

HITACHI INVERTER

SJ700-2 SERIES EASY-SEQUENCE
PROGRAMMING SOFTWARE EzSQ

INSTRUCTION MANUAL

Read through this Instruction Manual, and keep it handy for future reference.

NT***X

HITACHI

Introduction

Introduction

Thank you for purchasing the Hitachi SJ700 Series Inverter.

This Instruction Manual explains how to use the easy-sequence programming software (EzSQ) for the Hitachi SJ700 Series Inverter. Be sure to read this Instruction Manual carefully before using EzSQ, and keep it on hand for future reference.

Before creating user programs for the inverter, also refer to the SJ700 Inverter Instruction Manual for the necessary related knowledge, and ensure you understand and follow all safety information, precautions, and operating and handling instructions for the correct use of the inverter.

Always use the inverter strictly within the range of specifications described in the Inverter Instruction Manual and correctly implement maintenance and inspections to prevent faults from occurring.

When using the inverter together with optional products, also read the manuals for those products. Note that this Instruction Manual and the manual for each optional product to be used should be delivered to the end user of the inverter.

Handling of this Instruction Manual

- The contents of this Instruction Manual are subject to change without prior notice.
- Even if you lose this Instruction Manual, it will not be resupplied, so please keep it carefully.
- No part of this Instruction Manual may be reproduced in any form without the publisher's permission.
- If you find any incorrect description, missing description or have a question concerning the contents of this Instruction Manual, please contact the publisher.

Revision History

No.	Revision content	Date of issue	Manual code

- The current edition of this Instruction Manual also includes some corrections of simple misprints, missing letters, misdescriptions and certain added explanations other than those listed in the above Revision History table.

Safety Instructions

Be sure to read this Instruction Manual, SJ700 Instruction Manual, and appended documents thoroughly before using EzSQ and the inverter.


In these Instruction Manuals, safety instructions are classified into two levels: WARNING and CAUTION.



: Indicates that incorrect handling may cause hazardous situations, which may result in serious personal injury or death.



: Indicates that incorrect handling may cause hazardous situations, which may result in moderate or slight personal injury or physical damage alone.

Note that even a  level situation may lead to a serious consequence according to circumstances. Be sure to follow every safety instruction, which contains important safety information. Also focus on and observe the items and instructions described under "Notes" in the text.

WARNING

During trial operation of the inverter with a user program, a user program error may cause the motor driven by the inverter to run uncontrollably. Be sure to implement safety measures such as the emergency stop mechanism in your system before trial operation. Otherwise, system failure or personal injury may result.

CAUTION

To debug a user program, first conduct a trial operation of the inverter with an independent motor to confirm that the motor does not run uncontrollably. After that, install the motor in your system (machine), and start system operation. Otherwise, system failure or personal injury may result.

Contents

Chapter 1 Introduction

1.1	Installing and Uninstalling EzSQ	1 - 1
1.1.1	Installing EzSQ	1 - 1
1.1.2	Uninstalling EzSQ	1 - 3
1.2	Preparing for Programming	1 - 5

Chapter 2 Creation and Execution of a User Program

2.1	Language Specifications	2 - 1
2.2	Overview of System Configuration	2 - 2
2.3	General Flow of Operation and Setup	2 - 3
2.4	Creation of User Program	2 - 3
2.5	Syntax Check	2 - 3
2.6	Settings on the Inverter	2 - 4
2.7	Execution of User Program	2 - 5

Chapter 3 Syntax

3.1	Code Description Format	3 - 1
3.1.1	Line	3 - 1
3.1.2	"Label" field	3 - 1
3.1.3	"Mnemonic" field	3 - 1
3.1.4	"Parm 1 to 6" fields	3 - 1
3.1.5	"Comment" field	3 - 1
3.2	Data Description Format	3 - 2
3.2.1	"Variable" field	3 - 2
3.2.2	"Define" field	3 - 2
3.2.3	"Answer" field	3 - 2
3.2.4	"Comment" field	3 - 2
3.3	Variables and Ranges of Numerical Values	3 - 2
3.4	Operators	3 - 3
3.5	Conditions	3 - 3

Chapter 4 How To Use EzSQ

4.1	Starting and Terminating EzSQ	4 - 1
4.1.1	Starting EzSQ	4 - 1
4.1.2	Terminating EzSQ	4 - 2
4.2	Flow of Programming	4 - 3
4.3	File Management	4 - 4
4.4	Methods of Entering a Program in the "Code Window"	4 - 6
4.5	Program Editing Functions	4 - 10
4.6	Methods of Entering Data in the "Data Window"	4 - 12

4.7	Compiling a Program	4 - 14
4.8	Downloading a Program to the Inverter	4 - 15
4.9	Saving a Program in EEPROM	4 - 16
4.10	Compilation and Downloading	4 - 16
4.11	Uploading a Program from the Inverter	4 - 17
4.12	Reset	4 - 17
4.13	"Monitor Window"	4 - 18
4.14	Printing a Program	4 - 19
4.15	Referencing the Help Information	4 - 19
4.16	Version Information	4 - 19
4.17	Setting/Clearing a Password	4 - 20
	4.17.1 Setting a Password	4 - 20
	4.17.2 Clearing a Password	4 - 20

Chapter 5 Instruction Words

5.1	List of Instructions	5 - 1
5.2	Program Control Instructions	5 - 4
	entry and end statements	5 - 4
	sub and end sub statements	5 - 4
	goto statement	5 - 5
	on trip goto statement	5 - 5
	ifs-then-else-end if statements	5 - 6
	if statement	5 - 7
	for-next loop statements	5 - 8
	while loop statement	5 - 9
	until loop statement	5 - 10
	select case syntax statement	5 - 10
	call statement	5 - 11
	inc statement	5 - 12
	dec statement	5 - 12
	Label definition statement	5 - 13
	wait statement	5 - 13
5.3	Operators	5 - 14
5.4	Conditional Expressions	5 - 14
5.5	Input/Output Control Instructions	5 - 15
	X () or Xw (contact input)	5 - 15
	Y () or Yw (contact output)	5 - 16
	UB () or UBw (internal user contact control)	5 - 18
5.6	Timer Control Instructions	5 - 19
	timer set (timer-start instruction)	5 - 20
	timer off (timer-stop instruction)	5 - 21
	delay on or delay off (delay operation instruction)	5 - 21
5.7	Inverter Control Instructions	5 - 23
	Inverter operation command	5 - 23
	Inverter operation monitoring instruction	5 - 24
	user monitor	5 - 25
	user trip	5 - 25
	stop statement	5 - 26

Contents

chg param statement	5 - 26
mon param statement	5 - 26
5.8 Other Reserved Variables	5 - 28
U (00) to U (31)	5 - 28
UL (00) to UL (03)	5 - 28
SET-Freq	5 - 29
ACCEL	5 - 30
DECEL	5 - 31
XA (0) to XA (2)	5 - 32
YA (0) to YA (2)	5 - 33
TC (0) to TC (7)	5 - 34
TD (0) to TD (7), TDw	5 - 35
FM	5 - 36
Iout	5 - 37
Dir	5 - 38
PID-FB	5 - 39
F-CNV	5 - 39
Tmon	5 - 40
Vout	5 - 41
Power	5 - 41
PlsCnt	5 - 42
POS	5 - 42
STATUS	5 - 43
DCV	5 - 43
RUN-Time	5 - 44
ON-Time	5 - 44
ERR CNT	5 - 45
ERR (1) to ERR (6)	5 - 45

Chapter 6 Interface with the Inverter

6.1 Inverter Settings Related to the Easy Sequence Function	6 - 1
6.2 Switching of Operation	6 - 2
6.2.1 Easy sequence function selection (A017)	6 - 2
6.3 Switching of Input/Output Terminals	6 - 2
6.3.1 Program run signal input terminal (PRG terminal)	6 - 2
6.3.2 General-purpose contact input terminals	6 - 2
6.3.3 General-purpose contact output terminals	6 - 3
6.3.4 General-purpose analog input terminal (O terminal)	6 - 3
6.3.5 General-purpose analog input terminal (OI terminal)	6 - 3
6.3.6 General-purpose analog input terminal (O2 terminal)	6 - 4
6.3.7 General-purpose analog output terminal (FM terminal)	6 - 4
6.3.8 General-purpose analog output terminal (AM terminal)	6 - 5
6.3.9 General-purpose analog output terminal (AMI terminal)	6 - 5
6.4 Switching of Command Input Device	6 - 6
6.4.1 Frequency command selection (A001)	6 - 6
6.4.2 Operation command selection (A002)	6 - 6
6.4.3 Acceleration/deceleration time input selection (P031)	6 - 6
6.5 Others	6 - 7

6.5.1 User-defined variables "U (00)" to "U (31)" (P100 to P131)6 - 7

Chapter 7 Errors and Troubleshooting

7.1 Errors Specific to the Easy Sequence Function7 - 1
7.2 Troubleshooting7 - 2

Chapter 8 Appendix

8.1 Inverter Parameters and Available Settings8 - 1

Chapter 1 Introduction

This chapter explains how to install EzSQ, uninstall it, and prepare for programming with EzSQ.

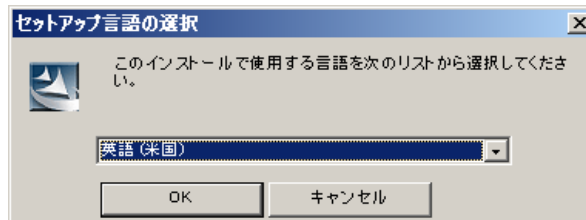
1.1	Installing and Uninstalling EzSQ	1 - 1
1.2	Preparing for Programming	1 - 5

1.1 Installing and Uninstalling EzSQ

1.1.1 Installing EzSQ

This section explains how to install the easy sequence programming software EzSQ.

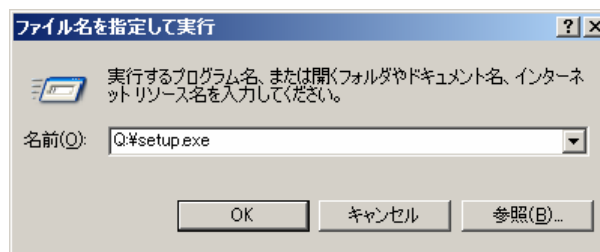
- (1) Insert the EzSQ installation CD-ROM into the CD-ROM drive of your personal computer.
- (2) With the CD-ROM drive in auto-starting mode, opening the "Select the Setup Language" window automatically starts up the EzSQ Installer.



Select "English(U.S)" and click the [OK] button.

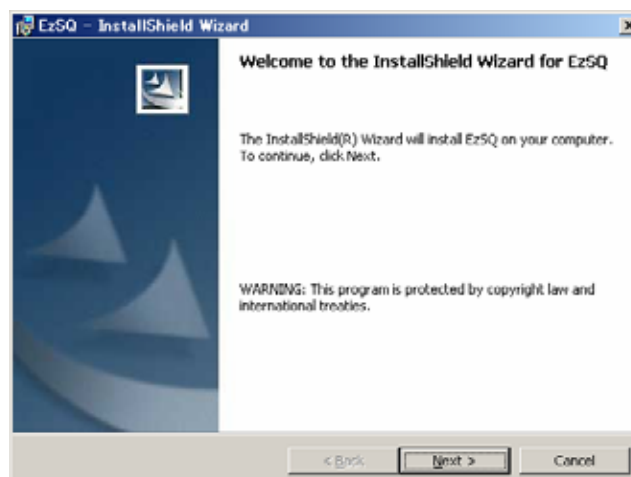
Note: If the CD-ROM drive is not in auto-starting mode, perform one of the following three operations to start up the EzSQ Installer:

- 1) Start Microsoft Explorer, browse the files on the CD-ROM drive, and then select and open the "setup.exe" file.
- 2) Select "Run" from the "Start" menu of Windows. Click the [Browse] button in the "Run" dialog box to open the "Browse File" dialog box, select "setup.exe" among the files on the CD-ROM drive, and then click the [Open] button.
- 3) Select "Run" from the "Start" menu of Windows. Enter "Q:¥setup.exe" in the "File name" field of the "Run" dialog box, and then click the [OK] button.



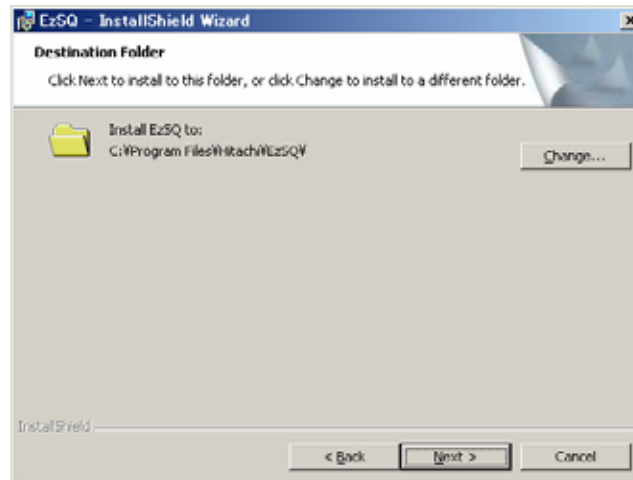
("Q" in the file name indicates the CD-ROM drive name that varies depending on the personal computer.)

- (3) The above operation starts up the EzSQ installation wizard ("EzSQ - InstallShield Wizard"). Click the [Next] button as instructed in the wizard window.

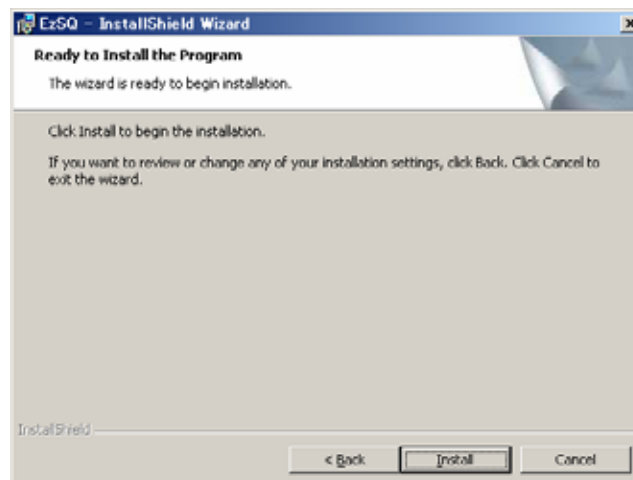


Chapter 1 Introduction

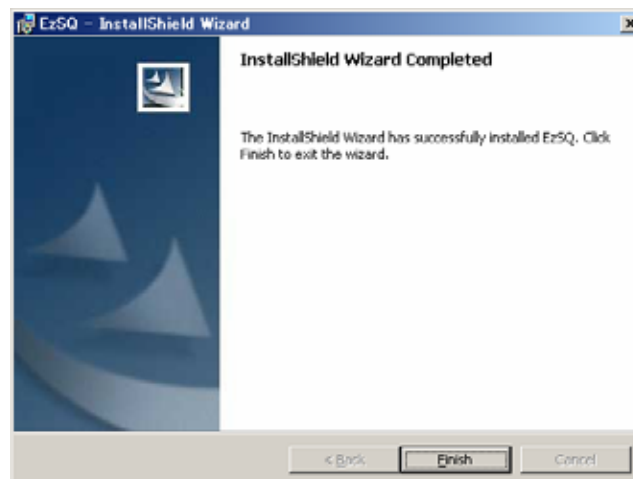
- (4) The wizard requests you to select the folder to install EzSQ.
If you don't need to change the default setting, click the [Next] button.
* To change the installation-destination folder from the default, click the [Change] button, and then select a desired folder.



- (5) The wizard indicates that it is ready to start installation. Click the [Install] button.



- (6) The wizard indicates that installation has been completed.
Click the [Finish] button to exit the wizard.



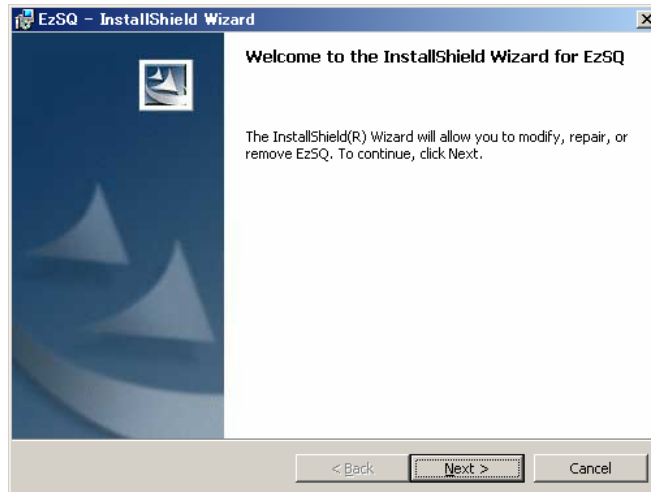
1.1.2 Uninstalling EzSQ

This section explains how to uninstall EzSQ. To uninstall EZSQ, use one of the following two methods:

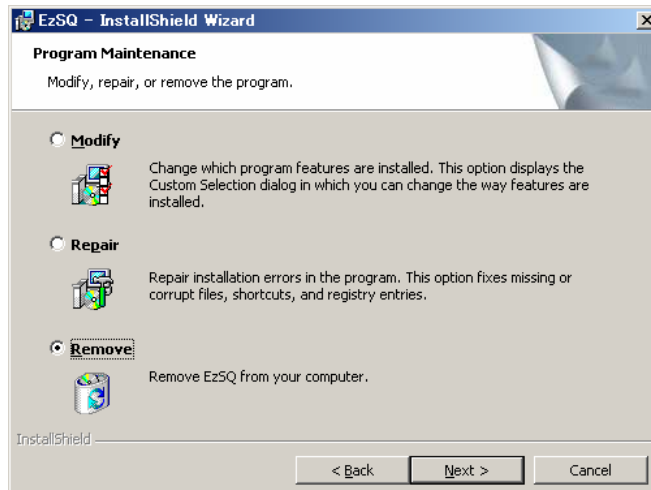
- Delete EzSQ by using "Add/Remove Programs" in the Control Panel.
- Delete EzSQ by using the EzSQ - InstallShield Wizard (on the EzSQ Installation CD-ROM).

To uninstall EzSQ by using the EzSQ - InstallShield Wizard, follow the procedure below.

- (1) Perform installation steps (1) and (2) described above to start up the installation wizard.
Click the [Next] button as instructed in the wizard window.

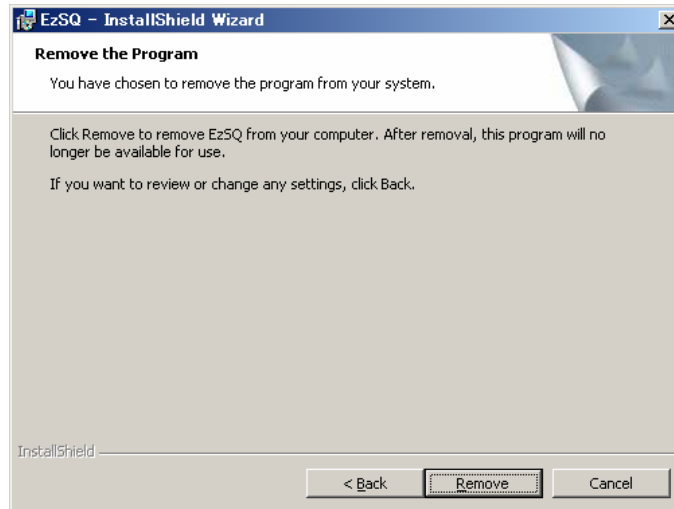


- (2) The wizard shows program maintenance options.
Select "Remove" and click the [Next] button.

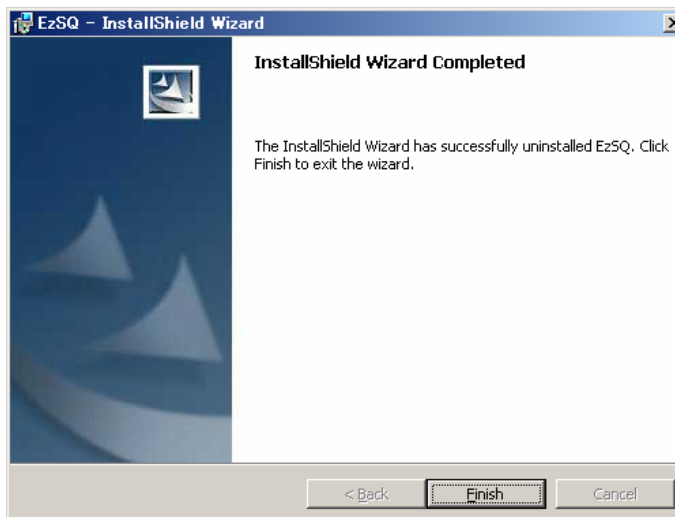


Chapter 1 Introduction

- (3) The wizard requests you to confirm program deletion.
Click the [Remove] button.



- (4) The wizard indicates that program deletion has been completed.



