets target station channel *)
    MOV(TRUE,1,Var_ControlData[4]); (* Sets target station network number *)
    MOV(TRUE,2,Var_ControlData[5]); (* Sets target station number *)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets monitoring time *)
    MOV(TRUE,4,Var_ControlData[9]); (* Sets data length by the word *)
    MOV(TRUE,0,Var_ControlData[10]);
END_IF;
IF (Var_Flag_Inst2=TRUE)THEN
    MOV(TRUE,10,D750); (*Sets send data to D750 to D753 *)
    MOV(TRUE,20,D751);
    MOV(TRUE,30,D752);
    MOV(TRUE,40,D753);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.1=FALSE)) THEN
    MOV(TRUE, 5, Var_ControlData[7]); (* Sets number of resends *)
    JP_SEND(TRUE,1,Var_ControlData,D750,Var_Result); (* Sends data *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of sending *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
      MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

5

J(P)_RECV, G(P)_RECV

CC IE C **CC IE F** **NET/H** **Ether**

Structured ladder/FBD



ST

ENO:= J_RECV (EN, Jn*, s, d1, d2);

ENO:= G_RECV (EN, Un*, s, d1, d2);

The following instruction can go in the dotted squares.

J_RECV, JP_RECV, G_RECV, GP_RECV

■Executing condition

Instruction	Executing condition
J_RECV G_RECV	
JP_RECV GP_RECV	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s	Variable that stores control data	Array of ANY16 [0..17]
Output argument	ENO	Execution result	Bit
	d1	Start number of the host station's device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads received data (for main program).

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Execution/error completion type	b15 … b7 … b0 0 (1) 0 Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s)[11]. 1: Clock data at the time of error completion is set in the area starting from (s)[11].	0000H, 0080H	User
(s)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Host station channel	Specify the channel of host station that stores receive data. Setting values are as follows. • Ethernet, MELSECNET/H, CC-Link IE Controller Network: 1 to 8 • CC-Link IE Field Network: 1 to 2	1 to 8	User
(s)[3]	Channel used by sending station	Channel used by the sending station is stored. 1 to 8: Channel	—	System
(s)[4]	Network No. of sending station	Network number of the sending station is stored. 1 to 239: Network number	—	System
(s)[5]	Sending station No.	Station number of the sending station is stored. Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120	—	System
(s)[6]	(Reserved)	—	—	—
(s)[7]	(Reserved)	—	—	—
(s)[8]	Arrival monitoring time	Specify the monitoring time required for the instruction completion. When the instruction is not completed within the monitoring time, it completes abnormally. Setting values are as follows. ■Ethernet • 0 to 16383 • 0 to TCP retransmission timer value: Monitoring is performed by the TCP retransmission timer value. • (TCP retransmission timer value + 1) to 16383: Monitoring time (unit: second) ■CC-Link IE MELSECNET/H • 0 to 32767 • 0: 10 seconds • 1 to 32767: 1 to 32767 seconds	0 to 32767	User
(s)[9]	Receive data length	The number of received data stored in (d1) to (d1)+n is stored. 0: No receive data 1 to 960: Number of words of receive data	—	System
(s)[10]	(Reserved)	—	—	—
(s)[11]	Clock set flag*1	Valid/invalid status of the data in the area starting from (s)[12] is stored. 0: Invalid 1: Valid	—	System
(s)[12] ⋮ (s)[15]	Clock data at the time of error completion*1	Clock data at the time of error completion are stored in BCD format. b15 to b8 / b7 to b0 (s)[12] Month (01H to 12H) / Year (00H to 99H) Last two digits (s)[13] Hour (00H to 23H) / Day (01H to 31H) (s)[14] Second (00H to 59H) / Minute (00H to 59H) (s)[15] Year (00H to 99H) First two digits / Day of week (00H to 06H) 00H (Sun.) to 06H (Sat.)	—	System
(s)[16]	Error-detected network No.*1	Network number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) 1 to 239: Network number	—	System

Device	Item	Setting data	Setting range	Setting side
(s)[17]	Error-detected station No.*1	Number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120	—	System

*1 Data are stored only when 1 is set in bit 7 of Error completion type ((s)[0]).

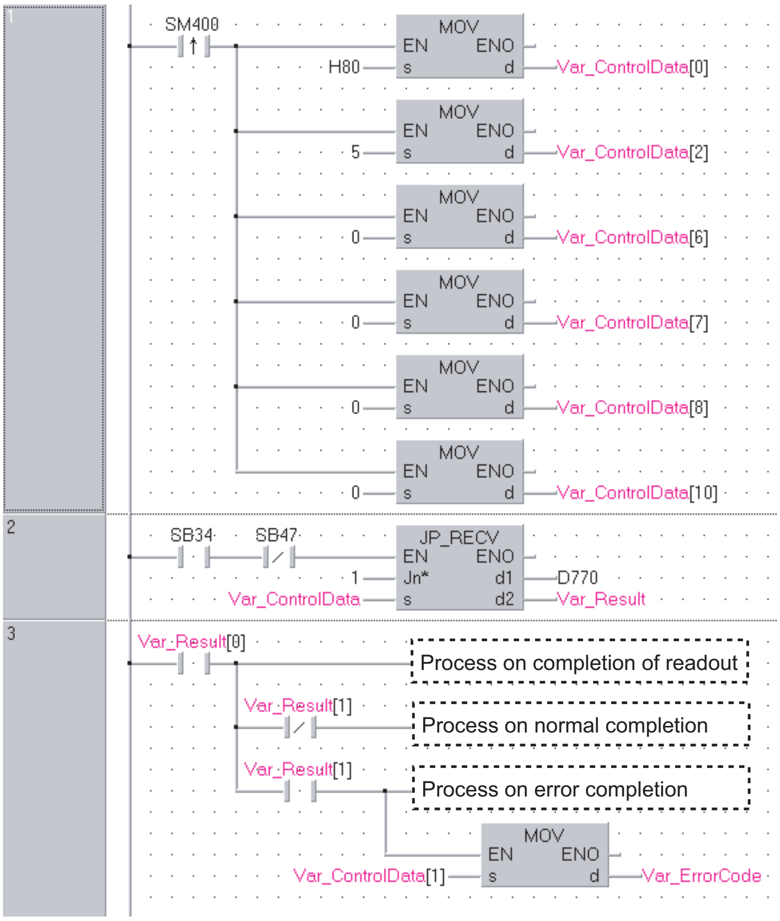
Program example

The following program reads out data, which is sent from the station number 1 by the SEND instruction, from the channel 5 of the station number 2 (host station) and stores the data to the devices from D770 to D773 of the station number 2 (host station) when SB0034 turns ON.

For the SEND instruction, refer to the following section.

☞ Message (user-specified data) communication

[Structured ladder/FBD]



Sets error completion type

Sets host station channel

Sets arrival monitoring time

Performs readout

Execution finished

Normal completion

Error completion

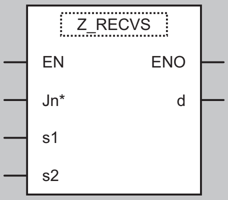
Stores error code

[ST]

```
IF(SM400=TRUE)THEN
    MOV(TRUE,H80,Var_ControlData[0]); (* Sets error completion type *)
    MOV(TRUE,5,Var_ControlData[2]); (* Sets host station channel *)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[7]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets arrival monitoring time *)
    MOV(TRUE,0,Var_ControlData[10]);
END_IF;
IF((SB34=TRUE) AND (SB47=FALSE)) THEN
    JP_RECV(TRUE,1,Var_ControlData,D770,Var_Result); (* Performs readout *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of readout *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
      IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
    END_IF;
END_IF;
```

Z_RECVS

CC IE C **CC IE F** **NET/H** **Ether**

Structured ladder/FBD	ST
	ENO:= Z_RECVS (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.
Z_RECVS

■Executing condition

Instruction	Executing condition
Z_RECVS	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Start number of the host station's device that stores read data	ANY16
Output argument	ENO	Execution result	Bit
	d	Dummy	Bit

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads received data (for interrupt program).

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Completion type	0 (Fixed)	0	User
(s1)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Host station channel	Specify the channel of host station that stores receive data. Setting values are as follows. • Ethernet, MELSECNET/H, CC-Link IE Controller Network: 1 to 8 • CC-Link IE Field Network: 1 to 2	1 to 8	User
(s1)[3]	Channel used by sending station	Channel used by the sending station is stored. 1 to 8: Channel	—	System
(s1)[4]	Network No. of sending station	Network number of the sending station is stored. 1 to 239: Network number	—	System
(s1)[5]	Sending station No.	Station number of the sending station is stored. Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Slave station in CC-Link IE Field Network: 1 to 120	—	System
(s1)[6] (s1)[7] (s1)[8]	System area	—	—	—
(s1)[9]	Receive data length	The number of received data stored in (s2) to (s2)+n is stored. 0: No receive data 1 to 960: Number of words of receive data	—	System
(s1)[10] ⋮ (s1)[17]	System area	—	—	—

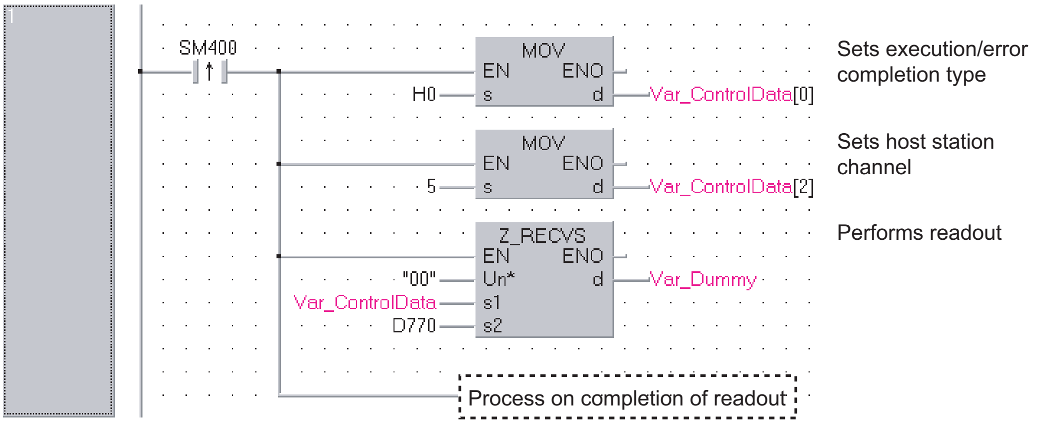
Program example

The following program reads data, which is sent from the station number 1 by the SEND instruction, from the channel 5 of the station number 2 (host station) and stores the data to the devices from D770 to D773 of the station number 2 (host station) when an interruption program starts up.

For the SEND instruction, refer to the following section.

☞ Page 144 Message (user-specified data) communication

[Structured ladder/FBD]



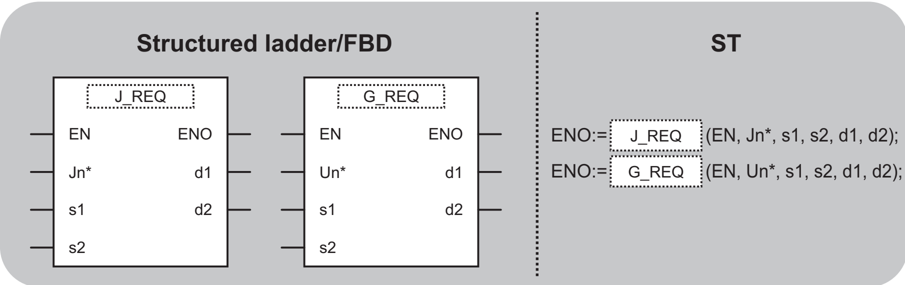
[ST]

```
IF(SM400=TRUE)THEN
    MOV(TRUE,H0,Var_ControlData[0]); (* Sets execution/error completion type *)
    MOV(TRUE,5,Var_ControlData[2]); (* Sets host station channel *)
    Z_RECVS(TRUE,"00",Var_ControlData,D770,Var_Dummy); (* Performs readout *)

 (* Process on completion of readout *)

END_IF;
```

Transient request to another station

J(P)_REQ, G(P)_REQ

CC IE C | CC IE F | NET/H | Ether



The following instruction can go in the dotted squares.
J_REQ, JP_REQ, G_REQ, GP_REQ

■Executing condition

Instruction	Executing condition
J_REQ G_REQ	
JP_REQ GP_REQ	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	ANY16
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..17]
	s2	Variable that stores request data	Array of ANY16 [0..5]
Output argument	ENO	Execution result	Bit
	d1	Variable that stores response data	Array of ANY16 [0..5]
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

Remotely runs or stops a programmable controller on another station.
Also, reads/writes clock data from/to a programmable controller on another station.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Error completion type	b15 … b7 … b4 … b0 0 | (1) | 0 | 1 | 0 | 1 Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s1)[11]. 1: Clock data at the time of error completion is set in the area starting from (s1)[11].	0011H, 0091H	User
(s1)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Channel used by host station	Specify the channel used by the host station. 1 to 8: Channel	1 to 8	User
(s1)[3]	Target station's CPU type	Specify the type of the target station CPU. Setting values are as follows. ■Ethernet • 0000H: Target station CPU/host system CPU (Specified data are the same as '03FFH'.) • 03FFH[*1]: Target station CPU/host system CPU ■MELSECNET/H CC-Link IE • 0000H: Target station CPU/host system CPU (Specified data are the same as '03FFH'.) • 03E0H[*2]: Multi-CPU No. 1/target station CPU (single CPU system) • 03E1H[*2]: Multi-CPU No. 2 • 03E2H[*2]: Multi-CPU No. 3 • 03E3H[*2]: Multi-CPU No. 4 • 03FFH[*1]: Target station CPU/host system CPU	■Ethernet 0000H,03FFH ■MELSECNET/H, CC-Link IE 0000H, 03E0H to 03E3H, 03FFH	User
(s1)[4]	Target station network No.	Specify the network number of the target station. 1 to 239: Network number 254: Specify this when 254 has been set in Jn. (Network specified in 'Valid module during other station access')	1 to 239, 254	User
(s1)[5]	Target station No.	Specify the station number of the target station. Setting values are as follows. ■Station number specification • MELSECNET/H: 1 to 64 • When the host station is Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to120 • When the host station is anything other than Universal model QCPU in Ethernet or CC-Link IE Controller Network: 1 to 64 • Master station in CC-Link IE Field Network: 125 (7DH) • Local station or the intelligent device station in CC-Link IE Field Network: 1 to 120 ■Group specification (target station is anything other than CC-Link IE Field Network) 81H to A0H: All stations in group numbers 1 to 32 (Available only at clock data writing and remote RUN/STOP) Group No.1 · · · 81H Group No.2 · · · 82H to Group No.32 · · · A0H ■All stations specification FFH: All stations of the target network number (Except the host station.) (Available only at clock data writing and remote RUN/STOP) To specify a group or all stations. • Specify '0000H' or '03FFH' for the target station's CPU type ((s1)[3]). • Group specification cannot be set for the station of the CC-Link IE Field Network. • It cannot be confirmed if the data are written to the target station normally. Confirm the device of the target station of the write destination.	1 to 120, 125 (7DH), 81H to A0H, FFH	User
(s1)[6]	—	(Fixed value)	0	User
(s1)[7]	Number of resends	• For instruction execution Specify the number of resends when the instruction is not completed within the monitoring time specified in (s1)[8].	0 to 15	User
		• At instruction completion The number of resends (result) is stored	0 to 15	System

Device	Item	Setting data	Setting range	Setting side
(s1)[8]	Arrival monitoring time	Specify the monitoring time required for the instruction completion. If the instruction is not completed within this time, it is resent by the number of times specified in (s1)[7]. Setting values are as follows. ■Ethernet • 0 to 16383 • 0 to TCP retransmission timer value: Monitoring is performed by the TCP retransmission timer value. • (TCP retransmission timer value + 1) to 16383: Monitoring time (unit: second) ■MELSECNET/H CC-Link IE • 0 to 32767 • 0: 10 seconds • 1 to 32767: 1 to 32767 seconds	0 to 32767	User
(s1)[9]	Request data length	Specify the number of request data (words). (Number of words of data stored in the request data storage device (s2)) 4: Remote RUN 3: Remote STOP 2: Clock data read 6: Clock data write	2 to 4, 6	User
(s1)[10]	Response data length	Number of response data (words) are stored. (Number of words of the data stored in response data storage device) 2: Remote RUN/STOP 6: Clock data read 2: Clock data write	—	System
(s1)[11]	Clock set flag*3	Valid/invalid status of the data in the area starting from (s1)[12] is stored. 0: Invalid 1: Valid	—	System
(s1)[12] ⋮ (s1)[15]	Clock data on error completion*3	Clock data at the time of error completion are stored in BCD format. b15 to b8 | b7 to b0 (s1)[12] Month (01H to 12H) | Year (00H to 99H) Last two digits (s1)[13] Hour (00H to 23H) | Day (01H to 31H) (s1)[14] Second (00H to 59H) | Minute (00H to 59H) (s1)[15] Year (00H to 99H) First two digits | Day of week (00H to 06H) 00H (Sun.) to 06H (Sat.)	—	System
(s1)[16]	Error-detected network No.*3	Network number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) 1 to 239: Network number	—	System
(s1)[17]	Error-detected station No.*3	Number of the station where an error was detected is stored. (However, when an error was detected at the host station, the network number is not stored.) Stored values are as follows. • MELSECNET/H: 1 to 64 • Ethernet, CC-Link IE Controller Network: 1 to 120 • Master station in CC-Link IE Field Network: 125 (7DH) • Slave station in CC-Link IE Field Network: 1 to 120	—	System

*1 Specification is possible when the host station is a network module or Ethernet module of function version D or later.
(Specification is not possible for other modules. An access is always made to the target station CPU.)

*2 Specification is possible when the versions of the QCPU and the network module on the host station and the target station are as indicated below.
(Specification is not possible for other modules. An access is always made to the target station CPU.)
· Network module: The first five digits of the serial number are '06092' or higher.
· QCPU: The first five digits of the serial number are '06092' or higher.

*3 This becomes valid only when 1 is set in bit 7 of Error completion type ((s1)[0]).

■Remote RUN/STOP

- Request data (all set by the user)

Device	Item	Description	Remote RUN	Remote STOP
(s2)[0]	Request type	0010H: When station number is specified in (s1)[5] 0030H: When all stations a group is specified in (s1)[5]	○	○
(s2)[1]	Sub-request type	0001H: Remote RUN 0002H: Remote STOP	○	○
(s2)[2]	Operation mode	Specify whether to forcibly execute remote RUN/STOP. The forced execution is a function that forces a station which has stopped by remote STOP to RUN remotely from another station. • For remote RUN 0001H: No forced execution 0003H: Forced execution (This setting can be specified for remote RUN.) • For remote STOP 0003H: (Fixed)	○	○
(s2)[3]	Clear mode	Specify the status of device memory in the CPU module only for remote RUN. 0000H: Not cleared (Note that the local devices are cleared.) 0001H: Cleared (excluding the latch range and settings in remote RUN) 0002H: Cleared (including the latch range and settings in remote RUN) Clear mode ((s2)[3]) allows specification to clear (initialize) the devices in the CPU module at the start of CPU module operation activated by remote RUN. After performing the specified clear processing, CPU module runs according to the setting that specified by Device Initial Value in GX Works2.	○	×

- Response data[*1] (all set by the system)

Device	Item	Description	Remote RUN	Remote STOP
(d1)[0]	Request type	0090H: When station number is specified in (s1)[5] 00B0H: When all stations or a group is specified in (s1)[5]	○	○
(d1)[1]	Sub-request type	0001H: Remote RUN 0002H: Remote STOP	○	○

*1 When "all stations or a group (81H to A0H, FFH)" is specified in (s1)[5], no response data will be stored.

■Reading/writing the clock data

- Request data (all set by the user)

Device	Item	Description	Read clock data	Write clock data
(s2)[0]	Request type	0001H: Clock data read 0011H: Clock data write (When station number is specified in (s1)[5]) 0031H: Clock data write (When all stations or a group is specified in (s1)[5])	○	○
(s2)[1]	Sub-request type	0002H: Clock data read 0001H: Clock data write	○	○
(s2)[2]	Change pattern, Clock data to be changed	Change pattern (bit 7 to bit 0) Specify the items to be written in high-order byte of (s2)[2] to (s2)[5]. 0: Not changed 1: Changed Year to be changed (bit 15 to bit 8)*1 Store the year (last two digits) in BCD format. b15 to b8: Year (00H to 99H); b7: 0; b6: Day of week; b5: Second; b4: Minute; b3: Hour; b2: Day; b1: Month; b0: Year (last two digits)	×	○
(s2)[3]	Clock data to be changed (continued)	High-order 8 bits: Day (01H to 31H), low-order 8 bits: Month (01H to 12H) b15 to b8: Day (01H to 31H); b7 to b0: Month (01H to 12H)	×	○
(s2)[4]		High-order 8 bits: Minute (00H to 59H), low-order 8 bits: Hour (00H to 23H) b15 to b8: Minute (00H to 59H); b7 to b0: Hour (00H to 23H)	×	○
(s2)[5]		High-order 8 bits: Day of week (00H (Sunday) to 06H (Saturday)), low-order 8 bits: Second (00H to 59H) b15 to b8: Day of week (00H to 06H) → 00H (Sun.) to 06H (Sat.); b7 to b0: Second (00H to 59H)	×	○

*1 This function cannot change the first two digits of year data.
To change the year data including the first two digits, set the clock data using another function (such as GX Works2).

• Response data (all set by the system)

Device	Item	Description	Read clock data	Write clock data
(d1)[0]	Request type	0081H: Clock data read 0091H: Clock data write (When station number is specified in (s1)[5]) 00B1H: Clock data write (When all stations or a group is specified in (s1)[5])[*2]	○	○
(d1)[1]	Sub-request type	0002H: Clock data read 0001H: Clock data write	○	○
(d1)[2]	Read clock data	High-order 8 bits: Month (01H to 12H), low-order 8 bits: Year (00H to 99H)[*3] b15 to b8 | b7 to b0 Month (01H to 12H) | Year (00H to 99H)	○	×
(d1)[3]		High-order 8 bits: Hour (00H to 23H), low-order 8 bits: Day (01H to 31H) b15 to b8 | b7 to b0 Hour (00H to 23H) | Day (01H to 31H)	○	×
(d1)[4]		High-order 8 bits: Second (00H to 59H), low-order 8 bits (00H to 59H) b15 to b8 | b7 to b0 Second (00H to 59H) | Minute (00H to 59H)	○	×
(d1)[5]		High-order 8 bits: (00H), low-order 8 bits: Day of week (00H (Sunday) to 06H (Saturday)) b15 to b8 | b7 to b0 00H | Day of week (00H to 06H) → 00H (Sun.) to 06H (Sat.)	○	×

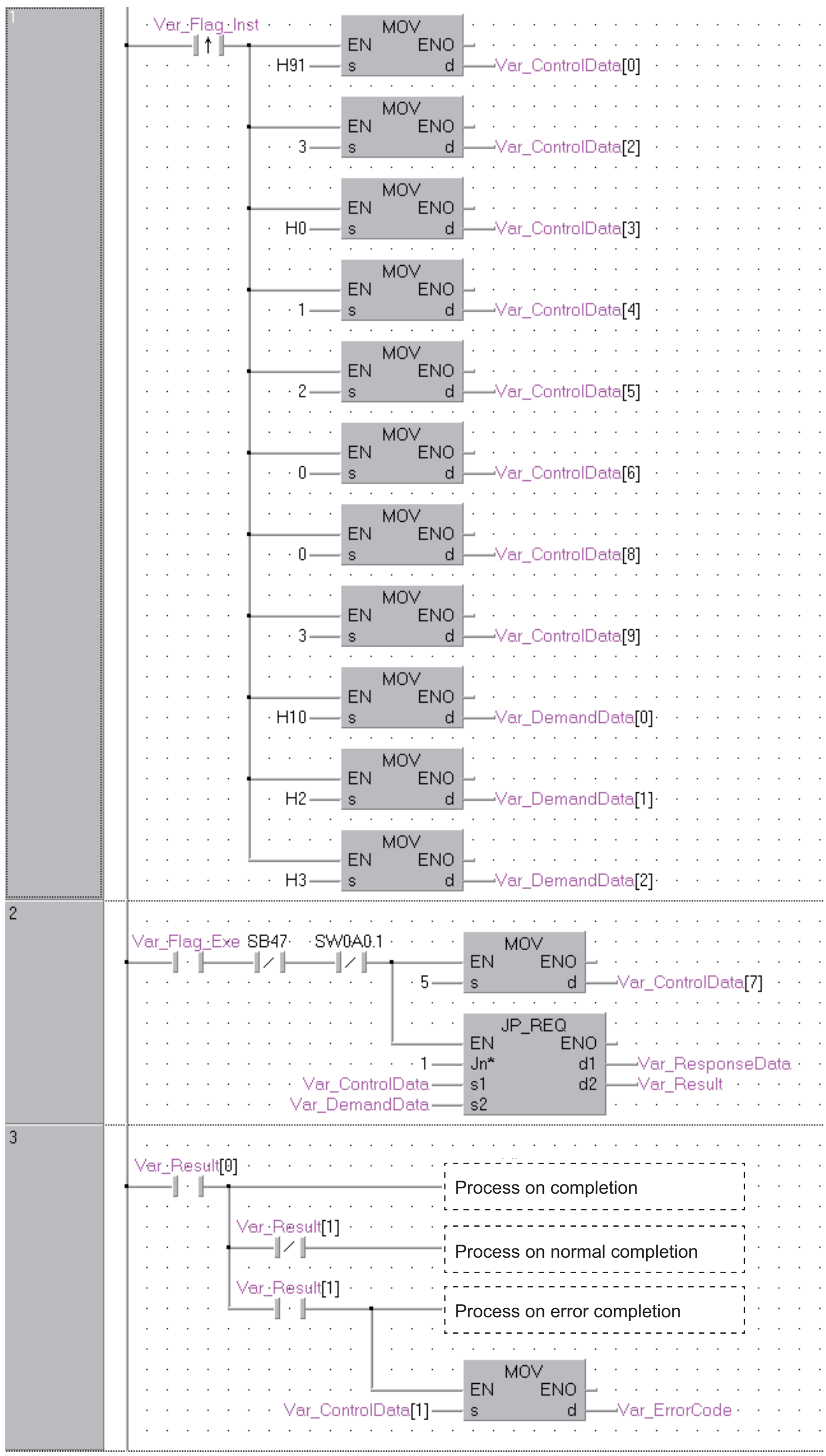
*2 When "all stations or a group (81H to A0H, FFH)" is specified in (s1)[5], no response data will be stored.

*3 Last two digits of year data

Program example

• The following program performs remote STOP to the QCPU, which is the station number 2 (target station).

[Structured ladder/FBD]



Sets execution/error completion type

Sets channel used by host station

Sets target station's CPU type

Sets target station network number

Sets target station number

Sets monitoring time

Sets data length by the word

Sets request data

Sets number of resends

Performs transient request to another station

Execution finished

Normal completion

Error completion

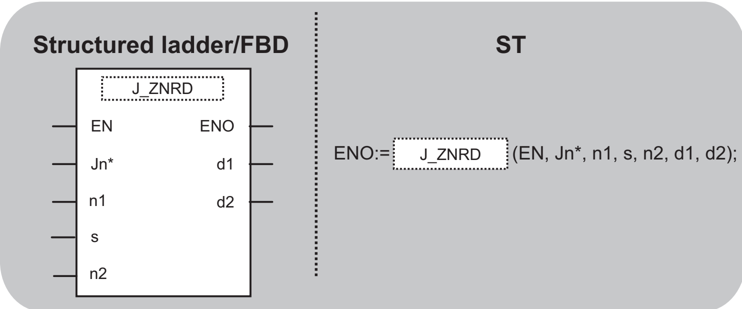
Stores error code

```
[ST]
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H91,Var_ControlData[0]); (* Sets execution/error completion type *)
    MOV(TRUE,3,Var_ControlData[2]); (* Sets channel used by host station *)
    MOV(TRUE,H0,Var_ControlData[3]); (* Sets target station's CPU type *)
    MOV(TRUE,1,Var_ControlData[4]); (* Sets target station network number *)
    MOV(TRUE,2,Var_ControlData[5]); (* Sets target station number *)
    MOV(TRUE,0,Var_ControlData[6]);
    MOV(TRUE,0,Var_ControlData[8]); (* Sets monitoring time *)
    MOV(TRUE,3,Var_ControlData[9]); (* Sets data length by the word *)
    MOV(TRUE,H10,Var_DemandData[0]); (* Sets request data *)
    MOV(TRUE,H2,Var_DemandData[1]);
    MOV(TRUE,H3,Var_DemandData[2]);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.1=FALSE)) THEN
    MOV(TRUE, 5, Var_ControlData[7]); (* Sets number of resends *)
    JP_REQ(TRUE,1,Var_ControlData,Var_DemandData,Var_ResponseData,Var_Result); (* Performs transient request to another station *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

Read from other station devices

J(P)_ZNRD

CC IE C NET/H Ether



The following instruction can go in the dotted squares.

J_ZNRD, JP_ZNRD

■Executing condition

Instruction	Executing condition
J_ZNRD	
JP_ZNRD	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	ANY16
	Jn*	Network number of the host station (1 to 239)	ANY16
	n1	Target station number (1 to 64)	ANY16
	s	Target station's start device number where data to be read are stored	ANY16
	n2	Read data length When the target station is Q/QnA/AnUCPU: 1 to 230 words When the target station is anything other than Q/QnA/AnUCPU: 1 to 32 words	ANY16
Output argument	ENO	Execution result	Bit
	d1	The host station's start device number where readout data will be stored A contiguous area for the read data length is required.)	ANY16
	d2	The host station's device that is turned on for one scan upon completion of the instruction d2[1] also turns ON if the instruction execution has failed.	Array of bit [0..1]

Setting data[*1*2]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○			—				○	—
(s)	—	○	—	—				—	—
n2	○			—				○	—
(d1)	—	○		—				—	—
(d2)	○			—				—	—

*1 Local devices and file registers per program cannot be used as setting data.
*2 In addition to the setting data, the ZNRD instruction is executed using the following fixed values.
Channel used by host station: Channel 1
Arrival monitoring time (monitoring time until instruction completion): 10 seconds
Number of resends for arrival monitoring timeout: 5 times

Processing details

This instruction reads data from devices of a programmable controller CPU on another station. (In units of words)

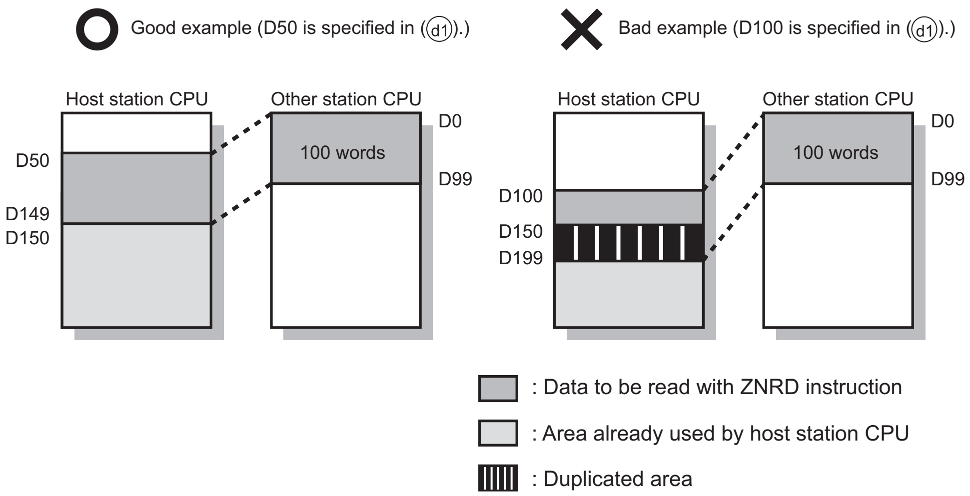
Point

- Specify devices of the target station's CPU within the range allowed for the host station CPU when reading data from the devices with the ZNRD instruction.

(Target station's start device number (s1) where data to be read are stored) + (Read points - 1) ≤ (End device No. of host station's CPU*3)

- Specify the host station's start device number (d1) within the range allowed for storing read data.

(Example) When D150 and after the area in the host station's CPU has been already used

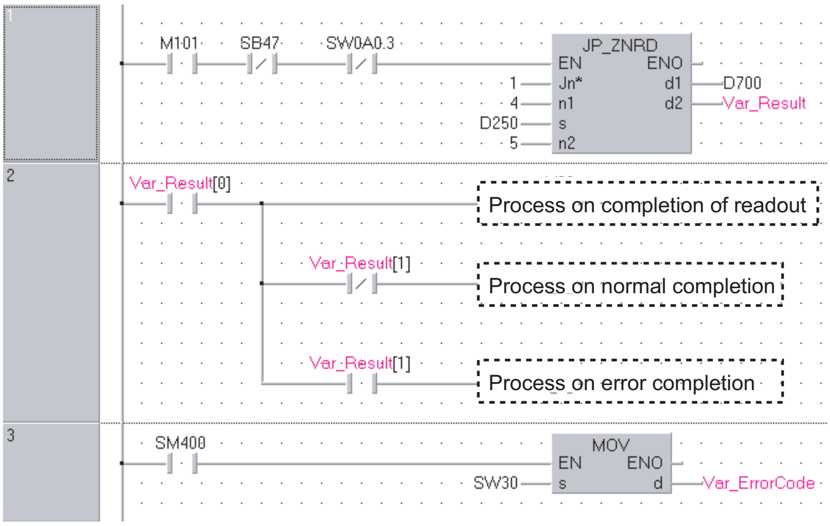


*3 End device No. of the device in the host station CPU, and whose device name is same as in (s1)

Program example

- n this program example, when M101 turns ON, data in D250 to D254 of station No.4 (target station) are read out to D700 to D704 of station No.1 (host station).

[Structured ladder/FBD]



Performs readout

Stores error code

[ST]

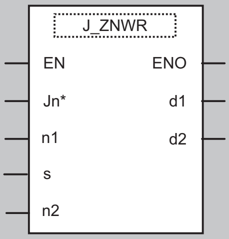
```
IF((M101=TRUE) &(SB47=FALSE) & (SW0A0.3=FALSE)) THEN
    JP_ZNRD(TRUE,1,4,D250,5,D700, Var_Result); (* Performs ZNRD instruction*)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of readout *)
        IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
        ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;

IF(SM400=TRUE)THEN
MOV(TRUE,SW30,Var_ErrorCode); (* Stores error code *)
END_IF;
```

Write to other station devices

J(P)_ZNWR

CC IE C NET/H Ether

Structured ladder/FBD



ST

ENO:= J_ZNWR (EN, Jn*, n1, s, n2, d1, d2);

The following instruction can go in the dotted squares.

J_ZNWR, JP_ZNWR

■Executing condition

Instruction	Executing condition
J_ZNWR	
JP_ZNWR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239)	ANY16
	n1	Target station number (1) Station No. specification 1 to 64: Station number (2) Group specification 81H to A0H: All stations of a group (No.1 to 32) (3) All stations FFH: All stations of the target network number (Except the host station)	ANY16
	s	Host station's start device number where data to be written are stored	ANY16
	n2	Write data length When the target station is Q/QnA/AnUCPU: 1 to 230 words When the target station is anything other than Q/QnA/AnUCPU: 1 to 32 words	ANY16
Output argument	ENO	Execution result	Bit
	d1	Target station's start device number where data is written (A contiguous area for the write data length is required.)	ANY16
	d2	The host station's device that is turned on for one scan upon completion of the instruction d2[1] also turns ON if the instruction execution has failed.	Array of bit [0..1]

Setting data*1*2	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	○			—				○	—
(s)	—	○	—	—				—	—
n2	○			—				—	—

Setting data*1*2	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(d1)	—	○		—				○	—
(d2)	○			—				—	—

*1 Local devices and file registers per program cannot be used as a device which is used in setting data.
*2 In addition to the setting data, the ZNWR instruction is executed using the following fixed values.
Channel used by host station: Channel 2
Arrival monitoring time (monitoring time until instruction completion): 10 seconds
Number of resends for arrival monitoring timeout: 5 times

Processing details

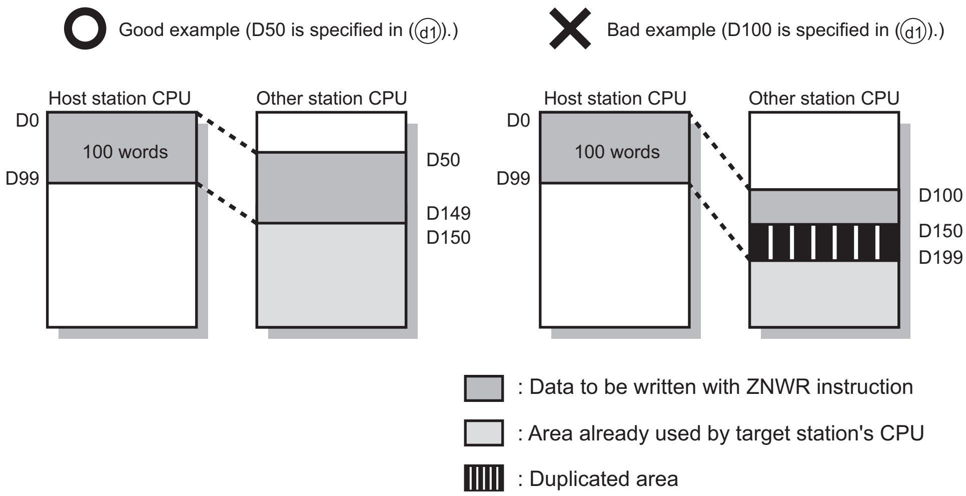
This instruction writes data to devices of a programmable controller CPU on another station. (In units of words)

Point

- Specify devices of the target station's CPU within the range allowed for the host station CPU when writing data to the devices with the ZNWR instruction.

(Target station's start device number (d1) where data are written) + (Write points - 1) ≤ (End device No. of host station's CPU*3)

- Specify the host station's start device number (d1) within the range allowed for storing write data. (Example) When D150 and after the area in the host station's CPU has been already used

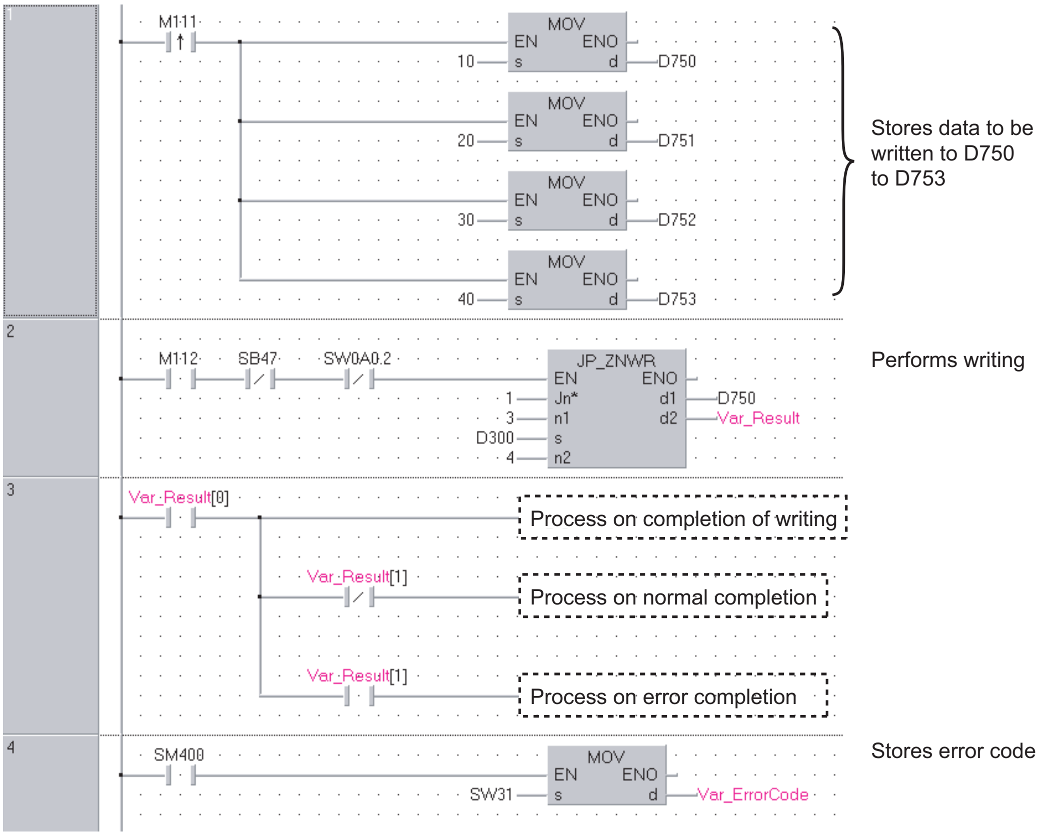


*3 End device No. of the device in the host station CPU, and whose device name is same as in (d1)

Program example

• In this program example, when M112 turns ON, data in D750 to D753 of station No.2 (host station) are written to D300 to D303 of station No.3 (target station).

[Structured ladder/FBD]



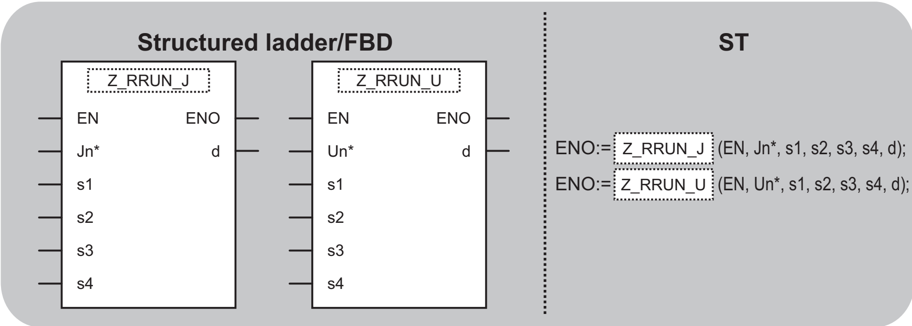
[ST]

```
IF(M111=TRUE)THEN (* Instruction flag ON *)
    MOV( TRUE, 10, D750);
    MOV( TRUE, 20, D751);
    MOV( TRUE, 30, D752 );
    MOV( TRUE, 40, D753 ); (* Stores data to be written to D750 to D753 *)
END_IF;

IF((M112=TRUE) &(SB47=FALSE) & (SW0A0.2=FALSE)) THEN
    JP_ZNWR(TRUE,1,3,D300,4, D750, Var_Result); (* Performs writing *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Completion of writing *)
(* Process on completion of writing *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;

IF(SM400=TRUE)THEN
    MOV(TRUE,SW31,Var_ErrorCode); (* Stores error code *)
END_IF;
```

RRUN instruction

Z(P)_RRUN_J, Z(P)_RRUN_U

CC IE C NET/H



The following instruction can go in the dotted squares.

Z_RRUN_J, ZP_RRUN_J, Z_RRUN_U, ZP_RRUN_U

■Executing condition

Instruction	Executing condition
Z_RRUN_J Z_RRUN_U	
ZP_RRUN_J ZP_RRUN_U	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the target station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	String
	Un*	Start I/O number of the host station network No. (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Channel used by host station For the RRUN instruction, specify the channel used by host station that is the same as the one used for the RSTOP instruction.	ANY16
	s2	Target station number (1) Station number specification Host station is Universal model QCPU: 1 to 120 Host station is anything other than Universal model QCPU: 1 to 64 (2) Group specification 81H to A0H: All stations of a group (No.1 to 32) (3) All stations FFH: All stations of the target network No. (Except the host station) To specify a group or all stations, specify '0000H' or '03FFH' for the target station's CPU type (s3).	ANY16
	s3	Target station's CPU type 0000H: Target station CPU/control CPU/host system CPU (Specified data are the same as '03FFH'.) 03E0H: Multi-CPU No. 1/target station CPU (single CPU system) 03E1H: Multi-CPU No. 2 03E2H: Multi-CPU No. 3 03E3H: Multi-CPU No. 4 03FFH: Target station CPU/control CPU/host system CPU	ANY16
	s4	Mode	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				○	—
(s3)	—	○		—				○	—
(s4)	—	○		—				○	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction remotely switches a CPU module on another station to RUN.

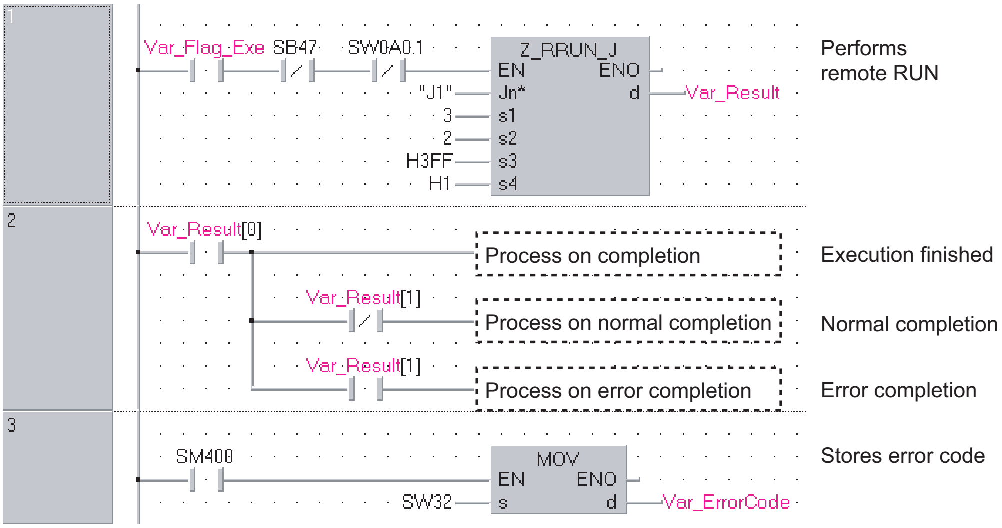
Precautions

This instruction is applicable to the QJ71LP21 or QJ71BR11 with the function version B or later.

Program example

- The following program remotely switches the QCPU on the station number 2 (target station) to RUN.

[Structured ladder/FBD]



[ST]

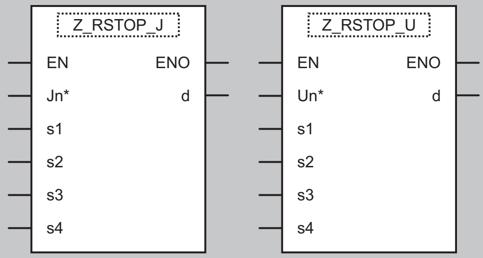
```
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.1=FALSE)) THEN
    Z_RRUN_J(TRUE,"J1",3,2,H3FF,H1,Var_Result); (* Performs remote RUN *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;

MOV(SM400,SW32,Var_ErrorCode); (* Stores error code *)
```

RSTOP instruction

Z(P)_RSTOP_J, Z(P)_RSTOP_U

CC IE C NET/H

Structured ladder/FBD



ST

ENO:= Z_RSTOP_J (EN, Jn*, s1, s2, s3, s4, d);

ENO:= Z_RSTOP_U (EN, Un*, s1, s2, s3, s4, d);

The following instruction can go in the dotted squares.

Z_RSTOP_J, ZP_RSTOP_J, Z_RSTOP_U, ZP_RSTOP_U

■Executing condition

Instruction	Executing condition
Z_RSTOP_J Z_RSTOP_U	
ZP_RSTOP_J ZP_RSTOP_U	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239) 254: Network specified in "Valid module during other station access"	String
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Channel used by host station	ANY16
	s2	Target station number (1) Station number specification Host station is Universal model QCPU: 1 to 120 Host station is anything other than Universal model QCPU: 1 to 64 (2) Group specification 81H to A0H: All stations of a group (No.1 to 32) (3) All stations FFH: All stations of the target network No. (Except the host station)	ANY16
	s3	Target station's CPU type 0000H: Target station CPU/control CPU/host system CPU (Specified data are the same as '03FFH'.) 03E0H: Multi-CPU No. 1/target station CPU (single CPU system) 03E1H: Multi-CPU No. 2 03E2H: Multi-CPU No. 3 03E3H: Multi-CPU No. 4 03FFH: Target station CPU/control CPU/host system CPU	ANY16
	s4	Specify options for the operation mode and clear mode. (1) Operation mode 1H: No forced execution 3H: Forced execution (2) Clear mode 0H: Do not clear (Note that the local devices are cleared.) 1H: Clear (excluding the latch range) 2H: Clear (including the latch range)	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				○	—
(s3)	—	○		—				○	—
(s4)	—	○		—				○	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction remotely switches a CPU module on another station to STOP.

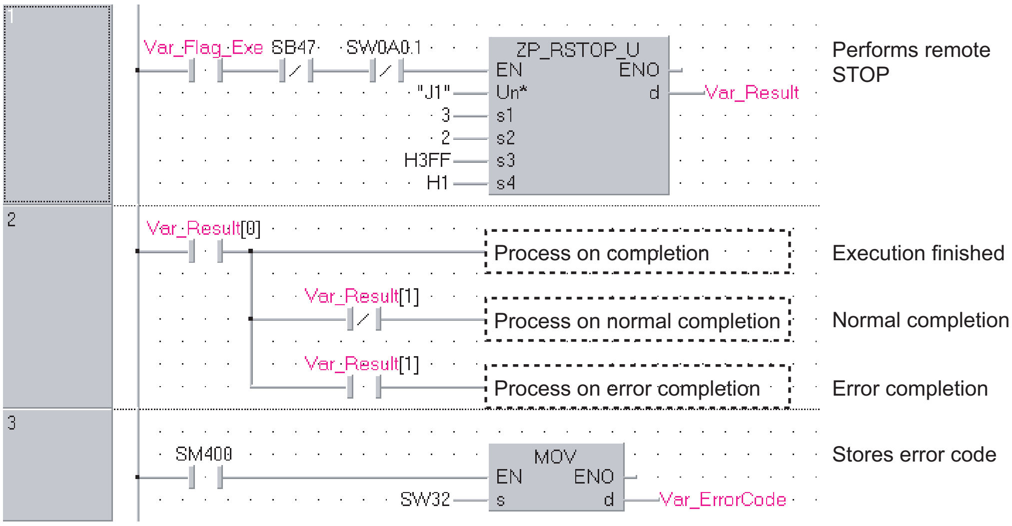
Precautions

This instruction is applicable to the QJ71LP21 or QJ71BR11 with the function version B or later.

Program example

• The following program remotely switches the QCPU on the station number 2 (target station) to STOP.

[Structured ladder/FBD]



[ST]

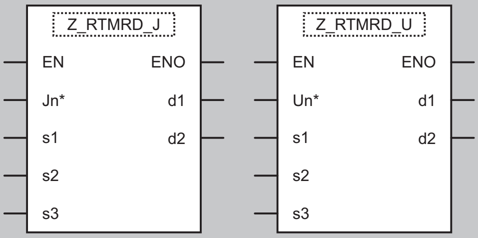
```
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.1=FALSE)) THEN
    ZP_RSTOP_J(TRUE,"J1",3,2,H3FF,H1,Var_Result); (* Performs remote STOP *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;

MOV(SM400, SW32, Var_ErrorCode); (* Stores error code *)
```

Reading clock data from another station

Z(P)_RTMRD_J, Z(P)_RTMRD_U

CC IE C NET/H

Structured ladder/FBD



ST

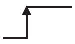
ENO:=Z_RTMRD_J(EN, Jn*, s1, s2, s3, d1, d2);

ENO:=Z_RTMRD_U(EN, Un*, s1, s2, s3, d1, d2);

The following instruction can go in the dotted squares.

Z_RTMRD_J, ZP_RTMRD_J, Z_RTMRD_U, ZP_RTMRD_U

■Executing condition

Instruction	Executing condition
Z_RTMRD_J Z_RTMRD_U	
ZP_RTMRD_J ZP_RTMRD_U	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239) 254: Network specified in "Valid module during other station access"	String
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Channel used by host station	ANY16
	s2	Target station number Host station is Universal model QCPU: 1 to 120 Host station is anything other than Universal model QCPU: 1 to 64	ANY16
	s3	Target station's CPU type 0000H: Target station CPU/control CPU/host system CPU (Specified data are the same as '03FFH'.) 03E0H: Multi-CPU No. 1/target station CPU (single CPU system) 03E1H: Multi-CPU No. 2 03E2H: Multi-CPU No. 3 03E3H: Multi-CPU No. 4 03FFH: Target station CPU/control CPU/host system CPU	ANY16
Output argument	ENO	Execution result	Bit
	d1	Variable that stores read clock data	Array of ANY16 [0..3]
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				○	—
(s3)	—	○		—				○	—
(d1)	—	○		—				—	—
(d2)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads clock data from a CPU module on another station.

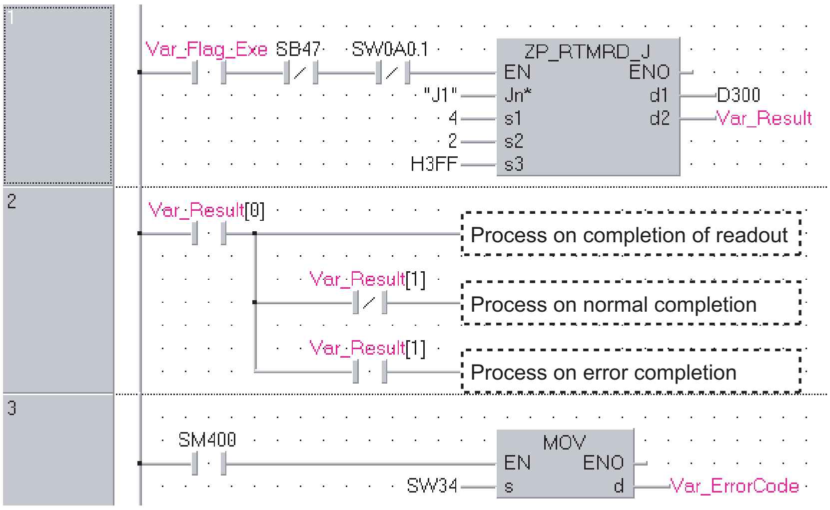
Precautions

This instruction is applicable to the QJ71LP21 or QJ71BR11 with the function version B or later.

Program example

- The following program reads out clock data from the QCPU on the station number 2 (target station) and stores the clock data in the station number 1 (host station).

[Structured ladder/FBD]



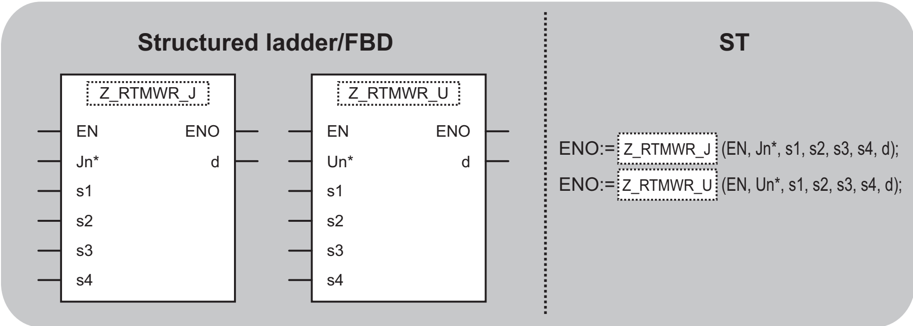
[ST]

```
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SW0A0.1=FALSE)) THEN
    ZP_RTMRD_J(TRUE,"J1",4,2,H3FF,D300,Var_Result); (* Reads clock data from another station *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)

(* Process on completion of readout *)

    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)

(* Process on normal completion *)

    ELSE (* Error completion *)

(* Process on error completion *)

    END_IF;
END_IF;
MOV(SM400, SW33, Var_ErrorCode); (* Stores error code *)
```

Writing clock data to another station

Z(P)_RTMWR_J, Z(P)_RTMWR_U

CC IE C NET/H



The following instruction can go in the dotted squares.
Z_RTMWR_J, ZP_RTMWR_J, Z_RTMWR_U, ZP_RTMWR_U

■Executing condition

Instruction	Executing condition
Z_RTMWR_J Z_RTMWR_U	
ZP_RTMWR_J ZP_RTMWR_U	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the target station (1 to 239, 254) 254: Network specified in "Valid module during other station access"	String
	Un*	Start I/O number of the host station network No. (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Channel used by host station (1 to 8)	ANY16
	s2	Target station number (1) Station number specification Host station is Universal model QCPU: 1 to 120 Host station is anything other than Universal model QCPU: 1 to 64 (2) Group specification 81H to A0H: All stations of a group (No.1 to 32) (3) All stations FFH: All stations of the target network No. (Except the host station) To specify a group or all stations, specify '0000H' or '03FFH' for the target station's CPU type (s3).	ANY16
	s3	Target station's CPU type 0000H: Target station CPU/control CPU/host system CPU (Specified data are the same as '03FFH'.) 03E0H: Multi-CPU No. 1/target station CPU (single CPU system) 03E1H: Multi-CPU No. 2 03E2H: Multi-CPU No. 3 03E3H: Multi-CPU No. 4 03FFH: Target station CPU/control CPU/host system CPU	ANY16
	s4	Variable that stores write clock data	Array of ANY16 [0..4]

Input/output argument	Name	Description	Data type
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				○	—
(s3)	—	○		—				○	—
(s4)	—	○		—				—	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction writes clock data to a CPU module on another station.

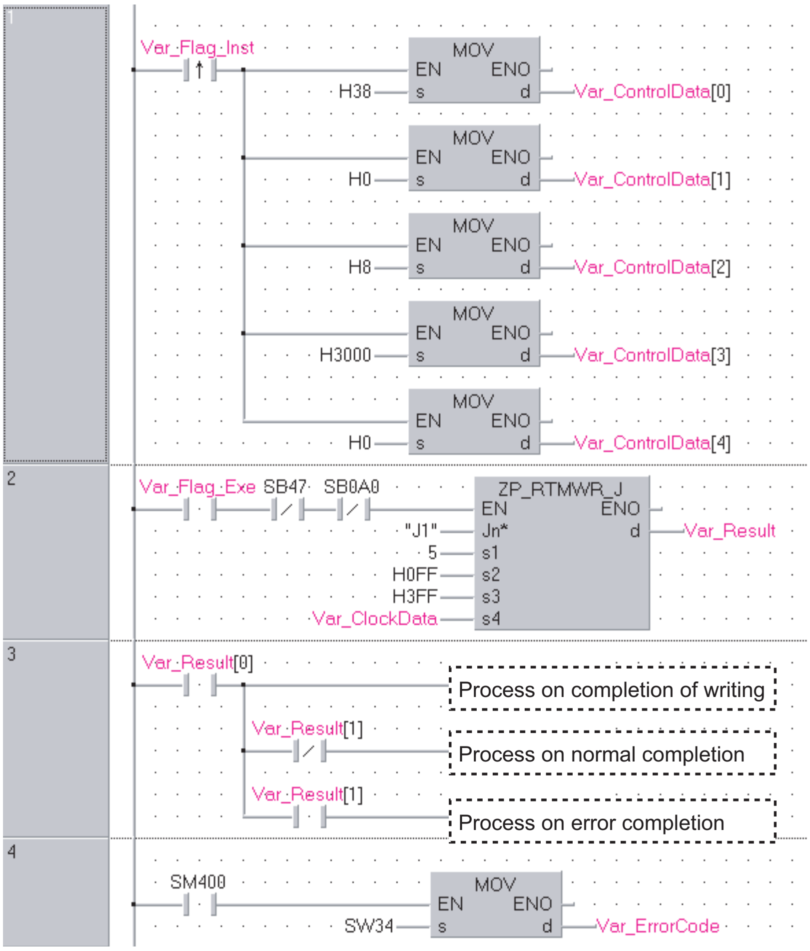
Precautions

This instruction is applicable to the QJ71LP21 or QJ71BR11 with the function version B or later.

Program example

• The following program writes the clock data (8:30:00) to all stations on the network number 1.

[Structured ladder/FBD]



Sets clock data

Writes clock data to other stations

Execution finished

Normal completion

Error completion

Stores error code

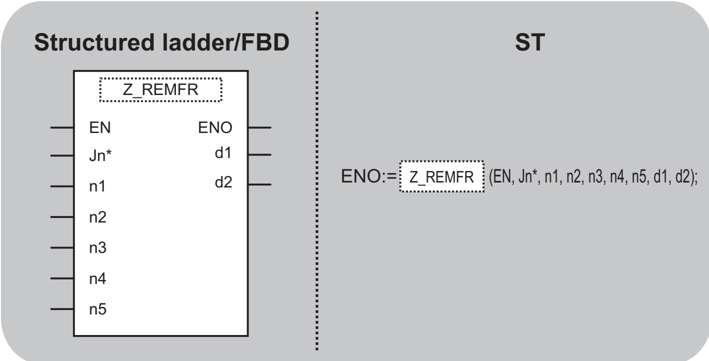
[ST]

```
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H38,Var_ClockData[0]); (* Sets clock data *)
    MOV(TRUE,H0,Var_ClockData[1]);
    MOV(TRUE,H8,Var_ClockData[2]);
    MOV(TRUE,H3000,Var_ClockData[3]);
    MOV(TRUE,H0,Var_ClockData[4]);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB47=FALSE) AND (SB0A0=FALSE)) THEN
    ZP_RTMWR_J(TRUE,"J1",5,H0FF,H3FF,Var_ClockData,Var_Result); (* Writes clock data to other stations*)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion of writing *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;
MOV(SM400, SW34, Var_ErrorCode); (* Stores error code *)
```

Reading from buffer memory of intelligent function module on remote I/O station

Z(P)_REMFR

CC IE F NET/H



The following instruction can go in the dotted squares.

Z_REMFR, ZP_REMFR

■Executing condition

Instruction	Executing condition
Z_REMFR	
ZP_REMFR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Target network number (1 to 239)	String
	n1	Channel number CC-Link IE Field Network: 1 to 32 MELSECNET/H: 1 to 8	ANY16
	n2	Target station number CC-Link IE Field Network: 1 to 120 MELSECNET/H: 1 to 64	ANY16
	n3	Start I/O number of the target intelligent function module For the CC-Link IE Field Network, the higher two digits when expressing the I/O number in three digits. For the MELSECNET/H, the higher three digits when expressing the I/O number in four digits.	ANY16
	n4	Read buffer memory start address Specifies the start address of the buffer memory for the read destination intelligent function module.	ANY16
	n5	Number of read points CC-Link IE Field Network: 1 to 240 MELSECNET/H: 1 to 960	ANY16

Input/output argument	Name	Description	Data type
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores read data (host station) Specifies the start number of the host station's device that stores read data.	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○		—				○	—
n2	—	○		—				○	—
n3	—	○		—				○	—
n4	—	○		—				○	—
n5	—	○		—				○	—
(d1)	—	○		—				—	—
(d2)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

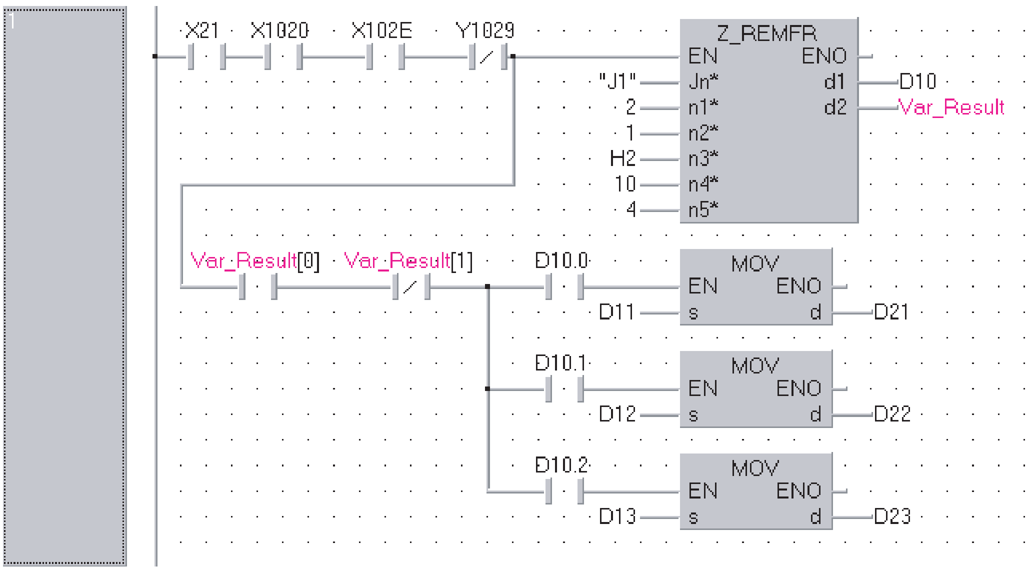
Processing details

This instruction reads data from the buffer memory of an intelligent function module to the host station's word device (starting from (d1)) on the intelligent device station/remote I/O station.

Program example

- The following program reads digital output values.

[Structured ladder/FBD]



Reads data from buffer memory

Reads CH1 digital value

Reads CH2 digital value

Reads CH3 digital value

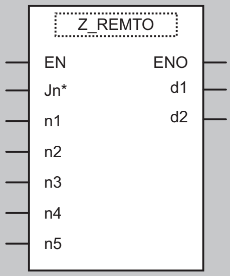
[ST]