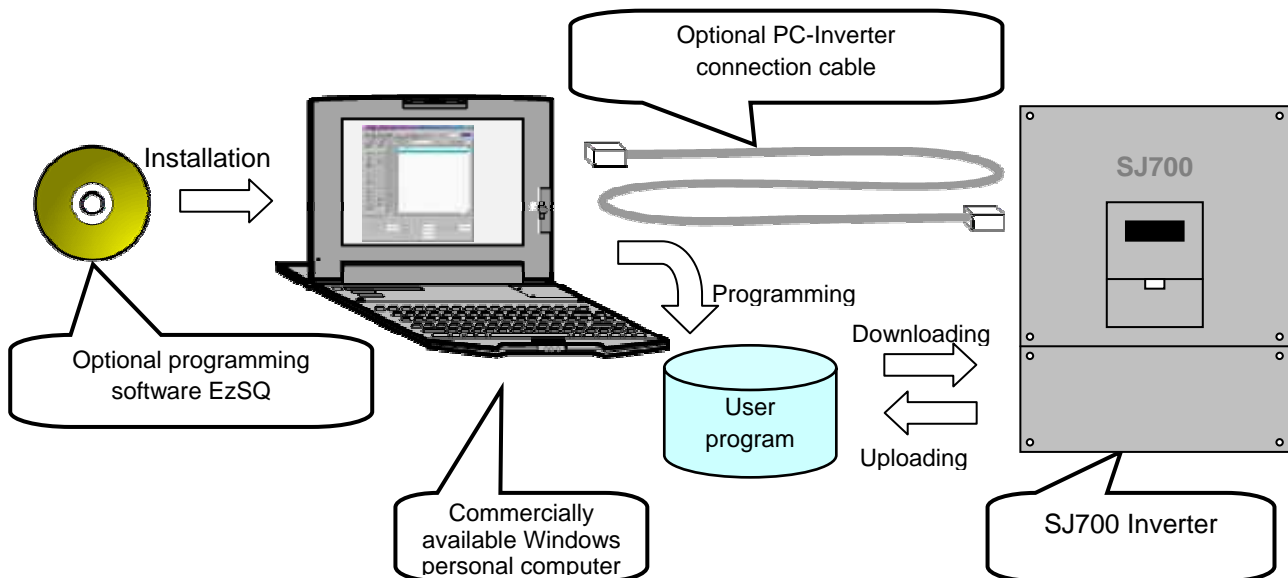
Uninstallation of EzSQ has been completed. Click the [Finish] button to exit the wizard.

1.2 Preparing for Programming

To create user programs with the easy sequence function of the SJ700 inverter, you must prepare the following devices and software:

- (1) SJ700 inverter
- (2) Personal computer (PC) (Windows system)
- (3) Optional programming software EzSQ
- (4) Optional PC-inverter connection cable
 - PC port: serial RS232C port
 - Inverter port: Operator-connection port

The following figure shows the basic system configuration for programming.



- Install EzSQ on your Windows personal computer, and connect the personal computer to the SJ700 inverter via the PC-inverter connection cable.
- After completing these preparations, you can operate EzSQ to create a user program and download it to the SJ700 inverter.

Chapter 2 Creation and Execution of a User Program

This chapter explains the general procedures for creating and executing a user program.

2.1	Language Specifications	2 - 1
2.2	Overview of System Configuration.....	2 - 2
2.3	General Flow of Operation and Setup.....	2 - 3
2.4	Creation of User Program	2 - 3
2.5	Syntax Check	2 - 3
2.6	Settings on the Inverter	2 - 4
2.7	Execution of User Program	2 - 5

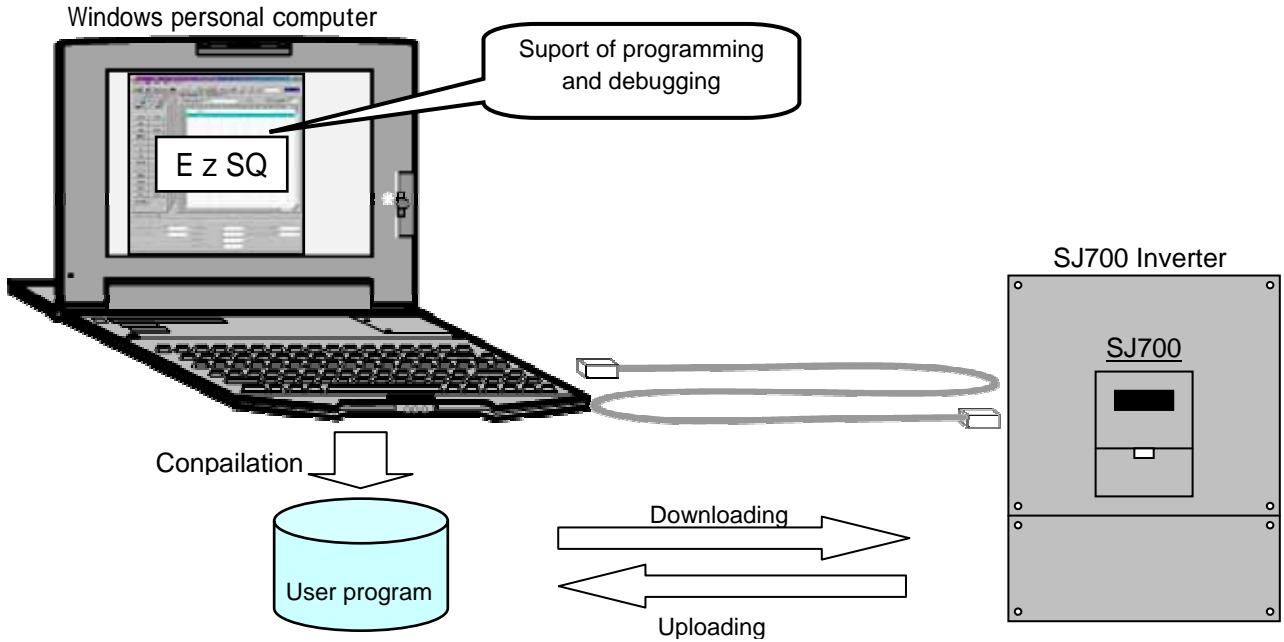
2.1 Language Specifications

The table below lists the programming-related specifications of the easy sequence function.

Item		Specification	
Language specification	Programming language	Basic-like language	
	Input device	Windows (DOS/V) personal computer (OS: Windows 98SE, Windows 2000, or Windows XP)	
	Max. program size	1024 steps (The internal storage capacity of the inverter is 1024 steps or 6 kilobytes.)	
	Programming support function (programming software)	- Editing (on Windows) - Display (on Windows) - Program syntax check (on Windows) - Downloading, uploading, and full clearance of program	
	Execution format	Execution by interpreter in an execution cycle of 2 ms per instruction (possible subroutine call with nesting in up to 8 layers)	
Input/output-related functions	External input	Contact signal	24 V open-collector input (using intelligent input terminals)
		Program run signal input	Always assigned to the FW terminal
		General-purpose input terminals	Up to 8 terminals (X (00) to X (07))
		General-purpose analog input	XA (0): 0 to 10 V (O terminal) XA (1): 4 to 20 mA (OI terminal) XA (2): 0 to 10 V (O2 terminal)
	External output	General-purpose output terminal	Up to 6 terminals (Y (00) to Y (05))
		General-purpose analog output	YA (0): Assignable to the FM terminal
			YA (1): Assignable to the AM terminal
			YA (2): Assignable to the AMI terminal
Reserved words	Instructions	(1) Program control instructions - Loop (for) - Unconditional branching (goto) - Conditional branching (if then, ifs then, select case, until, and while) - Time control (wait) - Subroutine (call, sub) - Others (entry, end, cont, inc, and dec)	
		(2) Arithmetic instructions - Arithmetic operation (+, -, *, /) - Remainder (mod) - Substitution (=) - Absolute value (abs) - Logic operation (or, and, xor, and not)	
		(3) Input/output control - General-purpose input/output (bit input, word input, bit output, and word output) - Reading of inverter input terminal	
		(4) Timer control - Delay operation - Timer control	
		(5) Parameter control - Rewriting of parameters by reselecting code on the operator's display	
	Number of variables	User-defined variable	U (00) to U (31) (32 variables)
		Internal user variable	UL (00) to UL (07) (8 variables)
		Set frequency	SET-Freq
		Acceleration time	ACCEL
		Deceleration time	DECEL
Monitoring variable		FM, Iout, Dir, PID-FB, F-CNV, Tmon, Vout, Power, RUN-Time, ON-Time, PlsCnt, POS, STATUS,DCV, ERR CNT, ERR (1), ERR (2), ERR (3), ERR (4), ERR (5), and ERR (6)	
General-purpose input contact		X (00) to X (07) (8 contacts)	
General-purpose output contact		Y (00) to Y (05) (6 contacts) (including a relay contact output)	
Internal user contact		UB (00) to UB (07) (8 contacts)	
Internal timer contact	TD (0) to TD (7) (8 contacts)		
Inverter input/output	Specification by code on the remote operator's display		

2.2 Overview of System Configuration

Programming software EzSQ enables you to edit and compile a user program that will use the easy sequence function of the inverter, download it from the personal computer to the inverter, and upload it from the inverter to the personal computer.



The table below lists the main functions of EzSQ.

Function	Description
Programming support	Supports the input, editing, saving, reading, and printing of user programs
Compilation	Compiles an edited user program
Downloading and uploading	Downloads a user program to the inverter Uploads a user program to from the inverter
Debugging support	Monitors program execution, inverter status, and others

2.3 General Flow of Operation and Setup

A general flow of operations from programming to program execution with the easy sequence function is as follows:

- (1) Create and compile a user program with the Program Editor (EzSQ).
- (2) Download the compiled user program to the inverter, and save it in EEPROM.
- (3) Configure the parameters required for the easy sequence function in the inverter.
- (4) Enable the easy sequence function (set "01" in parameter "A017").
- (5) Turn on the FW terminal (PRG signal) to execute the user program.

After having downloaded the user program to the inverter, you can disconnect the inverter from the personal computer and execute the easy-sequence program (i.e., user program) on the inverter alone. If the downloaded user program is saved in internal EEPROM of the inverter, you can execute the user program even after resetting the inverter power. If the downloaded user program is not saved in EEPROM, the user program will be deleted when the inverter power is fully shut off. You are recommended not to save a created user program when downloading it to the inverter for debugging purposes. You should save the user program when downloading it again after debugging.

2.4 Creation of User Program

To input a user program, use the EzSQ Program Editor on the Windows system (personal computer). The Program Editor has three functional windows: the "Code Window" (for code input), "Data Window" (for data input), and "Monitor Window" (for monitoring).

Use the "Code Window" to input program source codes to create a user program.

Use the "Code Window" to configure the initial values for program execution.

Use the "Monitor Window" to monitor the status of program execution and inverter operation, and debug a created user program.

After checking the program syntax and counting the program steps, download the user program to the inverter.

For details on the above operations, see Chapter 4, "How To Use EzSQ."

- (1) Code input (in the "Code Window")
 - A main routine must begin with an "entry" instruction and end with an "end" instruction. A user program can contain only one main routine.
 - A subroutine must begin with a "sub" instruction and end with an "End sub" instruction. A user program can include multiple subroutines.
- (2) Data input (in the "Data Window")
 - Define the initial values of the variables used in the user program you created. A variable must be defined on a line with a variable name, definition expression, and calculation result. The definition expression can be another variable.

2.5 Syntax Check

When a user program is compiled, the codes input in the "Code Window" are checked for validity. If a syntax error is detected, EzSQ stops compilation and displays an error message.

EzSQ checks whether the input codes conform to the parameter input restrictions and the limited number of steps. At the end of compilation, EzSQ generates the intermediate program codes.

The following input restrictions apply to code input in the "Code Window":

- (1) Label (Label)
 - All alphanumeric characters can be input.
- (2) Mnemonic code (Mnemonic)
 - Only instructions and writable variables can be input.
- (3) Parameter string (Parm 1 to 6)
 - Parameter data can be set according to an instruction input as a mnemonic string. If a blank or "<Instruction>" is input as a mnemonic string, parameter data cannot be input.
- (4) Comment string (Comment)
 - All alphanumeric characters can be input.

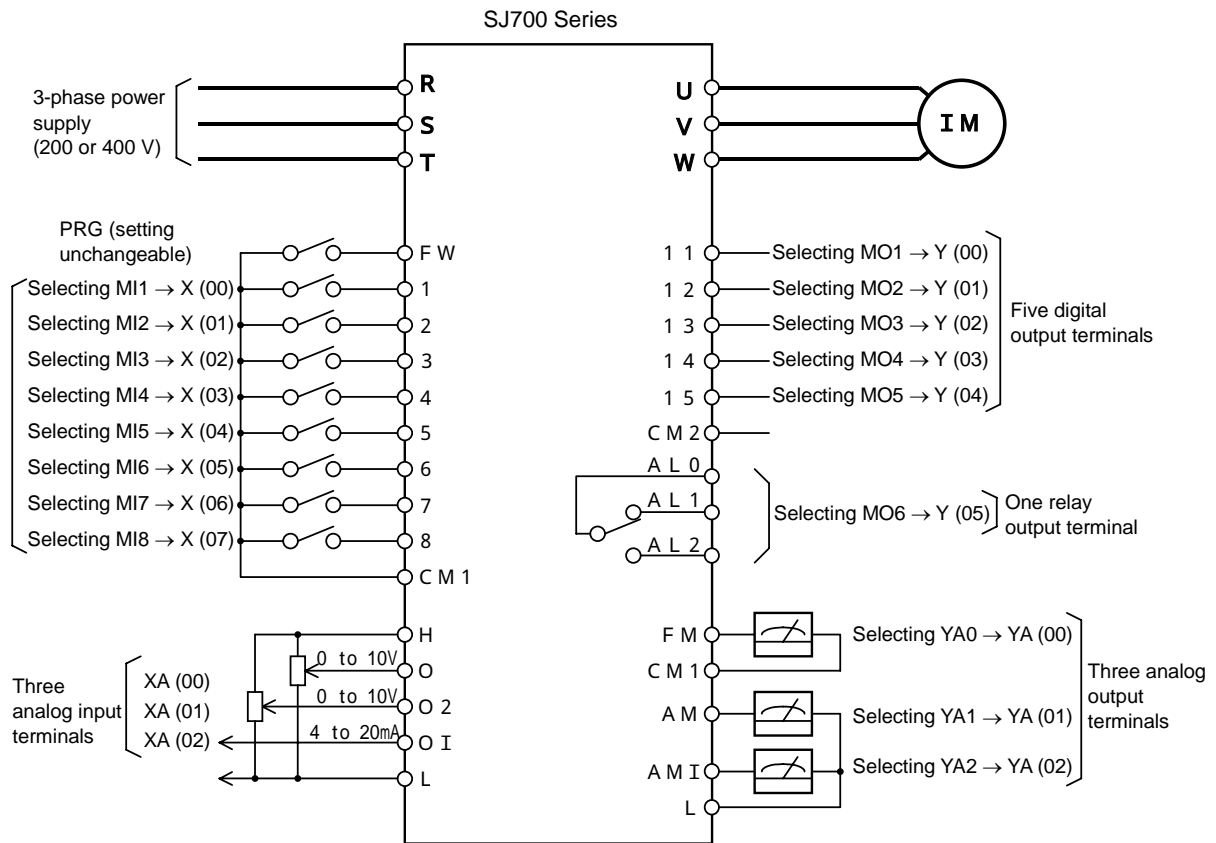
Chapter 2 Creation and Execution of a User Program

2.6 Settings on the Inverter

The easy sequence function is a feature of the SJ700 Series Inverter. The easy sequence function enables the inverter to access the variables written in a user program via inverter input/output terminals. Programming software EzSQ provides the reserved variables for control access to the inverter via external terminals. You can combine these variables as required to control inverter operation. Some reserved variables require the inverter settings to enable the variables. Also, the settings of frequency commands, operation commands, and input/output terminals on the inverter must follow those specified in the user program.

The table below lists the items, reserved variables, and function codes that must be configured when using the easy sequence function. For details on the reserved variables, see the explanation of settings related to the easy sequence function in Chapter 5.

Category	Item		Variable notation in program	Related function code	Variable/terminal use condition
		Terminal name			
Operation switching	Selection of easy sequence function		-	A017	-
Input/output switching	Program run signal	PRG (FW terminal)	-	A017	The PRG terminal is enabled when A017 = 01. (The FW function is disabled.)
	General-purpose input contacts (8 contacts)	Intelligent input terminals 1 to 8	X (00) to X (07) Xw	C001 to C008	Function code settings are required.
		General-purpose output contacts (6 contacts)	Intelligent output terminals 11 to 15	Y (00) to Y (05) Yw	
	Intelligent relay output terminals AL0 to AL2		C026		
	General-purpose analog inputs (3 terminals)	O terminal	XA (0)	-	No setting is required.
		O1 terminal	XA (1)	-	
		O2 terminal	XA (2)	-	
	General-purpose analog outputs (3 terminals)	FM terminal	YA (0)	C027	Function code settings are required.
		AM terminal	YA (1)	C028	
		AM1 terminal	YA (2)	C029	
Command switching	Frequency source setting		SET-Freq	A001	Related variables are valid only when A001 = 07.
	Run command source setting		FW, RV STA, STP, F/R	A002	Related variables are valid only when A002 = 01.
	Accel/decel time input selection		ACCEL DECEL	P031	Related variables are valid only when P031 = 03.
Other	User-defined variables (32 variables)		U (00) to U (31)	P100 to P131	The variables can be redefined by using the digital operator or EzSQ.



Input and output terminals available for general-purpose input/output settings

2.7 Execution of User Program

After downloading the user parameters to the inverter, enable the easy sequence function (by setting "01" in parameter "A017") to make the inverter ready for program execution. At the same time, the FW terminal is switched to the PRG terminal. (The FW terminal will no longer function as the terminal to input the forward-rotation command.) Then, turn on the PRG terminal to execute the user program downloaded to the inverter.

Chapter 3 Syntax

This chapter explains the syntax and definitions used for programming.

3.1	Code Description Format	3 - 1
3.2	Data Description Format	3 - 2
3.3	Variables and Ranges of Numerical Values	3 - 2
3.4	Operators	3 - 3
3.5	Conditions	3 - 3

3.1 Code Description Format

Each line of a program consists of the "Label," "Mnemonic," "Parm 1 to 6," and "Comment" fields. The "Mnemonic" field is used to describe an instruction word. Some instruction words do not require parameters.

(Example)

Line	Label	Mnemonic	Parm1	Parm2	Parm3	Parm4	Parm5	Parm6	Comment
001		entry							
002	LBL1	delay on	FW	TD(1)	1000				Example of comment
003		end							
004									

LBL1 delay on FW TD(1) 1000 Example of comment

Diagram labels: Label, Mnemonic instruction code, Parameters, Comment

3.1.1 Line

A line is the instruction unit of a program. You can describe one instruction word per line. It takes 2 ms to execute each line. One instruction corresponds to one step of the program.

3.1.2 "Label" field

Use the "Label" field to describe, for example, the branch destination for a branch instruction.

3.1.3 "Mnemonic" field

Use the "Mnemonic" field to describe the instruction to be executed. For details on the instructions, see Chapter 5, "Instruction Words."

3.1.4 "Parm 1 to 6" fields

Use the "Parm 1 to 6" fields to describe the arguments required to execute an instruction. Up to six arguments can be described as required for the instruction word described on the same line.

3.1.5 "Comment" field

Use the "Comment" field to describe a comment on each line.

3.2 Data Description Format

Each variable is described on a line that consists of the "Variable," "Define," "Answer," and "Comment" fields.

(Example)

Variable	Define	Answer	Comment
U(00)	5000*2+10	10010	Initial value
U(01)	U(00)+100	10110	
U(02)		0	

U() UL()

U(00) 5000*2+10 10010 Initial value

Variable name
Definition expression
Calculation result
Comment

3.2.1 "Variable" field

Use the "Variable" field to describe a variable name to be defined. Definable variable names are as follows:

Type	Variable name
User-defined variable	U (00) to U (31)

3.2.2 "Define" field

Use the "Definition" field to describe a definition expression for a variable. A definition expression can be a variable name. Clicking the [Calculate] button in the "Data Window" after entering a definition expression starts calculation.

3.2.3 "Answer" field

The "Answer" field displays a calculation result. You cannot rewrite the calculation result.

3.2.4 "Comment" field

You can describe a comment about the variable described on the line.

3.3 Variables and Ranges of Numerical Values

The tables below show the ranges of numerical values that can be specified in the "Code Window" and "Data Window."

(1) Variables and ranges of numeric values

Type of variable	Variable name	Range of numeric values
Bit and contact variables	FW, X (00), etc.	0, 1 (0: OFF, 1: ON)
User-defined variable	U (00) to U (31)	0 to 65535
Internal user variable	UL (00) to UL (07)	-2147483648 to 2147483647
Frequency setting variable	SET-Freq	0 to 40000
Acceleration/deceleration time setting variable	ACCEL, DECEL	0 to 360000
Monitoring and other variables	See Section 5.9, "Inverter Monitor Variables."	

(2) Numeric notation

Notation	Numeration	Remarks
(Omitted)	Decimal	Decimal number
&H	Hexadecimal	Hexadecimal number (specifiable only in the "Data Window")
&B	Binary	Binary number (specifiable only in the "Data Window")

Note: If the calculation executed for a variable by clicking the [Calculate] button results in a value that is outside the range of numeric values defined for the variable, the "User data range is invalid" message will appear and the "Answer" field will indicate "<Range invalid>."

3.4 Operators

You can use the following operators for a program:

Specification format	Description
<variable 1> = <variable 2> + <variable 3>	Addition
<variable 1> = <variable 2> - <variable 3>	Subtraction
<variable 1> = <variable 2> * <variable 3>	Multiplication
<variable 1> = <variable 2> / <variable 3>	Division
<variable 1> = <variable 2> mod <variable 3>	Remainder
<variable 1> = abs <variable 3>	Absolute value
<variable 1> = <variable 3>	Substitution
<variable 1> = <variable 2> or <variable 3>	OR (logical addition)
<variable 1> = <variable 2> and <variable 3>	AND (logical product)
<variable 1> = <variable 2> xor <variable 3>	XOR (exclusive-OR)
<variable 1> = not <variable 3>	NOT (negation)

Note 1: <variable 2> can be a constant ranging from 0 to 127.

Note 2: <variable 3> can be a constant ranging from -2^{31} to $2^{31}-1$.

3.5 Conditions

You can use a conditional expression to compare two variables. The operation result of a conditional expression is "true" (the condition is satisfied) or "false" (the condition is not satisfied).