```
IF((X21=TRUE) AND (X1020=TRUE) AND (X102E=TRUE) AND (Y1029=FALSE))THEN
    Z_REMFR(TRUE,"J1",2,1,H2,10,4,D10,Var_Result); (* Reads data from buffer memory *)
    (*Reads digital values of CH1 to CH3 at once*)
    IF((Var_Result[0]=TRUE) AND (Var_Result[1]=FALSE))THEN
        IF(D10.0=TRUE)THEN
        MOV(TRUE,D11,D21); (* Reads CH1 digital output value *)
        END_IF;
        IF(D10.1=TRUE)THEN
        MOV(TRUE,D12,D22); (* Reads CH2 digital output value *)
        END_IF;
        IF(D10.2=TRUE)THEN
        MOV(TRUE,D13,D23); (* Reads CH3 digital output value *)
        END_IF;
    END_IF;
END_IF;
```

Writing to buffer memory of intelligent function module on remote I/O station

Z(P)_REMTO

CC IE F NET/H

Structured ladder/FBD	ST
	ENO:= Z_REMTO (EN, Jn*, n1, n2, n3, n4, n5, d1, d2);

The following instruction can go in the dotted squares.

Z_REMTO, ZP_REMTO

■Executing condition

Instruction	Executing condition
Z_REMTO	
ZP_REMTO	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Jn*	Network number of the host station (1 to 239)	String
	n1	Channel number CC-Link IE Field Network: 1 to 32 MELSECNET/H: 1 to 8	ANY16
	n2	Target station number CC-Link IE Field Network: 1 to 120 MELSECNET/H: 1 to 64	ANY16
	n3	Start I/O number of the target intelligent function module For the CC-Link IE Field Network, the higher two digits when expressing the I/O number in three digits. For the MELSECNET/H, the higher three digits when expressing the I/O number in four digits.	ANY16
	n4	Write buffer memory start address Specifies the start address of the buffer memory for the write destination intelligent function module.	ANY16
	n5	Number of write points CC-Link IE Field Network: 1 to 240 MELSECNET/H: 1 to 960	ANY16

Input/output argument	Name	Description	Data type
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores write data (host station) Specifies the start number of the host station's device that stores write data.	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n1	—	○		—				○	—
n2	—	○		—				○	—
n3	—	○		—				○	—
n4	—	○		—				○	—
n5	—	○		—				○	—
(d1)	—	○		—				—	—
(d2)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

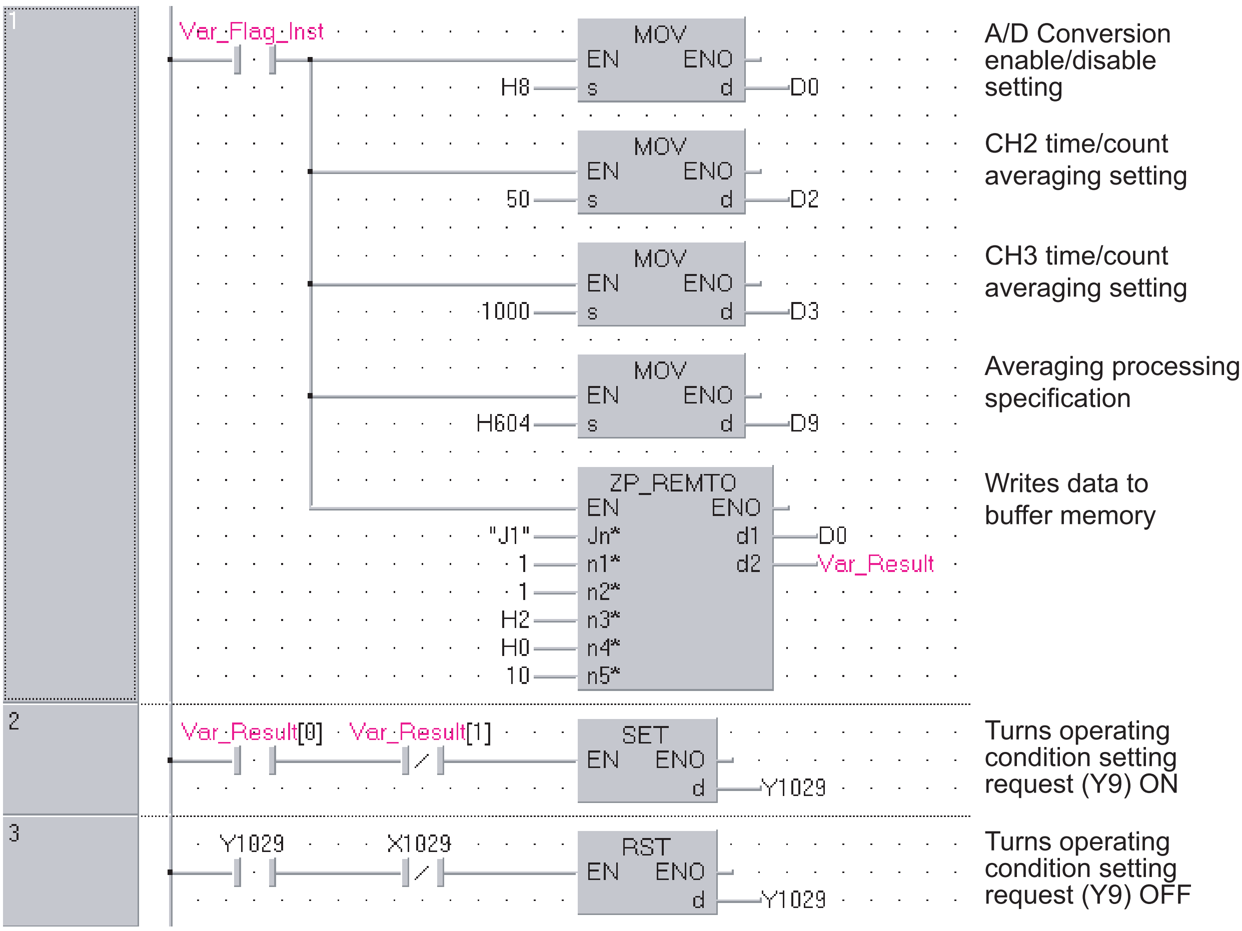
Processing details

This instruction writes data to the buffer memory of an intelligent function module on the intelligent device station/remote I/O station.

Program example

- The following program makes the A/D conversion enable setting on channels.

[Structured ladder/FBD]



[ST]

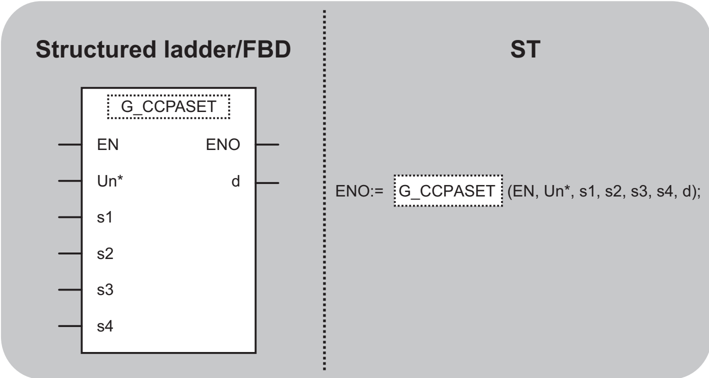
```
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H8,D0); (* A/D Conversion enable/disable setting *)
    MOV(TRUE,50,D2); (* CH2 time/count averaging setting *)
    MOV(TRUE,1000,D3); (* CH3 time/count averaging setting *)
    MOV(TRUE,H604,D9); (* Averaging processing specification *)
    ZP_REMTO(TRUE,"J1",1,1,H2,H0,10,D0,Var_Result); (* Writes data to buffer memory *)
END_IF;
IF((Var_Result[0]=TRUE) AND (Var_Result[1]=FALSE))THEN
    SET(TRUE,Y1029); (* Turns operating condition setting request (Y9) ON *)
END_IF;
IF((Y1029=TRUE) AND (X1029=FALSE))THEN
    RST(TRUE,Y1029); (* Turns operating condition setting request (Y9) OFF *)
END_IF;
```

5

Setting parameter

G(P)_CCPASET

CC IE F



The following instruction can go in the dotted squares.
G_CCPASET, GP_CCPASET

■Executing condition

Instruction	Executing condition
G_CCPASET	
GP_CCPASET	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	ANY16
	s1	Variable that stores control data	Array of ANY16 [0..3]
	s2	Start number of the host station's device that stores network configuration setting data.	Array of ANY16 [0..599]
	s3	Start number of the host station's device that stores reserved station specification data.	Array of ANY16 [0..7]
	s4	Start number of the host station's device that stores error invalid station setting data.	Array of ANY16 [0..7]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of Bit [0..1]

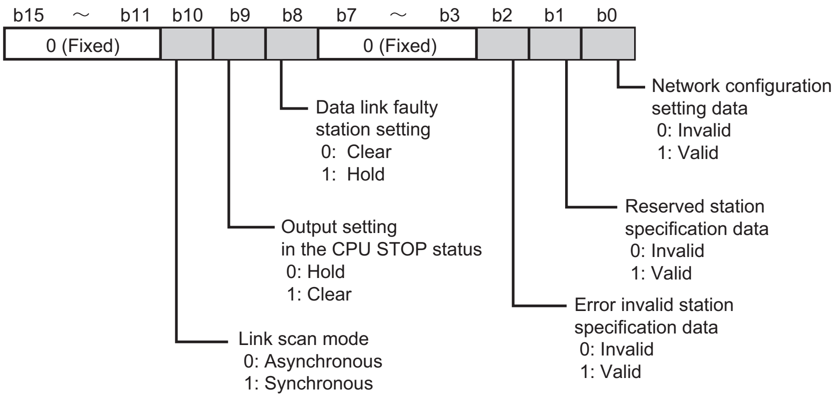
Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(s3)	—	○		—					
(s4)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sets parameters for master/local module (master station).

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[1]	Setting flag	Specify the validity of setting data from (s2) to (s4) in the range from b0 to b2. '0: Invalid' is specified, default parameter is applied. The supplementary setting and the network operation setting in the range from b8 to bA. 	Refer to the left.	User
(s1)[2]	Total number of slave station	Specify the number of connected slave stations.	1 to 120	User
(s1)[3]	Constant link scan time	Set the constant link scan time. 0: No setting 5 to 2000: Constant link scan time	5 to 2000(ms)	User

5

- Network configuration setting data

Set the network configuration settings when network configuration setting data (b0) is enabled in the setting flag ((s1)[1]).

Device	Item		Setting data	Setting range	Setting side
(s2)[0]	1st	Slave station setting information	Specify the station type and station number. b15 ～ b12 b11 ～ b8 b7 ～ b0 Station type | 1 (Fixed) | Station number Station type: 0 : Remote I/O station 1 : Remote device station 2 : Intelligent device station 3 : Local station Station number: 1 to 120 : Station number	Refer to the left.	User
(s2)[1]		RX/RY offset	Specify the start number of RX/RY in units of 16 points.	0 to 3FF0H	
(s2)[2]		RX/RY size	Specify the number of RX/RY in units of 16 points.	0 to 2048	
(s2)[3]		RWr/RWw offset	Specify the start number of RWr/RWw. in units of 4 points.	0 to 1FFCH	
(s2)[4]		RWr/RWw size	Specify the number of RWr/RWw. in units of 4 points.	0 to 1024	
(s2)[5] ⋮ (s2)[594]	· · · · ·				
(s2)[595]	120th	Slave station setting information	The same as from (s2)[0] to (s2)[4].		
(s2)[596]		RX/RY offset			
(s2)[597]		RX/RY size			
(s2)[598]		RWr/RWw offset			
(s2)[599]		RWr/RWw size			

- Reserved station specification data

Set the slave station as the reserved station when reserved station specification data (b1) is enabled in the setting flag ((s1)[1]).

Device	Item	Setting data	Setting side
(s3)[0] ⋮ (s3)[7]	Reserved station specification	Specify the reserved station. 0: Not specified (Default) 1: Specified	User

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(s3)[0]	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
(s3)[1]	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
(s3)[2]	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
(s3)[3]	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49
(s3)[4]	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65
(s3)[5]	96	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81
(s3)[6]	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97
(s3)[7]	—	—	—	—	—	—	—	—	120	119	118	117	116	115	114	113

Numbers in the table indicate the station numbers.

- Error invalid station setting data

Set the slave station as the error invalid station when error invalid station setting data(b2) is enabled in the setting flag ((s1)[1])

Device	Item	Setting data	Setting side
(s4)[0] ⋮ (s4)[7]	Error invalid station setting*1	Specify the error invalid station. 0: Not specified (Default) 1: Specified	User

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
s4[0]	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
s4[1]	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
s4[2]	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
s4[3]	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49
s4[4]	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65
s4[5]	96	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81
s4[6]	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97
s4[7]	—	—	—	—	—	—	—	—	120	119	118	117	116	115	114	113

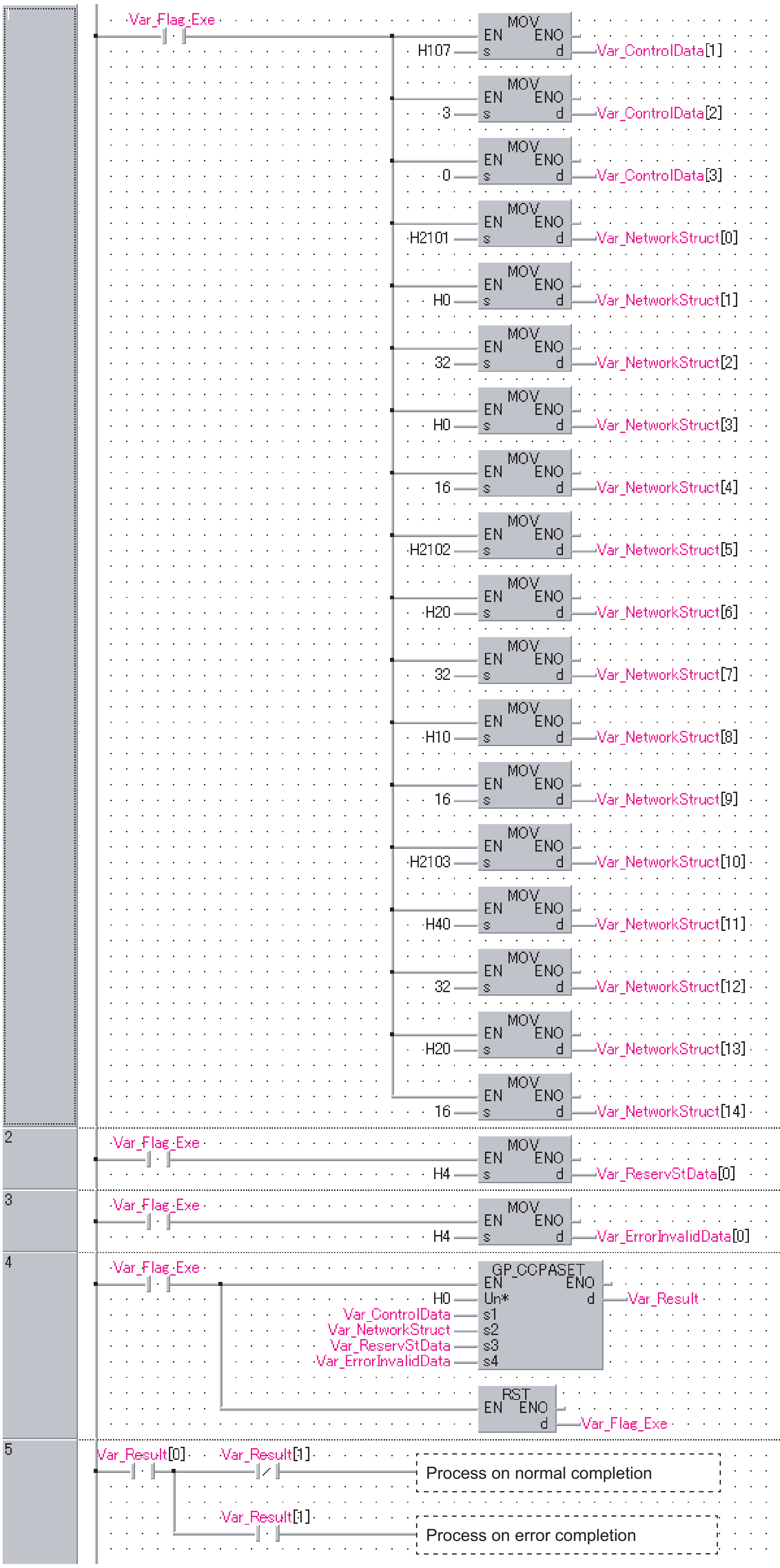
Numbers in the table indicate the station numbers.

*1 Reserved station specification has a priority when an error invalid station and reserved station are specified for the same station.

Program example

- The following program sets parameters for master station of network No.1 when Var_Flag_Exe turns ON. (Total number of slave stations is 3.)

[Structured ladder/FBD]



Set control data

Set network configuration setting data

Set reserved station specification data

Error invalid station specification data

Performs writing

Turns execution flag OFF

[ST]

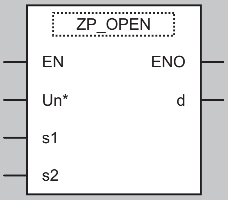
```
IF( Var_Flag_Exe = TRUE ) (* Execution flag *)
    MOV( TRUE, H107, Var_ControlData[1]); (* Sets control data *)
    MOV( TRUE, 3, Var_ControlData[2]);
    MOV( TRUE, 0, Var_ControlData[3]);
    MOV( TRUE, H2101, Var_NetworkStruct[0] ); (* Sets data of network configuration setting *)
    MOV( TRUE, H0, Var_NetworkStruct[1] );
    MOV( TRUE, 32, Var_NetworkStruct[2] );
    MOV( TRUE, H0, Var_NetworkStruct[3] );
    MOV( TRUE, 16, Var_NetworkStruct[4] );
    MOV( TRUE, H2102, Var_NetworkStruct[5] );
    MOV( TRUE, H20, Var_NetworkStruct[6 ] );
    MOV( TRUE, 32, Var_NetworkStruct[7] );
    MOV( TRUE, H10, Var_NetworkStruct[8] );
    MOV( TRUE, 16, Var_NetworkStruct[9] );
    MOV( TRUE, H2103, Var_NetworkStruct[10] );
    MOV( TRUE, H40, Var_NetworkStruct[11] );
    MOV( TRUE, 32, Var_NetworkStruct[12] );
    MOV( TRUE, H20, Var_NetworkStruct[13] );
    MOV( TRUE, 16, Var_NetworkStruct[14] );
END_IF;

IF( Var_Flag_Exe = TRUE ) (* Execution flag *)
    MOV( TRUE, H4, Var_ReservStData[0] ); (* Sets data of reserved station specification *)
END_IF;

IF( Var_Flag_Exe = TRUE ) (* Execution flag *)
    MOV( TRUE, H4, Var_ErrorInvalidData[0] ); (* Sets data of error invalid station setting *)
END_IF;

IF( Var_Flag_Exe = TRUE ) (* Execution flag *)
    GP_CCPASET(TRUE, H0, Var_ControlData, Var_NetworkStruct, Var_ReservStData, Var_ErrorInvalidData, Var_Result); (* Performs writing *)
    RST( TRUE, Var_Flag_Exe ); (* Turns execution flag OFF *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (*Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;

END_IF;
```

5

Connection opening or closing

ZP_OPEN

Ether

Structured ladder/FBD	ST
	ENO:= ZP_OPEN (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.

ZP_OPEN

■Executing condition

Instruction	Executing condition
ZP_OPEN	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..9]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				—	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction establishes (opens) a connection with external device for data communication.

Setting data

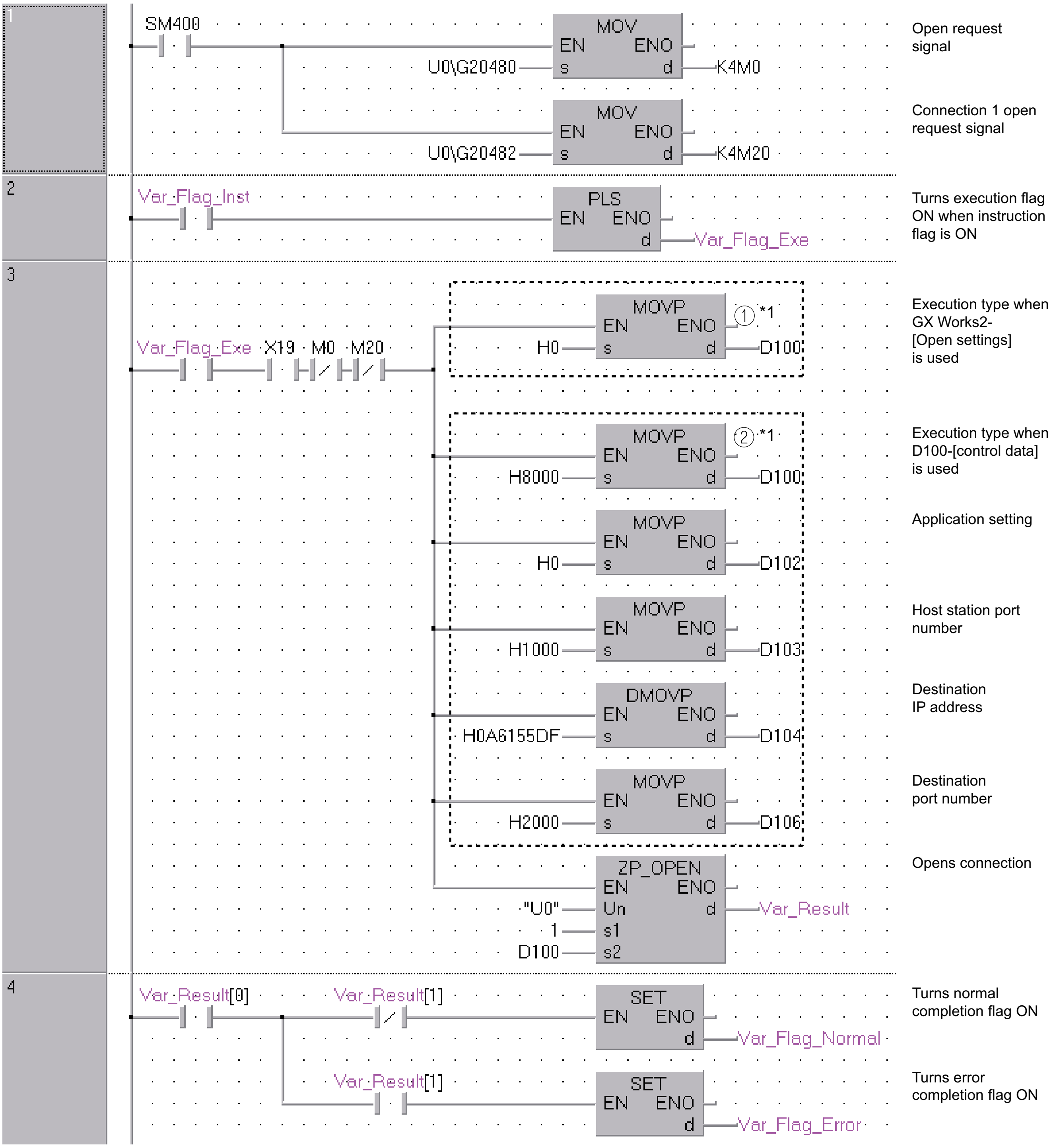
Device	Item	Setting data	Setting range	Setting side
(s2)[0]	Execution type/ Completion type	Specify whether to use the parameter values set by GX Works2 or the setting values of the following control data ((s2)[2] to (s2)[9]) at open processing of a connection. 0000H: Uses the parameter set in [Open settings] of GX Works2. 8000H: Uses the settings of control data (s2)[2] to (s2)[9].	0000H, 8000H	User
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2]	Application setting area	Specify the application of connection. b15 b14 … b9 b8 b7 … b1 b0 (6) | 0 | (5) | (4) | (3) | 0 | (2) | (1) (1) Application of fixed buffer (b0) 0: For sending, or fixed buffer is not used in communication 1: For receiving (2) Check of existence of the target (b1) 0: Not checked 1: Checked (3) Pairing open setting (b7) 0: No pairing open 1: Pairing open (4) Communication method (protocol) (b8) 0: TCP/IP 1: UDP/IP (5) With/without procedure in fixed buffer communication (b9) 0: Procedural communication Nonprocedural communication (6) Open system (b15, b14) 00: Active open or UDP/IP 10: Unpassive open 11: Fullpassive open	(Refer to the left.)	User
(s2)[3]	Host station port No.	Specify the port number of the host station.	401H to 1387H, 138BH to FFFEH	User
(s2)[4] (s2)[5]	Destination IP address	Specify the IP address of the external device.	1H to FFFFFFFFH (FFFFFFFFH: broadcast)	User
(s2)[6]	Destination port No.	Specify the port number of the external device.	401H to FFFFH (FFFFH: broadcast)	User
(s2)[7] ⋮ (s2)[9]	Destination MAC address	Specify the MAC address of the external device.	n 000000000000H FFFFFFFFFFFFH	User

Program example

- The following program opens the connection 1 for TCP/IP communication using the Active open process.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



*1 For divisions of (1) and (2) in the program, (1) is necessary when the [Open settings] of GX Works2 is used and (2) is necessary when it is not used.

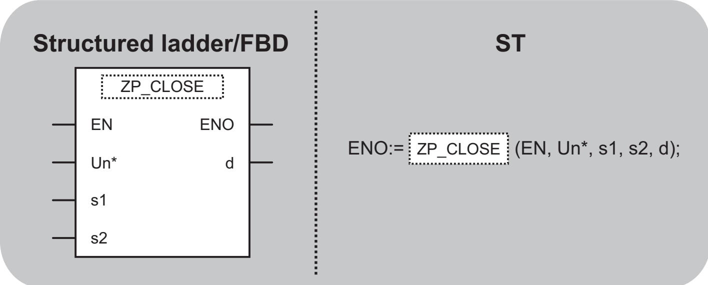
```
[ST]
IF(SM400=TRUE)THEN (* Always ON *)
    MOV(TRUE,U0\G20480,K4M0); (* Open completed signal/connection 1 open completion signal *)
    MOV(TRUE,U0\G20482,K4M20); (* Open request signal/connection 1 open request signal *)
END_IF;
IF(Var_Flag_Inst=TRUE)THEN (* When instruction flag is ON*)
    PLS(TRUE,Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;
IF((Var_Flag_Exe=TRUE) AND (X19=TRUE) (* Execution flag/initialization normal completion signal *)
    AND (M0=FALSE) AND (M20=FALSE))THEN (* Connection 1 open completion signal/connection 1 open request signal *)
①*1            (*Use GX Works2-[Open settings]*)
        MOVP(TRUE,H0,D100);
                (*Execution type*)

②*1            (*Use D100-[control data]*)
        MOVP(TRUE,H8000,D100);
                (*Execution type*)
        MOVP(TRUE,H0,D102);
                (*Application setting*)
        MOVP(TRUE,H1000,D103);
                (*Host station port number*)
        DMOVP(TRUE,H0A6155DF,D104);
                (*Destination IP address*)
        MOVP(TRUE,H2000,D106);
                (*Destination port number*)
    ZP_OPEN(TRUE,"U0",1,D100,Var_Result); (* Opens connection *)

END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    END_IF;
    IF(Var_Result[1]=TRUE)THEN (* Error completion *)
      SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

*1 For divisions of (1) and (2) in the program, (1) is necessary when the [Open settings] of GX Works2 is used and (2) is necessary when it is not used.

5

ZP_CLOSE

Ether



The following instruction can go in the dotted squares.

ZP_CLOSE

■Executing condition

Instruction	Executing condition
ZP_CLOSE	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				—	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction shuts off (closes) a connection with external device during data communication.

Setting data

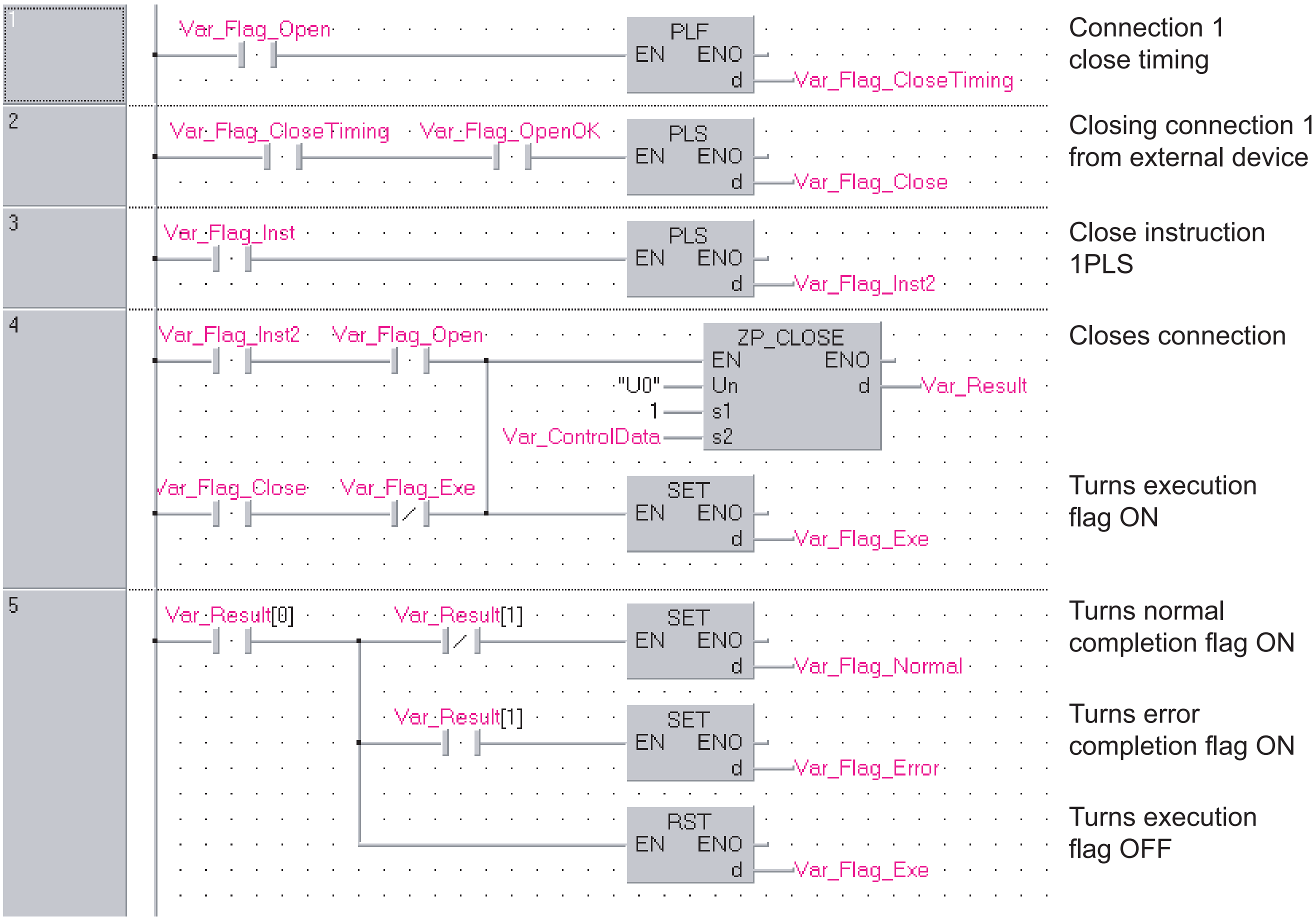
Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

Program example

- The following program closes the connection 1.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



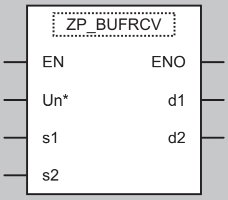
[ST]

```
IF(Var_Flag_Open=TRUE)THEN (* Connection 1 open completion signal *)
    PLF(TRUE,Var_Flag_CloseTiming); (* Connection 1 close timing *)
END_IF;
IF((Var_Flag_CloseTiming=TRUE) AND (Var_Flag_OpenOK=TRUE))THEN (* Connection 1 close timing/open instruction normal completion *)
    PLS(TRUE,Var_Flag_Close); (* Closing connection from external device *)
END_IF;
IF(Var_Flag_Inst=TRUE)THEN (* Close instruction *)
    PLS(TRUE,Var_Flag_Inst2); (* Close instruction 1PLS *)
END_IF;
IF(((Var_Flag_Inst2=TRUE) AND (Var_Flag_Open=TRUE)) (* Close instruction 1PLS/connection 1 open completion signal *)
    OR ((Var_Flag_Close=TRUE) AND (Var_Flag_Exe=FALSE)))THEN (* Closing connection 1 from external device/CLOSE instruction is in execution *)
    ZP_CLOSE(TRUE,"U0",1,Var_ControlData,Var_Result); (* Closes connection *)
    SET(TRUE,Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    END_IF;
    IF(Var_Result[1]=TRUE)THEN (* Error completion *)
      SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
    RST(TRUE,Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

Fixed buffer communication

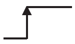
ZP_BUFRCV

Ether

Structured ladder/FBD	ST
	ENO:= ZP_BUFRCV (EN, Un*, s1, s2, d1, d2);

The following instruction can go in the dotted squares.
ZP_BUFRCV

■Executing condition

Instruction	Executing condition
ZP_BUFRCV	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores read data	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

<table>
<tr><th rowspan="2">Setting data*1</th><th colspan="2">Internal device</th><th rowspan="2">R, ZR</th><th colspan="2">J□\□</th><th rowspan="2">U□\G□</th><th rowspan="2">Zn</th><th rowspan="2">Constant K, H</th><th rowspan="2">Others</th></tr>
<tr><th>Bit</th><th>Word</th><th>Bit</th><th>Word</th></tr>
<tr><td>(s1)</td><td>—</td><td colspan="2">○</td><td colspan="4">—</td><td>○</td><td>—</td></tr>
<tr><td>(s2)</td><td>—</td><td colspan="2">○</td><td colspan="4">—</td><td>—</td><td>—</td></tr>
<tr><td>(d1)</td><td>—</td><td colspan="2">○</td><td colspan="4">—</td><td>—</td><td>—</td></tr>
<tr><td>(d2)</td><td>○</td><td colspan="2">○</td><td colspan="4">—</td><td>—</td><td>—</td></tr>
</table>

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads receive data from external device in fixed buffer communication.
This instruction is used in a main program.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

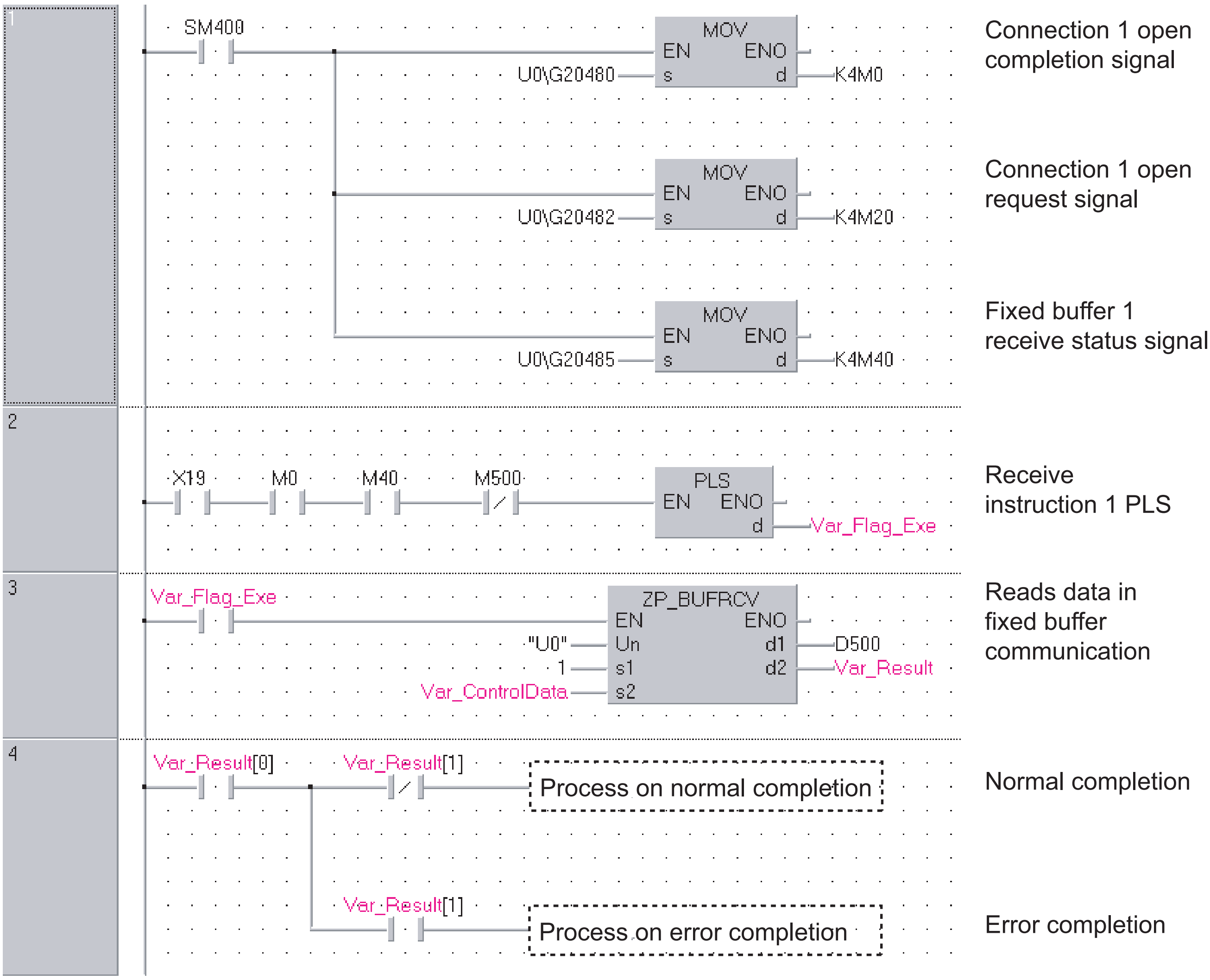
Device	Item	Setting data	Setting range	Setting side
(d1)+0	System area	Data length of the data read from the fixed buffer data area is stored. (Data length becomes the number of words or the number of bytes depending on the procedure used in fixed buffer communication.) • With procedure (communication in binary code): The number of words (1 to 1017) • With procedure (communication in ASCII code): The number of words (1 to 508) • Nonprocedural communication (communication in binary code): The number of bytes (1 to 2046)	(Refer to the left.)	System
(d1)+1 ⋮ (d1)+n	Receive data	Data read from the fixed buffer data area are stored in ascending address order.	—	System

Program example

- The following program reads out receive data from the fixed buffer of the connection 1.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]

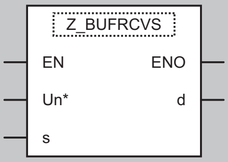


[ST]
```
IF(SM400=TRUE)THEN (* Always ON *)
    MOV(TRUE,U0\G20480,K4M0); (* Open completion signal/connection 1 open completion signal *)
    MOV(TRUE,U0\G20482,K4M20); (* Open request signal/connection 1 open request signal *)
    MOV(TRUE,U0\G20485,K4M40); (* Fixed buffer receive status signal/fixed buffer 1 receive status signal *)
END_IF;
(* Program to receive fixed buffer number 1 (main program) *)
IF((X19=TRUE) AND (M0=TRUE) AND (M40=TRUE) AND (M500=FALSE))THEN (* Initialization normal completion signal/connection 1 normal open completion signal *)
    (* Fixed buffer 1 receive status signal/receive instruction completion signal *)
    PLS(TRUE,Var_Flag_Exe); (* Receive instruction 1PLS *)
END_IF;
IF(Var_Flag_Exe=TRUE)THEN (* Receive instruction 1PLS *)
    ZP_BUFRCV(TRUE,"U0",1,Var_ControlData,D500,Var_Result); (* Reads data in fixed buffer communication *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;
```

5

Z_BUFRCVS

Ether

Structured ladder/FBD	ST
	ENO:= Z_BUFRCVS (EN, Un*, s, d);

The following instruction can go in the dotted squares.

Z_BUFRCVS

■Executing condition

Instruction	Executing condition
Z_BUFRCVS	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Connection number (1 to 16)	ANY16
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores read data	ANY16

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s)	—	○		—				○	—
(d)	—	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads receive data from external device in fixed buffer communication.

This instruction is used in an interrupt program.

Setting data

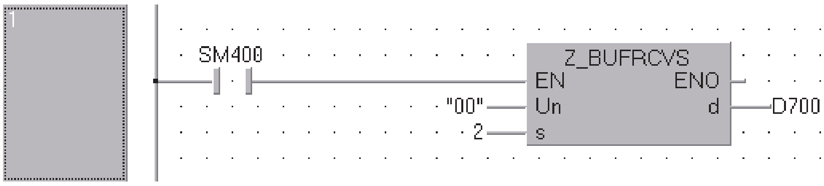
Device	Item	Setting data	Setting range	Setting side
(d1)+0	Receive data length	Data length of the data read from the fixed buffer data area is stored. (Data length becomes the number of words or the number of bytes depending on the procedure used in fixed buffer communication.) • With procedure (communication in binary code): The number of words (1 to 1017) • With procedure (communication in ASCII code): The number of words (1 to 508) • Nonprocedural communication (communication in binary code): The number of bytes (1 to 2046)	(Refer to the left.)	System
(d1)+1 ⋮ (d1)+n	Receive data	Data read from the fixed buffer data area are stored in ascending address order.	—	System

Program example

• The following program reads receive data from the fixed buffer of the connection 2.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



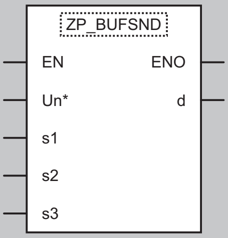
Performs readout in fixed buffer communication

[ST]

Z_BUFRCVS(SM400,"00",2,D700); (* Reads data in fixed buffer communication *)

ZP_BUFSND

Ether

Structured ladder/FBD	ST
	ENO:=ZP_BUFSND(EN, Un*, s1, s2, s3, d);

The following instruction can go in the dotted squares.

ZP_BUFSND

■Executing condition

Instruction	Executing condition
ZP_BUFSND	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
	s3	Start number of the device that stores write data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—				○	—
(s2)	—	○		—				—	—
(s3)	—	○		—				—	—
(d)	○	○		—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data to external device in fixed buffer communication.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

• Send data

Device	Item	Setting data	Setting range	Setting side
(s3)+0	Send data length	Data length of the data read from the fixed buffer data area is stored. (Data length becomes the number of words or the number of bytes depending on the procedure used in fixed buffer communication.) • With procedure (communication in binary code): The number of words (1 to 1017) • With procedure (communication in ASCII code): The number of words (1 to 508) • Nonprocedural communication (communication in binary code): The number of bytes (1 to 2046)	(Refer to the left.)	User
(s3)+1 ⋮ (s3)+n	Send data	Specify the send data.	—	User

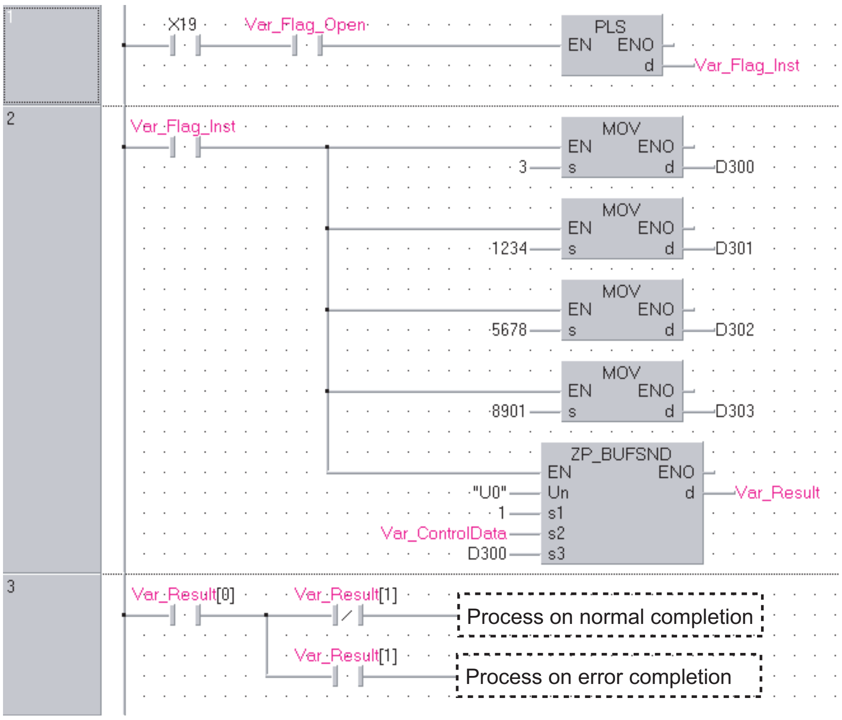
5

Program example

• The following program sends data from the fixed buffer of the connection 1.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



Send instruction 1PLS

Sets data length (number of words)

Sets send data

Sends data in fixed buffer communication

Normal completion

Error completion

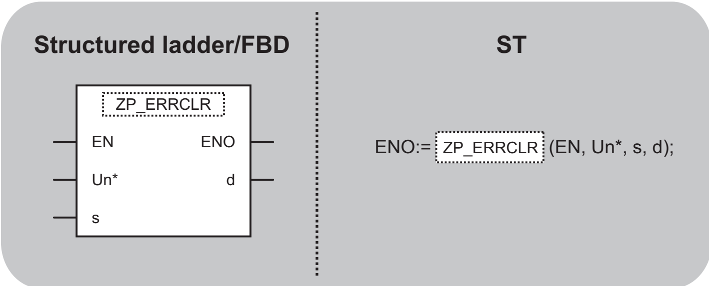
[ST]

```
IF((X19=TRUE) AND (Var_Flag_Open=TRUE))THEN (* Initialization normal completion signal/connection 1 open completion signal*)
    PLS(TRUE,Var_Flag_Inst); (* Send instruction 1PLS *)
END_IF;
IF(Var_Flag_Inst=TRUE)THEN (* Send instruction 1PLS *)
    MOV(TRUE,3,D300); (* Sets data length (number of words) *)
    MOV(TRUE,1234,D301); (* Sets send data *)
    MOV(TRUE,5678,D302); (* Sets send data *)
    MOV(TRUE,8901,D303); (* Sets send data *)
    ZP_BUFSND(TRUE,"U0",1,Var_ControlData,D300,Var_Result); (* Sends data in fixed buffer communication *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;
```

Reading or clearing error information

ZP_ERRCLR

Ether



The following instruction can go in the dotted squares.

ZP_ERRCLR

■Executing condition

Instruction	Executing condition
ZP_ERRCLR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..7]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction turns OFF the LED on Ethernet module and clears error information stored in the buffer memory.

5

Setting data

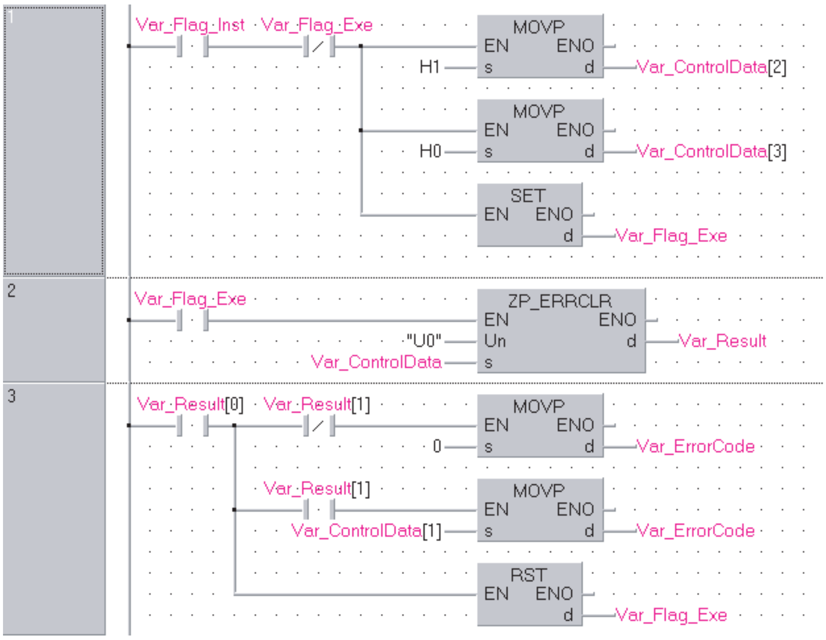
Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Clear target specification	Specify the error information to be cleared. 0000H: Initial error code 0001H to 0010H: Open error code of the corresponding connection 0100H: Error log block area 0101H: Communication status - Status by protocol 0102H: Communication status - E-mail reception status 0103H: Communication status - E-mail transmission status FFFFH: Clears all of the above.	(Refer to the left.)	User
(s)[3]	Clear function specification	Specify the function to be cleared. 0000H: [COM.ERR] LED is turned OFF and an error code is cleared. FFFFH: Error log clear	0000H, FFFFH	User
(s)[4] ⋮ (s)[7]	System area	—	—	—

Program example

- The following program clears the open error code of the connection 1.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



Sets clear target

Sets clear function

Turns execution flag ON

Clears error information

Normal completion Clears error code to 0

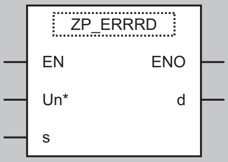
Error completion Stores error code

Turns execution flag OFF

```
[ST]
IF((Var_Flag_Inst=TRUE) AND (Var_Flag_Exe=FALSE))THEN
    MOVP(TRUE,H1,Var_ControlData[2]); (* Sets clear target *)
    MOVP(TRUE,H0,Var_ControlData[3]); (* Sets clear function *)
    SET(TRUE,Var_Flag_Exe); (* Turns execution flag ON *)
END_IF;
IF(Var_Flag_Exe=TRUE)THEN
    ZP_ERRCLR(TRUE,"U0",Var_ControlData,Var_Result); (* Clears error information *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        MOVP(TRUE,0,Var_ErrorCode); (* Clears error code to 0 *)
    END_IF;
    IF(Var_Result[1]=TRUE)THEN (* Error completion *)
        MOVP(TRUE,Var_ControlData[1],Var_ErrorCode);(* Stores error code *)
    END_IF;
    RST(TRUE,Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

ZP_ERRRD

Ether

Structured ladder/FBD	ST
	ENO:= ZP_ERRRD (EN, Un*, s, d);

The following instruction can go in the dotted squares.

ZP_ERRRD

■Executing condition

Instruction	Executing condition
ZP_ERRRD	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..7]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads the error information stored in the buffer memory of the Ethernet module.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0 : Error completion (error code)	—	System
(s)[2]	Read information specification	Specify the error information to be read. 0: Initial error code 1 to 16: Open error code of the corresponding connection	0, 1 to 16	User
(s)[3]	Read target information specification	Specify the target error information to be read. 0000H: Latest error information	0000H	User

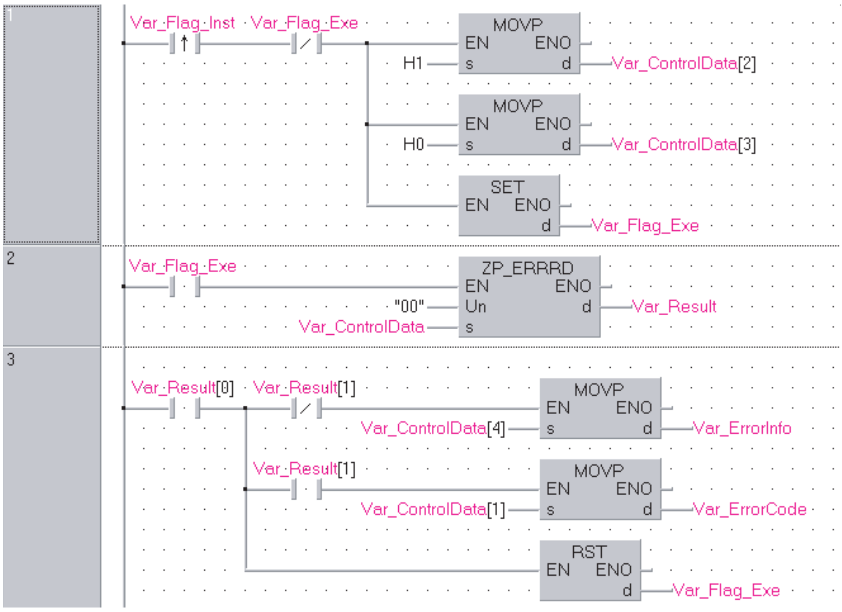
Device	Item	Setting data	Setting range	Setting side
(s)[4]	Error information	The read error information is stored. 0000H: No error Other than 0000H: Error code	—	System
(s)[5] ⋮ (s)[7]	System area	—	—	—

Program example

• The following program reads the open error code of the connection 1.

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



Sets open error code

Sets latest error information

Turns execution flag ON

Reads error information

Normal completion Stores error information

Error completion Stores error code

Turns execution flag OFF

[ST]

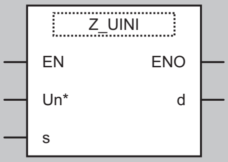
```
IF((Var_Flag_Inst=TRUE) AND (Var_Flag_Exe=FALSE))THEN
    MOVP(TRUE,H1,Var_ControlData[2]); (* Sets open error code of connection number 1 *)
    MOVP(TRUE,H0,Var_ControlData[3]); (* Sets latest error information *)
    SET(TRUE,Var_Flag_Exe); (* Turns execution flag ON*)
END_IF;
IF(Var_Flag_Exe=TRUE)THEN
    ZP_ERRRD(TRUE,"00",Var_ControlData,Var_Result); (* Reads error information *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        MOVP(TRUE,Var_ControlData[4],Var_ErrorInfo); (* Stores error information*)
    END_IF;
    IF(Var_Result[1]=TRUE)THEN (* Error completion *)
        MOVP(TRUE,Var_ControlData[1],Var_ErrorCode); (* Stores error code *)
    END_IF;
    RST(TRUE,Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

UINI instruction

Z(P)_UINI

CC IE C Ether*1

*1 ZP_UINI instruction only

Structured ladder/FBD	ST
	ENO:= Z_UINI (EN, Un*, s, d);

The following instruction can go in the dotted squares.
Z_UINI, ZP_UINI

■Executing condition

Instruction	Executing condition
Z_UINI	
ZP_UINI	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..9]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

Ethernet: This instruction reinitializes the Ethernet module.
CC-Link IE Controller Network: For Universal model QCPU, this instruction sets the station number of the CC-Link IE Controller Network module on normal station (host station).

Setting data

• Ethernet

Device	Item	Setting data	Setting range	Setting side
(s)[0]	System area	—	—	—
(s)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Modification specification	[When updating the address information of external devices which are held by the Ethernet module] • Specify '0H'.*1 [When modifying the host station IP address, operation settings, transmission speed, communication mode] • Specify the parameter to be modified. However, Modification specification of transmission speed, communication mode cannot be executed simultaneously with that of host station IP address, operation settings. If executed, only modification specification of host station IP address and operation settings will be set. b15 ··· b12 ··· b1 b0 (3) 0 (2) (1) (1) Modification specification of host station IP address (b0) Specify whether the host station IP address is modified or not. (To modify the IP address, specify the address in (s)[3] and (s)[4].) 0: Not changed 1: Changed (2) Modification specification of operation setting (b1) Specify whether the operation setting is modified or not. (To modify the operation setting, specify the operation setting in (s)[5].) 0: Not changed 1: Changed (3) Modification specification of transmission speed, communication mode (b15 to b12) Specify the transmission speed, communication mode. 0: not changed 1: automatic negotiation 2: 100Mbps/full-duplex communication 3: 100Mbps/half-duplex communication 4: 10Mbps/full-duplex communication 5: 10Mbps/half-duplex communication	(Refer to the left.)	User
(s)[3] (s)[4]	Host station IP address	Specify the IP address of the host station.	00000001H to FFFFFFFEH	User
(s)[5]	Operation setting	b15 ··· b8 ··· b6 b5 b4 ··· b1 b0 0 (5) 0 (4) (3) (2) 0 (1) 0 (1) Communication data code setting (b1) 0: Binary code 1: ASCII code (2) TCP existence confirmation setting (b4) 0: Use Ping 1: Use KeepAlive (3) Transmission frame setting (b5) 0: Ethernet frame 1: IEEE802.3 frame (4) Setting for enabling/disabling write during RUN (b6) 0: Disable 1: Enable (5) Initial timing setting (b8) 0: Do not wait for OPEN (communication impossible during STOP status) 1: Always wait for OPEN (communication possible during STOP status)	(Refer to the left.)	User
(s)[6] ⋮ (s)[9]	—	Specify 0.	0	User

*1 The Ethernet module enables data exchange to restart by clearing the address information retained in the module and by performing re-initial processing. (The Initial normal completion signal (X19) is on.)

• CC-link IE Controller Network

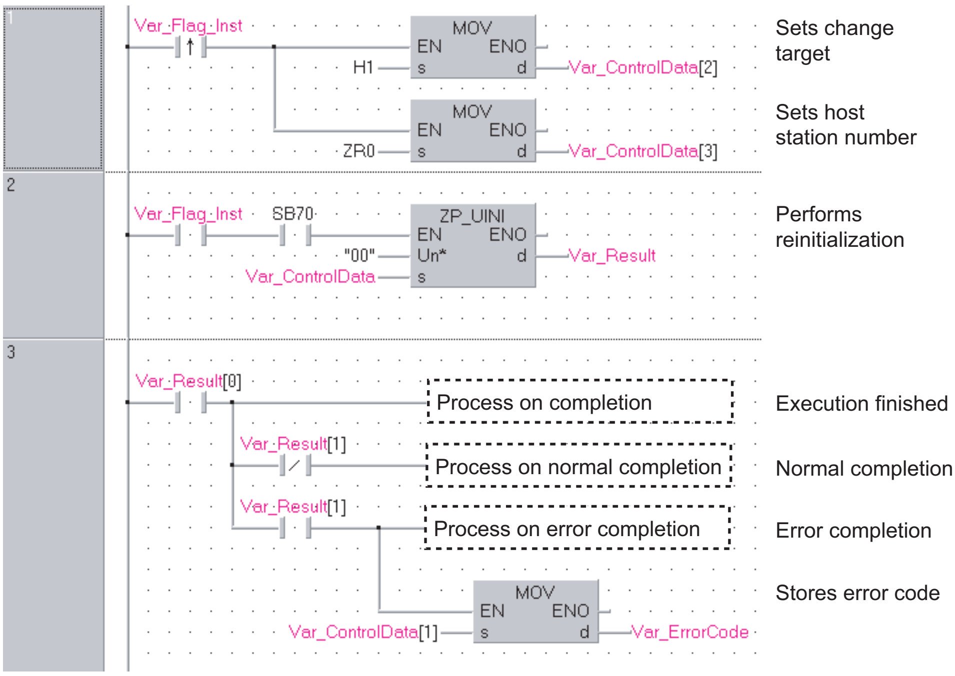
Device	Item	Setting data	Setting range	Setting side
(s)[0]	—	Specify 0.	0	User
(s)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	Modification specification	Specify the change target 0001H: With station number setting	0001H	User
(s)[3]	Host station No.	Specify the station number of the host station.	1 to 120	User
(s)[4] ⋮ (s)[9]	—	Specify 0.	0	User

Point

The UINI instruction can be executed only once. The UINI instruction cannot be executed again after determination of station number. (It caused an error completion.)
However, in the case of the UINI instruction with the error completion, execute the UINI instruction again after taking corrective action.

Program example

• The following program sets the station number 2. The following is an example for Ethernet.
[Structured ladder/FBD]



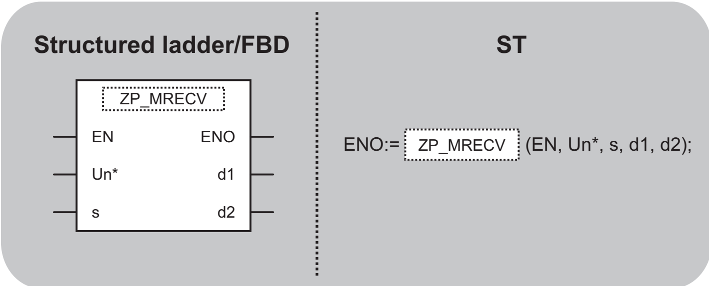
```
[ST]
IF(Var_Flag_Inst=TRUE)THEN
    MOV(TRUE,H1,Var_ControlData[2]); (* Sets change target *)
    MOV(TRUE,ZR0,Var_ControlData[3]); (* Sets host station number *)
END_IF;
IF((Var_Flag_Exe=TRUE) AND (SB70=TRUE))THEN
    ZP_UINI(TRUE,"00",Var_ControlData,Var_Result); (* Performs reinitialization *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
(* Process on completion *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
        MOV(TRUE, Var_ControlData[1], Var_ErrorCode); (* Stores error code *)
    END_IF;
END_IF;
```

5

E-mail communication

ZP_MRECV

Ether



The following instruction can go in the dotted squares.

ZP_MRECV

■Executing condition

Instruction	Executing condition
ZP_MRECV	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s	Variable that stores control data	Array of ANY16 [0..15]
Output argument	ENO	Execution result	Bit
	d1	Start number of the host station's device that stores the content of the received e-mail (header + attached file)	ANY16
	d2	Variable that turns ON upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data[*1]	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					
(d1)	—	○		—					
(d2)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads received e-mail.

Setting data

Device	Item		Setting data	Setting range	Setting side
(s)[0]	Execution/Error completion type		b15 … b9 b8 b7 … b0 0 | (2) | 0 | (1) | 0 (1) Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s)[11]. 1: Clock data at the time of error completion is set in the area starting from (s)[11]. (2) Execution type (bit 9)*1 Specify whether to inquire about existence of mails in the server after reading received mails. 0: Not requested (not read) 1: Requested (read)	0000H, 0080H, 0200H, 0280H	User
(s)[1]	Completion status		The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s)[2]	E-mail No. to be read		Specify the number of a mail to be read when multiple mails are received. 0: First mail 1 or more: Specified mail	0 or more	User
(s)[3] ⋮ (s)[8]	System area		—	—	—
(s)[9]	Receive data length	For instruction execution	Specify the data length (header + attached file) of the mail that can be stored in (d1) to (d1)+n. (Header: 1 to 373, attached file: 1 to 6144) 0: Adjust data length to that of the received mail. 1 to 6517: The number of data that can be stored in ((d1) to (d1)+n)	0 to 6517 (word) * Includes the header length explained below.	User
		At instruction completion	Data length (header + attached file) of the mail stored in (d1) to (d1)+n is stored. 1 to 6517: The number of receive data stored in ((d1) to (d1)+n)		System
(s)[10]	Header length	For instruction execution	Specify the header data length of the mail that can be stored in (d1) to (d1)+n. 0: Adjust header data length to that of the received mail. 1 to 373: The number of data that can be stored in ((d1) to (d1)+n)	0 to 373 (word)	User
		At instruction completion	Header data length of the mail stored in (d1) to (d1)+n is stored. 1 to 373: Number of receive data stored in ((d1) to (d1)+n)		System
(s)[11]	Clock set flag		Valid/invalid status of the data in the area starting from (s)[12] is stored. 0: Invalid 1: Valid	0, 1	System
(s)[12] ⋮ (s)[15]	Clock data (set only when errors occur)		Clock data at the time of error completion are stored in BCD format. b15 to b8 | b7 to b0 s [12] Month (01H to 12H) | Year (00H to 99H) Last two digits s [13] Hour (00H to 23H) | Day (01H to 31H) s [14] Second (00H to 59H) | Minute (00H to 59H) s [15] Year (00H to 99H) First two digits | Day of week (00H to 06H) 00H (Sun.) to 06H (Sat.)	—	System
(d1)+0 ⋮ (d1)+n	Receive data		Content (header + attached file) of the received mail are stored.	—	System

*1 The following table shows the processing that depends on the selection of the execution type after executing the MRECV instruction.

Setting option	Processing	Advantage	Disadvantage
No request (not read)	• Only e-mail read processing from the mail server is performed. • Inquiry (reading) for the information of received mails remaining in the mail server is performed after the time set in the GX Works2 parameter has elapsed.	Unnecessary read processing is not performed when the mail server has no mail.	Even if mails remain in the mail server, they cannot be read immediately. Mails tend to be accumulated in the mail server.

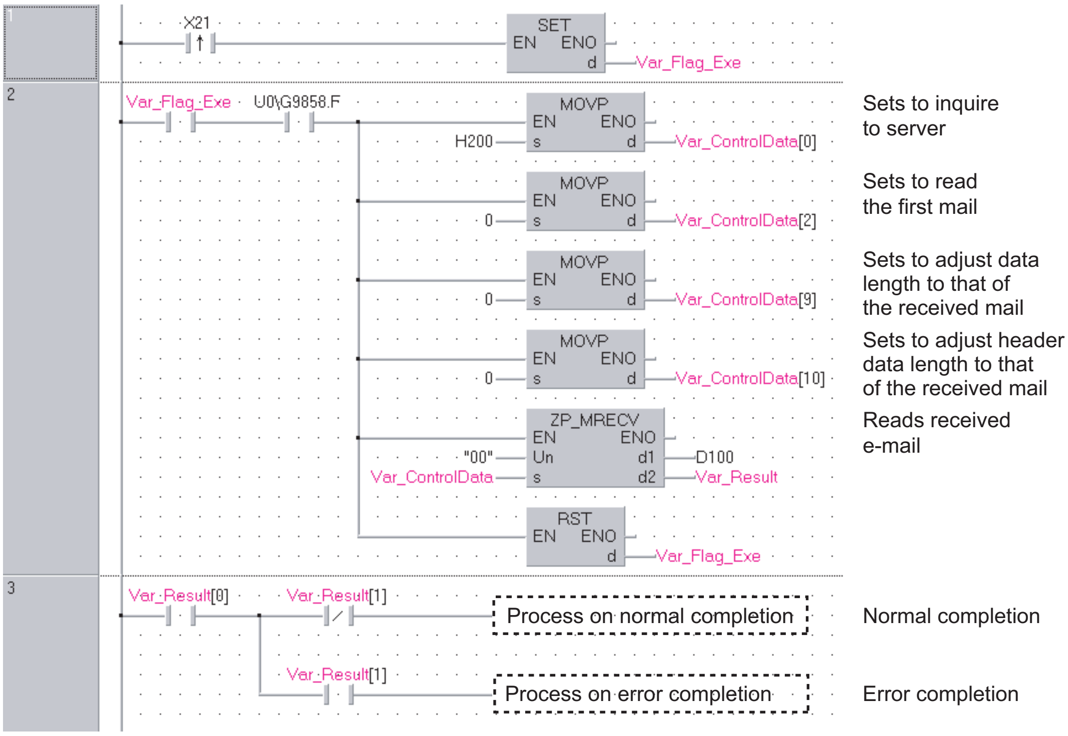
Setting option	Processing	Advantage	Disadvantage
Request (read)	• E-mail read processing from the mail server is performed. • After the execution of the MRECV instruction, inquiry (read) processing for information on the received mails remaining in the mail server is performed. (Inquiry for receiving of a mail is made immediately.)	Received mails stored in the mail server can be read in series.	Inquiries to the mail server are made more often. Internal processing of the module increases, which affects other internal processing to a certain degree.

Program example

• The following program performs the e-mail receiving process by the receive instruction (X21).

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

[Structured ladder/FBD]



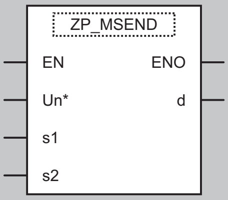
[ST]