Use a conditional expression with an "if" or "while" instruction to change the flow of program execution according to a condition.

Specification format	Description
<variable 1> = <variable 2>	"True" when <variable 1> is equal to <variable 2>
<variable 1> < <variable 2>	"True" when <variable 1> is less than <variable 2>
<variable 1> <= <variable 2>	"True" when <variable 1> is not greater than <variable 2>
<variable 1> > <variable 2>	"True" when <variable 1> is greater than <variable 2>
<variable 1> >= <variable 2>	"True" when <variable 1> is not less than <variable 2>
<variable 1> <> <variable 2>	"True" when <variable 1> is not equal to <variable 2>

Note: <variable 1> and <variable 2> can be constants ranging from 0 to 127.

Chapter 4 How To Use EzSQ

This chapter explains how to operate EzSQ.

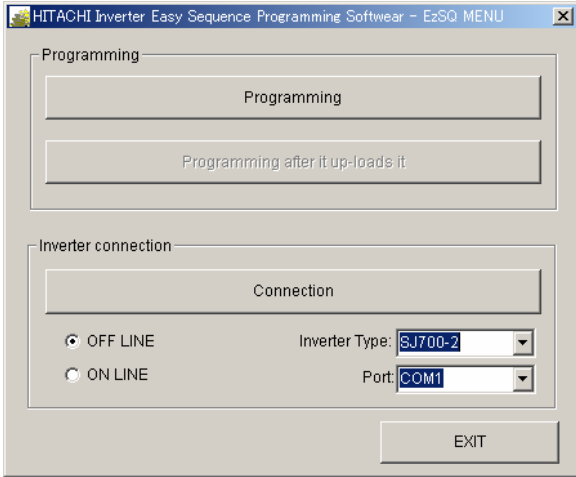
4.1	Starting and Terminating EzSQ	4 - 1
4.2	Flow of Programming	4 - 3
4.3	File Management.....	4 - 4
4.4	Methods of Entering a Program in the "Code Window"	4 - 6
4.5	Program Editing Functions.....	4 - 10
4.6	Methods of Entering Data in the "Data Window".....	4 - 12
4.7	Compiling a Program.....	4 - 14
4.8	Downloading a Program to the Inverter	4 - 15
4.9	Saving a Program in EEPROM	4 - 16
4.10	Compilation and Downloading	4 - 16
4.11	Uploading a Program from the Inverter.....	4 - 17
4.12	Reset	4 - 18
4.13	"Monitor Window"	4 - 18
4.14	Printing a Program	4 - 19
4.15	Referencing the Help Information	4 - 19
4.16	Version Information	4 - 19
4.17	Setting/Clearing a Password.....	4 - 20

4.1 Starting and Terminating EzSQ

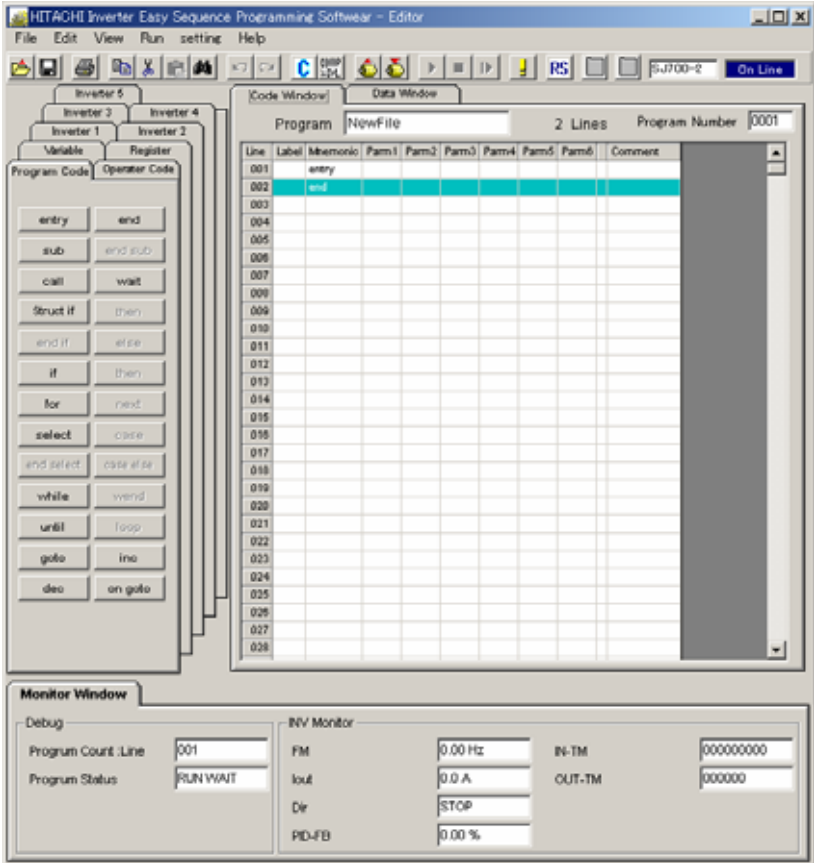
This section explains the procedures for starting and terminating EzSQ.

4.1.1 Starting EzSQ

- (1) On your personal computer, from the "Start" menu, select "Programs" - "Hitachi Industrial Equipment Systems Co., Ltd." - "EzSQProgramEditor.exe" to start up EzSQ.
- (2) The "EzSQ MENU" window appears.

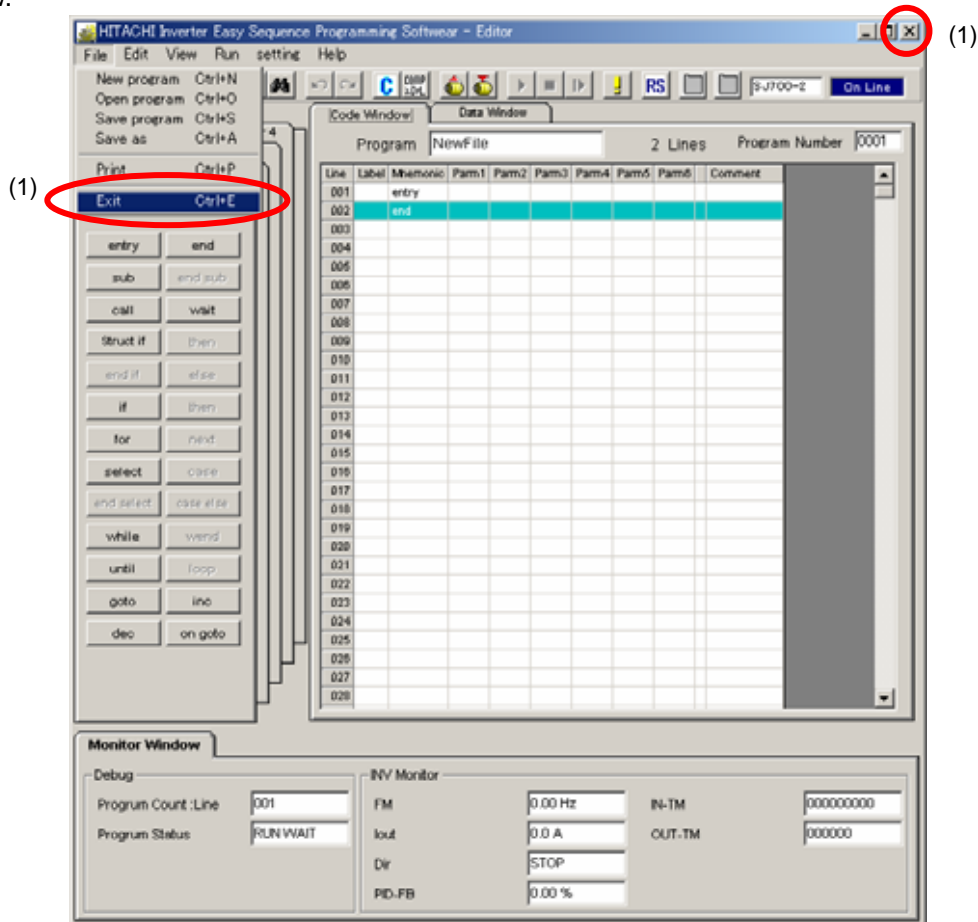


- (3) To connect the personal computer to the inverter, select the connection port to be used, and click the [Connection] button.
- (4) Click the [Programming] button. The "Editor" window appears.

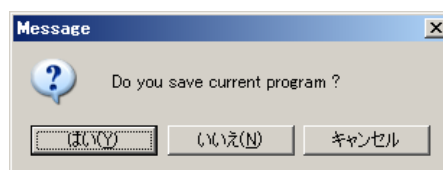


4.1.2 Terminating EzSQ

- (1) In the "Editor" window, select "Exit" from the "File" menu, or click  in the top-right corner of the window.

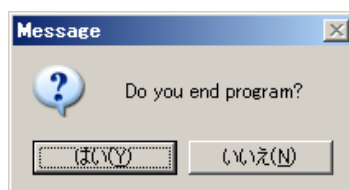


- (2) If you attempt to exit EzSQ before saving the currently edited program, the following message dialog box will appear:



- [Yes]: Saves the currently edited program, and then terminates EzSQ.
[No]: Terminates EzSQ without saving the currently edited program.
[Cancel]: Cancels the termination of EzSQ.

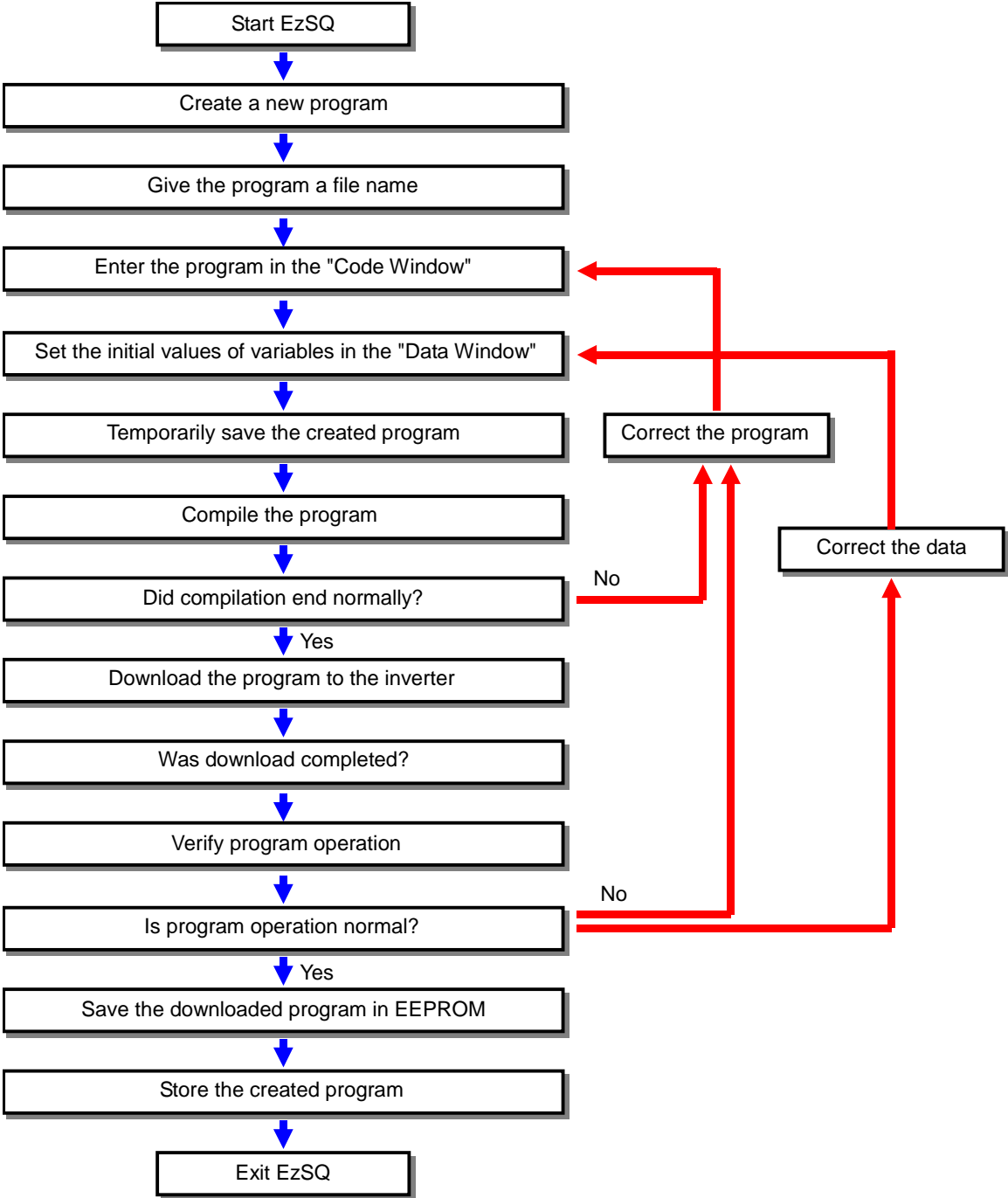
- (3) If you have already saved the currently edited program, the following message dialog box will appear:



- [Yes]: Terminates EzSQ.
[No]: Cancels the termination of EzSQ.

4.2 Flow of Programming

The following figure shows the flow of programming for the easy sequence function.

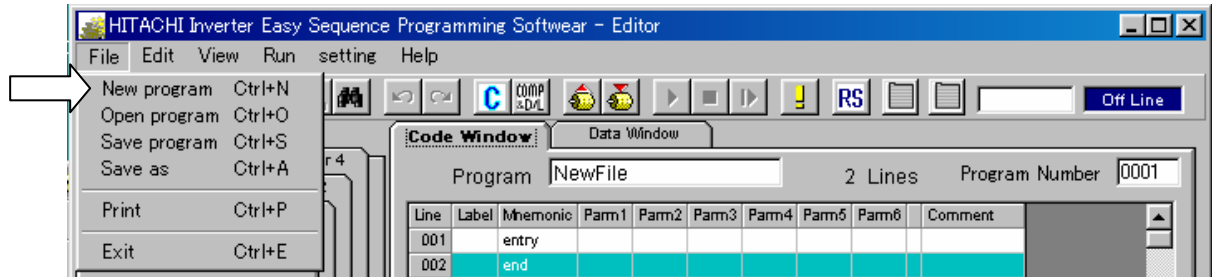


4.3 File Management

This section explains how to save and read a program you created with the Program Editor.

(1) Creating a new program

- 1) Select "New program" from the "File" menu on the menu bar of the "Editor" window.
(Shortcut command: Ctrl + N keys)

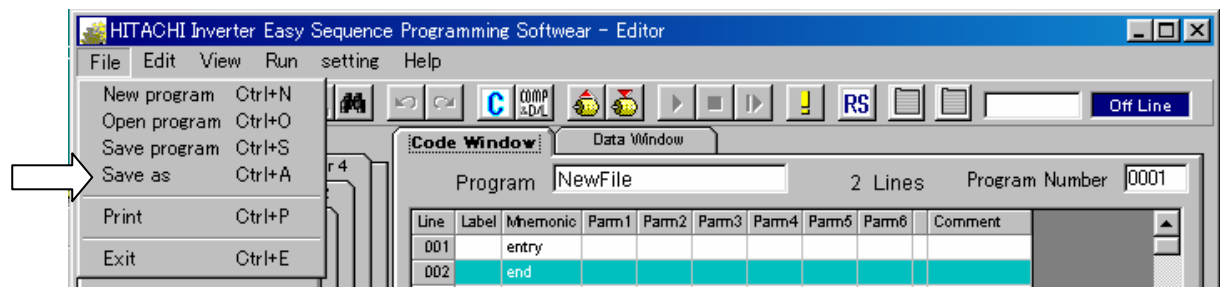


- 2) Discard the currently edited program, and then start editing a new program.

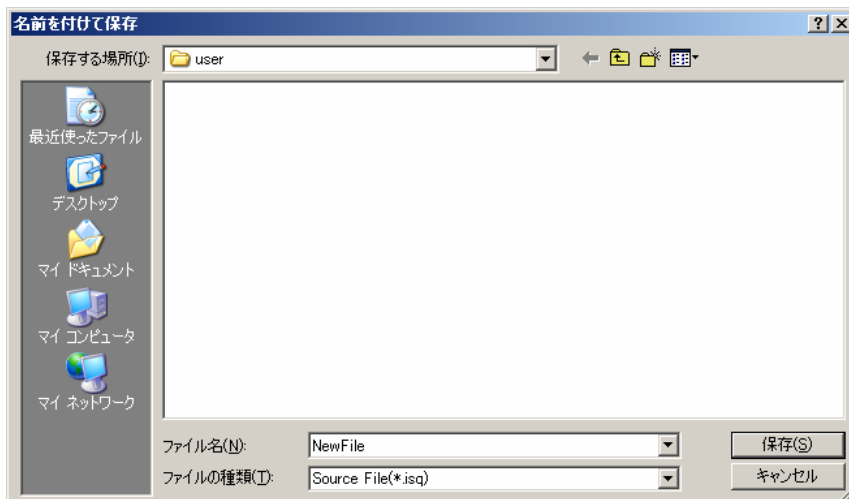
(2) Saving a new program as a file

To save a new program, follow the procedure below.

- 1) Select "Save as" from the "File" menu.
(Shortcut command: Ctrl + A keys)



- 2) The "Save as" dialog box appears.

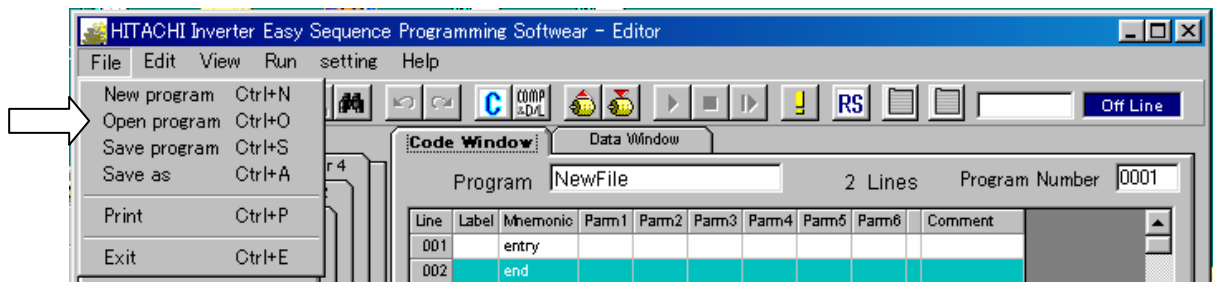


- 3) Click the [Save] button to save the new program.
The program name you gave to the new program is used as the file name (except the extension).
You can select ".isq" or ".csv" as the file type.

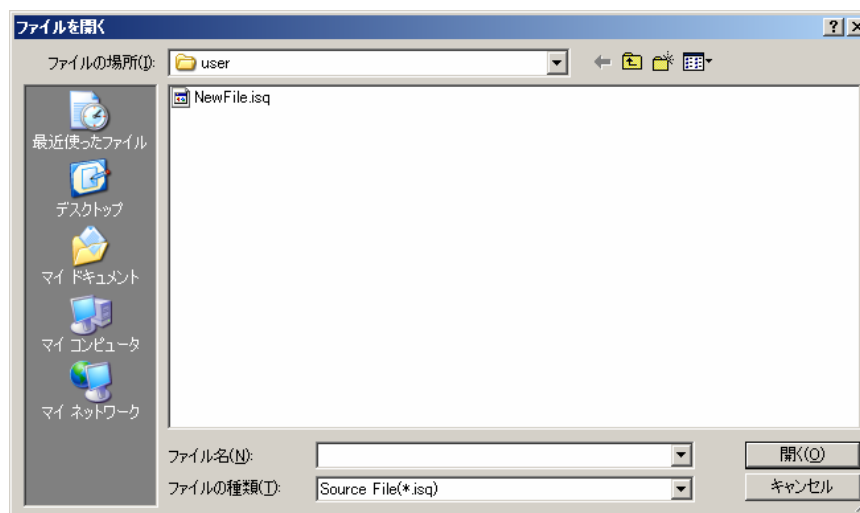
(3) Opening an existing program file

- 1) Select "Open program" from the "File" menu.

(Shortcut command: Ctrl + O keys or the file-opening icon button under the menu bar)



- 2) The "Open" dialog box appears.

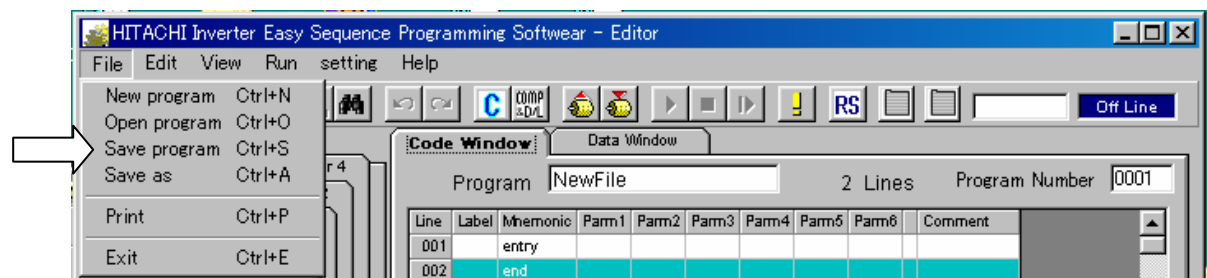


- 3) Select the file to be opened, and then click the [Open] button. The Program Editor reads the selected program file. You can select "*.isq" or "*.csv" as the file type.

(4) Overwriting a file

- 1) Select "Save program" from the "File" menu.