```
IF (X21=TRUE) THEN
    SET(TRUE,Var_Flag_Exe);
END_IF;
IF((Var_Flag_Exe=TRUE) AND (U0\G9858.F=TRUE))THEN
    MOVP(TRUE,H200,Var_ControlData[0]); (* Sets to inquire to server *)
    MOVP(TRUE,0,Var_ControlData[2]); (* Sets to read the first mail *)
    MOVP(TRUE,0,Var_ControlData[9]); (* Sets to adjust data length to that of the received mail *)
    MOVP(TRUE,0,Var_ControlData[10]); (* Sets to adjust header data length to that of the received mail *)
    ZP_MRECV(TRUE,"00",Var_ControlData,D100,Var_Result); (* Reads received e-mail *)
    RST(TRUE,Var_Flag_Exe);
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;
```

ZP_MSEND

Ether

Structured ladder/FBD



ST

ENO:= ZP_MSEND (EN, Un*, s1, s2, d);

The following instruction can go in the dotted squares.

ZP_MSEND

■Executing condition

Instruction	Executing condition
ZP_MSEND	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un*	Start I/O number of the module (00 to FE: Higher two digits when expressing the I/O number in three digits)	String
	s1	Variable that stores control data	Array of ANY16 [0..15]
	s2	Start number of the host station's device that stores the content of the sent e-mail (subject + attached file) or (subject + text)	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data*1	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	—	○		—					
(d)	○	○		—					

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends an e-mail.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0]	Execution/Error completion type Send data format	b15 … b12 … b8 b7 … b0 0 | (2) | (1) | 0 (1) Error completion type (bit 7) Specify the clock data setup status at the time of error completion. 0: Clock data at the time of error completion is not set in the area starting from (s1)[11]. 1: Clock data at the time of error completion is set in the area starting from (s1)[11]. (2) Send data format (bit 12 to bit 8) Specify the data format of the send data. (Sending the data as an attached file) • Binary data • ASCII data (converted from binary into ASCII) • CSV data (converted from binary into CSV) (Sending the data as a text) • Binary data [Precautions for specifying a text] • When a text is specified, setting at bit 11 to bit 8 is invalid. • Specify the text in ASCII characters in a sequence program. (Ethernet module does not convert text into ASCII characters.) • The following binary code data are treated as control codes. 0D0AH: Line feed code, CR+LF 00H: End of the text • The number of characters per line in a text to 78 characters or less (Enter the line feed code, CR+LF (0D0AH), at the last line of a text.)	(Refer to the left.)	User
(s1)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s1)[2]	Transmission destination No.	Specify the external device to which e-mails are to be sent by the setting number on [Send mail address setting] of GX Works2. 1 to 16: Setting number of the external device	1 to 16	User
(s1)[3] ⋮ (s1)[8]	System area	—	—	—
(s1)[9]	Send data length	Specify the data length ((subject + attached file) or (subject + text)) of the mail stored in (s2) to (s2)+n. • Sending the data as an attached file (subject: 0 to 373, attached file: 1 to 6144) 1 to 6517: Data length (word) of a mail • Sending the data as a text (subject: 0 to 373, text: 1 to 960) 1 to 1333: Data length (word) of a mail	0 to 6517 or 1 to 1333	User
(s1)[10]	Subject length	Specify the subject data length of the mail stored in (s2) to (s2)+n. 0 to 373: Data length (word) of subject	0 to 373	User
(s1)[11]	Clock set flag	Valid/invalid status of the data in the area starting from (s1)[12] is stored. 0: Invalid 1: Valid	—	System
(s1)[12] ⋮ (s1)[15]	Clock data (set only when errors occur)	Clock data at the time of error completion are stored in BCD format. b15 to b8 | b7 to b0 s1 [12] Month (01H to 12H) | Year (00H to 99H) Last two digits s1 [13] Hour (00H to 23H) | Day (01H to 31H) s1 [14] Second (00H to 59H) | Minute (00H to 59H) s1 [15] Year (00H to 99H) First two digits | Day of week (00H to 06H) 00v (Sun.) to 06H (Sat.)	—	System

• Send data

Device	Item	Setting data	Setting range	Setting side
(s2)+0 ⋮ (s2)+n	Send data	Specify the content of ((subject + attached file) or (Subject + text)) of a mail to be sent.	—	User

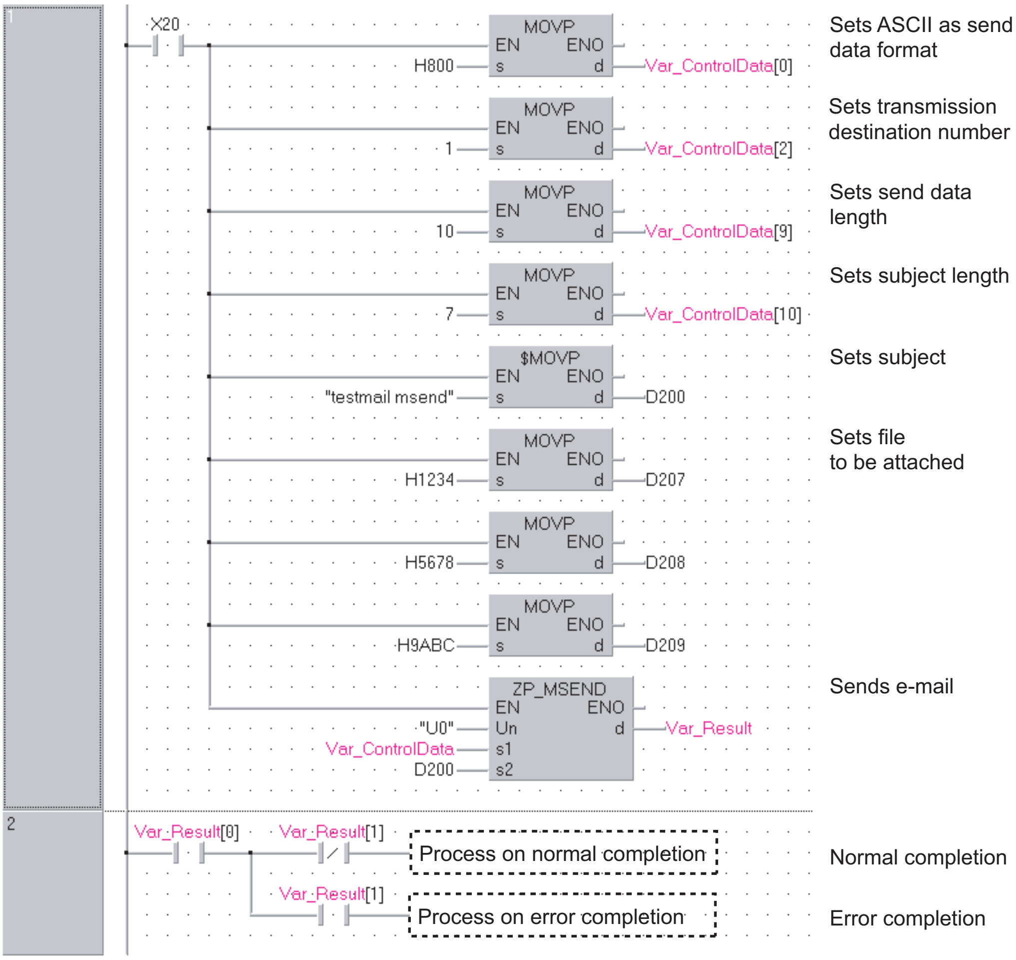
Program example

- The following program performs e-mail sending process by the send instruction (X20).

(The I/O signals of the Ethernet module are X/Y00 to X/Y1F)

- Sending the data as an attached file

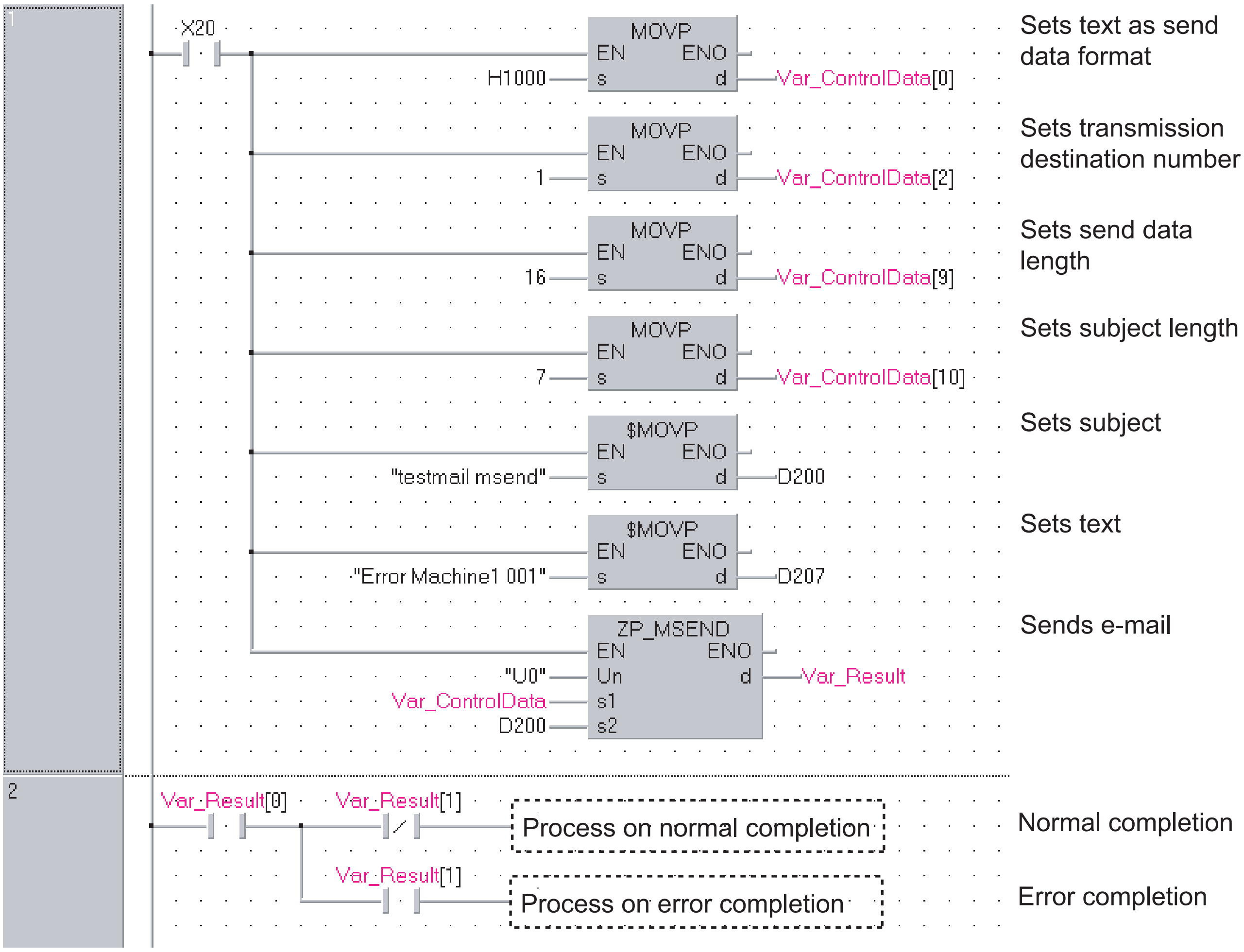
[Structured ladder/FBD]



```
[ST]
IF(X20=TRUE)THEN
    MOVP(TRUE,H800,Var_ControlData[0]); (* Sets ASCII as send data format *)
    MOVP(TRUE,1,Var_ControlData[2]); (* Sets transmission destination number *)
    MOVP(TRUE,10,Var_ControlData[9]); (* Sets send data length *)
    MOVP(TRUE,7,Var_ControlData[10]); (* Sets subject length *)

    Int_Msg[0] := H6574; (* te *)
    Int_Msg[1] := H7473; (* st *)
    Int_Msg[2] := H616d; (* ma *)
    Int_Msg[3] := H6c69; (* il *)
    Int_Msg[4] := H6d20; (* m *)
    Int_Msg[5] := H6573; (* se *)
    Int_Msg[6] := H646e; (* nd *) (* Sets subject *)

    MOVP(TRUE,H1234,Int_Msg[7]); (* Sets file to be attached *)
    MOVP(TRUE,H5678,Int_Msg[8]);
    MOVP(TRUE,H9ABC,Int_Msg[9]);
    ZP_MSEND(TRUE,"U0",Var_ControlData,Int_Msg[0],Var_Result); (* Sends e-mail *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)

(* Process on normal completion *)

    ELSE (* Error completion *)

(* Process on error completion *)

    END_IF;
END_IF;
```

• Sending the data as a text

[Structured ladder/FBD]



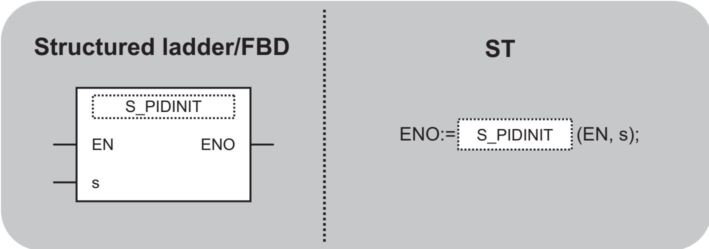
```
[ST]
IF(X20=TRUE)THEN
    MOVP(TRUE,H1000,Var_ControlData[0]); (* Sets text as send data format *)
    MOVP(TRUE,1,Var_ControlData[2]); (* Sets transmission destination number *)
    MOVP(TRUE,16,Var_ControlData[9]); (* Sets send data length *)
    MOVP(TRUE,7,Var_ControlData[10]); (* Sets subject length *)

    Int_Msg[0] := H6574; (* te *)
    Int_Msg[1] := H7473; (* st *)
    Int_Msg[2] := H616d; (* ma *)
    Int_Msg[3] := H6c69; (* il *)
    Int_Msg[4] := H6d20; (* m *)
    Int_Msg[5] := H6573; (* se *)
    Int_Msg[6] := H646e; (* nd *) (* Sets subject *)

    Int_Msg[7] := H7274; (* Er *)
    Int_Msg[8] := H6f72; (* ro *)
    Int_Msg[9] := H2072; (* r *)
    Int_Msg[10] := H614d; (* Ma *)
    Int_Msg[11] := H6863; (* ch *)
    Int_Msg[12] := H6e69; (* in *)
    Int_Msg[13] := H3165; (* e1 *)
    Int_Msg[14] := H3020; (* 0 *)
    Int_Msg[15] := H3130; (* 01 *) (* Sets text *)