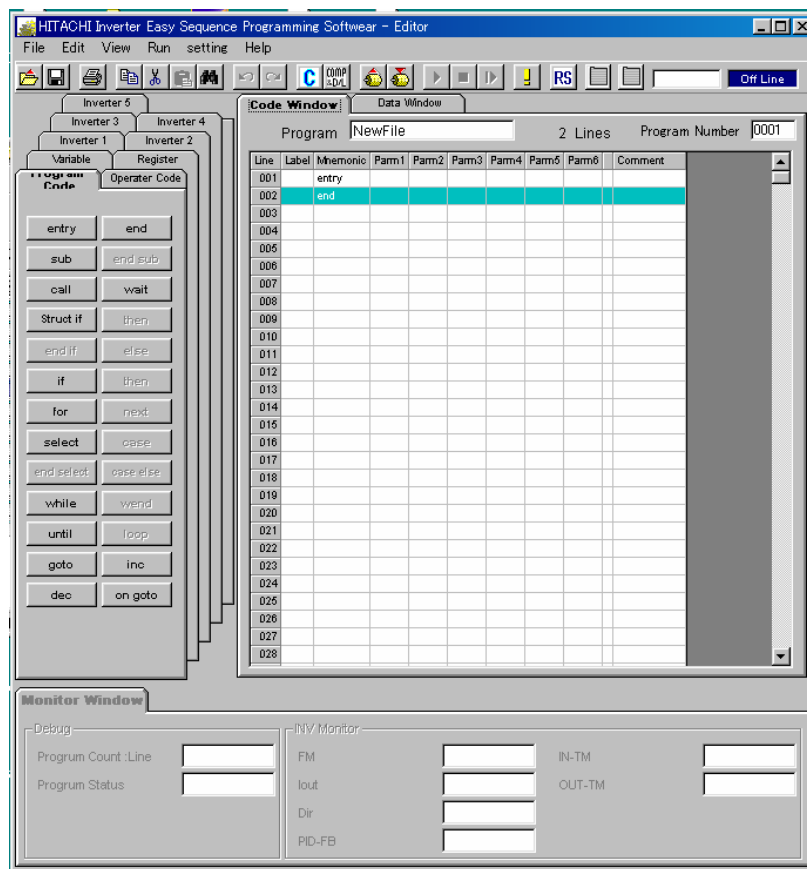
(Shortcut command: Ctrl + S keys or the file-saving icon button under the menu bar)



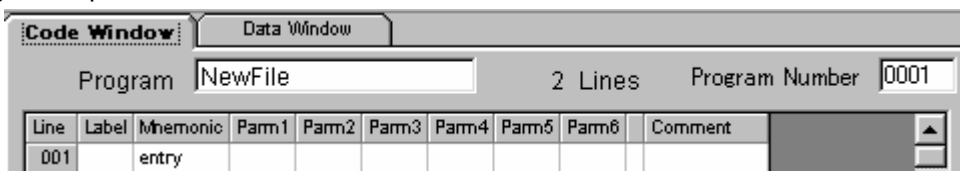
- 2) The currently edited program is saved by overwriting the file having the same file name as the program name of the currently edited program.

4.4 Methods of Entering a Program in the "Code Window"

If the "Editor" window shows the "Data Window," click the "Code Window" tab to switch to the "Code Window."



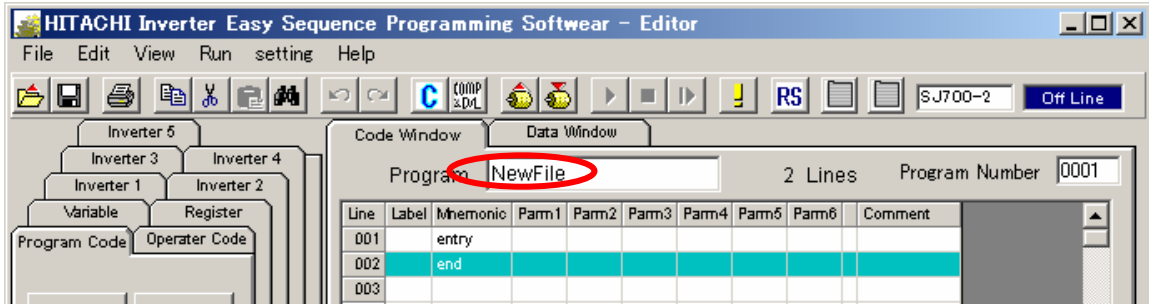
(1) Display and input fields



- "Program": This field displays the name of the program being edited and allows you to change it.
- "Lines": This field displays the total number of lines of the program being edited.
- "Program Number": This field allows you to enter an arbitrary 4-digit number as a program number. The program number can be monitored (with function code "d024") on the digital operator. Use the program number for program management.
- "Comment": This field allows you to describe a comment (e.g., a note on programming).
- "Parm 1" to "Parm 6": These fields allow you to enter the parameters required for the instruction entered in the "Mnemonic" field.
- "Mnemonic": This field allows you to enter a mnemonic instruction code or the variable to store an operation result.
- "Label": This field allows you to enter a label name that indicates the corresponding line. The label name will be used as the jump destination in an "if" or "goto" statement. Specifying the same label name for two or more lines causes a compilation error.
- "Line": This field indicates a line number. You can enter up to 512 lines of instructions in the code area. A compiled program must not have more than 512 steps. The number of steps of a compiled program will be displayed when compilation ends normally.

(2) Entering a program name

The "Program" field displays the name of the program being edited. (This field displays "NewFile" as the initial program name after the Program Editor is started or when a new program is created.) First change the program name to avoid losing an already created program. (You can also change the program name [file name] when saving the edited program file with the "Save as" command. A program name must be an alphanumeric string of up to 16 characters.)



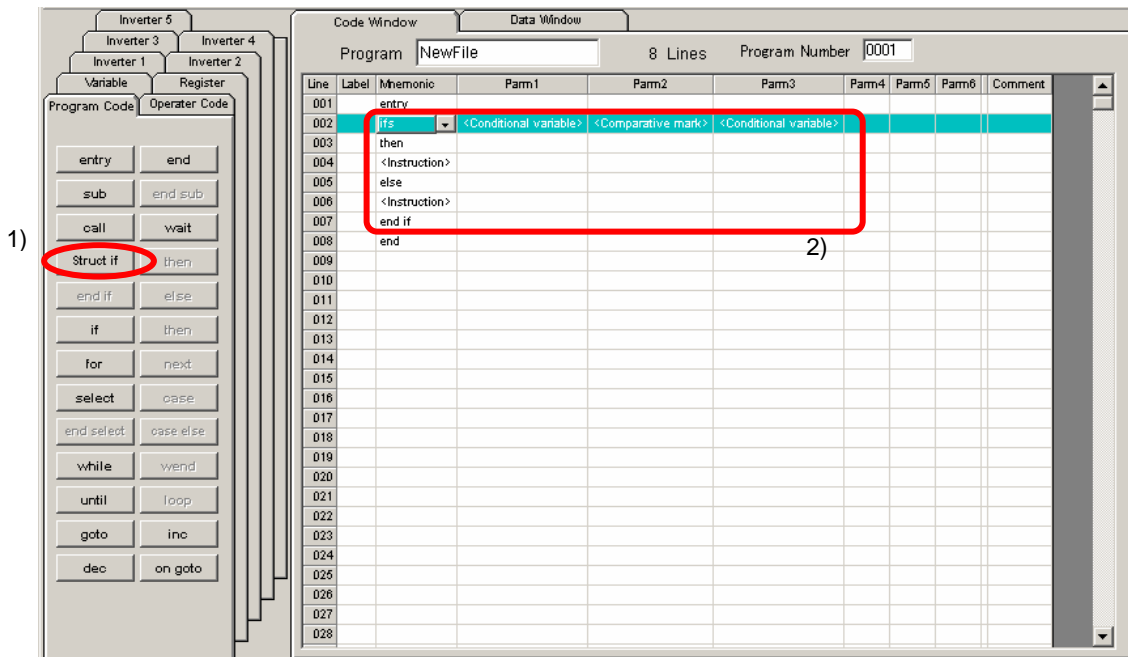
(3) Methods of describing a program

(a) Entering instructions from the code input tabs

You can easily enter instruction statements by using the code input tabs of the "Editor" window. Clicking a button on a code input tab inserts or overwrites an instruction statement or variable in a cell selected in the "Code Window".

(Example of operation)

- 1) In the "Program Code" input tab, click the "Struct if" button.
- 2) A structured if instruction is inserted in the cell currently selected in the "Code Window".



- 3) If you temporarily enter a descriptive statement enclosed by "<" and ">" in place of an instruction or parameter in a cell, you must enter the instruction or parameter in the cell later. If you do not enter the instruction or parameter in the cell, that cell is assumed to contain no description during compilation.


(Examples of descriptive statements)


- <Conditional variable> :Indicates a variable or numeric (0 to 127)
- <Comparative mark> :Indicates "=", "<", "<=", ">", ">=", or "<>".
- <Instruction> :Indicates a group of instructions or operational variables.

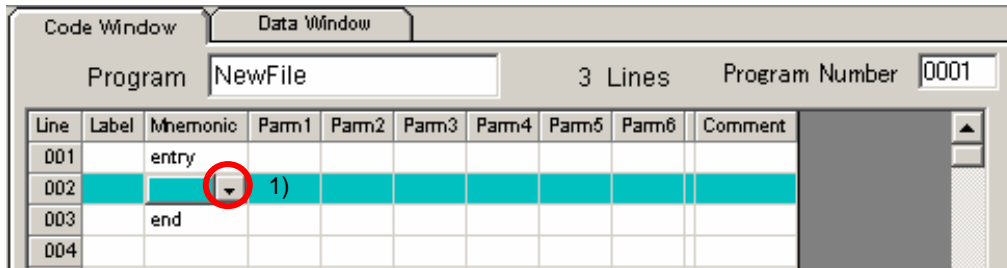
Note: On each code input tab, the buttons for instructions and variables that cannot be entered in the currently selected cell are disabled.

Chapter 4 How To Use EzSQ

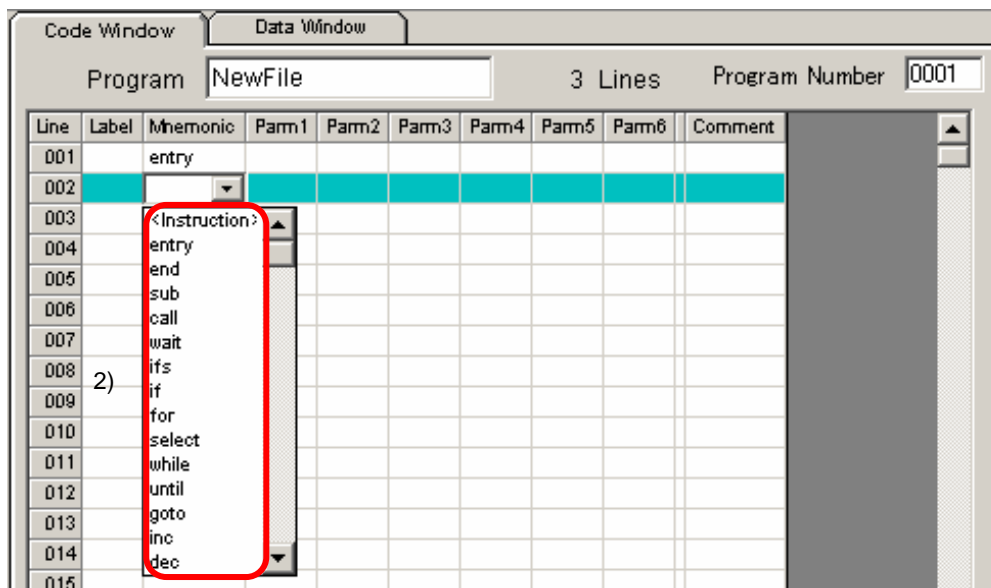
(b) Entering instructions from the lists

When selected, specific cells show a pull-down list mark  at the right end. You can select and enter an instruction, variable, or other parameter from the pull-down list.

- 1) Click the  mark or press the space bar on the keyboard.



- 2) A pull-down list appears to show the instructions that can be entered in the cell.

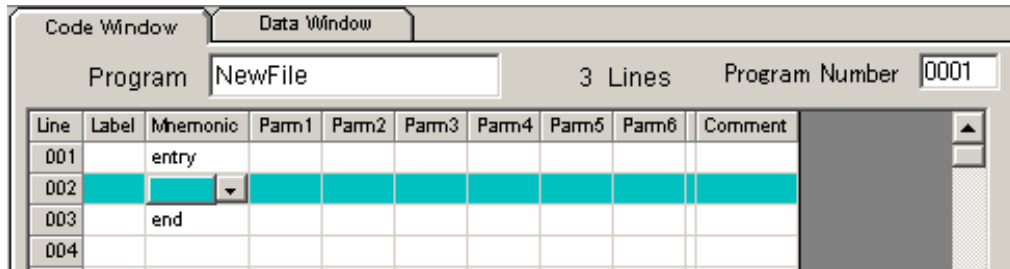


- 3) If the desired instruction is not shown, use the vertical scroll bar along the right edge of the list to scroll through the list.
- 4) Select and click the instruction to be entered. (You can also use the arrow keys ([↑] and [↓]) on the keyboard to select an instruction, and then press the Enter key to enter it.)
- 5) The instruction selected in the list is entered in the selected cell.

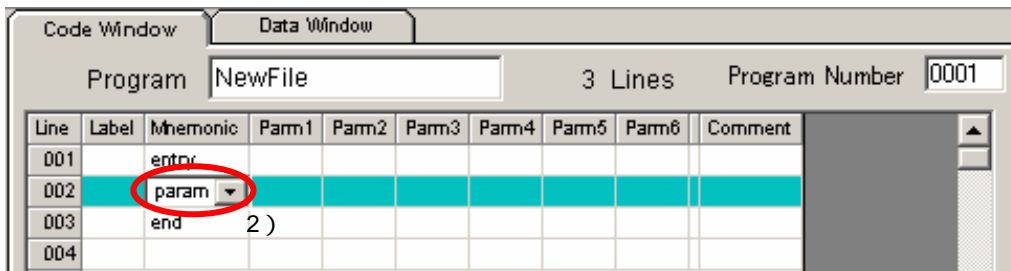
(c) Entering instructions from the keyboard

You can enter instructions directly from the keyboard in selected cells.

- 1) Click in the "Mnemonic" cell in which to enter an instruction.

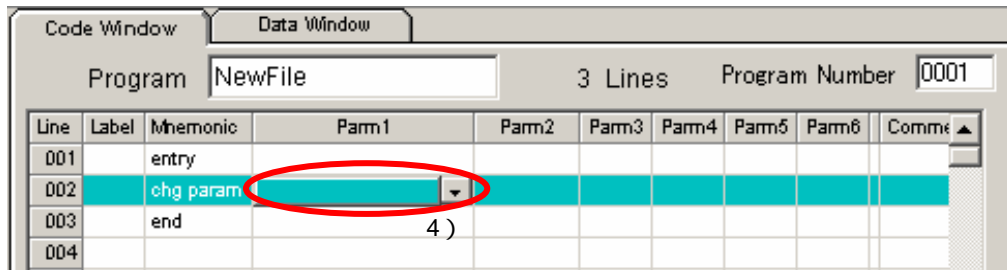


- 2) Type in the instruction statement from the keyboard.



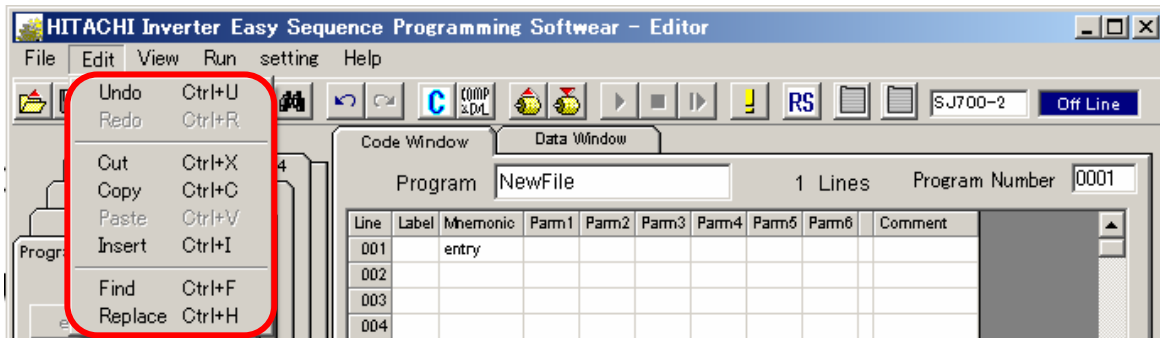
- 3) Press the Tab or arrow ([→]) key.

- 4) The "Parm" cell next to the cell in which you entered the instruction changes to the selected state.




4.5 Program Editing Functions

This section explains the edit functions (commands) you can use when describing a program. You can use the edit commands in both the "Code Window" and "Data Window."




- (1) "Undo"


The "Undo" command is effective after an editing operation (e.g., input) has been performed. The command cancels the preceding editing operation.

 - Select "Undo" from the "Edit" menu on the menu bar.
(You can also execute this command by clicking the  icon under the menu bar or pressing the "Ctrl + U" keys on the keyboard.)
- (2) "Redo"


The "Redo" command is effective after the "Undo" command has been executed. This command restores the undone editing operation (e.g., input).

 - Select "Redo" from the "Edit" menu on the menu bar.
(You can also execute this command by clicking the  icon under the menu bar or pressing the "Ctrl + R" keys on the keyboard.)
- (3) "Cut"


The "Cut" command cuts out one or more selected lines. You can use this command to delete program lines.

 - 1) Select the line to be deleted.
 - 2) Select "Cut" from the "Edit" menu on the menu bar.
(You can also execute this command by clicking the  icon under the menu bar or pressing the "Ctrl + X" keys on the keyboard.)
 - 3) The selected line is deleted. The deleted line can be pasted with the "Paste" command as described below.
The lines following the deleted line are shifted upward.
- (4) "Copy"

The "Copy" command copies one or more selected lines.

 - 1) Select the line to be copied.
 - 2) Select "Copy" from the "Edit" menu on the menu bar.
(You can also execute this command by clicking the  icon under the menu bar or pressing the "Ctrl + C" keys on the keyboard.)
 - 3) The selected line is copied. The copied line can be pasted with the "Paste" command as described below.
- (5) "Paste"

The "Paste" command is effective after the "Cut" or "Copy" command has been executed. This command pastes one or more lines that were cut out or copied in the program.

 - 1) Select the line where to paste the cut-out or copied line(s).
 - 2) Select "Paste" from the "Edit" menu on the menu bar.
(You can also execute this command by clicking the  icon under the menu bar or pressing the "Ctrl + V" keys on the keyboard.)
 - 3) The cutout or copied line(s) is (are) inserted in (and after) the selected line.


(6) "Insert"

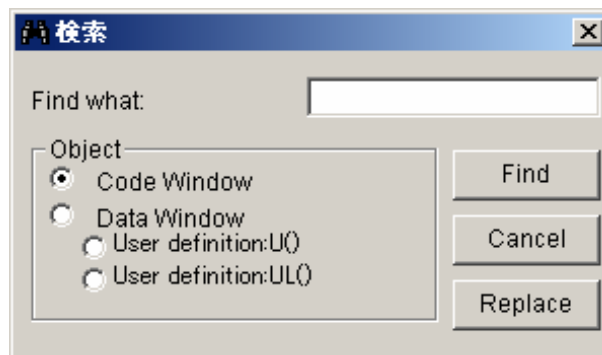
The "Insert" command inserts a blank line in the program.

- 1) Select the line where to insert a blank line.
- 2) Select "Insert" from the "Edit" menu on the menu bar.
(You can also execute this command by pressing the "Ctrl + I" keys on the keyboard.)
- 3) A blank line is inserted between the selected line and the preceding line.

(7) "Find"

The "Find" command searches the selected area ("Code Window" or "Data Window") to find a specified string.

- 1) Select "Find" from the "Edit" menu on the menu bar.
(You can also execute this command by clicking the  icon under the menu bar or pressing the "Ctrl + F" keys on the keyboard.)
- 2) The "Find" dialog box appears.

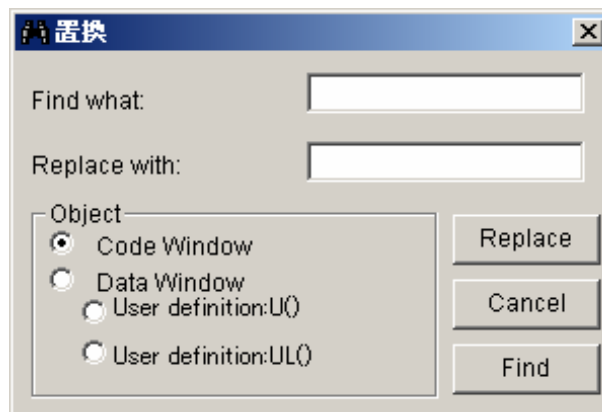


- 3) Enter the string to be found, and then click the [Find] button to start searching.
- 4) Clicking the [Cancel] button closes the "Find" dialog box.
- 5) Clicking the [Replace] button switches to the "Replace" dialog box.

(8) "Replace"

The "Replace" command searches the selected area ("Code Window" or "Data Window") to find a specified string and replace it with another specified string.

- 1) Select "Replace" from the "Edit" menu on the menu bar.
(You can also execute this command by pressing the "Ctrl + H" keys on the keyboard.)
- 2) The "Replace" dialog box appears.

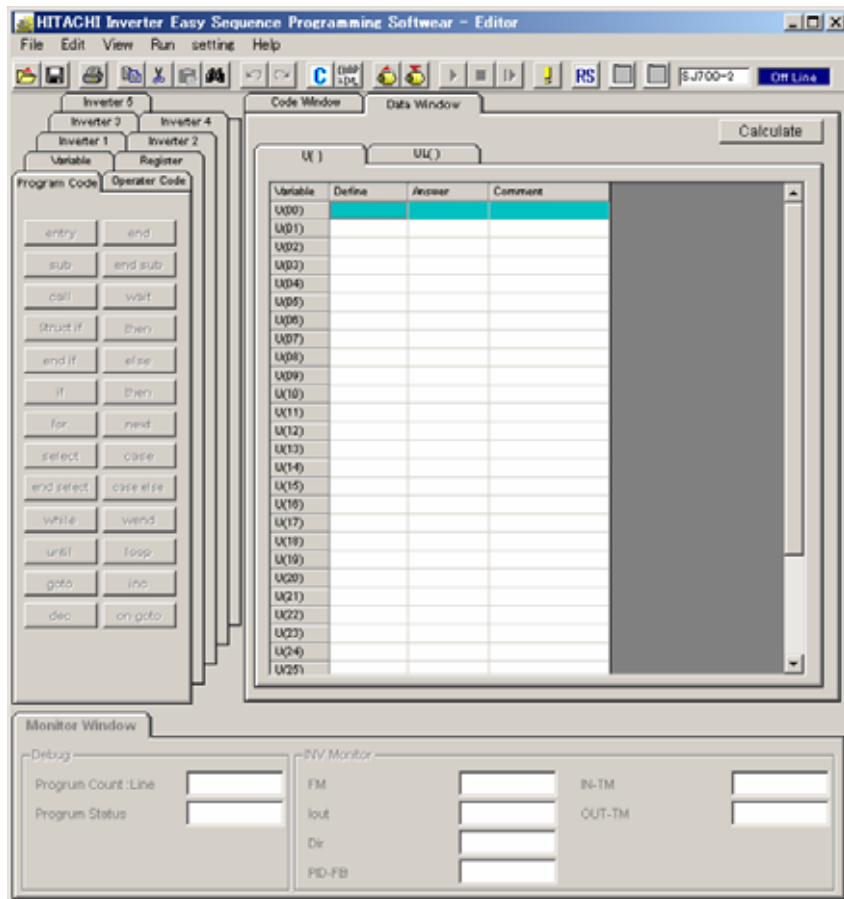


- 3) Enter the string to be found and the string to replace, and then click the [Find] button to start replacement.
- 4) Clicking the [Cancel] button closes the "Replace" dialog box.
- 5) Clicking the [Find] button switches to the "Find" dialog box.

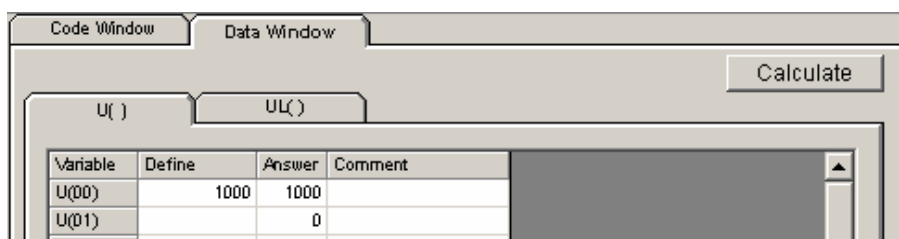
4.6 Methods of Entering Data in the "Data Window"

Use the "Data Window" (data input window) to set the initial values of variables to be applied when executing the program.

If the "Program Editor" window shows the "Code Window," click the "Data Window" tab to switch to the "Data Window."



(1) Display and input fields



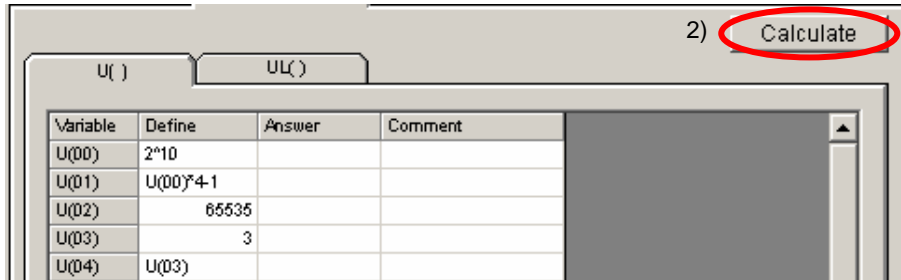
- Variable tab: "U ()" ,"UL()" indicates that the "Data Window" currently shows the variable tab for the variables with code "U (XX)" ,"UL(XX)".
- "Comment": This field allows you to describe a comment (e.g., a note on programming). The entered comment will be saved when the program file is saved, but will not be reflected in the program compilation result. The comment will be deleted when the compiled program is uploaded after being downloaded once.
- "Answer": This field displays the initial variable settings to be applied when executing the program.
- "Define": This field allows you to enter the value or calculation expression to be set in a variable.
- "Variable": This field displays a variable name.

Note: If the desired variable is not shown in the window, use the vertical scroll bar along the right edge of the window to scroll through the display.

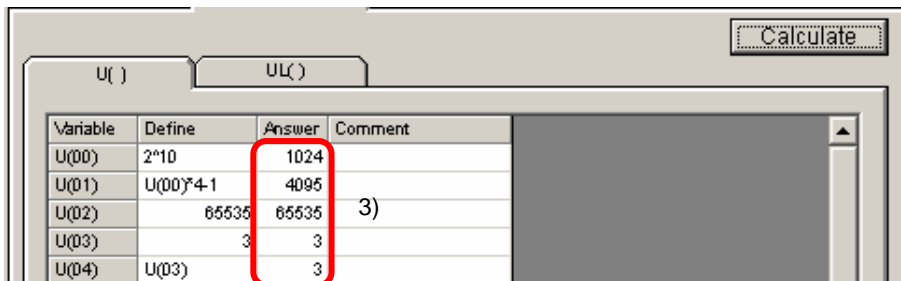
(2) Method of setting values in the variables

Follow the procedure below to set values in the variables. You can enter a calculation expression or immediate value in the "Define" field for each variable.

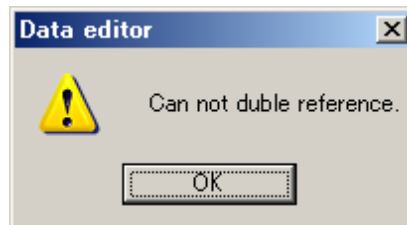
- 1) Enter a calculation expression or immediate value in the "Define" field.
You can use the arithmetic operators and parentheses for the calculation expression. Each immediate value to be entered and the calculation result must be within the upper and lower limits set for the target variable.
- 2) After entering expressions or values for the variables, click the [Calculate] button.



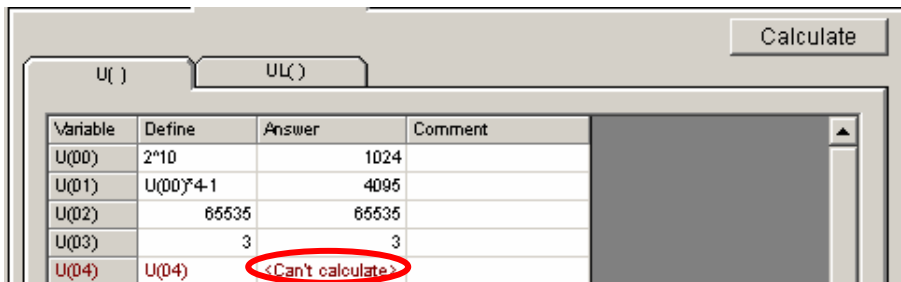
3) The "Answer" field shows the calculation result for each variable.



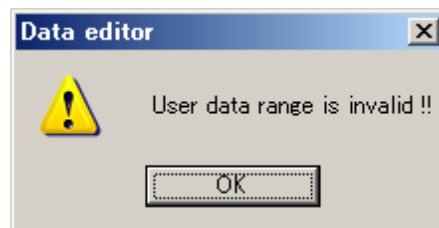
* If the calculation expression entered for a variable references the same variable, an error message appears.



Also, the "Result" field for the variable indicates "<Can't calculate>."



* If the value entered for a variable is not within the maximum and minimum limits set for the variable, an error message appears.




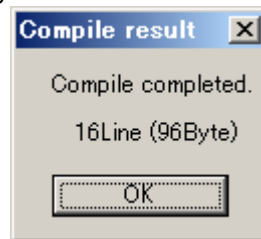
Also, the "Result" field for the variable indicates "<Range invalid>."

Variable	Define	Answer	Comment
U(00)	2^{10}	1024	
U(01)	$U(00)^4 - 1$	4095	
U(02)	65535	65535	
U(03)	3	3	
U(04)	$U(02) + U(03)$	<Range invalid>	

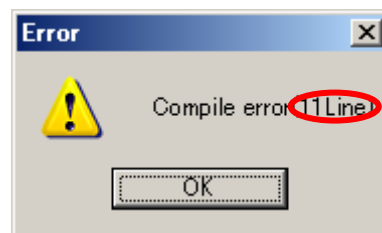
4.7 Compiling a Program

After entering a program and setting the values of variables, compile the program as follows:

- (1) Select "Compile" from the "Run" menu on the menu bar.
(You can also execute the "Compile" command by clicking the  icon under the menu bar or pressing the "Ctrl + F9" keys on the keyboard.)
- (2) EzSQ compiles the program and displays a normal-end message when no error is detected. Then, the program is ready for downloading.




Note: If EzSQ detects an error during compilation, it displays the line where the error is detected (in case of an instruction format error) or both the line and string where the error is detected (in case of a parameter error). In either case, correct the indicated line of the program.



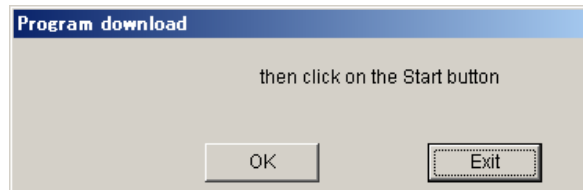
4.8 Downloading a Program to the Inverter

After the program has been compiled normally, download it from the personal computer to the inverter. Downloading is enabled only when the program run signal terminal (FW terminal) of the inverter is off or the inverter is in program-stopped status. You cannot download the program in any other status.

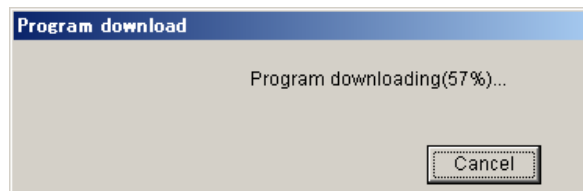
- (1) Select "Download" from the "Run" menu on the menu bar.

(You can also execute the "Download" command by clicking the  icon under the menu bar or pressing the "Ctrl + F11" keys on the keyboard.)

- (2) The "Program Download" dialog box appears.

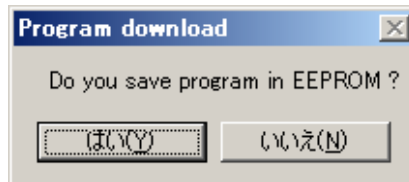


- (3) Click the [OK] button to start downloading.

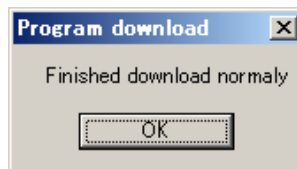


- (4) For downloading, EzSQ transfers the program from the personal computer to the memory buffer of the inverter. The program is not stored in internal memory (EEPROM) of the inverter. After the transfer ends, a dialog box appears to ask you whether to save the downloaded program in EEPROM.

Click the [Yes] button to save the downloaded program in the inverter's EEPROM. Click the [No] button to omit saving.



- (5) After the download process ends, an end message dialog box appears. Click the [OK] button.

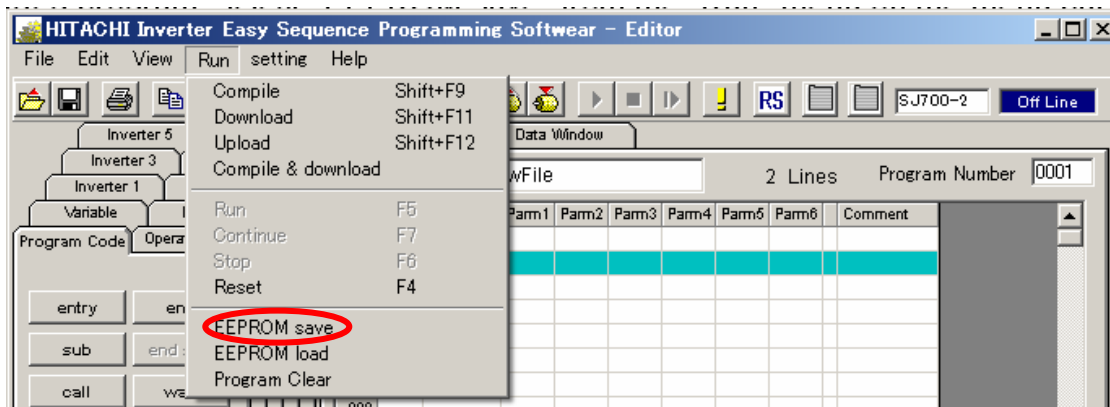


Note: If you do not save the downloaded program in the inverter's EEPROM, the downloaded program will be deleted from the inverter when the inverter power is reset. Therefore, be sure to save the downloaded program in the inverter's EEPROM. Even if you do not save the downloaded program immediately after transferring it to the inverter, you can save it later by using the save operation described below.

4.9 Saving a Program in EEPROM


After a program has been downloaded normally to the inverter, you can save it in internal memory (EEPROM) of the inverter.

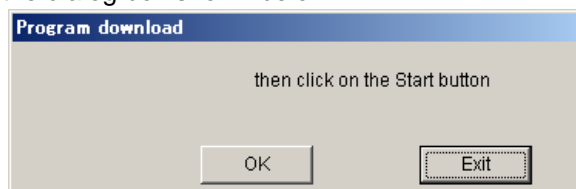
To save the downloaded program, select "EEPROM save" from the "Run" menu on the menu bar.



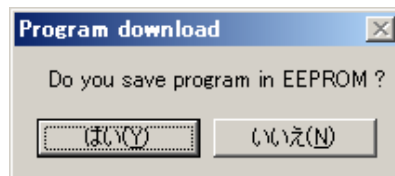
4.10 Compilation and Downloading

You can compile and download a program as a series of processes with a single command.

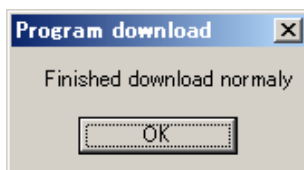
- (1) Select "Compile & download" from the "Run" menu on the menu bar.
(You can also execute the "Compile & download" command by clicking the  icon under the menu bar.)
- (2) EzSQ compiles the program and, if no error is detected, downloads it to the inverter successively. Click the [OK] button in the dialog box shown below.



- (3) After the program is downloaded (transferred) to the inverter, a dialog box appears to ask you whether to save the downloaded program in the inverter's EEPROM. Click the [Yes] button to save the downloaded program in the inverter's EEPROM. Click the [No] button to omit saving.




- (4) After the download process ends, an end message dialog box appears. Click the [OK] button.

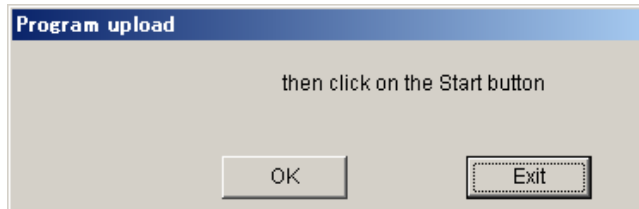


Note: If EzSQ detects an error during compilation, it displays the line where the error is detected. In such case, correct the indicated line of the program.

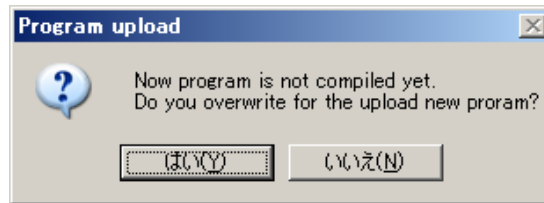
4.11 Uploading a Program from the Inverter

You can upload a downloaded program from the inverter to the personal computer to search for the source file that matches the program data. The search result will be displayed.

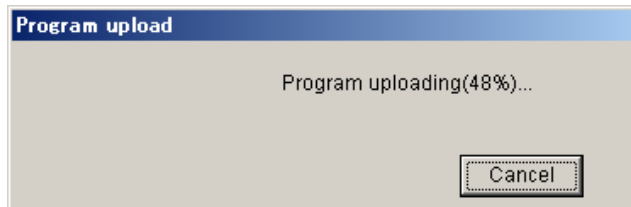
- (1) Select "Upload" from the "Run" menu on the menu bar.
(You can also execute the "Upload" command by clicking the  icon under the menu bar or pressing the "Ctrl + F12" keys on the keyboard.)
- (2) The "Program Upload" dialog box appears.



Note: If the program currently being edited has not been compiled yet, a dialog box appears to ask you whether to overwrite the program being edited. If you intend to overwrite, click the [Yes] button to proceed.



- (3) Click the [OK] button in the "Program Upload" dialog box to start uploading.



- (4) After uploading ends, an end message dialog box appears. Click the [OK] button.



End message indicating that the uploaded program matched a file in the source list



End message indicating that the uploaded program did not match any file in the source list

Note 1: The source files to be searched on the personal computer by this operation are those contained in the "user" folder created in the same folder where EzSQ was installed. On the personal computer, be sure to store all source files downloaded to the inverter in the "user" folder.

Note 2: If you have changed any settings of variables "U (00)" to "U (31)" in a program downloaded to the inverter by using the digital operator of the inverter, any attempt to upload the program and search for a matching source file ends with an unmatched result.

4.12 Reset

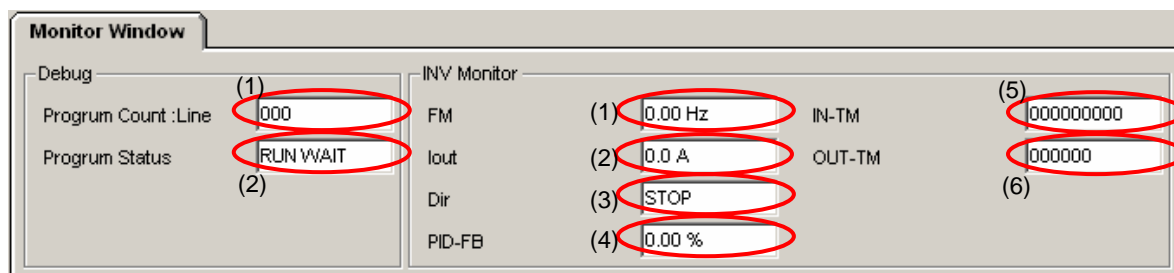
- You can reset the tripping inverter by performing a reset operation on EzSQ.

- 1) - Select "Reset" from the "Run" menu on the menu bar.

(You can also execute this command by clicking the  icon under the menu bar or pressing the F4 key on the keyboard.)

4.13 "Monitor Window"

- (1) Display fields



"Debug" section

- "Program Count" (1): Displays the program line number of the line that is currently being executed.
- "Program Status" (2): Displays the program execution status.


"INV Monitor" section

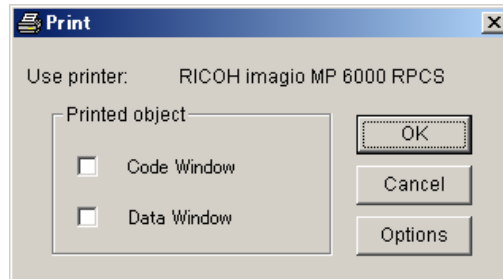
You can monitor various items of the inverter on the personal computer without having to connect the digital operator to the inverter. (For details, refer to the SJ700 Series Inverter Instruction Manual.)

- "FM" (1): Monitors the inverter output frequency in the same way as the output frequency monitoring function (code "d001") of the inverter.
- "lout" (2): Monitors the inverter output current in the same way as the output current monitoring function (code "d002") of the inverter.
- "Dir" (3): Monitors the motor-rotation direction in the same way as the rotation direction monitoring function (code "d003") of the inverter.
- "PID-FB" (4): Monitors the PID feedback data in the same way as the process variable (PV), PID feedback monitoring function (code "d004") of the inverter.
- "IN-TM" (5): Monitors the intelligent input terminal status in the same way as the intelligent input terminal status function (code "d005") of the inverter.
- "OUT-TM" (6): Monitors the intelligent output terminal status in the same way as the intelligent output terminal status function (code "d006") of the inverter.

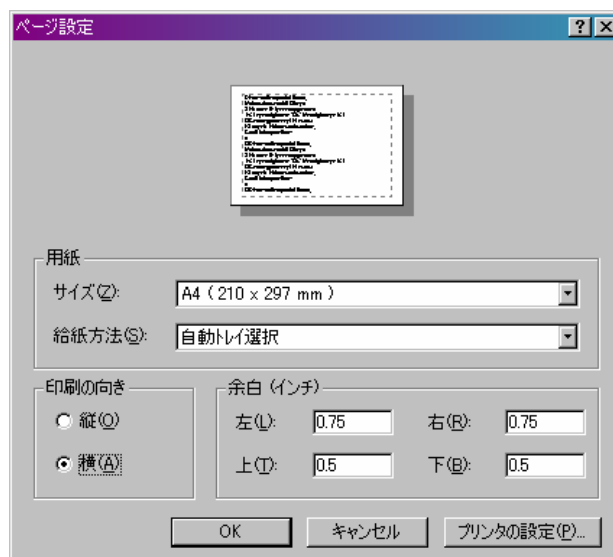
4.14 Printing a Program

You can print out the currently edited program.

- (1) Select "Print" from the "Run" menu on the menu bar.
(You can also execute the "Print" command by clicking the  icon under the menu bar or pressing the "Ctrl + P" keys on the keyboard.)
- (2) The "Print" dialog box appears.