    ZP_MSEND(TRUE,"U0",Var_ControlData,Int_Msg[0],Var_Result); (* Sends e-mail *)
END_IF;
IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
(* Process on normal completion *)
    ELSE (* Error completion *)
(* Process on error completion *)
    END_IF;
END_IF;
```

6 PID CONTROL INSTRUCTION

6.1 PID Control Instruction (Inexact Differential)

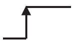
Data setting

S(P)_PIDINIT



The following instruction can go in the dotted squares.
S_PIDINIT, SP_PIDINIT

■Executing condition

Instruction	Executing condition
S_PIDINIT	
SP_PIDINIT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device that stores PID control data	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

This instruction enables PID control by registering the PID control data for the number of loops to be used to the CPU module in batch.

Setting data

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
Common setting data (device: (s)+0 to (s)+1)						
(s)+0	Number of loops	Set the number of loops for PID operation.	1 to 32		User	An error occurs and the PID operation for all loops is not performed.
(s)+1	Number of loops in one scan	Set the number of loops for PID operation in one scan if multiple loops have reached the sampling cycle time.	1 to 32		User	
Setting data for No. 1 loop (device: (s)+2 to (s)+15)						
(s)+2	Operational expression selection	Select the PID operational expression.*1	0: Forward operation 1: Reverse operation	0: Forward operation 1: Reverse operation	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+3	Sampling cycle (T_S)	Set the PID operation cycle.	1 to 6000 (unit: 10ms)	1 to 6000 (unit: 10ms)	User	
(s)+4	Proportional constant (K_P)	Proportional gain of PID operation	1 to 10000 (unit: 0.01)	1 to 10000 (unit: 0.01)	User	
(s)+5	Integral constant (T_I)	Constant that expresses the magnitude of the integral action (I action) effect. Increasing the integral constant slows down the manipulated value change.	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	User	
(s)+6	Derivative constant (T_D)	Constant that expresses the magnitude of the derivative action (D action) effect. Increasing the derivative constant causes a significant change in the manipulated value even with a slight change of the control target.	0 to 30000 (unit: 10ms)	0 to 30000 (unit: 10ms)	User	
(s)+7	Filter coefficient (α)	Set the degree of filtering to be applied to the process value. The filtering effect decreases as the value gets closer to 0.	0 to 100	0 to 100	User	
(s)+8	MV lower limit (MVLL)	Set the lower limit for the manipulated value (MV) calculated in PID operation in automatic mode. If the MV is less than the set lower limit value (MVLL), the value is clipped to the MVLL.	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the MVLL or MVHL is less than -50, the value is clipped to -50. • If the MVLL or MVHL is greater than 2050, the value is clipped to 2050.
(s)+9	MV upper limit (MVHL)	Set the upper limit for the manipulated value calculated in PID operation in automatic mode. If the MV is greater than the set upper limit value (MVHL), the value is clipped to the MVHL.	-50 to 2050	-32768 to 32767	User	

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+10	MV change rate limit (ΔMVL)	Set the variation limit between the previous MV and the present MV. When the MV variation is greater than the limit value, bit 1 (b1) of the alarm device is set to '1'. MV variation is not limited. (Even if the MV variation exceeds the limit value, the actual MV variation is used as it is for calculating the MV.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the ΔMVL value is less than 0, the value is clipped to 0. • If the ΔMVL value is greater than 2000, the value is clipped to 2000.
(s)+11	PV change rate limit (ΔPVL)	Set the variation limit between the previous PV and the present PV. When the PV variation is greater than the limit value, bit 0 (b0) of the alarm device is set to '1'. PV variation is not limited. (Even if the PV variation exceeds the limit value, the actual PV variation is used as it is for performing the PID operation.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the ΔPVL value is less than 0, the value is clipped to 0. • If the ΔPVL value is greater than 2000, the value is clipped to 2000.
(s)+12	(Fixed value)	—	0	0	User	—
(s)+13	Derivative gain (K_D)	Set a duration (delay in action) for derivative action. As the setting value increases, the duration becomes smaller and action becomes closer to exact differential. Ideal value K_D = 8.00	0 to 32767 (unit: 0.01) If setting value > 30000 K_D = Infinite (∞)	0 to 32767 (unit: 0.01) If setting value > 30000 K_D = Infinite (∞)	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+14	(Fixed value)	—	0	0	User	—
(s)+15	(Fixed value)	—	0	0	User	—
Setting data for No. 2 loop (device: (s)+16 to (s)+29)						

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+16	Operational expression selection	The same as Setting data for No. 1 loop				
(s)+17	Sampling cycle (T_S)					
(s)+18	Proportional constant (K_P)					
(s)+19	Integral constant (T_I)					
(s)+20	Derivative constant (T_D)					
(s)+21	Filter coefficient (α)					
(s)+22	MV lower limit (MVLL)					
(s)+23	MV upper limit (MVHL)					
(s)+24	MV change rate limit (ΔMVL)					
(s)+25	PV change rate limit (ΔPVL)					
(s)+26	(Fixed value)					
(s)+27	Derivative gain (K_D)					
(s)+28	(Fixed value)					
(s)+29	(Fixed value)					
Setting data for No. n loop						
(s)+(m+0)	Operational expression selection	The same as Setting data for No. 1 loop m=(n-1)×14+2 n: number of loops				
(s)+(m+1)	Sampling cycle (T_S)					
(s)+(m+2)	Proportional constant (K_P)					
(s)+(m+3)	Integral constant (T_I)					
(s)+(m+4)	Derivative constant (T_D)					
(s)+(m+5)	Filter coefficient (α)					
(s)+(m+6)	MV lower limit (MVLL)					
(s)+(m+7)	MV upper limit (MVHL)					
(s)+(m+8)	MV change rate limit (ΔMVL)					
(s)+(m+9)	PV change rate limit (ΔPVL)					
(s)+(m+10)	(Fixed value)					
(s)+(m+11)	Derivative gain (K_D)					
(s)+(m+12)	(Fixed value)					
(s)+(m+13)	(Fixed value)					

*1 For the PID operational expression set in the operational expression selection, refer to the following.
MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)

Precautions

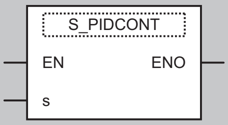
The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

○: Applicable, ×: Not applicable

PID operation

S(P)_PIDCONT

Structured ladder/FBD	ST
	ENO:= S_PIDCONT (EN, s);

The following instruction can go in the dotted squares.
S_PIDCONT, SP_PIDCONT

■Executing condition

Instruction	Executing condition
S_PIDCONT	
SP_PIDCONT	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device that is assigned in I/O data area	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

- This instruction measures sampling cycle and performs PID operation at instruction execution.
- This instruction performs PID operation based on the set value (SV) and process value (PV) in the I/O data area set to the device number specified by (s) or later, and stores the operation result to the automatic manipulated value (MV) area in the I/O data area.
- PID operation is performed in response to the first execution of the PIDCONT instruction after the set sampling cycle time has elapsed.

Setting data

Device	Data name		Description	Setting range		Setting side	Processing when the setting data are outside the setting range
				With PID limits	Without PID limits		
(s)+0	Initial processing flag		Processing method at the start of PID operation	0: PID operation for the number of loops to be used is batch-processed in one scan. Other than 0: PID operation for the number of loops to be used is processed in several scans.		User	—
(s)+1 ⋮ (s)+9	PID control work area (reserved by the system)			—	—	—	—
I/O data area for No. 1 loop (device: (s)+10 to (s)+27)							
(s)+10	Setting value	SV	• PID control target value	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If SV is less than 0, the value is clipped to 0. • If SV is greater than 2000, the value is clipped to 2000.
(s)+11	Process value	PV	• Feedback data from the control target to the A/D conversion module	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If PV is less than -50, the value is clipped to -50. • If PV is greater than 2050, the value is clipped to 2050.
(s)+12	Automatic manipulated value	MV	• Manipulated value obtained by PID operation • The value is output from the D/A conversion module to the control target.	-50 to 2050	-32768 to 32767	System	—
(s)+13	Process value after filtering	PVf	• Process value obtained by calculation using operational expression.[*1]	-50 to 2050	-32768 to 32767	System	—
(s)+14	Manual manipulated value	MV_{MAN}	• Store the data output from the D/A conversion module in manual operation.	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If MV_{MAN} is less than -50, the value is clipped to -50. • If MV_{MAN} is greater than 2050, the value is clipped to 2050.
(s)+15	Manual/ automatic selection	MAN/ AUTO	• Select whether the output to the D/A conversion module is a manual manipulated value or an automatic manipulated value. • In manual operation, the automatic manipulated value remains unchanged.	0: Automatic manipulated value 1: Manual manipulated value		User	When other than 0 or 1 is selected, an error occurs and the operation for the number corresponding loop is not performed.

Device	Data name		Description	Setting range		Setting side	Processing when the setting data are outside the setting range
				With PID limits	Without PID limits		
(s)+16	Alarm	ALARM	• Used to determine if the change rate of the MV (manipulated value) and the PV (process value) is within or outside the limit value range. • Once set, the alarm data are maintained until the user resets it.	b15 … b1 b0 0 (2) (1) When the MV variation is outside the limit range, bit 1 (b1) is set to '1'. When the PV variation is outside the limit range, bit 0 (b0) is set to '1'.		User System	—
(s)+17 ⋮ (s)+32	PID control work area (reserved by the system)			—		—	—
I/O data area for No. 2 loop (device: (s)+28 to (s)+45)							
(s)+33	Setting value	SV	The same as I/O data area for No. 1 loop				
(s)+34	Process value	PV					
(s)+35	Automatic manipulated value	MV					
(s)+36	Process value after filtering	PVf					
(s)+37	Manual manipulated value	MV_{MAN}					
(s)+38	Manual/ automatic selection	MAN/ AUTO					
(s)+39	Alarm	ALARM					
(s)+40 ⋮ (s)+55	PID control work area (reserved by the system)			—		—	—
I/O data area for No. n loop							
(s)+(m+0)	Setting value	SV	The same as I/O data area for No. 1 loop m=(n-1) × 23+10 n: number of loops				
(s)+(m+1)	Process value	PV					
(s)+(m+2)	Automatic manipulated value	MV					
(s)+(m+3)	Process value after filtering	PVf					
(s)+(m+4)	Manual manipulated value	MV_{MAN}					
(s)+(m+5)	Manual/ automatic selection	MAN/ AUTO					
(s)+(m+6)	Alarm	ALARM					
(s)+(m+7) ⋮ (s)+(m+22)	PID control work area (reserved by the system)			—		—	—

*1 For process value after filtering (PVf), the value calculated based on the process value of input data are stored.
For the operational expression, refer to the following.
MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)

Precautions

The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

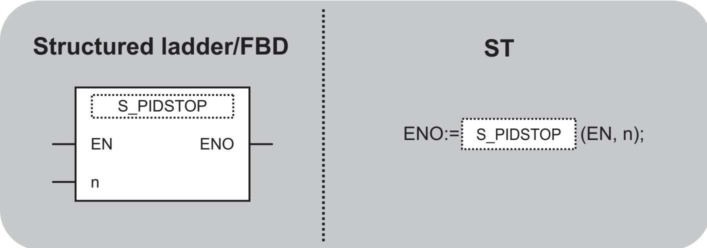
CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

○: Applicable, ×: Not applicable

6

PIDSTOP instruction and PIDRUN instruction

S_PIDSTOP, S_PIDRUN



The following instruction can go in the dotted squares.
S_PIDSTOP, SP_PIDSTOP, S_PIDRUN, SP_PIDRUN

■Executing condition

Instruction	Executing condition
S_PIDSTOP, S_PIDRUN	
SP_PIDSTOP, SP_PIDRUN	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Loop number for stop/start	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(n)	○			—				○	—

Processing details

- S(P)_PIDSTOP

This instruction stops the PID operation for the loop number specified by 'n'.

- S(P)_PIDRUN

This instruction starts the PID operation for the loop number specified by 'n'.

Precautions

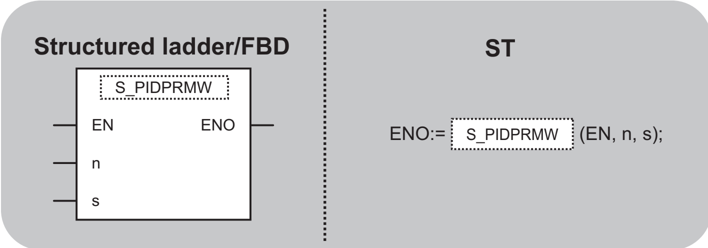
The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

○: Applicable, ×: Not applicable

Operation parameter change

S(P)_PIDPRMW



The following instruction can go in the dotted squares.
S_PIDPRMW, SP_PIDPRMW

■Executing condition

Instruction	Executing condition
S_PIDPRMW	
SP_PIDPRMW	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Loop number to be changed	ANY16
	s	Start number of the device that stores PID control data to be changed	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(n)	○	○		—				○	—
(s)	—	○		—				—	—

Processing details

This instruction changes the operation parameter of the loop number specified by 'n' to the PID control data stored in the devices starting from the device number specified by (s).

6

Setting data

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+0	Operational expression selection	Select the PID operational expression.[*1]	0: Forward operation 1: Reverse operation	0: Forward operation 1: Reverse operation	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+1	Sampling cycle (T_S)	Set the PID operation cycle.	1 to 6000 (unit: 10ms)	1 to 6000 (unit: 10ms)	User	
(s)+2	Proportional constant (K_P)	Proportional gain of PID operation	1 to 10000 (unit: 0.01)	1 to 10000 (unit: 0.01)	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+3	Integral constant (T_I)	Constant that expresses the magnitude of the integral action (I action) effect. Increasing the integral constant slows down the manipulated value change.	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	User	
(s)+4	Derivative constant (T_D)	Constant that expresses the magnitude of the derivative action (D action) effect. Increasing the derivative constant causes significant changes in the manipulated value even with a slight change of the control target.	0 to 30000 (unit: 10ms)	0 to 30000 (unit: 10ms)	User	
(s)+5	Filter coefficient (α)	Set the degree of filtering to be applied to the process value. The filtering effect decreases as the value gets closer to 0.	0 to 100	0 to 100	User	
(s)+6	MV lower limit (MVLL)	Set the lower limit for the manipulated value (MV) calculated in PID operation in automatic mode. If the MV is less than the set lower limit value (MVLL), the value is clipped to the MVLL.	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If MVLL or MVHL value is less than -50, the value is clipped to -50. • If MVLL or MVHL value is greater than 2050, the value is clipped to 2050.
(s)+7	MV upper limit (MVHL)	Set the upper limit for the manipulated value calculated in PID operation in automatic mode. If the MV is greater than the set upper limit value (MVHL), the value is clipped to the MVHL.	-50 to 2050	-32768 to 32767	User	
(s)+8	MV change rate limit (ΔMVL)	Set the variation limit between the previous MV and the present MV. When the MV variation is greater than the limit value, bit 1 (b1) of the alarm device is set to '1'. MV variation is not limited. (Even if the MV variation exceeds the limit value, the actual MV variation is used as it is for calculating the MV.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the ΔMVL value is less than 0, the value is clipped to 0. • If the ΔMVL value is greater than 2000, the value is clipped to 2000.

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+9	PV change rate limit (ΔPVL)	Set the variation limit between the previous PV and the present PV. When the PV variation is greater than the limit value, bit 0 (b0) of the alarm device is set to '1'. PV variation is not limited. (Even if the PV variation exceeds the limit value, the actual PV variation is used as it is for performing the PID operation.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the ΔPVL value is less than 0, the value is clipped to 0. • If the ΔPVL value is greater than 2000, the value is clipped to 2000.
(s)+10	(Fixed value)	—	0	0	User	—
(s)+11	Derivative gain (K_D)	Set a duration (delay in action) for derivative action. As the setting value increases, the duration becomes smaller and action becomes closer to exact differential. Ideal value K_D = 8.00	0 to 32767 (unit: 0.01) If setting value > 30000 K_D = Infinite (∞)	0 to 32767 (unit: 0.01) If setting value > 30000 K_D = Infinite (∞)	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+12	(Fixed value)	—	0	0	User	—
(s)+13	(Fixed value)	—	0	0	User	—

*1 For the PID operational expression set in the operational expression selection, refer to the following.
📖 MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)

Precautions

The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

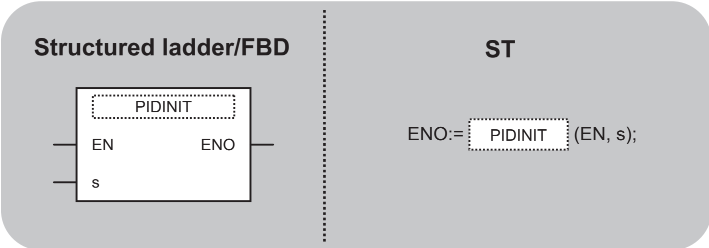
○: Applicable, ×: Not applicable

6

6.2 PID Control Instruction (Exact Differential)

Data setting

PIDINIT(P)



The following instruction can go in the dotted squares.
PIDINIT, PIDINITP

■Executing condition

Instruction	Executing condition
PIDINIT	
PIDINITP	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device that stores PID control data	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

This instruction enables PID control by registering the PID control data for the number of loops to be used to the CPU module in batch.

Setting data

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
Common setting data (device: (s)+0 to (s)+1)						
(s)+0	Number of loops	Set the number of loops for PID operation.	1 to 32		User	An error occurs and the PID operation for all loops is not performed.
(s)+1	Number of loops in one scan	Set the number of loops for PID operation in one scan if multiple loops have reached the sampling cycle time.	1 to 32		User	
Setting data for No. 1 loop (device: (s)+2 to (s)+11)						
(s)+2	Operational expression selection	Select the PID operational expression.[*1]	0: Forward operation 1: Reverse operation	0: Forward operation 1: Reverse operation	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+3	Sampling cycle (T_S)	Set the PID operation cycle.	1 to 6000 (unit: 10ms)	1 to 6000 (unit: 10ms)	User	
(s)+4	Proportional constant (K_P)	Proportional gain of PID operation	1 to 10000 (unit: 0.01)	1 to 10000 (unit: 0.01)	User	
(s)+5	Integral constant (T_I)	Constant that expresses the magnitude of the integral action (I action) effect. Increasing the integral constant slows down the manipulated value change.	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	User	
(s)+6	Derivative constant (T_D)	Constant that expresses the magnitude of the derivative action (D action) effect. Increasing the derivative constant causes a significant changes in the manipulated value even with a slight change of the control target.	0 to 30000 (unit: 10ms)	0 to 30000 (unit: 10ms)	User	
(s)+7	Filter coefficient (α)	Set the degree of filtering to be applied to the process value. The filtering effect decreases as the value gets closer to 0.	0 to 100	0 to 100	User	
(s)+8	MV lower limit (MVLL)	Set the lower limit for the manipulated value (MV) calculated in PID operation in automatic mode. If the MV is less than the set lower limit value (MVLL), the value is clipped to the MVLL.	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If MVLL or MVHL value is less than -50, the value is clipped to -50. • If MVLL or MVHL value is greater than 2050, the value is clipped to 2050.
(s)+9	MV upper limit (MVHL)	Set the upper limit for the manipulated value calculated in PID operation in automatic mode. If the MV is greater than the set upper limit value (MVHL), the value is clipped to the MVHL.	-50 to 2050	-32768 to 32767	User	

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+10	MV change rate limit (ΔMVL)	Set the variation limit between the previous MV and the present MV. When the MV variation is greater than the limit value, bit 1 (b1) of the alarm device is set to '1'. MV variation is not limited. (Even if the MV variation exceeds the limit value, the actual MV variation is used as it is for calculating the MV.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the ΔMVL value is less than 0, the value is clipped to 0. • If the ΔMVL value is greater than 2000, the value is clipped to 2000.
(s)+11	PV change rate limit (ΔPVL)	Set the variation limit between the previous PV and the present PV. When the PV variation is greater than the limit value, bit 0 (b0) of the alarm device is set to '1'. PV variation is not limited. (Even if the PV variation exceeds the limit value, the actual PV variation is used as it is for performing the PID operation.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If the ΔPVL value is less than 0, the value is clipped to 0. • If the ΔPVL value is greater than 2000, the value is clipped to 2000.
Setting data for No. 2 loop (device: (s)+12 to (s)+21)						
(s)+12	Operational expression selection	The same as Setting data for No. 1 loop				
(s)+13	Sampling cycle (T_S)					
(s)+14	Proportional constant (K_P)					
(s)+15	Integral constant (T_I)					
(s)+16	Derivative constant (T_D)					
(s)+17	Filter coefficient (α)					
(s)+18	MV lower limit (MVLL)					
(s)+19	MV upper limit (MVHL)					
(s)+20	MV change rate limit (ΔMVL)					
(s)+21	PV change rate limit (ΔPVL)					
Setting data for No. n loop						

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+(m+0)	Operational expression selection	The same as Setting data for No. 1 loop m=(n-1)×10+2 n: number of loops				
(s)+(m+1)	Sampling cycle (T_S)					
(s)+(m+2)	Proportional constant (K_P)					
(s)+(m+3)	Integral constant (T_I)					
(s)+(m+4)	Derivative constant (T_D)					
(s)+(m+5)	Filter coefficient (α)					
(s)+(m+6)	MV lower limit (MVLL)					
(s)+(m+7)	MV upper limit (MVHL)					
(s)+(m+8)	MV change rate limit (ΔMVL)					
(s)+(m+9)	PV change rate limit (ΔPVL)					

*1 For the PID operational expression set in the operational expression selection, refer to the following.
📖 MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)

Precautions

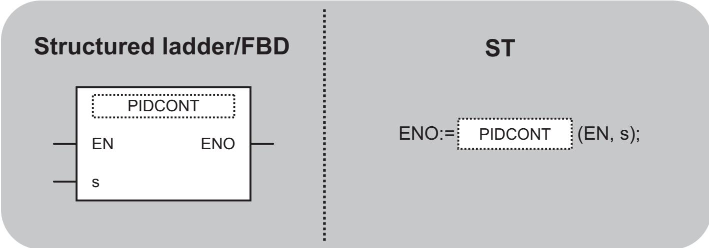
The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

○: Applicable, ×: Not applicable

PID operation

PIDCONT(P)



The following instruction can go in the dotted squares.

PIDCONT, PIDCONTP

■Executing condition

Instruction	Executing condition
PIDCONT	
PIDCONTP	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device that is assigned in I/O data area	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

- This instruction measures sampling cycle and performs PID operation at instruction execution.
- This instruction performs PID operation based on the set value (SV) and process value (PV) in the I/O data area set to the device number specified by (s) or later, and stores the operation result to the automatic manipulated value (MV) area in the I/O data area.
- PID operation is performed in response to the first execution of the PIDCONT instruction after the set sampling cycle time has elapsed.

Setting data

Device	Data name		Description	Setting range		Setting side	Processing when the setting data are outside the setting range
				With PID limits	Without PID limits		
(s)+0	Initial processing flag		Processing method at the start of PID operation	0: PID operation for the number of loops to be used is batch-processed in one scan. Other than 0: PID operation for the number of loops to be used is processed in several scans.		User	—
(s)+1 ⋮ (s)+9	PID control work area (reserved by the system)			—	—	—	—
I/O data area for No. 1 loop (device: (s)+10 to (s)+27)							
(s)+10	Setting value	SV	• PID control target value	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If SV is less than 0, the value is clipped to 0. • If SV is greater than 2000, the value is clipped to 2000.
(s)+11	Process value	PV	• Feedback data from the control target to the A/D conversion module	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If PV is less than -50, the value is clipped to -50. • If PV is greater than 2050, the value is clipped to 2050.
(s)+12	Automatic manipulated value	MV	• Manipulated value obtained by PID operation • The value is output from the D/A conversion module to the control target.	-50 to 2050	-32768 to 32767	System	—
(s)+13	Process value after filtering	PVf	• Process value obtained by calculation using operational expression.[*1]	-50 to 2050	-32768 to 32767	System	—
(s)+14	Manual manipulated value	MV_{MAN}	• Store the data output from the D/A conversion module in manual operation.	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • If MV_{MAN} is less than -50, the value is clipped to -50. • If MV_{MAN} is greater than 2050, the value is clipped to 2050.
(s)+15	Manual/ automatic selection	MAN/ AUTO	• Select whether the output to the D/A conversion module is a manual manipulated value or an automatic manipulated value. • In manual operation, the automatic manipulated value remains unchanged.	0: Automatic manipulated value 1: Manual manipulated value		User	When other than 0 or 1 is selected, an error occurs and the operation for the corresponding loop is not performed.

Device	Data name		Description	Setting range		Setting side	Processing when the setting data are outside the setting range
				With PID limits	Without PID limits		
(s)+16	Alarm	ALARM	• Used to determine if the change rate of the MV (manipulated value) and the PV (process value) is within or outside the limit value range. • Once set, the alarm data are maintained until the user resets it.	b15 … b1 b0 0 | (2) | (1) When the MV variation is outside the limit range, bit 1 (b1) is set to '1'. When the PV variation is outside the limit range, bit 0 (b0) is set to '1'.		User System	—
(s)+17 ⋮ (s)+27	PID control work area (reserved by the system)			—		—	—
I/O data area for No. 2 loop (device: (s)+28 to (s)+45)							
(s)+28	Setting value	SV	The same as I/O data area for No. 1 loop				
(s)+29	Process value	PV					
(s)+30	Automatic manipulated value	MV					
(s)+31	Process value after filtering	PVf					
(s)+32	Manual manipulated value	MV_{MAN}					
(s)+33	Manual/ automatic selection	MAN/ AUTO					
(s)+34	Alarm	ALARM					
(s)+35 ⋮ (s)+45	PID control work area (reserved by the system)			—		—	—
I/O data area for No. n loop							
(s)+(m+0)	Setting value	SV	The same as I/O data area for No. 1 loop m=(n-1) × 18+10 n: number of loops				
(s)+(m+1)	Process value	PV					
(s)+(m+2)	Automatic manipulated value	MV					
(s)+(m+3)	Process value after filtering	PVf					
(s)+(m+4)	Manual manipulated value	MV_{MAN}					
(s)+(m+5)	Manual/ automatic selection	MAN/ AUTO					
(s)+(m+6)	Alarm	ALARM					
(s)+(m+7) ⋮ (s)+(m+17)	PID control work area (reserved by the system)			—		—	—

*1 For process value after filtering (PVf), the value calculated based on the process value of input data are stored.
For the operational expression, refer to the following.
MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)

Precautions

The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

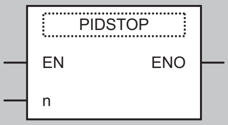
CPU module model		**Inexact differential**	**Exact differential**
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

○: Applicable, ×: Not applicable

6

PIDSTOP instruction and PIDRUN instruction

PIDSTOP, PIDRUN

Structured ladder/FBD	ST
	ENO:= PIDSTOP (EN, n);

The following instruction can go in the dotted squares.
PIDSTOP, PIDSTOPP, PIDRUN, PIDRUNP

■Executing condition

Instruction	Executing condition
PIDSTOP, PIDRUN	
PIDSTOPP, PIDRUNP	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Loop number for stop/start	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(n)	○								—

Processing details

- PIDSTOP(P)

This instruction stops the PID operation for the loop number specified by 'n'.

- PIDRUN(P)

This instruction starts the PID operation for the loop number specified by 'n'.

Precautions

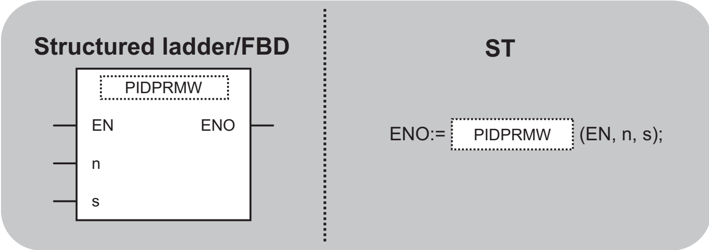
The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

○: Applicable, ×: Not applicable

Operation parameter change

PIDPRMW(P)



The following instruction can go in the dotted squares.
PIDPRMW, PIDPRMWP

■Executing condition

Instruction	Executing condition
PIDPRMW	
PIDPRMWP	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Loop number to be changed	ANY16
	s	Start number of the device that stores PID control data to be changed	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(n)	○	○		○				○	—
(s)	—	○		—				—	—

Processing details

This instruction changes the operation parameter of the loop number specified by 'n' to the PID control data stored in the devices starting from the device number specified by (s).

Setting data

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+0	Operational expression selection	Select the PID operational expression.*1	0: Forward operation 1: Reverse operation	0: Forward operation 1: Reverse operation	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+1	Sampling cycle (T_S)	Set the PID operation cycle.	1 to 6000 (unit: 10ms)	1 to 6000 (unit: 10ms)	User	
(s)+2	Proportional constant (K_P)	Proportional gain of PID operation	1 to 10000 (unit: 0.01)	1 to 10000 (unit: 0.01)	User	An error occurs and the PID operation for the corresponding loop is not performed.
(s)+3	Integral constant (T_I)	Constant that expresses the magnitude of the integral action (I action) effect. Increasing the integral constant slows down the manipulated value change.	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	1 to 32767 (unit: 100ms) If setting value > 30000 T_I = Infinite (∞)	User	
(s)+4	Derivative constant (T_D)	Constant that expresses the magnitude of the derivative action (D action) effect. Increasing the derivative constant causes significant changes in the manipulated value even with a slight change of the control target.	0 to 30000 (unit: 10ms)	0 to 30000 (unit: 10ms)	User	
(s)+5	Filter coefficient (α)	Set the degree of filtering to be applied to the process value. The filtering effect decreases as the value gets closer to 0.	0 to 100	0 to 100	User	
(s)+6	MV lower limit (MVLL)	Set the lower limit for the manipulated value (MV) calculated in PID operation in automatic mode. If the MV is less than the set lower limit value (MVLL), the value is clipped to the MVLL.	-50 to 2050	-32768 to 32767	User	In the case of "With PID limits", the PID operation is not performed after values are replaced as follows: • If MVLL or MVHL value is less than -50, the value is clipped to -50. • If MVLL or MVHL value is greater than 2050, the value is clipped to 2050.
(s)+7	MV upper limit (MVHL)	Set the upper limit for the manipulated value calculated in PID operation in automatic mode. If the MV is greater than the set upper limit value (MVHL), the value is clipped to the MVHL.	-50 to 2050	-32768 to 32767	User	
(s)+8	MV change rate limit (ΔMVL)	Set the variation limit between the previous MV and the present MV. When the MV variation is greater than the limit value, bit 1 (b1) of the alarm device is set to '1'. MV variation is not limited. (Even if the MV variation exceeds the limit value, the actual MV variation is used as it is for calculating the MV.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed after values are replaced as follows: • ΔMVL value is less than 0, the value is clipped to 0. • ΔMVL value is greater than 2000, the value is clipped to 2000.

Device	Data item	Description	Setting range		Setting side	Processing when the setting data are outside the setting range
			With PID limits	Without PID limits		
(s)+9	PV change rate limit (ΔPVL)	Set the variation limit between the previous PV and the present PV. When the PV variation is greater than the limit value, bit 0 (b0) of the alarm device is set to '1'. PV variation is not limited. (Even if the PV variation exceeds the limit value, the actual PV variation is used as it is for performing the PID operation.)	0 to 2000	-32768 to 32767	User	In the case of "With PID limits", the PID operation is performed values are replaced as follows: • If the ΔPVL value is less than 0, the value is clipped to 0. • If the ΔPVL value is greater than 2000, the value is clipped to 2000.

*1 For the PID operational expression set in the operational expression selection, refer to the following.
📖 MELSEC-Q/L/QnA Programming Manual (PID Control Instructions)

Precautions

The following table shows the CPU modules applicable to the PID control instructions (inexact differential) and the PID control instructions (exact differential).

CPU module model		Inexact differential	Exact differential
Basic model QCPU	The first five digits of the serial number are '04121' or lower.	×	×
	The first five digits of the serial number are '04122' or higher	○	○
High Performance model QCPU	The first five digits of the serial number are '05031' or lower.	×	○
	The first five digits of the serial number are '05032' or higher.	○	○
Redundant CPU		○	○
Universal model QCPU		○	○
LCPU		○	○

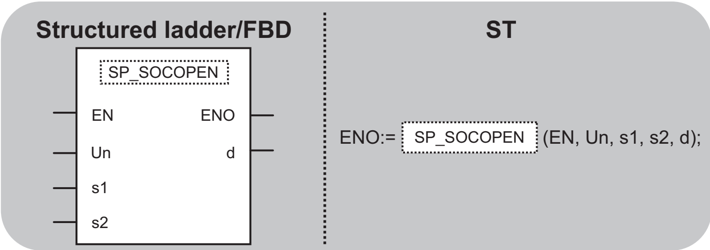
○: Applicable, ×: Not applicable

7 SOCKET COMMUNICATION FUNCTION INSTRUCTION

7.1 Opening/Closing Connection

SP_SOCOPEN

QnUDE(H) LCPU



The following instruction can go in the dotted squares.
SP_SOCOPEN

■Executing condition

Instruction	Executing condition
SP_SOCOPEN	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..9]
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON during one scan upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	△*1	△*1	—				—	—
(d)	△*1	—	△*1	—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction establishes a connection.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	Execution type/Completion type	Specify which to use the parameter values set by GX Works2 or the setting values of the following control data ((s2)[2] to (s2)[9]) at open processing of a connection. 0000H: Uses the parameter set in [Open settings] of GX Works2. 8000H: Uses the settings of control data (s2)[2] to (s2)[9].	0000H 8000H	User
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2]	Application setting area	b15 b14 ··· b9 b8 ··· b0 (3) | 0 | (2) | (1) | 0 (1) Communication method (protocol) (b8) 0: TCP/IP 1: UDP/IP (2) With/without procedure in socket communication function (b9) 1: Nonprocedural communication (3) Open system (b15, b14) 00: Active open or UDP/IP 10: Unpassive open 11: Fullpassive open	(Refer to the left.)	User
(s2)[3]	Host station port No.	Specify the port number of the host station.	1H to 1387H 1392H to FFFEH (400H or later is recommended)	User
(s2)[4] (s2)[5]	Destination IP address*2	Specify the IP address of the external device.	1H to FFFFFFFFH (FFFFFFFFH : broadcast)	User
(s2)[6]	Destination port No.*2	Specify the port number of the external device.	1H to FFFFH (FFFFH: broadcast)	User
(s2)[7] to (s2)[9]	—	Unavailable	—	System

*2 "Destination IP address" and "Destination port No" are neglected at Unpassive open.

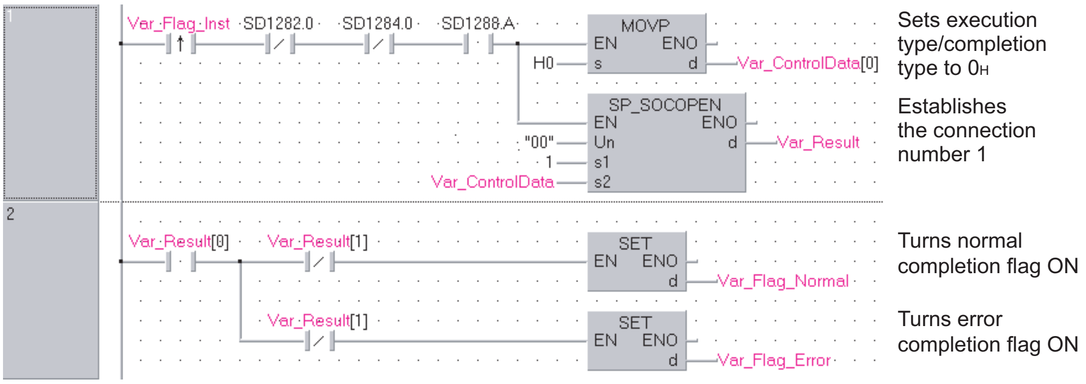
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program opens the connection 1.

[Structured ladder/FBD]

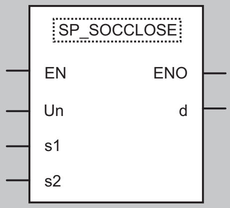


[ST]

```
IF((LDP( TRUE, Var_Flag_Inst ))
&(SD1282.0=FALSE) &(SD1284.0=FALSE) &(SD1288.A=TRUE))THEN
    MOVP(TRUE, H0, Var_ControlData[0]); (* Sets execution type/completion type to 0H *)
    SP_SOCOPEN( TRUE, "00", 1, Var_ControlData, Var_Result ); (* Establishes the connection number 1 *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET( FALSE, Var_Flag_Normal ); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

SP_SOCCLOSE

QnUDE(H) LCPU

Structured ladder/FBD	ST
	ENO:= SP_SOCCLOSE (EN, Un, s1, s2, d);

The following instruction can go in the dotted squares.

SP_SOCCLOSE

■Executing condition

Instruction	Executing condition
SP_SOCCLOSE	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 (0..1)
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON during one scan upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	△*1	△*1	—				—	—
(d)	△*1	—	△*1	—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction shuts off a specified connection.

7

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

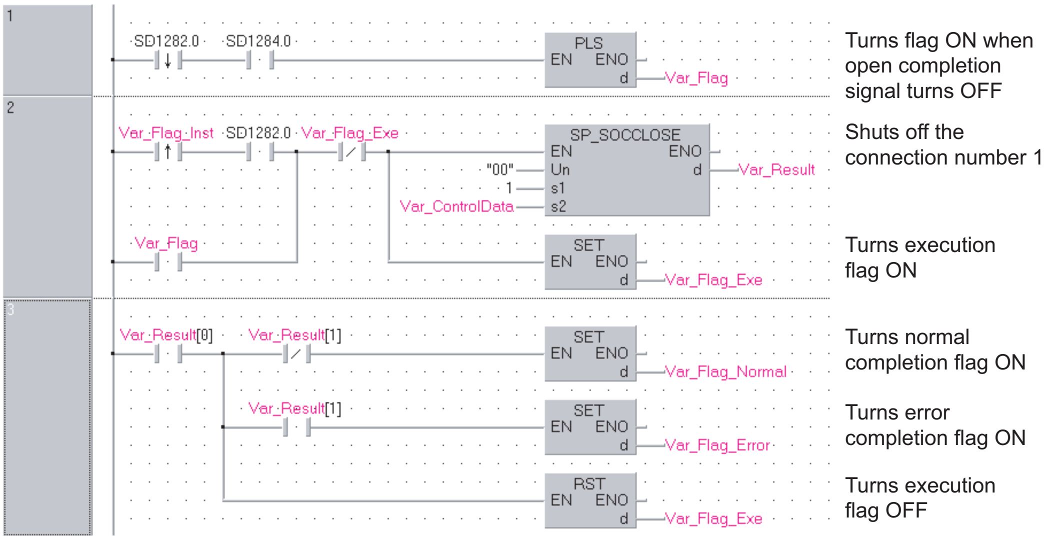
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program shuts off the connection 1 when the disconnect request flag turns ON or the external device closes the connection 1.

[Structured ladder/FBD]



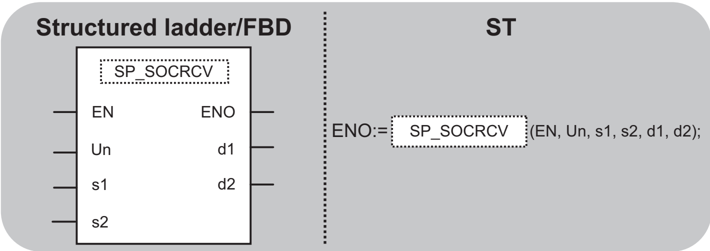
[ST]

```
IF((LDF( TRUE, SD1282.0 ))
&(SD1284.0=TRUE))THEN (* When open completion signal turns OFF *)
    PLS(TRUE, Var_Flag); (* Turns flag ON *)
END_IF;

IF(((LDP(TRUE, Var_Flag_Inst) & SD1282.0) OR Var_Flag) & (NOT Var_Flag_Exe)) THEN
    SP_SOCCLOSE(TRUE, "00", 1, Var_ControlData, Var_Result); (* Shuts off the connection number 1 *)
    SET(TRUE, Var_Flag_Exe); (* Turns execution flag ON *)
ELSE
    SP_SOCCLOSE(FALSE, "00", 1, Var_ControlData, Var_Result);
    SET(FALSE, Var_Flag_Exe);
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN
        SET(FALSE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
    RST(TRUE, Var_Flag_Exe); (* Turns execution flag OFF *)
END_IF;
```

7.2 SOCRCV Instruction

SP_SOCRCV

QnUDE(H) LCPU



The following instruction can go in the dotted squares.
SP_SOCRCV

■Executing condition

Instruction	Executing condition
SP_SOCRCV	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
Output argument	ENO	Execution result	Bit
	d1	Start number of the device that stores receive data	ANY16
	d2	Variable that turns ON during one scan upon completion of the instruction d2[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	△*1	△*1	—				—	—
(d1)	—	△*1	△*1	—				—	—
(d2)	△*1	—	△*1	—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction reads receive data of a specified connection from the socket communication receive data area at the end process performed after the instruction execution.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

Device	Item	Setting data	Setting range	Setting side
(d1)+0	Receive data length	Data length of the data read from the socket communication receive data area is stored. (number of bytes)	0 to 2046	System
(d1)+1 to (d1)+n	Receive data	Data read from the socket communication receive data area are stored in ascending address order.	—	System

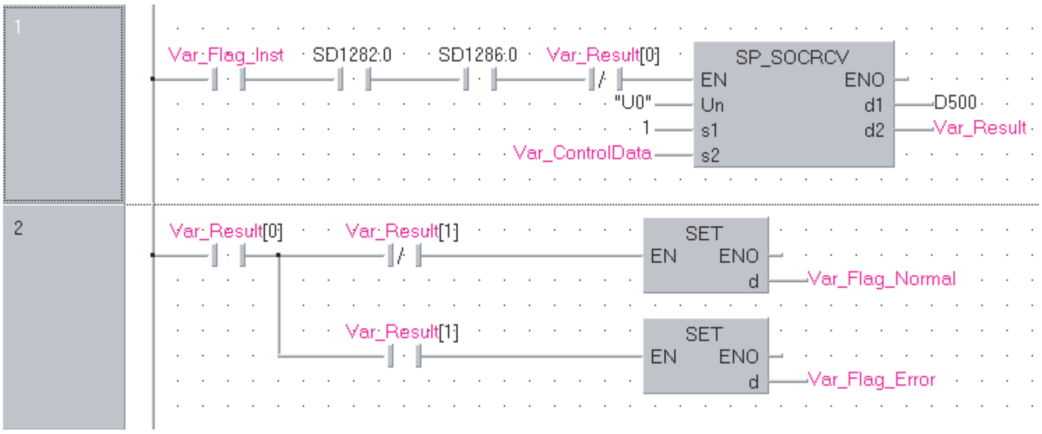
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program reads data received from the external device.

[Structured ladder/FBD]



Reads data from the connection number 1

Turns normal completion flag ON

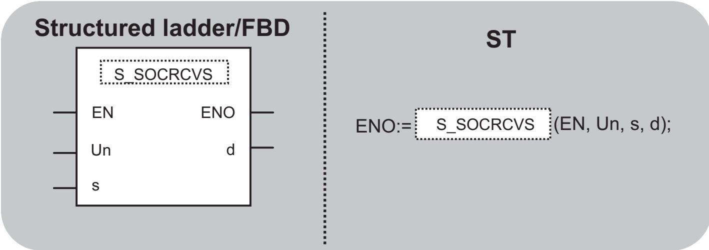
Turns error completion flag OFF

[ST]

```
IF((Var_Flag_Inst=TRUE) &(SD1282.0=TRUE) &(SD1286.0=TRUE) &(Var_Result[0]=FALSE))THEN
    SP_SOCRCV ( TRUE, "U0", 1, Var_ControlData, D500, Var_Result ); (* Reads data from the connection number 1 *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
        SET(TRUE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
        SET(TRUE, Var_Flag_Error); (* Turns error completion flag ON *)
    END_IF;
END_IF;
```

S_SOCRCVS

QnUDE(H) LCPU



The following instruction can go in the dotted squares.

S_SOCRCVS

■Executing condition

Instruction	Executing condition
S_SOCRCVS	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s	Connection number (1 to 16)	ANY16
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores receive data	ANY16

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s)	—	○	○	—				○	—
(d)	—	○	○	—				—	—

Processing details

This instruction reads receive data of a specified connection from the socket communication receive data area.

Setting data

Device	Item	Setting data	Setting range	Setting side
(d)+0	Receive data length	Data length of the data read from the socket communication receive data area is stored. (number of bytes)	0 to 2046	System
(d)+1 to (d)+(n)	Receive data	Data read from the socket communication receive data area are stored in ascending address order.	—	System

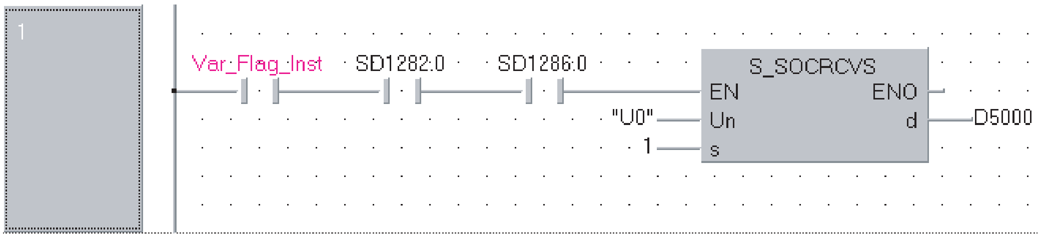
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program reads data received from the external device.

[Structured ladder/FBD]



Reads data from the connection number 1

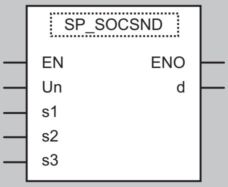
[ST]

```
IF((Var_Flag_Inst=TRUE) &(SD1282.0=TRUE) &(SD1286.0=TRUE) )THEN
    S_SOCRCVS( TRUE, "U0", 1, D5000 ); (* Reads data from the connection number 1 *)
END_IF;
```

7.3 Sending Data

SP_SOCSND

QnUDE(H) LCPU

Structured ladder/FBD	ST
	ENO:= SP_SOCSND (EN, Un, s1, s2, s3, d);

The following instruction can go in the dotted squares.
SP_SOCSND

■Executing condition

Instruction	Executing condition
SP_SOCSND	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
	s3	Start number of the device that stores send data	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that turns ON during one scan upon completion of the instruction d[1] also turns ON at the time of error completion.	Array of bit [0..1]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	△*1	△*1	—				—	—
(s3)	—	○	○	—				—	—
(d)	△*1	—	△*1	—				—	—

*1 Local devices and file registers per program cannot be used as setting data.

Processing details

This instruction sends data to the external device of a specified connection.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

Device	Item	Setting data	Setting range	Setting side
(s3)+0	Send data length	Data length of the data read from the fixed buffer data area is stored. (number of bytes)	0 to 2046	User
(s3)+1 to (s3)+n	Send data	Specify the send data.	—	User

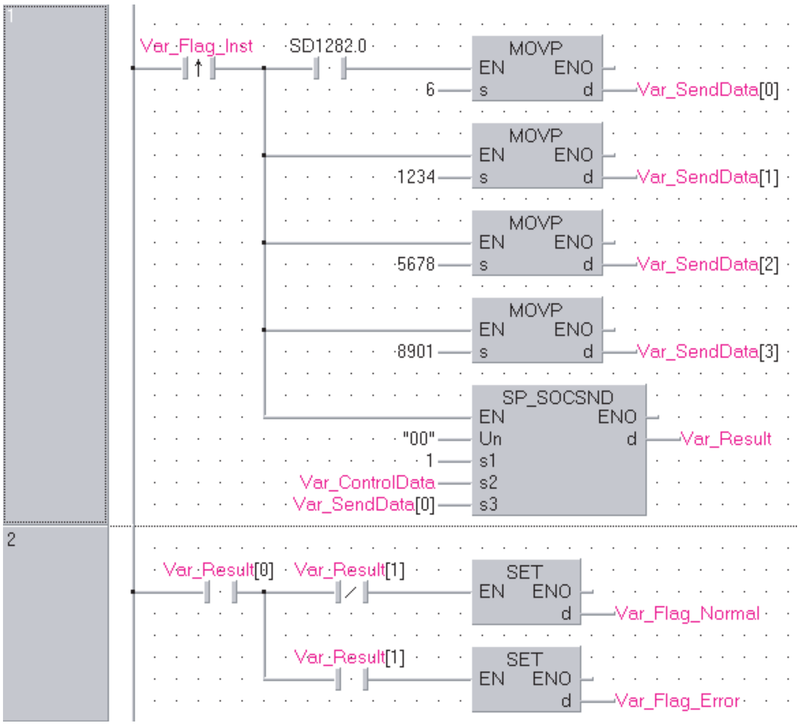
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program sends data (1234, 5678, and 8901) to the external device using the socket communication function.