- (3) Select the checkbox for the print-target ("Code Window" or "Data Window"). Printing of the code or data part of the program is enabled.
- (4) Click the [OK] button to start printing.
- (5) To close the "Print" dialog box, click the [Cancel] button.
- (6) Clicking the [Options] button opens the "Page Setup" dialog box.



4.15 Referencing the Help Information

You can reference the electronic data of this Instruction Manual from the "Program Editor" window.

- (1) Select "Help" from the "Help" menu on the menu bar.
(You can also execute the "Help" command by pressing the F1 key on the keyboard.)
 - (2) The PDF file containing the electronic data of this Instruction Manual opens.
- Note: Opening the PDF file requires Adobe Acrobat Reader to be installed on your personal computer.

4.16 Version Information

You can reference the version information on EzSQ.

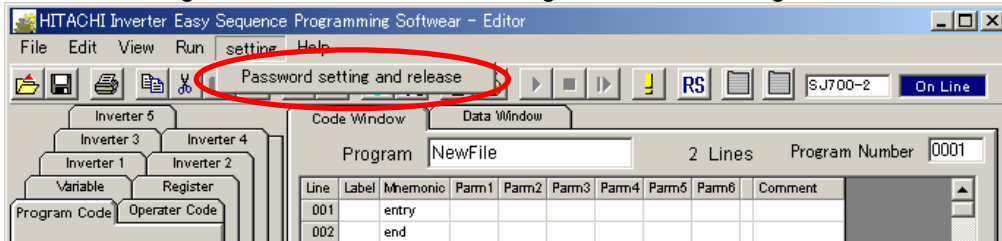
- (1) Select "About Program Editor" from the "Help" menu on the menu bar.
- (2) The "Version" dialog box appears.
- (3) To close the dialog box, click the any button.

4.17 Setting/Clearing a Password

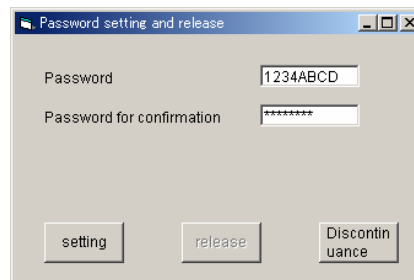
- You can set a password to lock a downloaded program.
- The program locked with a password cannot be uploaded or downloaded.

4.17.1 Setting a password

- Select "Password setting and release" from the "Setting" menu in the "Program Editor" window.



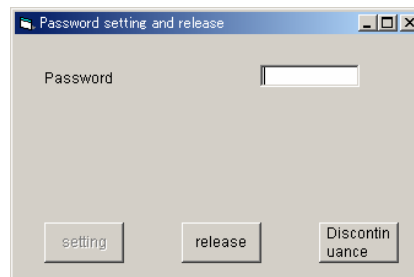
- The " Password setting and release " window appears.
(The "Password for confirmation" field appears only when no password has been set for the relevant program. If a password has already been set for the program, Only the "Password" field described below for clearing a password appears.)



- <1> Enter a password that consists of up to eight alphanumeric characters in the "Password" and " Password for confirmation" fields.
 - <2> Click the "setting" button to register the password in the inverter.
- Note: You cannot clear the set password by user-setting initialization on the inverter.
To clear the set password, perform the password clearance operation described below.

4.17.2 Clearing a password

- <1> Enter a password that consists of up to eight alphanumeric characters in the "Password" fields.
- <2> Click the "release" button to clear the password in the inverter.



Note: You can set or clear a password only while the inverter is online. Any attempt to set or clear a password while the inverter is offline will result in an error message being displayed



Chapter 5 Instruction Words

This chapter explains the detailed specifications of instruction words.

5.1	List of Instructions	5 - 1
5.2	Program Control Instructions	5 - 4
5.3	Operators	5 - 14
5.4	Conditional Expressions.....	5 - 14
5.5	Input/Output Control Instructions	5 - 15
5.6	Timer Control Instructions	5 - 20
5.7	Inverter Control Instructions.....	5 - 24
5.8	Other Reserved Variables	5 - 29
5.9	Inverter Monitor Variables	5 - 37

5.1 List of Instructions

This section lists the instructions that can be used in a program.

(1) Program control instructions

Instruction name	Instruction format						Description
	Mnemonic code	First argument	Second argument	Third argument	Fourth argument	Fifth argument	
entry statement	entry						Indicates the beginning of the main program.
	end						Indicates the end of the main program.
sub statement	sub	<subroutine name>					Indicates the beginning of a subroutine.
	end sub						Indicates the end of a subroutine.
call statement	call	<subroutine name>					Branches processing to the subroutine specified by <subroutine name>.
cont statement	cont	on					Resumes execution, after an interruption, from the step where program execution was interrupted.
		off					Resumes execution, after an interruption, from the beginning of the program.
for loop statement	for	<variable>	<start value>	<end value>	<incremental value>		Executes <instruction set> repeatedly until <variable> reaches <end value>. Note that <variable>, which initially contains <start value>, is incremented by <incremental value> each time <instruction set> is executed.
	<instruction set>						Indicates the instructions to be executed repeatedly.
	next						Ends the "for" loop.
goto statement	goto	<label name>					Branches processing unconditionally to the step labeled with <label name>.
on trip goto statement	on	trip	goto	<label name>			Branches processing to the step labeled with <label name> when the inverter trips.
if statement	if	<condition>			then	<label name>	Branches processing to the step labeled with <label name> when the <condition> is met.
ifs (structured if) statement	ifs	<condition>					Starts the structured if statement.
	then						Indicates the beginning of instructions to be executed when <condition> is met.
	<instruction set>						Indicates the instructions to be executed when <condition> is met.
	else						Indicates the beginning of instructions to be executed when <condition> is not met.
	<instruction set>						Indicates the instructions to be executed when <condition> is not met.
	end if						Ends the structured if statement.
select case syntax statement	select	<conditional variable>					Executes the instructions specified after "case" when the value of <conditional variable> is <conditional value>.
	case	<conditional value>					Indicates the conditional value and the beginning of instructions to be executed.
	[case else]						Indicates the beginning of instructions to be executed when the value of <conditional variable> is not <conditional value>.
	end select						Ends the select case syntax statement.
until loop statement	until	<condition>					Executes <instruction set> repeatedly until <condition> is met.
	<instruction set>						Indicates the instructions to be executed while <condition> is not met.
	loop						Ends the "until" loop.
wait statement	wait	***.**	<condition>				Waits for "***.**" seconds.
		<condition>					Waits until <condition> is met.
while loop statement	while	<condition>					Executes <instruction set> while <condition> is met.
	<instruction set>						Indicates the instructions to be executed while <condition> is met.
	wend						Ends the "while" loop.
inc statement	inc	<variable>					Increments the value of <variable> by 1.
dec statement	dec	<variable>					Decrements the value of <variable> by 1.

Chapter 5 Instruction Words

(2) Conditional expressions

The table below lists the conditional expressions that can be used for the <condition> parameters in program control instructions.

Instruction name	Instruction format						Description
	Mnemonic code	First argument	Second argument	Third argument	Fourth argument	Fifth argument	
Comparison		<variable 2/constant>	=	<variable 3/constant>			"True" when <variable 2/constant> is equal to <variable 3/constant>
		<variable 2/constant>	<	<variable 3/constant>			"True" when <variable 2/constant> is less than <variable 3/constant>
		<variable 2/constant>	<=	<variable 3/constant>			"True" when <variable 2/constant> is not greater than <variable 3/constant>
		<variable 2/constant>	>	<variable 3/constant>			"True" when <variable 2/constant> is greater than <variable 3/constant>
		<variable 2/constant>	>=	<variable 3/constant>			"True" when <variable 2/constant> is not less than <variable 3/constant>
		<variable 2/constant>	<>	<variable 3/constant>			"True" when <variable 2/constant> is not equal to <variable 3/constant>

(3) Operational instructions

Instruction name	Instruction format						Description
	Mnemonic code	First argument	Second argument	Third argument	Fourth argument	Fifth argument	
Arithmetic operation	<variable 1>= =	<variable 2/constant>	+	<variable 3/constant>			Adds <variable 2/constant> and <variable 3/constant> and assigns the result to <variable 1>.
	<variable 1>= =	<variable 2/constant>	-	<variable 3/constant>			Subtracts <variable 3/constant> from <variable 2/constant> and assigns the result to <variable 1>.
	<variable 1>= =	<variable 2/constant>	*	<variable 3/constant>			Multiplies <variable 2/constant> by <variable 3/constant> and assigns the result to <variable 1>.
	<variable 1>= =	<variable 2/constant>	/	<variable 3/constant>			Divides <variable 2/constant> by <variable 3/constant> and assigns the result to <variable 1>.
Remainder	<variable 1>= =	<variable 2/constant>	mod	<variable 3/constant>			Divides <variable 2/constant> by <variable 3/constant> and assigns the remainder to <variable 1>.
Absolute value	<variable 1>= =		abs	<variable 3/constant>			Assigns the absolute value of <variable 3/constant> to <variable 1>.
Substitution	<variable 1>= =			<variable 3/constant>			Assigns <variable 3/constant> to <variable 1>.
Logic operation	<variable 1>= =	<variable 2/constant>	or	<variable 3/constant>			Assigns the OR of <variable 2/constant> and <variable 3/constant> to <variable 1>.
	<variable 1>= =	<variable 2/constant>	and	<variable 3/constant>			Assigns the AND of <variable 2/constant> and <variable 3/constant> to <variable 1>.
	<variable 1>= =	<variable 2/constant>	xor	<variable 3/constant>			Assigns the XOR of <variable 2/constant> and <variable 3/constant> to <variable 1>.
	<variable 1>= =		not	<variable 3/constant>			Inverts the bits of <variable 3/constant> and assigns the inverted bits to <variable 1>.

(4) Input/output control, timer control, and inverter control instructions

Instruction name	Instruction format						Description
	Mnemonic code	First argument	Second argument	Third argument	Fourth argument	Fifth argument	
General-purpose contact input	<variable> =	X (ii)					Fetches general-purpose contact information and stores it in <variable>. (0 = off, 1 = on)
	<variable> =	Xw					Fetches general-purpose contact information and stores it as word data in <variable>.
General-purpose contact output	Y (ii) =	<variable/constant>					Outputs bit data to a general-purpose contact. (0 = off, 1 = on)
	Yw =	<variable/constant>					Outputs word data to a general-purpose contact.
Inverter operation command	<input terminal> =	<variable/constant>					Operates an inverter input terminal. (0 = off, 1 = on)
Inverter operation monitoring	<variable> =	<output terminal>					Fetches information from an inverter output terminal.
	<variable> =	<input terminal> =					Fetches information from an inverter input terminal.
Delay operation	delay on	<variable 1>	TD (k)	<variable 2/constant>			Turns on the terminal specified by <variable 1> after the time specified by <variable2/constant> elapses.
	delay off	<variable 1>	TD (k)	<variable 2/constant>			Turns off the terminal specified by <variable 1> after the time specified by <variable2/constant> elapses.
Timer control	timer set	TD (k)	<variable/constant>				Sets <variable/constant> in a specified timer and starts the timer.
	timer off	TD (k)					Stops the specified timer.
Internal user contact control	<variable> =	UB (ii)					Fetches internal user contact information and stores it in <variable>. (0 = off, 1 = on)
	<variable> =	UBw					Fetches internal user contact information and stores it as word data in <variable>.
	UB (ii) =	<variable/constant>					Outputs bit data to an internal user contact. (0 = off, 1 = on)
	UBw =	<variable/constant>					Outputs word data to an internal user contact.
Parameter change	chg param	<display code>	<variable/constant>				Replaces the content of <display code> with <variable/constant>.
Parameter reading	mon param	<display code>	<variable>				Reads the content of <display code> into <variable>.
Stop inverter	stop						the inverter decelerate and stop the motor
User monitor	Umon(ii) =			<variable>			Displays <variable> on user monitor (ii)
	Umon(ii) =	<variable1>	<operators>	<variable2>			Displays the result of operation with <variable1> and <variable2> on user monitor (ii)
	<variable>=			Umon(ii)			Value of user monitor (ii) is read out to <variable>
User trip	trip	<variable>					Makes the inverter trip

Chapter 5 Instruction Words

5.2 Program Control Instructions

This section explains the details of program control instructions.

entry and end statements

Instructions to start and end the main program

- Format

Format	Description
entry	This instruction indicates the beginning of the main program. (This instruction must be described at the top of the main program.)
end	This instruction indicates the end of the main program.

- Explanation

The entry and end statements indicate the beginning and end of the main program, respectively. Each program always requires these instructions.

- Sample program

```
entry                : The main program begins.
:
FW = 1              : Start forward rotation of the motor.
wait 10.00         : Wait 10 seconds.
stop                : Stop inverter output.
wait 10.00         : Wait 10 seconds.
:
end                 : The main program ends.
```

sub and end sub statements

Instructions to start and end a subroutine

- Format

Format	Description
sub <subroutine name>	This instruction indicates the beginning of a subroutine.
end sub	This instruction indicates the end of a subroutine. Control is returned to the calling routine.

- Explanation

The sub and end sub statements indicate the beginning and end of a subroutine, respectively. <subroutine name>: Specifies the name of a called subroutine. This subroutine name is the first argument (branch destination) of the call instruction in the calling routine.

Note: Subroutines can be nested in up to eight layers. A subroutine programmed with a structured instruction (i.e., sub, for, while, until, select, or ifs) is counted as one nesting layer. Therefore, when a for-next loop statement is described in a subroutine, there are two nesting layers.

- Sample program

```
sub sub1            : Subroutine "sub1" begins.
:
FW = 1              : Start forward rotation of the motor.
wait 10.00         : Wait 10 seconds.
stop                : Stop inverter output.
wait 10.00         : Wait 10 seconds.
:
end sub            : Subroutine "sub1" ends.
```

goto statement	Instruction to branch processing unconditionally
-----------------------	--

- Format

Format	Description
goto <label name>	This instruction branches processing unconditionally to the step labeled with <label name>.

- Explanation

Use this instruction to branch processing unconditionally to the step labeled with <label name>. <label name>: Specifies the label name of the branch-target step (line).

- Sample program

```

      goto      LABEL1           : Unconditionally branch to step "LABEL1".
      :
LABEL1  FW =    1               : Branch target
    
```

on trip goto statement	Instruction to branch processing upon the occurrence of an event
-------------------------------	--

- Format

Format	Description
on trip goto <label name>	This instruction branches processing to the step labeled with <label name> when the inverter trips.

- Explanation

Use this instruction to branch processing to the step labeled with <label name> when the inverter trips.

- Sample program

```

      on      trip   goto      LABEL1   : Branch to step "LABEL1" when the inverter trips.
      :
LABEL1  Y(00) =    1               : Branch target
    
```

Chapter 5 Instruction Words

ifs-then-else-end if statements	Structured if instruction
--	---------------------------

- Format

Format	Description
ifs <condition> [then] <instruction set 1> [else] <instruction set 2> end if	When <condition> is met, this instruction executes <instruction set 1> described between "then" and "else." When <condition> is not met, this instruction executes <instruction set 2> described between "else" and "end if."

- Explanation

This instruction executes different sets of instructions according to whether <condition> is met. When <condition> is met, this instruction executes <instruction set 1>. When <condition> is not met, this instruction executes <instruction set 2>.
If neither <instruction set 1> nor <instruction set 2> is described, the ifs statement jumps to the end if statement.

- <condition>: Specifies a conditional expression among those listed in Section 3.5, "Conditions," of Chapter 3, "Syntax."
- <instruction set 1>: Specifies the instructions to be executed when <condition> is met. The instructions may be described on two or more lines. The instructions are executed in units of lines in a cycle as explained below.
- <instruction set 2>: Specifies the instructions to be executed when <condition> is met. The instructions may be described on two or more lines. The instructions are executed in units of lines in a cycle as explained below.

- Processing cycle

Note that <condition> is checked in the first cycle, and the first instruction in <instruction set 1> or <instruction set 2> is executed in the second cycle. In the third cycle, the second instruction <instruction set 1> or <instruction set 2> is executed or, if no other instruction remains in the instruction set, processing jumps to the end if statement. Therefore, the routine from "ifs" to "end if" is executed in three cycles when the instruction set contains only one instruction.
Refer to the statement execution sequence indicated by parenthesized numbers in the comment fields of the sample programs below.

- Sample programs

		When <condition> is met	When <condition> is not met
ifs	X(00) = 1	: (1)	(1)
	then		
	Y(00) = 1	: (2)	
	Y(01) = 0	: (3)	
	else		
	Y(00) = 0	:	(2)
	Y(01) = 1	:	(3)
	end if	: (4)	(4)

if statement

Instruction to branch processing unconditionally

- Format

Format	Description
if <condition> then <label name>	When <condition> is met, processing branches to the step labeled with <label name>. When <condition> is not met, processing proceeds to the next step (line).

- Explanation

Use this instruction to branch processing conditionally.

When <condition> is met, processing branches to the step labeled with <label name> described after "then."

<condition>: Specifies a conditional expression among those listed in Section 3.5, "Conditions," of Chapter 3, "Syntax."

<label name>: Specifies the label name of the branch-target step (line).

- Processing cycle

Note that <condition> check and branch processing are executed in the same cycle.

Refer to the statement execution sequence indicated by parenthesized numbers in the comment fields of the sample programs below.

- Sample program

	Yw = 0		:	Turn off terminals Y (00) to Y (05).	
	if X(00) = 1 then LABEL1	:	:	(1) When <condition> is met, branch to step "LABEL1."	
	if X(01) = 1 then LABEL2	:	:	(2) When <condition> is met, branch to step "LABEL2."	
	Y(00) = 1	:	:	(3)	
	goto LABEL3	:	:	(4)	
LABEL1	Y(01) = 1	:	:	(3)	
	goto LABEL3	:	:		
LABEL2	Y(02) = 1	:	:	(3)	
LABEL3	Y(03) = 1	:	:	(5) (4) (4)	

Chapter 5 Instruction Words

for-next loop statements for loop instruction

- Format

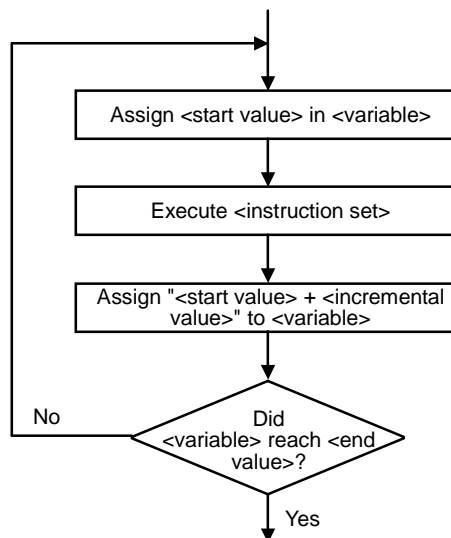
Format	Description
for <variable> <start value> <end value> [<incremental value>] <instruction set> next	This loop instruction executes <instruction set> repeatedly until <variable> reaches <end value>. Note that <variable>, which initially contains <start value>, is incremented by <incremental value> each time <instruction set> is executed.

- Explanation

Use the for loop statement to effectively describe a process for which the number of execution times is predetermined.

As a loop process, <instruction set> is executed and <variable> incremented by <incremental value> from <start value>. If <variable> reaches or exceeds <end value>, processing exits the loop. Otherwise, the loop process is repeated. Therefore, <instruction set> is always executed at least once.

The following chart shows the flow of processing.



- <variable>: Specifies the name of the variable to be used for the loop.
- <start value>: Specifies the initial value of <variable> to be applied at the beginning of the loop. You can specify a variable name or immediate value (i.e., a value that can be entered directly). The immediate value must be an integer ranging from 0 to 127. To use a larger numerical value, preset the value in a variable and specify the variable as <start value>.
- <end value>: Specifies the limit value at which to exit the loop. Processing exits the loop when <variable> reaches or exceeds <end value>. You can specify a variable name or immediate value (i.e., a value that can be entered directly). The immediate value must be an integer ranging from 0 to 127. To use a larger numerical value, preset the value in a variable and specify the variable as <end value>.
- <incremental value>: Specifies the value to be added to <variable> each time the loop is executed. You can specify a variable name or immediate value (i.e., a value that can be entered directly). The immediate value must be an integer ranging from 0 to 127. To use a larger numerical value, preset the value in a variable and specify the variable as <incremental value>.
- <instruction set>: Describes the set of instructions to be executed in one loop process. The instructions may be described on two or more lines. The instructions are executed in units of lines in a cycle as explained below.

- Processing cycle

Refer to the statement execution sequence indicated by parenthesized numbers in the comment fields of the sample programs below.

- (1): The "for" line is executed only once.
- (2) and (3): <instruction set> is executed.
- (4): <variable> is incremented in the cycle that follows the cycle in which the last instruction of <instruction set> is executed. Then, <variable> is checked to determine whether to exit the loop (in other words, the next statement is executed). When repeating the loop, processing returns to the first instruction of <instruction set> in this cycle.
- (5): This step is executed in the next cycle.
- (6) to (10): These steps are repeated in the same way as the preceding loop execution.
- (11): Processing proceeds to the following step (line).

- Sample program

					Sequence of execution
for	U(00)	0	3	1	: (1)
Y(00) =	1				: (2) (5) (8)
Y(00) =	0				: (3) (6) (9)
next					: (4) (7) (10)
Y(00) =	1				: (11)

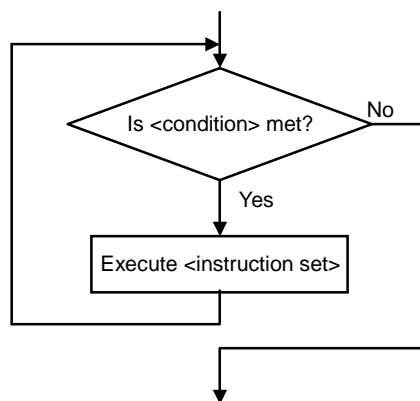
while loop statement	Instruction to conditionally execute a pre-conditioned loop
-----------------------------	---

- Format

Format	Description
while <condition> <instruction set> wend	This instruction executes <instruction set> while <condition> is met. Note that <condition> is checked before the execution of <instruction set>.

- Explanation

This instruction executes <instruction set> repeatedly as long as <condition> is met. Note that <condition> is checked before the execution of <instruction set>. If <condition> is not met, processing proceeds to the wend statement without executing <instruction set>.



- Sample program (Condition "X (00) = 0" is met after the loop is executed twice.)

					Sequence of execution
while	X(00) =	1			: (1) (5) (9)
Y(00) =	1				: (2) (6)
Y(00) =	0				: (3) (7)
wend					: (4) (8)
Y(00) =	1				: (10)

Chapter 5 Instruction Words

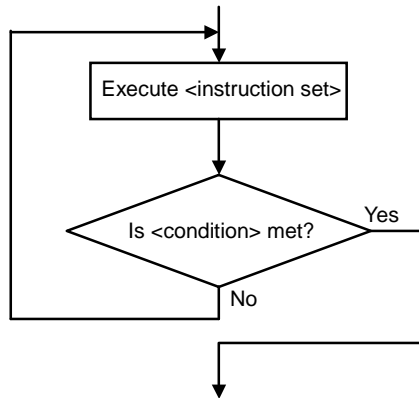
until loop statement Instruction to conditionally execute a post-conditioned loop

- Format

Format	Description
until <condition> <instruction set> loop	This instruction executes <instruction set> until <condition> is met. Note that <condition> is checked after the execution of <instruction set>.

- Explanation

This instruction executes <instruction set> repeatedly until <condition> is met. Note that <condition> is checked after the execution of <instruction set>.



- Sample program (Condition "X (00) = 0" is met after the loop is executed three times.)

until	X(00) = 1	Sequence of execution	: (1) (5) (9)
Y(00) =	1		: (2) (6) (10)
Y(00) =	0		: (3) (7) (11)
loop			: (4) (8) (12)
Y(00) =	1		: (13)

select case syntax statement Instruction to branch under multiple conditions

- Format

Format	Description
select <conditional variable> case <conditional value 1> <instruction set 1> case <conditional value 2> <instruction set 2> ... [case else] [<instruction set n>] end select	This instruction executes <instruction set 1> to <instruction set n-1> described in a case statement when <conditional variable> matches <conditional value 1> to <conditional value n-1> in the case statement, respectively. If the case else statement is described, <instruction set n> is executed when <conditional variable> does not match any of <conditional value 1> to <conditional value n-1>.

- Explanation

This instruction executes <instruction set 1> to <instruction set n-1> described in a case statement when <conditional variable> matches <conditional value 1> to <conditional value n-1> in the case statement, respectively. If the case else statement is described, <instruction set n> is executed when <conditional variable> does not match any of <conditional value 1> to <conditional value n-1>.

- **Sample program** (when Xw = 2)

U(00) =	Xw	Sequence of execution	: (1)
select	U(00)		: (2)
case	0		: (3)
Yw =	U(00)		:
case	1		:
Yw =	U(00)		:
case	2		: (4)
Yw =	U(00)		: (4)
case	4		:
Yw =	U(00)		:
case else			:
Yw =	0		:
end select			: (5)
Y(00) =	1		: (6)

call statement

Instruction to unconditionally branch to a subroutine

- **Format**

Format	Description
call <subroutine name>	This instruction branches processing unconditionally to the subroutine specified by <subroutine name>.

- **Explanation**

This instruction branches processing unconditionally to the subroutine specified by <subroutine name>. After the subroutine is executed, processing proceeds to the instruction that follows the calling step.

- **Sample program**

```

entry
call      SUB1                : Call subroutine "SUB1".
:
end

sub      SUB1                : Called subroutine
:
Y(00) =  1
:
sub end
    
```

Chapter 5 Instruction Words

inc statement	Instruction to increment a variable
----------------------	-------------------------------------

- Format

Format	Description
inc <variable>	This instruction increments <variable> by 1.

- Explanation

This instruction adds 1 to the value of <variable>.

- Sample program

(Code area [Code Window])

```

entry
LOOP   inc      U(00)           : Assign "U (00) + 1" to U (00).
        U(00) =  U(00)  and  U(01) : Mask the 6 low-order bits of U (00).
        Yw =    U(00)           : Output the content of U (00) to terminals Y (00) to Y (05).
        wait   0.5             : Wait 0.5 second.
        goto   LOOP
        end
    
```

(Data area [Data Window])

```

U(00) = 255
U(01) = 63
    
```

dec statement	Instruction to decrement a variable
----------------------	-------------------------------------

- Format

Format	Description
dec <variable>	This instruction decrements <variable> by 1.

- Explanation

This instruction subtracts 1 from the value of <variable>.

- Sample program

(Code area [Code Window])

```

entry
LOOP   dec      U(00)           : Assign "U (00) - 1" to U (00).
        U(00) =  U(00)  and  U(01) : Mask the 6 low-order bits of U (00).
        Yw =    U(00)           : Output the content of U (00) to terminals Y (00) to Y (05).
        wait   0.5             : Wait 0.5 second.
        goto   LOOP
        end
    
```

(Data area [Data Window])

```

U(00) = 255
U(01) = 63
    
```

Label definition statement

Statement to define a label

- Format

Format	Description
<label name>	This statement defines <label name>.

- Explanation

Use this statement to define <label name> to be used in the goto or other instructions. The statement is not executed when described alone.

- Sample program

```

:
goto LABEL1 : Branch to step "LABEL1".
:
LABEL1 Y(00) = 1 : Branch-target step (line)
:
    
```

wait statement

Instruction to make processing wait

- Formats

Format	Description
1 wait iii.ii	This instruction makes processing wait for "iii.ii" seconds.
2 wait <condition>	This instruction makes processing wait until <condition> is met.

- Explanation

Format 1: This instruction makes processing wait for "iii.ii" seconds. After "iii.ii" seconds elapse, the next instruction is executed.

Format 2: This instruction makes processing wait until <condition> is met. After <condition> is met, the next instruction is executed.

- Sample programs

```

Sample 1: Format 1
:
wait 1.00 : Wait 1 second.
Y(00) = 1
:
    
```

```

Sample 2: Format 2
:
wait X(00) = 1 : Wait until condition "X (00) = 1" is met.
LABEL1 Y(00) = 1
:
    
```

Chapter 5 Instruction Words

5.3 Operators

You can describe the following dyadic operations using the operators:

Format	Description
<variable 1> = <variable 2> + <variable 3>	Addition
<variable 1> = <variable 2> - <variable 3>	Subtraction
<variable 1> = <variable 2> * <variable 3>	Multiplication
<variable 1> = <variable 2> / <variable 3>	Division
<variable 1> = <variable 2> mod <variable 3>	Remainder
<variable 1> = abs <variable 3>	Absolute value
<variable 1> = <variable 3>	Substitution
<variable 1> = <variable 2> or <variable 3>	OR (logical addition)
<variable 1> = <variable 2> and <variable 3>	AND (logical product)
<variable 1> = <variable 2> xor <variable 3>	XOR (exclusive-OR)
<variable 1> = not <variable 3>	NOT (negation)

Note 1: <variable 2> can be a constant ranging from 0 to 127.

Note 2: <variable 3> can be a constant ranging from -2^{31} to $2^{31}-1$.

5.4 Conditional Expressions

You can use the following conditional expressions (as <condition>) in instruction statements:

Format	Description
<variable 1> = <variable 2>	"True" when <variable 1> is equal to <variable 2>
<variable 1> < <variable 2>	"True" when <variable 1> is less than <variable 2>
<variable 1> <= <variable 2>	"True" when <variable 1> is not greater than <variable 2>
<variable 1> > <variable 2>	"True" when <variable 1> is greater than <variable 2>
<variable 1> >= <variable 2>	"True" when <variable 1> is not less than <variable 2>
<variable 1> <> <variable 2>	"True" when <variable 1> is not equal to <variable 2>

Note: <variable 1> and <variable 2> can be constants ranging from 0 to 127.

5.5 Input/Output Control Instructions

This section describes the details of input/output control instructions.

X () or Xw (contact input)	Instruction to access contact inputs
------------------------------------	--------------------------------------

- Formats

	Format	Description
1	<variable>= X (ii) (ii = 00 to 07)	This instruction assigns the ii'th bit of contact input data to <variable>.
2	<variable>= Xw	This instruction assigns contact input data as word data to <variable>.

- Explanation

This instruction fetches the status of contact input terminals X (00) to X (07) and stores it in <variable> in units of bits or words. You cannot write data to <variable>, which is read-only. Details of the formats are explained below.

Format 1: With this format, the instruction assigns the status of the ii'th bit of contact input data to <variable>. (0 = off, 1 = on)

(Examples)

When terminal X (00) is off: UB= X (00) (UB (00) = 0)

When terminal X (00) is on: UB= X (01) (UB (00) = 1)

Format 2: With this format, the instruction assigns the status of contact input data as word data to <variable>. (0 = off, 1 = on)

(Examples)

When terminals X (00) to X (03) are on and terminals X (04) to X (07) are off: Uw= Xw (Uw = 15)

When terminals X (00) to X (02) are off and terminals X (03) to X (07) are on: Uw= Xw (Uw = 248)

Note 1: The setting of terminal active state (C011 to C018) is reflected in the polarity (on or off) of contact inputs X (00) to X (07) and Xw. When creating a user program, consider the on and off states of actual intelligent input terminals 1 to 8.

Note 2: Since this instruction reads the internal input terminal data at least twice (in two execution cycles), storing the read data in <variable> is delayed by at least two execution cycles.

Note 3: Wiring noise or switch chattering may cause incorrect read data to be set in <variable>. To avoid such problems, design your program so that it will verify the read data.

- Sample programs

Sample 1: Program to invert the status data of input terminal X (01) and output it to output terminal Y (05)

```

entry
LOOP   UB(00)=  X(01)           : Fetch the status of X (01) and store it as internal
                                     user contact data.
       ifs      UB(00)  =    0           : Branch according to the conditional expression.
       then
       Y(05)=   1                   : Turn Y (05) on when the condition is met.
       else
       Y(05)=   0                   : Turn Y (05) off when the condition is not met.
       end if
       goto    LOOP
end
    
```

Sample 2: Program to acquire input terminal status as word data and output only the status of terminals X (02) to X (05) as word data to output terminals Y (00) to Y (03)

```

LOOP   U(00)=           Xw           : Fetch input terminal status and store it in the user
                                     variable.
       U(00)=   U(00)   /    4           : Cut off the data of X (00) to X (01).
       U(00)=   U(00)   and  15          : Mask the data of X (06) to X (07).
       Yw=     U(00)           : Output the data of X (02) to X (05) as word data.
       goto    LOOP
end
    
```

Chapter 5 Instruction Words

Y () or Yw (contact output)

Instruction to access contact outputs

- Formats

	Format	Description
1	Y (ii)= <variable> or <constant> (ii = 00 to 07)	This instruction assigns <variable> or <constant> to the ii'th bit of contact output data.
2	Yw= <variable> or <constant>	This instruction assigns <variable> or <constant> as word data to contact outputs.

- Explanation

This instruction writes <variable> or <contact> to contact output terminals Y (00) to X (05) in units of bits or words to output the data. You can write and read data to and from <variable> or <contact>. You can also fetch and store the status data of contact output terminals Y (00) to Y (05) in <variable>. Details of the formats are explained below.

Format 1: With this format, the instruction outputs <variable> to the ii'th bit of contact output terminal.
(0 = off, 1 = on, 2 or more = off)

(Examples)

To turn terminal Y (00) off: Y (00)= 0

To turn terminal Y (01) on: Y (01)= 1

Format 2: With this format, the instruction outputs <variable> as word data to contact output terminals.
(Examples)

To turn terminal Y (00) on and turn terminals Y (01) to Y (05) off: Yw= 1

To turn terminals Y (00) to Y (04) off and turn terminal Y (05) on: Yw= U (00) (U (00) = 32)

Note: The setting of terminal active state (C031 to C036) is reflected in the polarity (on or off) of contact inputs Y (00) to Y (05) and Yw when the data is output to intelligent output terminals 11 to 15 and the relay output terminal. When creating a user program, consider the on and off states of actual intelligent output terminals.

- Sample programs

Sample 1: Program to turn terminals Y (00) to Y (05) on sequentially while the output frequency is increased in 10-Hz steps. (The inverter operation is the same as that programmed in sample 2.)

(Code area [Code Window])

```

entry
  SET-Freq=      6000      : Set the output frequency to 60 Hz.
  ACCEL=         3000      : Set the acceleration time to 30 seconds.
  DECEL=         3000      : Set the deceleration time to 30 seconds.
  Yw=    0
  FW=    1      : Start forward rotation of the motor.
LOOP
  if    FM    <    U(00)  then  LBL1  : When the output frequency is less than 10
  Hz,
    Y(00)= 1      : turn Y (00) on.
  if    FM    <    U(01)  then  LBL1  : When the output frequency is less than 20
  Hz,
    Y(01)= 1      : turn Y (01) on.
  if    FM    <    U(02)  then  LBL1  : When the output frequency is less than 30
  Hz,
    Y(02)= 1      : turn Y (02) on.
  if    FM    <    U(03)  then  LBL1  : When the output frequency is less than 40
  Hz,
    Y(03)= 1      : turn Y (03) on.
  if    FM    <    U(04)  then  LBL1  : When the output frequency is less than 50
  Hz,
    Y(04)= 1      : turn Y (04) on.
  if    FM    <    U(05)  then  LBL2  : When the output frequency is less than 60
  Hz,
    Y(05)= 1      : turn Y (05) on.
  LBL1  goto    LOOP

```

```
LBL2      FW=  0  
          Wait RUN  =  0  
          end
```

: Decelerate and stop the motor.

(Data area [Data Window])

```
U(00) = 1000      U(01) = 2000  
U(02) = 3000      U(03) = 4000  
U(04) = 5000      U(05) = 6000
```

Chapter 5 Instruction Words

Sample 2: Program to output codes sequentially to terminal Yw while the output frequency is increased in 10-Hz steps. (The inverter operation is the same as that programmed in sample 1.)

(Code area [Code Window])

```

entry
SET-Freq=      6000      : Set the output frequency to 60 Hz.
ACCEL=         3000      : Set the acceleration time to 30
                        seconds.
DECEL=         3000      : Set the deceleration time to 30
                        seconds.

Yw=            0
FW=            1
LOOP          if      FM    <    U(00)  then  LBL1  : Start forward rotation of the motor.
                        : When the output frequency is less than
                        10 Hz
                        if      FM    <    U(01)  then  LBL2  : When the output frequency is less than
                        20 Hz
                        if      FM    <    U(02)  then  LBL3  : When the output frequency is less than
                        30 Hz
                        if      FM    <    U(03)  then  LBL4  : When the output frequency is less than
                        40 Hz
                        if      FM    <    U(04)  then  LBL5  : When the output frequency is less than
                        50 Hz
                        if      FM    <    U(05)  then  LBL6  : When the output frequency is less than
                        60 Hz
LBL1          if      FM    =    U(05)  then  LBL8  : When the output frequency is 60 Hz
                        : Output "1" to Yw.
LBL2          Yw=      1
                        goto    LBL7
LBL3          Yw=      2
                        goto    LBL7
LBL4          Yw=      3
                        goto    LBL7
LBL5          Yw=      4
                        goto    LBL7
LBL6          Yw=      5
                        goto    LBL7
LBL7          Yw=      6
                        goto    LBL7
LBL8          Yw=      0
                        goto    LOOP
LBL2          FW=      0
                        Wait    RUN    =    0
end

```

(Data area [Data Window])

```

U(00) = 1000
U(01) = 2000
U(02) = 3000
U(03) = 4000
U(04) = 5000
U(05) = 6000

```


UB () or UBw (internal user contact control)

Instruction to access internal user contacts

Symbol	Variable name	Range of values	Default	Unit	Data size	Attribute
UB (00) to UB (07)	Internal user contact (bit access)	0: OFF 1: ON	0	-	Unsigned 1-word data	Readable and writable
UBw	Internal user contact (word access)	0 to 255	0	-	Unsigned 1-word data	Readable and writable

- Formats

	Format	Description
1	<variable>= UB (ii) (ii = 00 to 07)	This instruction assigns the ii'th bit of internal user contact data to <variable>.
2	UB (ii)= <variable> or <constant> (ii = 00 to 07)	This instruction assigns <variable> or <constant> to the ii'th bit of internal user contact data.
3	<variable>= UBw	This instruction assigns internal user contact data as word data to <variable>.
4	UBw= <variable> or <constant>	This instruction assigns <variable> or <constant> as word data to internal user contact data.

- Explanation

Use this instruction to control the internal contacts that the user can use for general purposes. The inverter has eight general-purpose contacts that are writable and readable by bit access (UB (00) to UB (07)) or word access (UBw). Details of the formats are explained below.

Format 1: With this format, the instruction reads the status of the ii'th bit of internal user contact data into <variable>. (0 = off, 1 = on)

Format 2: With this format, the instruction writes <variable> or <constant> to the ii'th bit of internal user contact data. (0 = off, 1 = on, 2 or more = off)

Format 3: With this format, the instruction reads internal user contact data as word data into <variable>.

Format 4: With this format, the instruction writes <variable> or <constant> as word data to internal user contact data.

- Sample programs

Sample 1: Statement to read internal user contact status as bit data (format 1)

```

:
U(00)=          UB(00)          : Assign bit data of UB (00) to U (00).
:

```

Sample 2: Statement to turn an internal user contact on (format 2)

```

:
UB(00)=         1                : Assign "1" (bit-on status) to UB (00).
:

```

Sample 3: Statement to read internal user contact status as word data (format 3)

```

:
U(00)=          UBw              : Assign word data of UBw to U (00).
:

```

Sample 4: Statement to change internal user contact status in units of words (format 4)

```

:
UBw(00)=        U(00)            : Write word data of U (00) to UBw.
:

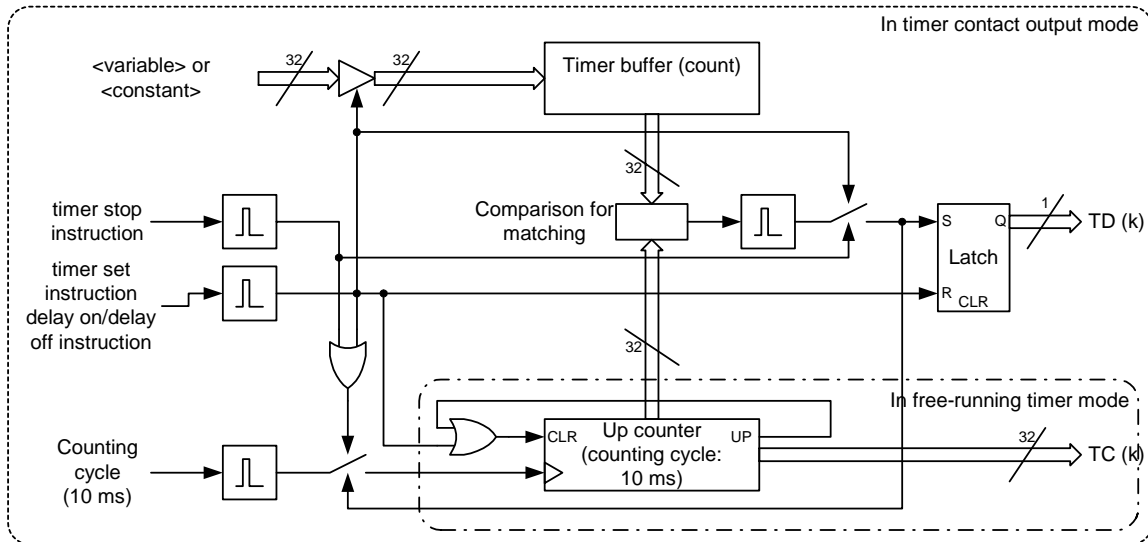
```

5.6 Timer Control Instructions

The easy sequence function of the inverter has a timer function that can be used in the following two modes:

- (1) Free-running timer mode
- (2) Timer contact output mode (timer-start, timer-stop, and delay operations)

The timer function uses eight timer counter circuits that are configured as shown in the figure below.