[Structured ladder/FBD]



Sets data length by the byte

Sets send data

Sends data to the connection number 1

Turns normal completion flag ON

Turns error completion flag ON

[ST]

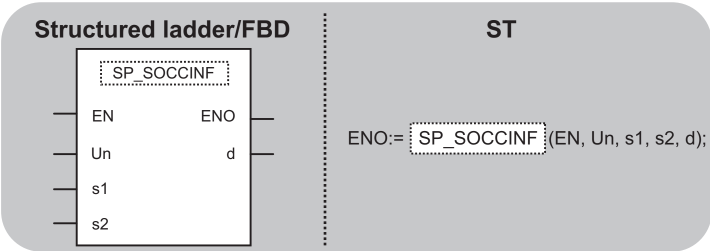
```
IF((Var_Flag_Inst=TRUE)&(SD1282.0=TRUE))THEN
    MOVP(TRUE, 6, Var_SendData[0]); (* Sets data length by the byte *)
    MOVP(TRUE, 1234, Var_SendData[1]); (* Sets send data *)
    MOVP(TRUE, 5678, Var_SendData[2]);
    MOVP(TRUE, 8901, Var_SendData[3]);
    SP_SOCSND( TRUE, "00", 1, Var_ControlData, Var_SendData[0], Var_Result ); (* Sends data to the connection number 1 *)
END_IF;

IF(Var_Result[0]=TRUE)THEN (* Execution finished *)
    IF(Var_Result[1]=FALSE)THEN (* Normal completion *)
      SET(FALSE, Var_Flag_Normal); (* Turns normal completion flag ON *)
    ELSE (* Error completion *)
      SET(TRUE, Var_Flag_Error); (* Turns error completion flag OFF *)
    END_IF;
END_IF;
```

7

7.4 SOCCINF Instruction

SP_SOCCINF

QnUDE(H) LCPU



The following instruction can go in the dotted squares.
SP_SOCCINF

■Executing condition

Instruction	Executing condition
SP_SOCCINF	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
Output argument	ENO	Execution result	Bit
	d	Variable that stores connection information	Array of ANY16 [0..4]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	○	○	—				—	—
(d)	—	○	○	—				—	—

Processing details

This instruction reads connection information of a specified connection.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

Device	Item	Setting data	Setting range	Setting side
(d)[0] (d)[1]	Destination IP address	The IP address of the external device is stored.	1H to FFFFFFFFH 0H : No destination (FFFFFFFFH : broadcast)	System
(d)[2]	Destination port No.	The port number of the external device is stored.	1H to FFFFH (FFFFH: broadcast)	System
(d)[3]	Host station port No.	The port number of the host station is stored.	1H to 1387H 1392H to FFFEH	System
(d)[4]	Application setting area	b15 b14 … b9 b8 … b0 (3) | 0 | (2) | (1) | 0 (1) Communication method (protocol) (b8) 0: TCP/IP 1: UDP/IP (2) With/without procedure in socket communication function (b9) 1: Nonprocedural communication (3) Open system (b15, b14) 00: Active open or UDP/IP 10: Unpassive open 11: Fullpassive open	—	System

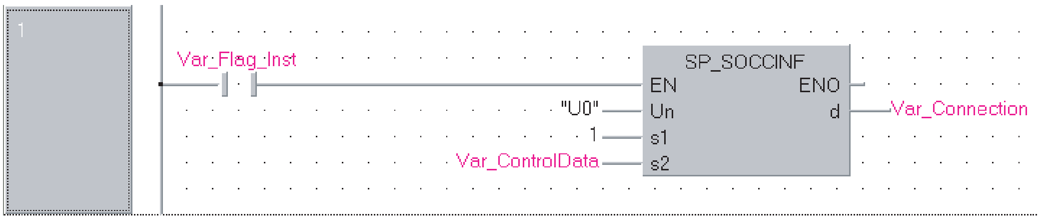
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program reads connection information of the connection number 1.

[Structured ladder/FBD]



Reads data from the connection number 1

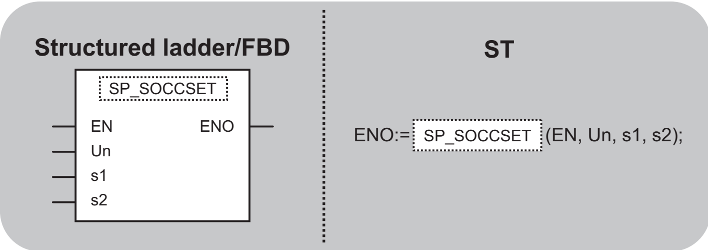
[ST]

```
IF(Var_Flag_Inst=TRUE)THEN
    SP_SOCCINF( TRUE, "U0", 1, Var_ControlData, Var_Connection ); (* Reads data from the connection number 1 *)
END_IF;
```

7.5 Changing Destination

SP_SOCCSET

QnUDE(H) LCPU



The following instruction can go in the dotted squares.
SP_SOCCSET

■Executing condition

Instruction	Executing condition
SP_SOCCSET	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..4]
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	○	○	—				—	—

Processing details

This instruction changes the IP address and port number of the external device of a specified connection.
(Available only with a UDP/IP connection)

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2] (s2)[3]	Destination IP address	Specify the IP address of the external device.	1H to FFFFFFFFH 0H : No destination (FFFFFFFFH : broadcast)	User
(s2)[4]	Destination port No.	Specify the port number of the external device.	1H to FFFFH (FFFFH: broadcast)	User

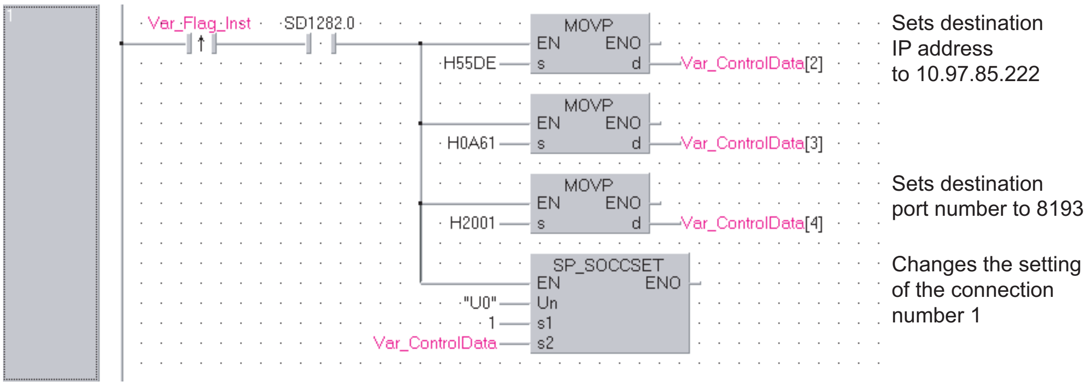
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program changes the destination (destination IP address and port number) of the connection number 1 which is being open.

[Structured ladder/FBD]



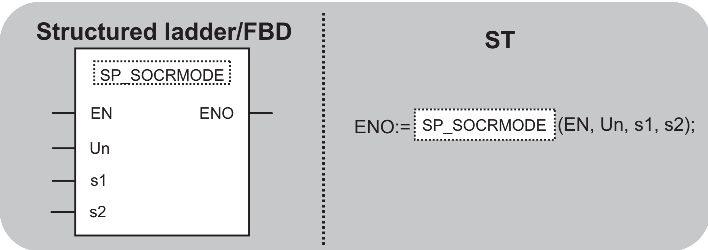
[ST]

```
IF((LDP( TRUE, Var_Flag_Inst )) &(SD1282.0=TRUE))THEN
    MOVP( TRUE, H55DE, Var_ControlData[2] );
    MOVP( TRUE, H0A61, Var_ControlData[3] ); (* Sets destination IP address to 10.97.85.222 *)
    MOVP(TRUE, H2001, Var_ControlData[4]); (* Sets destination port number to 8193 *)
    SP_SOCCSET( TRUE, "U0", 1, Var_ControlData ); (* Changes the setting of the connection number 1 *)
END_IF;
```

7.6 Changing Receive Mode

SP_SOCRMODE

QnUDE(H) LCPU



The following instruction can go in the dotted squares.

SP_SOCRMODE

■Executing condition

Instruction	Executing condition
SP_SOCRMODE	(rising edge)

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..3]
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	○	○	—				—	—

Processing details

This instruction changes the TCP receive mode (unavailable for a UDP connection) and receive data size.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction completion status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System
(s2)[2]	TCP Receive Mode[*1]	Specify the TCP receive mode. 0: TCP normal receive mode 1: TCP fixed length receive mode	0, 1	User
(s2)[3]	Receive Data Size	Specify the receive data size of the socket communication. (number of bytes)	1 to 2046	User

*1 Unavailable for a UDP connection.

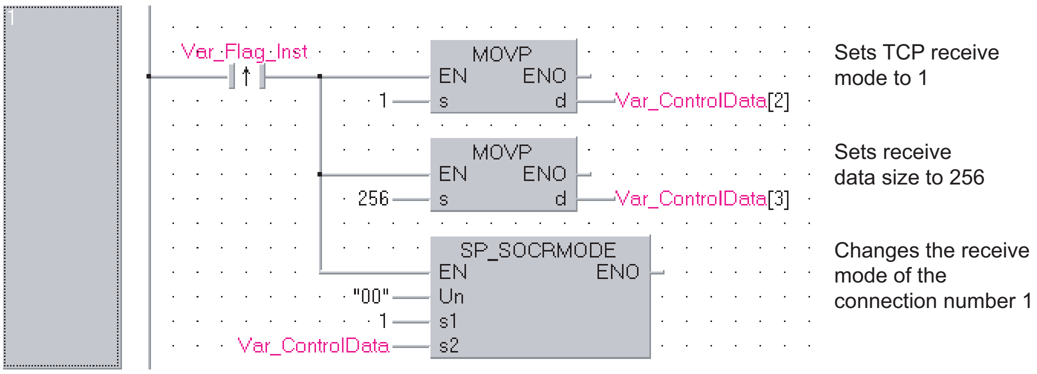
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program changes the receive mode of the connection number 1 to TCP fixed length receive mode and changes its receive data length to 256 bytes.
- After instruction execution, the connection number 1 turns the receive status signal ON when the length of receive data reaches 256 bytes.

[Structured ladder/FBD]



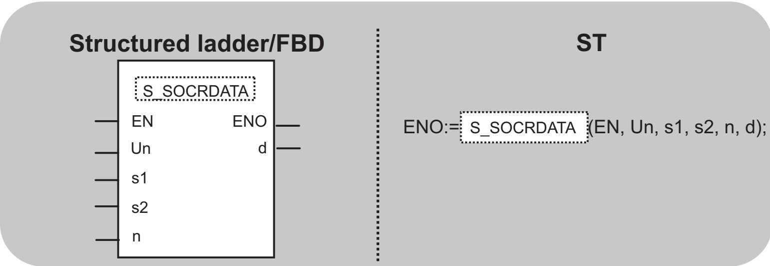
[ST]

```
IF(Var_Flag_Inst=TRUE)THEN
    MOVP(TRUE, 1, Var_ControlData[2]); (* Sets TCP receive mode to 1 *)
    MOVP(TRUE, 256, Var_ControlData[3]); (* Sets receive data size to 256 *)
    SP_SOCRMODE( TRUE, "00", 1, Var_ControlData); (*Changes the receive mode of the connection number 1 *)
END_IF;
```

7.7 SOCRDATA Instruction

S(P)_SOCRDATA

QnUDE(H) LCPU



The following instruction can go in the dotted squares.
S_SOCRDATA, SP_SOCRDATA

■Executing condition

Instruction	Executing condition
S_SOCRDATA	
SP_SOCRDATA	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	Un	Dummy ("U0")	String
	s1	Connection number (1 to 16)	ANY16
	s2	Variable that stores control data	Array of ANY16 [0..1]
	n	Number of read data (1 to 1024 words)	ANY16
Output argument	ENO	Execution result	Bit
	d	Variable that stores read data	ANY16

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
(s1)	—	○	○	—				○	—
(s2)	—	○	○	—				—	—
n	—	○	○	—				○	—
(d)	—	○	○	—				—	—

Processing details

This instruction reads data for the specified number of words from the socket communication receive data area of a specified connection, and stores it.

Setting data

Device	Item	Setting data	Setting range	Setting side
(s2)[0]	System area	—	—	—
(s2)[1]	Completion status	The instruction application status is stored. 0: Normal completion Other than 0: Error completion (error code)	—	System

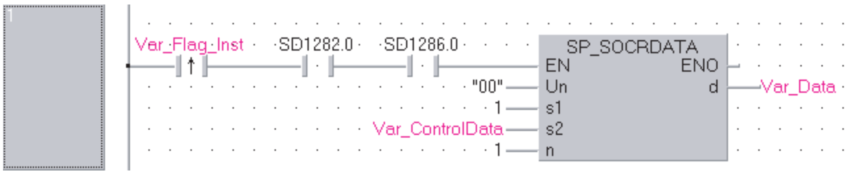
Precautions

- Use the Built-in Ethernet port QCPU of which the function version is B or later and the first five digits of the serial number are '11012' or higher.
- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program reads the receive data length of the connection number 1.

[Structured ladder/FBD]



Reads receive data length of the connection number 1

[ST]

```
IF((Var_Flag_Inst=TRUE) &(SD1282.0=TRUE) &(SD1286.0=TRUE))THEN
    SP_SOCRDATA( TRUE, "00", 1, Var_ControlData, 1, Var_Data); (* Reads receive data length of connection number 1 *)
END_IF;
```

7

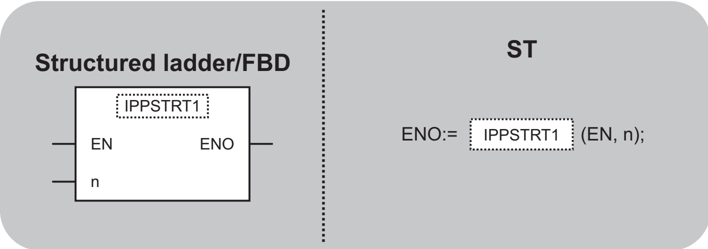
8 BUILT-IN I/O FUNCTION INSTRUCTION

8.1 Positioning Function Dedicated Instruction

Positioning start

IPPSTRT1, IPPSTRT2

LCPU



The following instruction can go in the dotted squares.
IPPSTRT1, IPPSTRT1P, IPPSTRT2, IPPSTRT2P

■Executing condition

Instruction	Executing condition
IPPSTRT1 IPPSTRT2	
IPPSTRT1P IPPSTRT2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Positioning data number (Setting range: 1 to 10)	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
n	—	○		—			○		—

Processing details

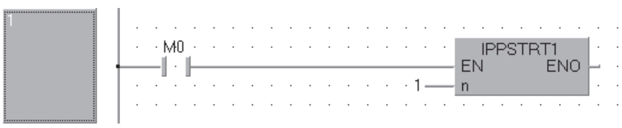
This instruction specifies a data number to be executed for 'n' from the positioning data No. 1 to No. 10 which are previously set in GX Works2, and starts the specified axis (refer to the following).

- IPPSTRT1(P): Axis 1
- IPPSTRT2(P): Axis 2

Program example

- The following program starts the "Positioning Data" No. 1 of the Axis 1 when M0 turns ON.

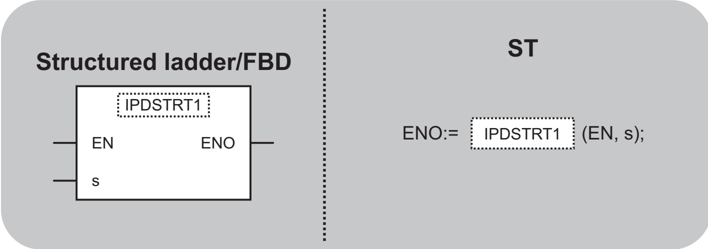
[Structured ladder/FBD]



[ST]
IPPSTRT1(M0, 1);

IPDSTRT1, IPDSTRT2

LCPU



The following instruction can go in the dotted squares.
IPDSTRT1, IPDSTRT1P, IPDSTRT2, IPDSTRT2P

■Executing condition

Instruction	Executing condition
IPDSTRT1 IPDSTRT2	
IPPSTRT1P IPDSTRT2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device in which the control data are stored	Array of ANY16 [0..7]
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

Regardless of "Positioning Data" No. 1 to No. 10 which are previously set in GX Works2, this instruction starts the positioning of the specified axis (refer to the following) using the data stored in the devices starting from (s).

- IPDSTRT1(P): Axis 1
- IPDSTRT2(P): Axis 2

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Control system	1: Positioning control (ABS) 2: Positioning control (INC) 3: Speed/position switching control (forward RUN) 4: Speed/position switching control (reverse RUN) 5: Current value change 6: Speed control (forward RUN) 7: Speed control (reverse RUN)	1 to 7	User
(s)[1]	Acceleration/deceleration time	—	0 to 32767(ms)	
(s)[2]	Deceleration stop time	—	0 to 32767(ms)	
(s)[3]	Dwell time	—	0 to 65535(ms)*1	
(s)[4] (s)[5]	Command speed	—	0 to 200000 (pulse/s)*2	
(s)[6] (s)[7]	Positioning address/movement amount	—	-2147483648 to 2147483647(pulse)	

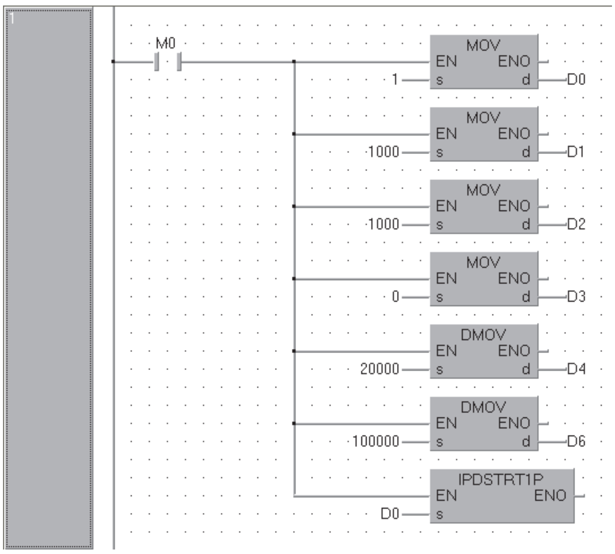
*1 Enter the setting value to the program as described below.
1 to 32767: Enter in decimal
32768 to 65535: Enter after converting it to hexadecimal
*2 The restricted speed value may be applied when the set value of the command speed is not within 0 to 200000.

Program example

- The following program sets the following positioning data and starts the axis 1 when M0 turns ON.

Device	Item	Setting data
D0	Control system	Positioning control (ABS)
D1	Acceleration/deceleration time	1000(ms)
D2	Deceleration stop time	1000(ms)
D3	Dwell time	0(ms)
D4, D5	Command speed	20000(pulse/s)
D6, D7	Positioning address/movement amount	100000(pulse)

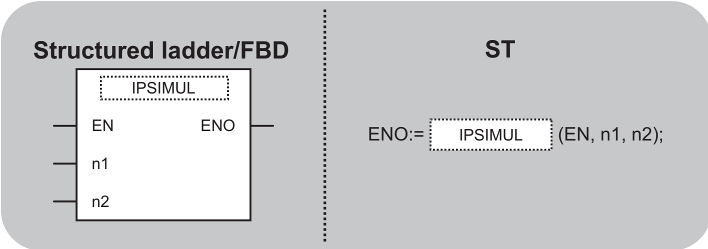
[Structured ladder/FBD]



[ST]
```
MOV( M0, 1, D0);
MOV( M0, 1000, D1);
MOV( M0, 1000, D2);
MOV( M0, 0, D3);
DMOV( M0, 20000, D4);
DMOV( M0, 100000, D6);
IPDSTRT1P(M0, D0);
```

IPSIMUL(P)

LCPU



The following instruction can go in the dotted squares.
IPSIMUL, IPSIMULP

■Executing condition

Instruction	Executing condition
IPSIMUL	
IPSIMULP	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n1	Axis 1 positioning data number	ANY16
	n2	Axis 2 positioning data number	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
n1	—	○		—			○		—
n2	—	○		—			○		—

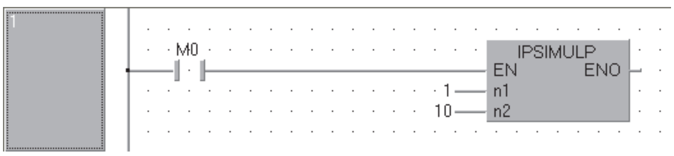
Processing details

This instruction simultaneously starts the positioning of the axis 1 positioning data number specified by n1 and the axis 2 positioning data number specified by n2.

Program example

- The following program simultaneously starts the axis 1 positioning data No. 1 and the axis 2 positioning data No. 10 when M0 turns ON.

[Structured ladder/FBD]

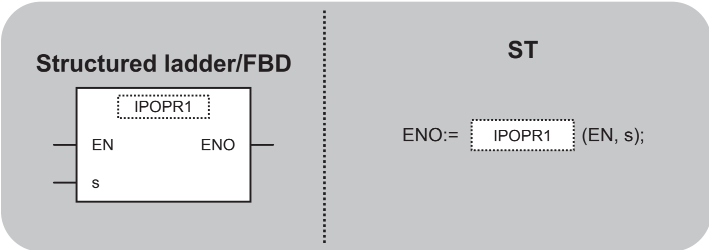


[ST]
IPSIMULP(M0, 1, 10);

OPR start

IPOPR1, IPOPR2

LCPU



The following instruction can go in the dotted squares.
IPOPR1, IPOPR1P, IPOPR2, IPOPR2P

■Executing condition

Instruction	Executing condition
IPOPR1 IPOPR2	
IPOPR1P IPOPR2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device in which the control data are stored	Array of ANY16 [0..2]
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

This instruction starts the OPR of which type is specified by (s) on the specified axis (refer to the following).

- IPOPR1(P): Axis 1
- IPOPR2(P): Axis 2

Setting data

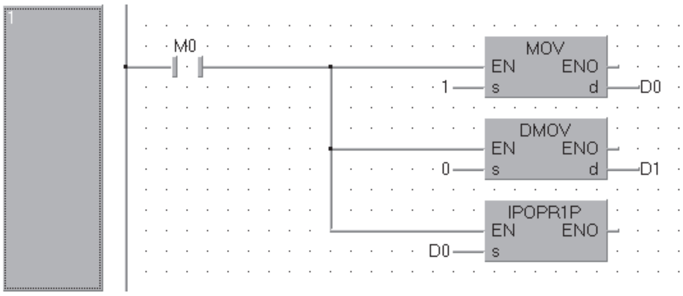
Device	Item	Setting data	Setting range	Setting side
(s)[0]	OPR type	1: Machine OPR 2: Fast OPR (OP address) 3: Fast OPR (standby address)	1 to 3	User
(s)[1] (s)[2]	Standby address (Set only when Fast OPR (standby address (3)) is set for the OPR type)	—	-2147483648 to 2147483647(pulse) (Ignored when other than standby address (3))	

Program example

- The following program starts the machine OPR of the axis 1 when M0 turns ON.

Device	Item	Setting data
D0	OPR type	Machine OPR
D1, D2	Standby address	0 (Ignored)

[Structured ladder/FBD]



[ST]

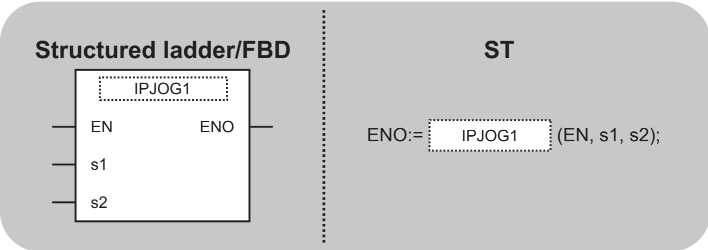
```
MOV(M0, 1, D0);
DMOV(M0, 0, D1);
IPOPR1P(M0, D0);
```

8

JOG start

IPJOG1, IPJOG2

LCPU



The following instruction can go in the dotted squares.
IPJOG1, IPJOG2

■Executing condition

Instruction	Executing condition
IPJOG1 IPJOG2	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s1	Start number of the device in which the control data are stored	Array of ANY16 [0..3]
	s2	Direction specification of the JOG operation 0: Forward RUN 1: Reverse RUN	Bit
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—					
(s2)	○	—	○	—					

Processing details

This instruction starts the JOG operation of the specified axis (refer to the following).

- IPJOG1: Axis 1
- IPJOG2: Axis 2

The JOG operation is executed in the direction specified by (s2), using the JOG speed, JOG acceleration/deceleration time stored in the devices starting from (s1).

Setting data

Device	Item	Setting data	Setting range	Setting side
(s1)[0] (s1)[1]	JOG speed	—	0 to 200000(pulse/s)*1	User
(s1)[2]	JOG acceleration time	—	0 to 32767(ms)	
(s1)[3]	JOG deceleration time	—		

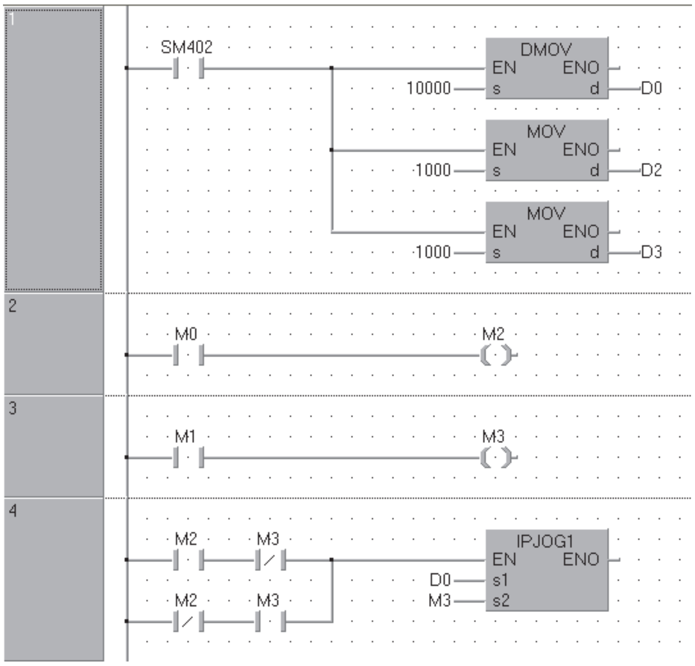
*1 The restricted speed value may be applied when the set value of the JOG speed is not within 0 to 200000.

Program example

- The following program starts the forward JOG operation when M0 turns ON, and starts the reverse JOG operation when M1 turns ON.

Device	Item	Setting data
D0, D1	JOG speed	10000(pulse/s)
D2	JOG acceleration time	1000(ms)
D3	JOG deceleration time	

[Structured ladder/FBD]



[ST]

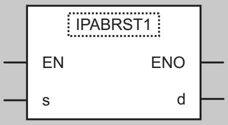
```
DMOV(SM402, 10000, D0);
MOV(SM402, 1000, D2);
MOV(SM402, 1000, D3);
OUT( M0, M2 );
OUT(M1,M3);
IPJOG1((M2 AND (NOT M3)) OR( (NOT M2) AND M3), D0 , M3);
```

Absolute position restoration

IPABRST1, IPABRST2

LCPU

Structured ladder/FBD

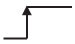


ST

ENO:= IPABRST1 (EN, s, d);

The following instruction can go in the dotted squares.
IPABRST1, IPABRST2

■Executing condition

Instruction	Executing condition
IPABRST1 IPABRST2	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device for input	Array of bit [0..2]
Output argument	ENO	Execution result	Bit
	d	Start number of the device for output	Array of bit [0..2]

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	○	—							
(d)	○	—							

Processing details

This instruction executes the absolute position restoration of the specified axis (refer to the following) by communicating with the servo amplifier using the input device specified by (s) and output device specified by (d).

- IPABRST1: Axis 1
- IPABRST2: Axis 2

Setting data

- Signals imported from servo amplifier

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Signals imported from servo amplifier	ABS send data bit0	0, 1	User
(s)[1]		ABS send data bit1		
(s)[2]		ABS send data ready		

- Signals exported to servo amplifier

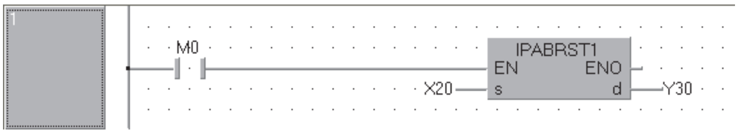
Device	Item	Setting data	Setting range	Setting side
(d)[0]	Signals exported to servo amplifier	Servo ON	—	System
(d)[1]		ABS transfer mode		
(d)[2]		ABS request flag		

Program example

This instruction executes the absolute position restoration of the axis 1 when M0 turns ON.

- X20 to X22: Signals imported from the servo amplifier
- Y30 to Y32: Signals exported to the servo amplifier

[Structured ladder/FBD]



[ST]

```
IPABRST1( M0 , X20, Y30 );
```

8

IPSTOP instruction

IPSTOP1, IPSTOP2

LCPU

The following instruction can go in the dotted squares.
IPSTOP1, IPSTOP2

■Executing condition

Instruction	Executing condition
IPSTOP1 IPSTOP2	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

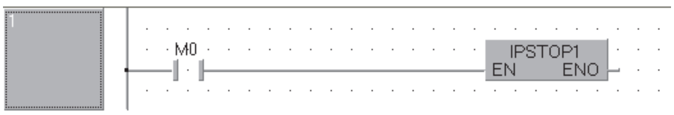
Processing details

This instruction stops the positioning of the specified axis (refer to the following).
- IPSTOP1: Axis 1
- IPSTOP2: Axis 2

Program example

- The following program stops the axis 1 when M0 turns ON.

[Structured ladder/FBD]

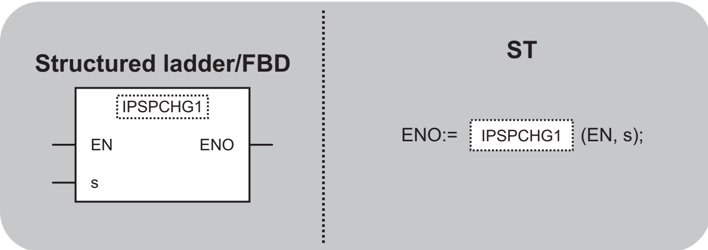


[ST]
IPSTOP1(M0);

Speed change

IPSPCHG1, IPSPCHG2

LCPU



The following instruction can go in the dotted squares.
IPSPCHG1, IPSPCHG1P, IPSPCHG2, IPSPCHG2P

■Executing condition

Instruction	Executing condition
IPSPCHG1 IPSPCHG2	
IPSPCHG1P IPSPCHG2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Start number of the device in which the control data are stored	Array of ANY16 [0..3]
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—					

Processing details

This instruction changes the speed of the specified axis (refer to the following) using the acceleration/deceleration time at speed change, deceleration stop time at speed change, and new speed value stored in the devices starting from (s).

- IPSPCHG1(P): Axis 1
- IPSPCHG2(P): Axis 2

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)[0]	Acceleration/deceleration time at speed change	—	0 to 32767(ms)	User
(s)[1]	Deceleration stop time at speed change	—		
(s)[2]	New speed value	—	0 to 200000 (pulse/s)*1	
(s)[3]				

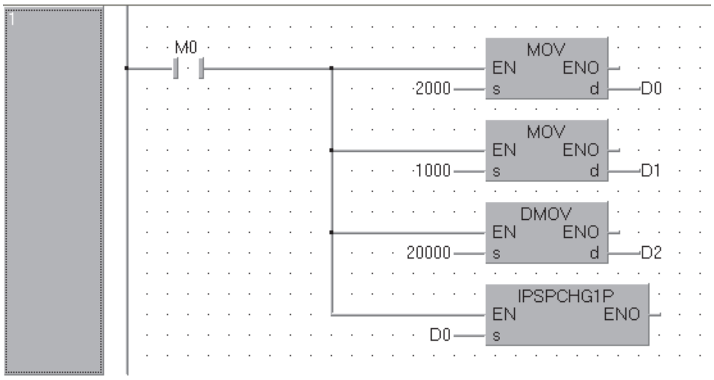
*1 The restricted speed value may be applied when the set value of the new speed is not within 0 to 200000.