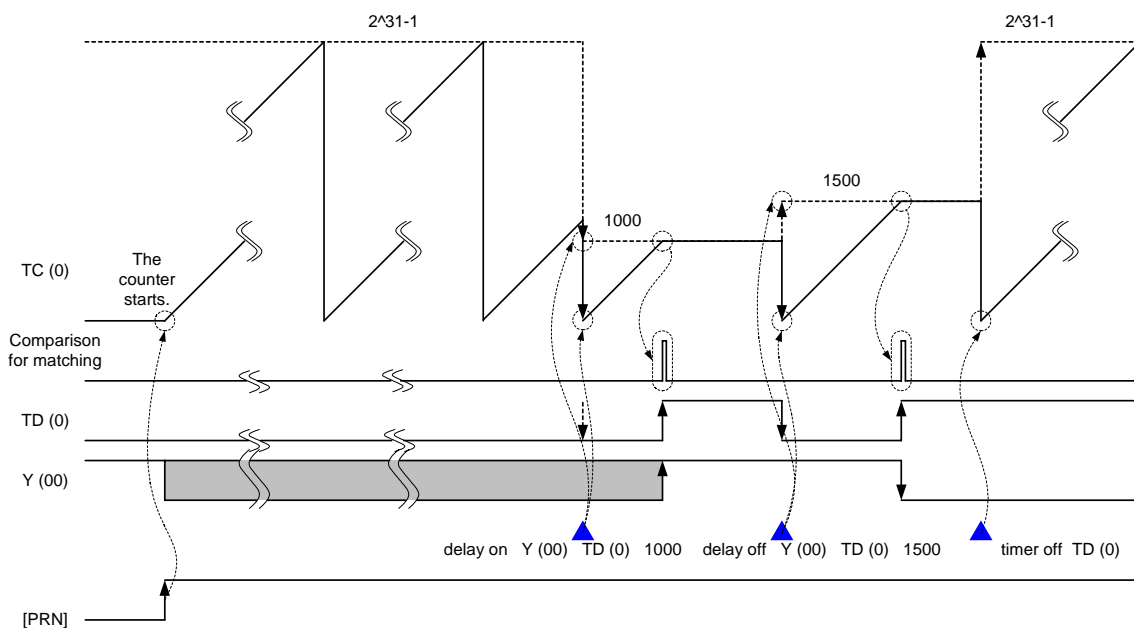
TC (k): Timer counter variable (up counter)
 TD (k): Timer contact (one-shot timer)

Block diagram for timer function

The timer counter is a 31-bit up counter that runs in a 10-ms cycle (1 count per 10 ms), and operates as a free-running timer when the execution of a easy sequence program is started.

When the timer set, delay on, or delay off instruction is executed, the timer counter is cleared and restarted. While the timer counter is operating, its count is compared with the count specified by a variable or constant to determine whether they match. When the counts match each other, the timer counter stops counting.

When the timer off instruction is executed, the timer counter is cleared and restarted. Subsequently, the timer counter operates as a free-running timer.



Example of timer function operation

timer set (timer-start instruction) Instruction to set and start the timer counter

- Format

Format	Description
timer set TD (k) <variable> or <constant>	This instruction sets <variable> or <constant> in the k'th timer and starts the timer counter.

- Explanation

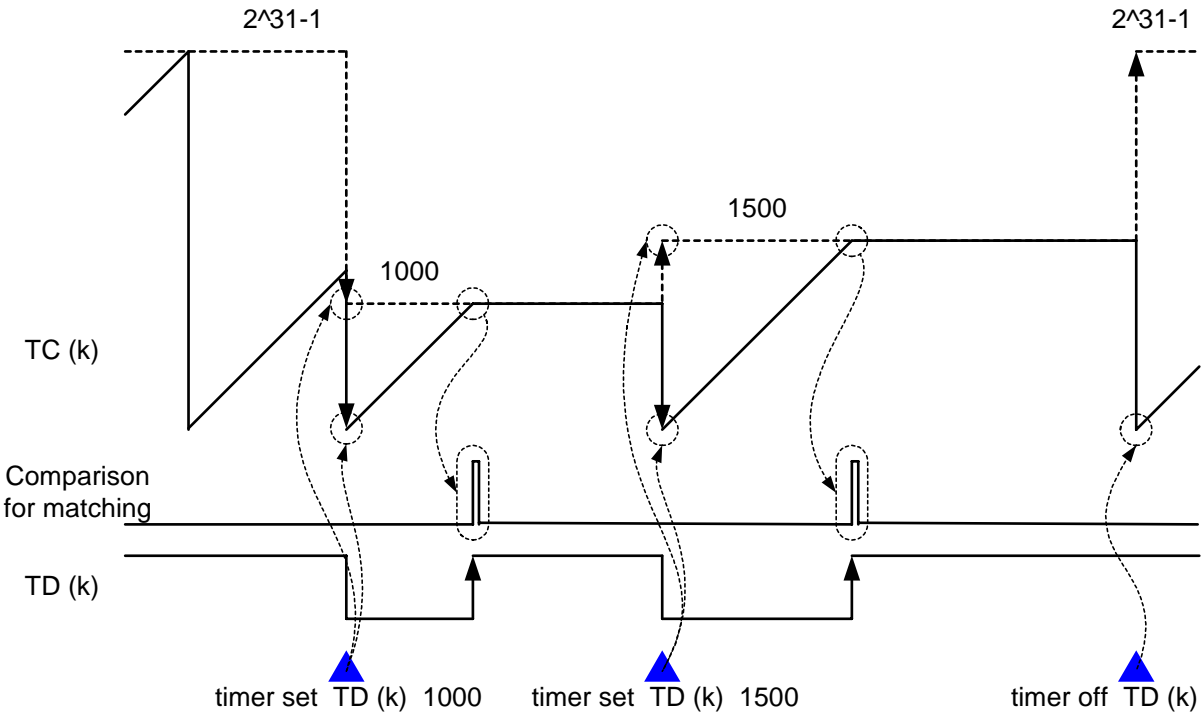
- (1) The timer set instruction sets <variable> or <constant> in the k'th timer buffer, clears the timer counter (up counter) "TC (k)" to zero, and then initiates counting by the timer counter. Then, the value of timer contact variable "TD (k)" is "0" (off).
- (2) Subsequently, the instructions described after the timer set instruction are executed.
- (3) When the timer counter "TC (k)" reaches the specified count, the value of timer contact variable "TD (k)" changes to "1" (on) (only once). Then, the timer counter "TC (k)" stops counting.

- Sample program

```

:
FW=      1
timer set TD(0)   5.00           : Start the 5-second timer counter.
wait     TD(0)   =      1       : Wait until "1" (on) is set in TD (0).
Y(00)=   1
:
    
```

- Example of operation (timing chart)



Timing chart for operation using the timer set instruction

Chapter 5 Instruction Words

timer off (timer-stop instruction)	Instruction to stop the timer
---	-------------------------------

- Format

Format	Description
timer off TD (k)	This instruction clears the k'th timer and operates it as a free-running timer.

- Explanation

This instruction clears the k'th timer counter (up counter) "TC (k)" to zero, and starts the timer counter in free-running timer mode. Then, the value of timer contact variable "TD (k)" is not changed. The timer counter "TC (k)" is switched from timer contact output mode to free-running timer mode.

- Sample program

```

:
wait      X(00) =      1           : Wait until terminal X (00) is turned on.
Y(00)=    1           : Turn terminal Y (00) on.
delay on  Y(01) TD(1) 1000       : Turn terminal Y (01) on with a delay.
timer set TD(0) 15.00           : Start the 15-second timer counter.
if        X(01) =      0   then  LBL1 : Wait when terminal X (01) is off.
timer off TD1                  : Clear timer counter TC (1).
LBL1     wait   TD(0) =      1     : Wait until the 15-second timer counter ends
                                         counting.
Y(02)=    1           : Turn terminal Y (02) on when the process ends.
:

```

delay on or delay off (delay operation instruction)	Instruction to turn a variable on or off with a delay
--	---

- Formats

Format	Description
delay on <variable 1> TD (k) <variable 3> or <constant>	This instruction sets the count of the k'th timer in <variable 3> or <constant> and starts the timer counter. When timer output "TD (k)" is turned on, <variable 3> is turned on.
delay off <variable 1> TD (k) <variable 3> or <constant>	This instruction sets the count of the k'th timer in <variable 3> or <constant> and starts the timer counter. When timer output "TD (k)" is turned on, <variable 3> is turned off.

- Explanation

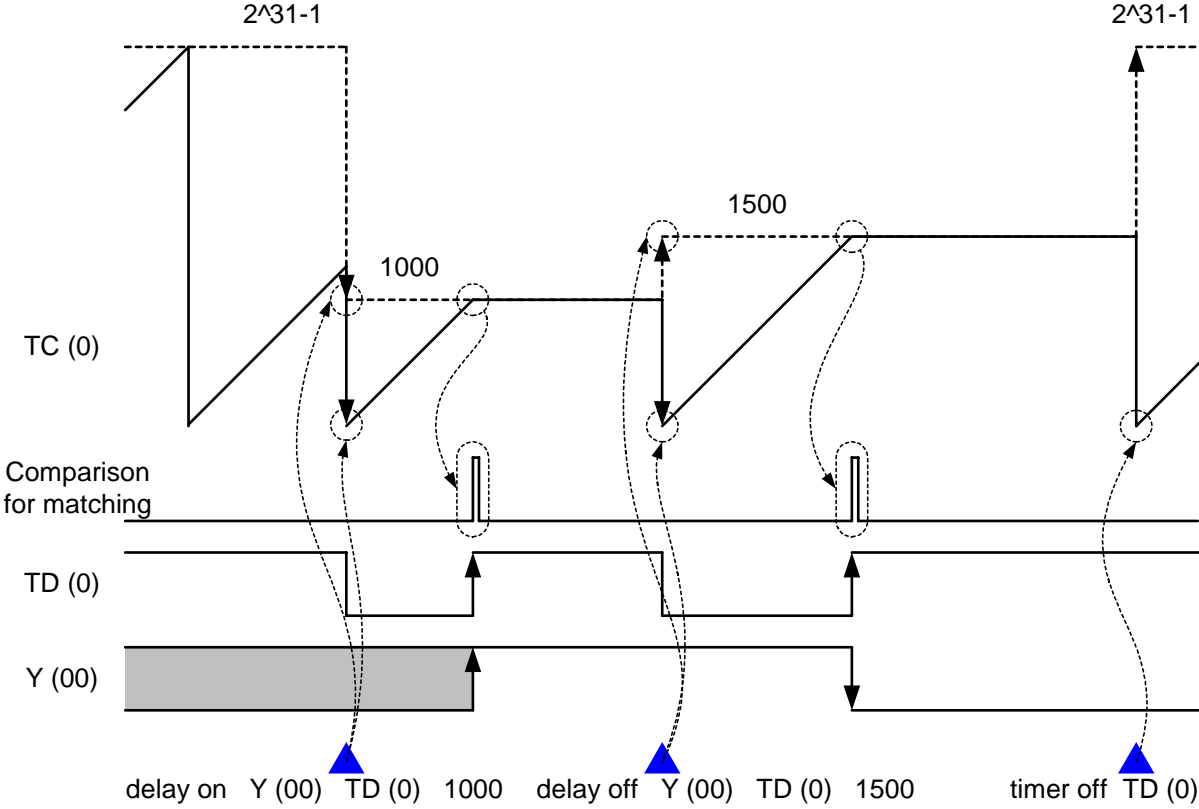
- (1) The delay on (or delay off) instruction sets <variable> or <constant> in the k'th timer buffer, clears the timer counter (up counter) "TC (k)" to zero, and then initiates counting by the timer counter. Then, the value of timer output variable "TD (k)" is "0" (off).
- (2) Subsequently, the instructions described after the delay on (or delay off) instruction are executed.
- (3) When the count of timer counter "TC (k)" matches the count preset in the timer buffer, the value of timer output variable "TD (k)" changes to "1" (on) (only once), and <variable 1> is turned on (or off). Then, the timer counter "TC (k)" stops counting.

- **Sample program:** Program to make the inverter alternately repeat forward rotation of the motor at 60 Hz and reverse rotation of the motor at 10 Hz

```

:
U(00)=          1000      : The output frequency for reverse rotation is 10 Hz.
ACCEL=          1000      : Set the acceleration time to 10 seconds.
DECEL=          1000      : Set the deceleration time to 10 seconds.
LOOP SET-Freq=    6000
FW=             1         : Accelerate the forward rotation speed up to 60 Hz.
wait RUN =      1         : Wait until the motor operates at 60 Hz.
delay off FW TD(0) 15.00  : Start the 15-second timer counter.
:                   : Turn the FW terminal off after 15 seconds elapse.
wait RUN =      0         : Wait until the motor stops.
delay on RV TD(0) 1.00   : Start reverse rotation after 1 second elapses.
SET-Freq=       1000      : Accelerate the reverse rotation speed up to 10 Hz.
:
wait FM =        U(00)    : Wait until the output frequency reaches 10 Hz.
RV=             0         : Decelerate and stop the motor.
Wait RUN =      0         : Wait until the motor stops.
goto LOOP
:
    
```

- **Example of operation (timing chart)**



Timing chart for operation using the delay on and delay off instructions

5.7 Inverter Control Instructions

Inverter operation command	Instruction to turn the input terminal function on or off
-----------------------------------	---

- Format

Format	Description
<input terminal function>= <variable> or <constant>	This instruction turns <input terminal function> of the inverter on or off according to the value of <variable> or <constant>.

- Explanation

This instruction turns the inverter input terminal specified by <input terminal function> on or off according to the value of <variable> or <constant>. When the value of <variable> or <constant> is 0, 1, or 2 or more, the input terminal specified by <input terminal function> is turned off, on, or off, respectively.

The function and operation of the specified input terminal are the same as those that can be specified by the terminal functions (C001 to C008) on the inverter. For details, refer to the SJ700 Series Inverter Instruction Manual.

- Sample program: Program to make the inverter alternately repeat forward acceleration and deceleration, and reverse acceleration and deceleration of the motor at 60 Hz

```

:
SET-Freq=      6000
LOOP  FW=   1           : Turn the FW terminal on.
      wait X(01) = 1    : Wait until X (01) is turned on.
      FW=   0           : Turn the FW terminal off.
      wait RUN  = 0    : Wait until the motor stops.
      RV=   1           : Turn the RV terminal on.
      wait X(02) = 1    : Wait until X (02) is turned on.
      RV=   0           : Turn the RV terminal off.
      wait RUN  = 0    : Wait until the motor stops.
      goto LOOP
:

```

Inverter operation monitoring instruction

Instruction to monitor the output terminal function

- Format

Format	Description
<variable 1>= <output terminal function>	This instruction fetches the on/off status of <output terminal function> of the inverter and stores it in <variable 1>.

- Explanation

This instruction fetches the on/off status of the inverter output terminal specified by <output terminal function> and stores it in <variable 1>. When the specified output terminal is off, the value of <variable 1> is "0"; when it is on, the value of <variable 1> is "1".

The function and operation of the specified output terminal are the same as those that can be specified by the terminal function (C021 to C026) on the inverter. For details, refer to the SJ700 Series Inverter Instruction Manual.

- Sample program

(Code area [Code Window])

```

:
SET-Freq=      6000      : Set the output frequency to 60 Hz.
ACCEL=         1000      : Set the acceleration time to 10 seconds.
DECEL=         1000      : Set the deceleration time to 10 seconds.
FW=           1
wait          X(01) = 1   : Wait until X (01) is turned on.
ACCEL=        3000      : Increase the acceleration time to 30
                        seconds.
wait          FA1 = 1    : Wait until the output frequency reaches
                        the set frequency.
LOOP  if      X(02) <> 1   then  LBL1 : Wait until X (02) is turned on.
      SET-Freq= 500      : Reduce the set frequency to 5 Hz.
LBL1  UB(00)= ZS
      if      UB(00) <> 1   then  LOOP
      DECEL=   3000      : Increase the deceleration time to 30
                        seconds when ZS is on.
      wait    10.00      : Operate the motor at 5 Hz for 10
                        seconds.
      FW=     0           : Decelerate and stop the motor.
:

```

(Parameter)

C063 = 5.0Hz

Chapter 5 Instruction Words

User Monitor		Operator display variable				
Umon(00) to Umon(02)	Variable name	Range of values	Default	Unit	Data size	Attribute
	User monitor 0 to 2	$-2^{31} - 2^{31}-1$	0	-	Signed 2-word data	Readable and writable

- Format

Format	Description
Umon(ii) = <variable>	Displays <variable> on user monitor (ii)
Umon(ii) = <variable1> <operator> <variable2>	Displays the result of operation with <variable1> and <variable2> on user monitor (ii)
<variable> = Umon(ii)	Value of user monitor (ii) is read out to <variable>

- Explanation

These variables can be used to monitor the causes of the last six trips made by the inverter. The data monitored with this variable corresponds to the data monitored by trip monitoring functions 1 to 6 (d081 to d086). These variables are read-only.

- **Sample program:** Program to display the summation of U(01) and U(02) on user monitor 2 (d027) (Code area [Code Window])

```

:
Umon(02)= U(01) + U(02)

```

User Trip	User trip issue command
-----------	-------------------------

- Format

Format	Description
trip <variable>	Makes the inverter trip

- Explanation

This instruction makes inverter trip. Range of <variable> is 0 to 9.

- **Sample program:** Program to issue the user trip 2 (E52) when the summation of variable 1 and variable2 exceeds 20 (Code area [Code Window])

```

:
U(00)= U(01) + U(02)
if U(00) > 20
then
trip 2
else
:

```


stop statement

Instruction to stop motor operation by the inverter

- Format

Format	Description
stop	This instruction makes the inverter decelerate and stop the motor.

- Explanation

This instruction makes the inverter decelerate and stop the motor.

When the FW terminal is on (FW = 1) or the RV terminal is on (RV = 1), this instruction turns off the FW terminal (FW = 0) or RV terminal (RV = 0).

- Sample program: Program to make the inverter operate the motor for forward or reverse rotation at a constant speed for 10 seconds

```

:
if      X(00)    <>    1    then  LBL1
FW=    1
: When X (00) is on, operate the motor for forward
: rotation.
LBL1   goto    LBL2
RV=    1
: When X (00) is on, operate the motor for reverse
: rotation.
LBL2   wait    FA1    =    1
: Wait until the motor rotates at a constant speed.
: Operate the motor at 5 Hz for 10 seconds.
wait   10.00
stop
:
: Decelerate and stop the motor. (FW = 0 or RV = 0)

```

chg param statement

Instruction to change a parameter setting

- Format

Format	Description
chg param <display code> <variable> or <constant>	This instruction changes the setting of the inverter parameter specified by <display code> to <variable> or <constant>.

- Explanation

<display code> specifies the parameter number of the inverter parameter of which the setting is to be changed. The range of parameter settings depends on the standard inverter specifications. For the inverter parameters and ranges of their settings, refer to the SJ700 Series Inverter Instruction Manual. Specify an integer as the desired new setting of the parameter in <variable> or <constant>. To specify a numerical value other than 0 to 127, preset the value in a variable and specify the variable as <variable>. The changed parameter setting is reflected in the inverter in a 40-ms cycle, which conforms to the standard inverter specifications. If, however, you directly access the inverter's EEPROM, the change is reflected in the inverter in the same cycle as that of instruction execution.

Note: You cannot specify any of parameters "U001" to "U012" in <display code>.

- Sample program: Program to change the overload restriction level according to output frequency (Code area [Code Window])

```

:
U(00)=                2000      : Set 200% in variable "U (00)."  

U(01)=                1500      : Set 150% in variable "U (01)."  

U(02)=                1000      : Set 100% in variable "U (02)."  

U(03)=                1000      : Set 10 Hz in variable "U (03)."  

chg param   b022   U(00)        : Change the setting of "b022" to 200.0%  

FW=        1  

wait       FM     >=          U(03) : Wait until the output frequency reaches 10 Hz.  

chg param   b022   U(01)        : Change the setting of "b022" to 150.0%.  

wait       FA1    =            1    : Wait until acceleration ends.  

chg param   b022   U(02)        : Change the setting of "b022" to 100.0%.  

:

```

(Parameter)

b031 = 10 (can be updated during operation)

Chapter 5 Instruction Words

mon param statement	Instruction to read a parameter
----------------------------	---------------------------------

- Format

Format	Description
mon param <display code> <variable>	This instruction assigns the content of the inverter parameter specified by <display code> to <variable>.

- Explanation

This instruction reads the content of the inverter parameter specified by <display code>, and assigns the read content to <variable>. The range of parameter settings depends on the standard inverter specifications. For the inverter parameters and ranges of their settings, refer to the SJ700 Series Inverter Instruction Manual.

- **Sample program:** Program to check whether the inputs of frequency command and acceleration/deceleration time are assigned to the easy sequence function

(Code area [Code Window])

```

entry
Yw=      0
mon param A001  U(00)      : Assign the content of A001 to U (00).
mon param P031  U(01)      : Assign the content of P031 to U (01).
if        U(00) <> 7   then  SKIP  : When A001 is not "07"
if        U(01) <> 3   then  SKIP  : When P031 is not "03"
Y(00)=    1                : Turn Y (00) on.
goto      LBL1
SKIP     Y(01)=    1                : Turn Y (01) on when the process ends.
LBL1     end

```

5.8 Other Reserved Variables

U (00) to U (31)	Variable name	Range of values	Default	Unit	Data size	Attribute
	User-defined variable	0 to 65535	Data stored in P100 to P131	-	Unsigned 1-word data	Readable and writable

- Explanation

User-designed variables are the general-purpose functions that can be used as unsigned 1-word variables regardless of format. The data written from a sequence program to the user-defined variables is not stored in the inverter's EEPROM. The variables will restore the initial settings when the inverter power is turned off. The user-defined variables correspond to inverter parameters "P100" to "P131". You can also change the settings of user-defined variables from the digital operator. The changes made from the digital operator will be stored in EEPROM.

- Sample program

```

:
U(00)=      U(00)  +      U(01)
U(02)=      U(00)  *      U(02)
U(03)=      U(00)  mod    U(01)
:
    
```

UL (00) to UL (07)	Variable name	Range of values	Default	Unit	Data size	Attribute
	Internal user variable	-2^{31} to $2^{31}-1$	0	-	Signed 2-word data	Readable and writable

- Explanation

Internal user variables are the general-purpose functions that can be used as unsigned 2-word variables, for example, to temporarily store arithmetic operation results.

Note: If an arithmetic operation causes data overflow, an execution error (E45) will result.

- Sample program

```

:
UL(00)=      Tmon           : Acquire the output torque data.
if      UL(00) >= 0      then SKIP : When the output torque is a positive value
UL(01)=      -1
UL(00)=      U(01) *      UL(00)   : When the output torque is a negative value (x -1)
SKIP    U(05)=      UL(00)
U(05)=      U(05) *      100       : Convert the scale.
U(05)=      U(05) /      300
YA(1)=      U(05)           : Output the data to general-purpose analog output.
:
    
```

Chapter 5 