Program example

- The following program changes the speed of the axis 1 when M0 turns ON.

Device	Item	Setting data
D0	Acceleration/deceleration time at speed change	2000(ms)
D1	Deceleration stop time at speed change	1000(ms)
D2, D3	New speed value	200000(pulse/s)

[Structured ladder/FBD]



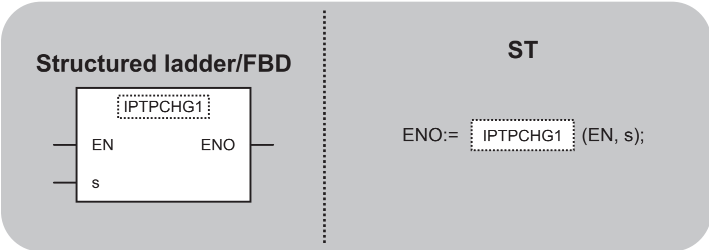
[ST]

```
MOV(M0, 2000, D0);
MOV(M0, 1000, D1);
DMOV(M0, 20000, D2);
IPSPCHG1P( M0 , D0 );
```

Target position change

IPTPCHG1, IPTPCHG2

LCPU



The following instruction can go in the dotted squares.

IPTPCHG1, IPTPCHG1P, IPTPCHG2, IPTPCHG2P

■Executing condition

Instruction	Executing condition
IPTPCHG1 IPTPCHG2	
IPTPCHG1P IPTPCHG2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Target position change value (constant), or start number of the device in which the control data are stored.	ANY32
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s)	—	○		—			○		—

Processing details

This instruction changes the position of the specified axis (refer to the following) to the new target position specified by (s).

- IPTPCHG1(P): Axis 1
- IPTPCHG2(P): Axis 2

Setting data

Device	Item	Setting data	Setting range	Setting side
(s)+0	Target position change value	—	-2147483648 to 2147483647(pulse/s)	User
(s)+1				

Program example

- The following program changes the target position of the axis 1 to 2000 when M0 turns ON.

[Structured ladder/FBD]

[ST]

```
IPTPCHG1P( M0 , 2000 );
```

8.2 Counter Function Dedicated Instruction

Current value read

ICCNTRD1, ICCNTRD2

LCPU

The following instruction can go in the dotted squares.
ICCNTRD1, ICCNTRD1P, ICCNTRD2, ICCNTRD2P

■Executing condition

Instruction	Executing condition
ICCNTRD1 ICCNTRD2	
ICCNTRD1P ICCNTRD2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
—	—								

Processing details

This instruction stores a value at the time of instruction execution to the current value of the specified CH (refer to the following).

Instruction	CH	Device in which the current value is stored
ICCNTRD1(P)	CH1	SD1880, SD1881
ICCNTRD2(P)	CH2	SD1900, SD1901

Program example

- The following program stores the most recent value to the CH 1 current value (SD1880, SD1881) when M0 turns ON.

[Structured ladder/FBD]

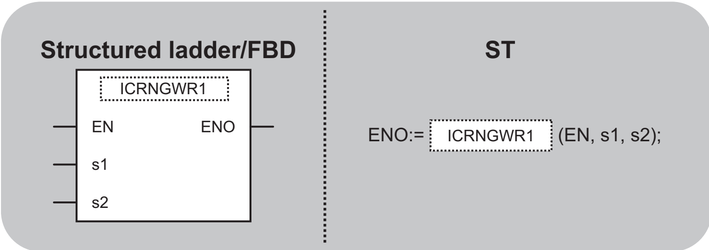
[ST]

```
ICCNTRD1( M0 );
```

Ring counter upper/lower limit value write

ICRNGWR1, ICRNGWR2

LCPU



The following instruction can go in the dotted squares.
ICRNGWR1, ICRNGWR1P, ICRNGWR2, ICRNGWR2P

■Executing condition

Instruction	Executing condition
ICRNGWR1 ICRNGWR2	
ICRNGWR1P ICRNGWR2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s1	Ring counter lower limit value (constant), or start number of the device that stores the ring counter lower limit value • Constant: Settings which is within the range of -2147483648 to 2147483647 and ((s1), (s1)+1) ≤ ((s2), (s2)+1) • Device: Within the range of specified device	ANY32
	s2	Ring counter upper limit value (constant), or start number of the device that stores the ring counter upper limit value • Constant: Settings which is within the range of -2147483648 to 2147483647 and ((s1), (s1)+1) ≤ ((s2), (s2)+1) • Device: Within the range of specified device	ANY32
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—			○		—
(s2)	—	○		—			○		—

Processing details

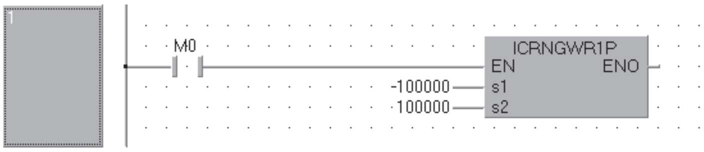
This instruction sets the ring counter lower limit value and the ring counter upper limit value of the specified CH (refer to the following).

- ICRNGWR1(P): CH1
- ICRNGWR2(P): CH2

Program example

- The following program sets -100000 for the ring counter lower limit value and 100000 for the ring counter upper limit value of CH 1 when M0 turns ON.

[Structured ladder/FBD]



[ST]

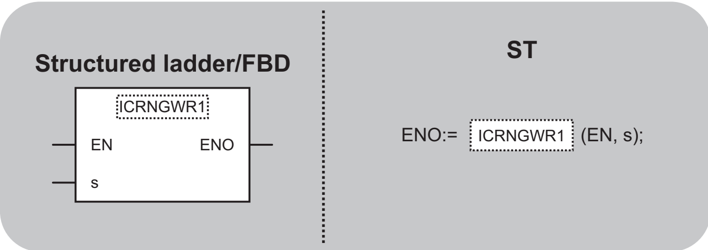
```
ICRNGWR1P( M0 , -100000 , 100000 );
```

8

Preset value write

ICPREWR1, ICPREWR2

LCPU



The following instruction can go in the dotted squares.
ICPREWR1, ICPREWR1P, ICPREWR2, ICPREWR2P

■Executing condition

Instruction	Executing condition
ICPREWR1 ICPREWR2	
ICPREWR1P ICPREWR2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s	Preset value (constant), or start number of the device that stores the preset value • Constant: Settings which is within the range of -2147483648 to 2147483647 • Device: Within the range of specified device	ANY32
Output argument	ENO	Execution result	Bit

<table>
<tr><th rowspan="2">Setting data</th><th colspan="2">Internal device</th><th rowspan="2">R, ZR</th><th colspan="2">J□\□</th><th rowspan="2">U□\G□</th><th rowspan="2">Zn</th><th rowspan="2">Constant</th><th rowspan="2">Others</th></tr>
<tr><th>Bit</th><th>Word</th><th>Bit</th><th>Word</th></tr>
<tr><td>(s)</td><td>—</td><td colspan="2">○</td><td colspan="3">—</td><td colspan="2">○</td><td>—</td></tr>
</table>

Processing details

This instruction sets a preset value of the specified CH (refer to the following).
- ICPREWR1(P): CH1
- ICPREWR2(P): CH2

Program example

- The following program sets 10000 for the preset value of CH 1 when M0 turns ON.

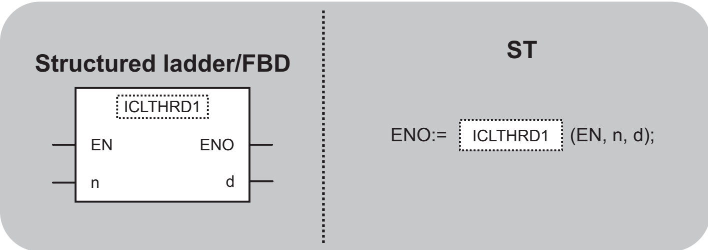
[Structured ladder/FBD]

[ST]
ICPREWR1(M0 , 10000);

Latch counter value read

ICLTHRD1, ICLTHRD2

LCPU



The following instruction can go in the dotted squares.

ICLTHRD1, ICLTHRD1P, ICLTHRD2, ICLTHRD2P

■Executing condition

Instruction	Executing condition
ICLTHRD1 ICLTHRD2	
ICLTHRD1P ICLTHRD2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Latch count value (1,2)	ANY16
Output argument	ENO	Execution result	Bit
	d	Start number of the device in which the latch count value is stored	ANY32

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
n	—	○		—			○		—
(d)	—	○		—			○	—	

Processing details

This instruction stores a latch count value n of the specified CH (refer to the following) to (d).

- ICLTHRD1(P): CH1
- ICLTHRD2(P): CH2

Program example

- The following program stores the latch count value 1 of CH 1 to D100 and D101 when M0 turns ON.

[Structured ladder/FBD]

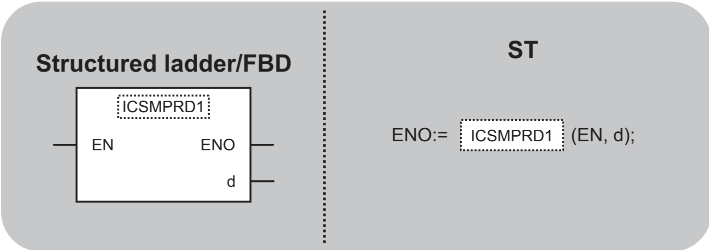
[ST]

ICLTHRD1(M0 , 1 , D100);

Sampling counter value read

ICSMPRD1, ICSMPRD2

LCPU



The following instruction can go in the dotted squares.
ICSMPRD1, ICSMPRD1P, ICSMPRD2, ICSMPRD2P

■Executing condition

Instruction	Executing condition
ICSMPRD1 ICSMPRD2	
ICSMPRD1P ICSMPRD2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
Output argument	ENO	Execution result	Bit
	d	Start number of the device in which the sampling count value is stored	ANY32

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(d)	—	○		—			○	—	

Processing details

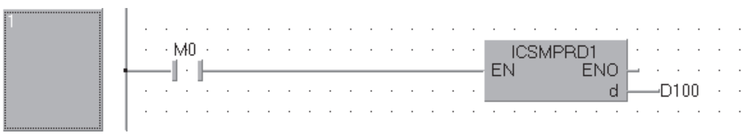
This instruction stores a sampling count value of the specified CH (refer to the following) to (d).

- ICSMPRD1(P): CH1
- ICSMPRD2(P): CH2

Program example

- The following program stores the sampling count value of CH 1 to D100 and D101 when M0 turns ON.

[Structured ladder/FBD]



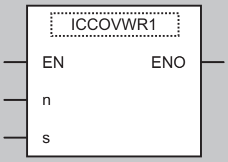
[ST]
ICSMPRD1(M0, D100);

Coincidence output point write

ICCOVWR1, ICCOVWR2

LCPU

Structured ladder/FBD



ST

ENO:= ICCOVWR1 (EN, n, s);

The following instruction can go in the dotted squares.
ICCOVWR1, ICCOVWR1P, ICCOVWR2, ICCOVWR2P

■Executing condition

Instruction	Executing condition
ICCOVWR1 ICCOVWR2	
ICCOVWR1P ICCOVWR2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Coincidence output No. n point (1,2)	ANY16
	s	Coincidence output No. n point (constant), or start number of the device in which coincidence output No. n point is stored • Constant: Settings which is within the range of -2147483648 to 2147483647 • Device: Within the range of specified device	ANY32
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
n	—	○		—			○		—
(s)	—	○		—			○		—

Processing details

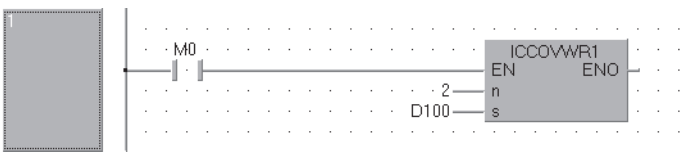
This instruction stores a coincidence output No. n point of the specified CH (refer to the following).

- ICCOVWR1(P): CH1
- ICCOVWR2(P): CH2

Program example

- The following program sets the value of D100 and D101 to the coincidence output No. 2 point of CH 1 when M0 turns ON.

[Structured ladder/FBD]



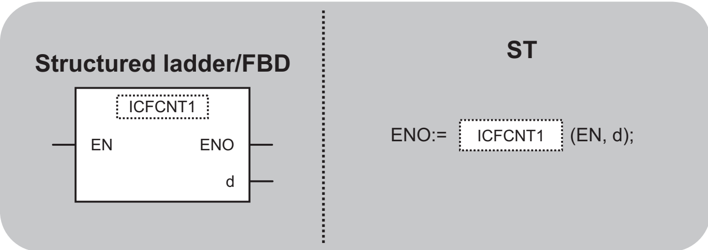
[ST]

```
ICCOVWR1( M0 , 2 , D100 );
```

Frequency measurement

ICFCNT1, ICFCNT2

LCPU



The following instruction can go in the dotted squares.
ICFCNT1, ICFCNT2

■Executing condition

Instruction	Executing condition
ICFCNT1 ICFCNT2	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores the measured frequency value	ANY32

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(d)	—	○		—			○	—	

Processing details

This instruction measures a frequency of the specified CH (refer to the following) according to the settings such as the frequency measurement unit time setting.

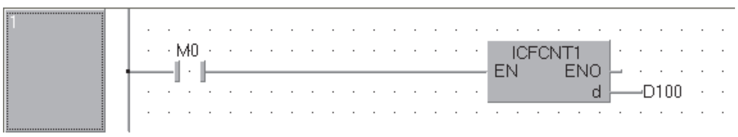
- ICFCNT1: CH1
- ICFCNT2: CH2

The measured value is stored to (d) at the ICFCNT instruction execution. The measurement starts at the rising pulse of the ICFCNT instruction execution command, and ends at the falling pulse.

Program example

- The following program executes the frequency measurement of CH 1 while M0 is ON.

[Structured ladder/FBD]

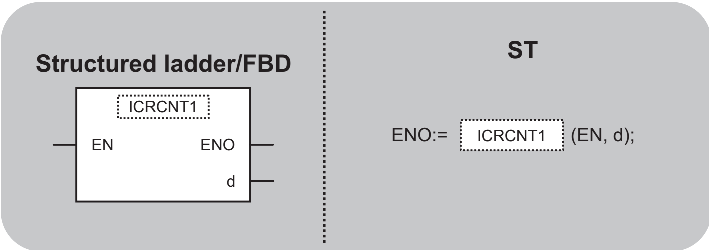


[ST]
ICFCNT1(M0 , D100);

Rotation speed measurement

ICRCNT1, ICRCNT2

LCPU



The following instruction can go in the dotted squares.
ICRCNT1, ICRCNT2

■Executing condition

Instruction	Executing condition
ICRCNT1 ICRCNT2	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores the measured rotation speed	ANY32

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(d)	—	○		—			○	—	

Processing details

This instruction measures a rotation speed of the specified CH (refer to the following) according to the settings such as the rotation speed measurement unit time setting.

- ICRCNT1: CH1
- ICRCNT2: CH2

The measured value is stored to (d) at the ICRCNT instruction execution. The measurement starts at the rising pulse of the ICRCNT instruction execution command, and ends at the falling pulse.

Program example

- The following program stores the rotation speed measurement value of CH 1 to D100 and D101 while M0 is ON.

[Structured ladder/FBD]

[ST]

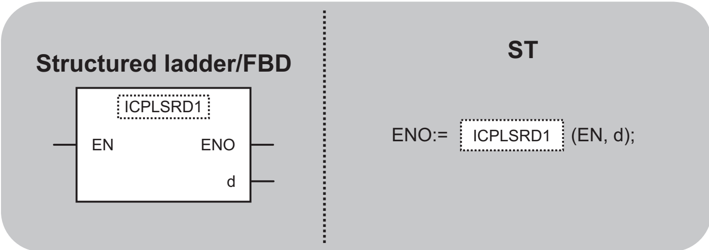
```
ICRCNT1( M0 , D100 );
```


8

Pulse measurement read

ICPLSRD1, ICPLSRD2

LCPU



The following instruction can go in the dotted squares.
ICPLSRD1, ICPLSRD1P, ICPLSRD2, ICPLSRD2P

■Executing condition

Instruction	Executing condition
ICPLSRD1 ICPLSRD2	
ICPLSRD1P ICPLSRD2P	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
Output argument	ENO	Execution result	Bit
	d	Start number of the device that stores the measured pulse value	ANY32

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(d)	—	○		—			○		—

Processing details

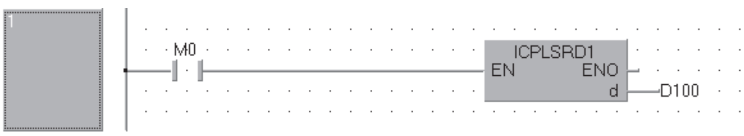
This instruction stores a measured pulse value of the specified CH (refer to the following) to (d).

- ICPLSRD1(P): CH1
- ICPLSRD2(P): CH2

Program example

- The following program stores the measured pulse value of CH 1 to D100 and D101 when M0 turns ON.

[Structured ladder/FBD]



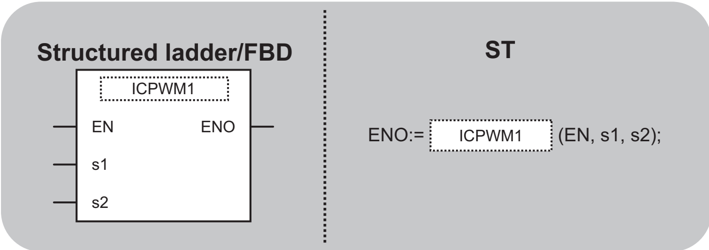
[ST]

ICPLSRD1(M0 , D100);

PWM output

ICPWM1, ICPWM2

LCPU



The following instruction can go in the dotted squares.
ICPWM1, ICPWM2

■Executing condition

Instruction	Executing condition
ICPWM1 ICPWM2	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	s1	PWM output ON time setting value (constant), or start number of the device that stores the PWM output ON time setting value • Constant: Settings which is 0 or within the range of 10 to 10^7 (0.1μs) and ((s1), (s1)+1) ≤ ((s2), (s2)+1) • Device: Within the range of specified device	ANY32
	s2	PWM output cycle time setting value (constant), or start number of the device that stores the PWM output cycle time setting value • Constant: Settings which is 0 or within the range of 50 to 10^7 (0.1μs) and ((s1), (s1)+1) ≤ ((s2), (s2)+1) • Device: Within the range of specified device	ANY32
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word				
(s1)	—	○		—			○		—
(s2)	—	○		—			○		—

Processing details

This instruction outputs a PWM waveform of the specified CH (refer to the following).

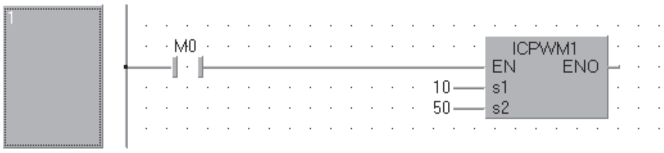
- ICPWM1: CH1
- ICPWM2: CH2

The PWM waveform with the ON time ((s1)) and the cycle time ((s2)) is output from the coincidence output No.1 signal during the ICPWM instruction execution. The output of the PWM waveform starts from OFF.

Program example

- The following program outputs the PWM waveform with 1μs ON time and 5μs cycle time from CH 1 while M0 is ON.

[Structured ladder/FBD]



[ST]

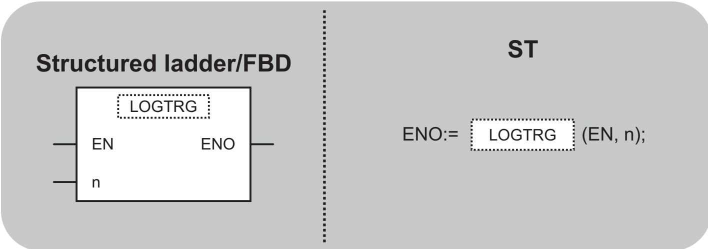
```
ICPWM1( M0 , 10 , 50 );
```

9 DATA LOGGING FUNCTION INSTRUCTION

9.1 Trigger Logging Set/Reset

LOGTRG Instruction, LOGTRGR Instruction

QnUDV LCPU



The following instruction can go in the dotted squares.
LOGTRG, LOGTRGR

■Executing condition

Instruction	Executing condition
LOGTRG	
LOGTRGR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n	Data logging configuration number	ANY16
Output argument	ENO	Execution result	Bit

Setting data	Internal device		R, ZR	J□\□		U□\G□	Zn	Constant	Others
	Bit	Word		Bit	Word			K, H	
n	—	○		—			○		—

Processing details

■LOGTRG

- The LOGTRG instruction generates a trigger in the trigger logging of the data logging configuration number specified by 'n'.
- A value from 1 to 10 is set for 'n'.
- When the LOGTRG instruction is executed, the special relay (data logging trigger) of the data logging configuration number specified by 'n' turns ON. After executing the trigger logging for the number of times set for "Number of records", the instruction latches the data and stops the trigger logging.
- Validated when "When trigger instruction executed" is selected as the trigger condition.
- No processing is performed with the following condition.
  - Specifying a data logging configuration number for which other than "When trigger instruction executed" is specified as the trigger condition.
  - Specifying a data logging configuration number which is not configured.
  - Specifying a data logging configuration number which is currently used for continuous logging.
  - Executing the LOGTRG instruction again without executing the LOGTRGR instruction after the LOGTRG instruction.

■LOGTRGR

- The LOGTRGR instruction resets the LOGTRG instruction of the specified data logging configuration number.
- When the LOGTRGR instruction is executed, the special relay (data logging trigger, trigger logging complete) of the data logging configuration number specified by 'n' turns OFF.
- When the instruction is executed while transferring data in the buffer memory to the SD memory card, the instruction process is held until data transfer is complete.

Operation error

- In the following case, an operation error occurs, the error flag (SM0) is turned ON, and the corresponding error code is stored to SD0.

Error code	Description
4100	The value for n is outside the range of 1 to 10

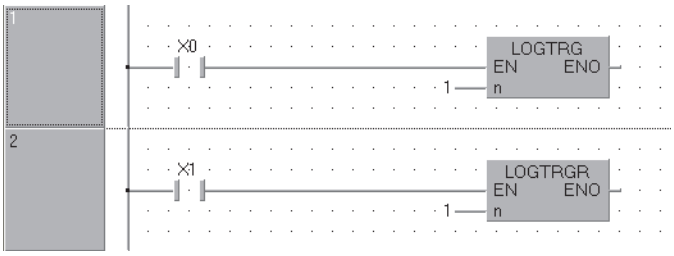
Precautions

- Use the LCPU other than L02SCPU and L02SCPU-P.

Program example

- The following program executes the LOGTRG instruction on the data logging configuration No. 1 when X0 turns ON, and resets the trigger condition with the LOGTRGR instruction when X1 turns ON.

[Structured ladder/FBD]



[ST]

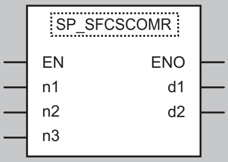
```
LOGTRG(X0,1);
LOGTRGR(X1,1);
```

10 SFC CONTROL INSTRUCTION

10.1 SFC Step Comment Read

S(P)_SFCSCOMR

High performance | Process | Redundant | Universal

Structured ladder/FBD	ST
	ENO:= SP_SFCSCOMR (EN, n1, n2, n3, d1, d2);

The following instruction can go in the dotted squares.
S_SFCSCOMR, SP_SFCSCOMR

■Executing condition

Instruction	Executing condition
S_SFCSCOMR	
SP_SFCSCOMR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n1	Block No. of an SFC program that read comments or device number where block No. is stored.	ANY16
	n2	The device number where the number of comments to read or the number of comments is stored.	ANY16
	n3	The number of comments to read in a single scan or device number where the number of comments is stored.	ANY16
Output argument	ENO	Execution result	Bit
	d1	The first number of device that stores comment read.	ANY16
	d2	A device that turns ON for 1 scan at completion of the instruction.	Bit

Setting data	Internal device		R	J□\□ U□\G□ Zn	Constant K, H	Expansion SFC BLm\Sn	Others	Sequence Program	SFC Program		Execution Site		
	Bit	Word							Step	Transition Condition	Block	Step	Transition Condition
n1	—	○		—	○	—		○		—	○	—	—
n2	—	○		—	○	—		○		—		—	—
n3	—	○		—	○	—		○		—		—	—
(d1)	—	○*1		—	—	—		○		—		—	—
(d2)	○*1	—		—	—	—		○		—		—	—

*1 Local device cannot be used.

Processing details

This function reads step comments being activated in the SFC block specified at (n1), by the number of comment specified at (n2), and stores those to the device number of after specified at (d1).

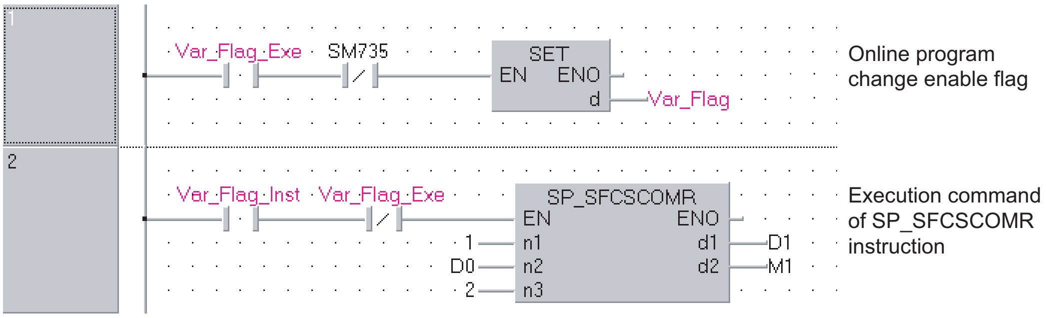
Precautions

- For High Performance model QCPU, use the function version is B or later and the first five digits of the serial number are '07012' or higher.
- For Process CPU and Redundant CPU, use the first five digits of the serial number are '07032' or higher.
- For Universal CPU, use the first five digits of the serial number are '12052' or higher. Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU can not be used.

Program example

- This program reads 2 comments being activated at the SFC block No.1 when X1 is turned ON, and stores those to the storage device after D0. (The number of comment to be read (n3) in a single scan is also set in 2.)

[Structured ladder/FBD]



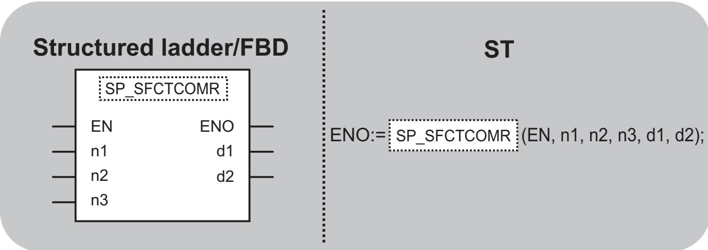
[ST]

```
IF((Var_Flag_Exe=TRUE) & (SM735=FALSE))THEN (*Online program change execution command*)
    SET(TRUE, Var_Flag); (*Online program change enable flag*)
END_IF;
IF((Var_Flag_Inst=TRUE) & (Var_Flag=FALSE))THEN (*Execution command of SP_SFCSCOMR instruction*)
    SP_SFCSCOMR(TRUE, 1, D0, 2, D1, M1);
END_IF;
```

10.2 SFC Transition Condition Comment Read

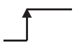
S(P)_SFCTCOMR

High performance | Process | Redundant | Universal



The following instruction can go in the dotted squares.
S_SFCTCOMR, SP_SFCTCOMR

■Executing condition

Instruction	Executing condition
S_SFCTCOMR	
SP_SFCTCOMR	

■Argument

Input/output argument	Name	Description	Data type
Input argument	EN	Executing condition	Bit
	n1	Block No. of an SFC program that read comments or device number where block No. is stored.	ANY16
	n2	The device number where the number of comments to read or the number of comments is stored.	ANY16
	n3	The number of comments to read in a single scan or device number where the number of comments is stored.	ANY16
Output argument	ENO	Execution result	Bit
	d1	The first number of device that stores comment read.	ANY16
	d2	A device that turns ON for 1 scan at completion of the instruction.	Bit

Setting data	Internal device		R	J□\□ U□\G□ Zn	Constant K, H	Expansion SFC BLm\Sn	Others	Sequence Program	SFC Program		Execution Site		
	Bit	Word							Step	Transition Condition	Block	Step	Transition Condition
n1	—	○		—	○	—		○		—	○	—	—
n2	—	○		—	○	—		○		—		—	—
n3	—	○		—	○	—		○		—		—	—
(d1)	—	○*1		—	—	—		○		—		—	—
(d2)	○*1	—		—	—	—		○		—		—	—

*1 Local device cannot be used.

Processing details

This function reads comments of the transition condition 1 associated with steps activated in the SFC block specified at (n1) with the number of comments specified at (n2), and stores those to the device number of after specified at (d1).

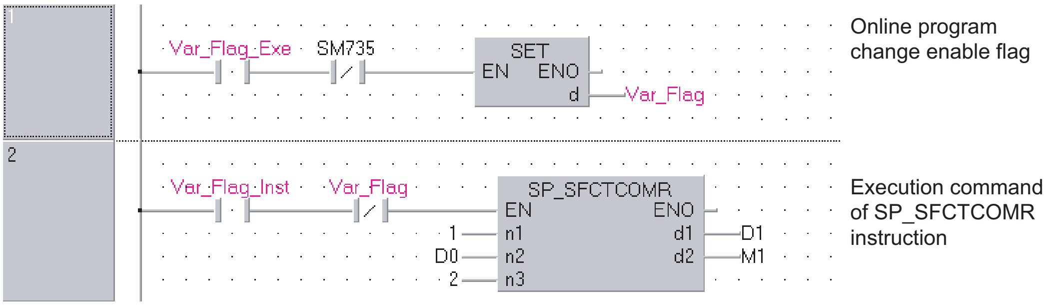
Precautions

- For High Performance model QCPU, use the function version is B or later and the first five digits of the serial number are '07012' or higher.
- For Process CPU and Redundant CPU, use the first five digits of the serial number are '07032' or higher.
- For Universal CPU, use the first five digits of the serial number are '12052' or higher. Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU can not be used.

Program example

- This program reads 2 comments being activated at the SFC block No.1 when X1 is turned ON, and stores those to the storage device after D0. (The number of comment to be read (n3) in a single scan is also set in 2.)

[Structured ladder/FBD]



[ST]

```
IF((Var_Flag_Exe=TRUE) & (SM735=FALSE))THEN (*Online program change execution command*)
    SET(TRUE, Var_Flag); (*Online program change enable flag*)
END_IF;
IF((Var_Flag_Inst=TRUE) & (Var_Flag=FALSE))THEN (*Execution command of SP_SFCTCOMR instruction*)
    SP_SFCTCOMR(TRUE, 1, D0, 2, D1, M1);
END_IF;
```

MEMO

INDEX

MEMO

I

INSTRUCTION INDEX

REVISIONS

*The manual number is given on the bottom left of the back cover.

Revision date	*Manual number	Description
July, 2008 ⋮ June, 2013	SH(NA)-080785ENG-A ⋮ SH(NA)-080785ENG-K	Due to the transition to the e-Manual, the details of revision have been deleted.
February, 2017	SH(NA)-080785ENG-L	Complete revision (layout change)
September, 2018	SH(NA)-080785ENG-M	Descriptions regarding the QnUDPVCPU is added.

Japanese manual version SH-080738-Q

WARRANTY

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

(1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.

(2) Even within the gratis warranty term, repairs shall be charged for in the following cases.

1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
2. Failure caused by unapproved modifications, etc., to the product by the user.
3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

2. Onerous repair term after discontinuation of production

(1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.

(2) Product supply (including repair parts) is not available after production is discontinued.

3. Overseas service

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

4. Exclusion of loss in opportunity and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

(1) Damages caused by any cause found not to be the responsibility of Mitsubishi.

(2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.

(3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.

(4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

5. Changes in product specifications

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

TRADEMARKS

Ethernet is a registered trademark of Fuji Xerox Co., Ltd. in Japan.
Microsoft, Microsoft Access, Excel, SQL Server, Visual Basic, Visual C++, Visual Studio, Windows, Windows NT, Windows Server, Windows Vista, and Windows XP are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.
In some cases, trademark symbols such as '™' or '®' are not specified in this manual.

SH(NA)-080785ENG-M

SH(NA)-080785ENG-M(1809)KWIX
MODEL: Q-KP-TM-E
MODEL CODE: 13JW09

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN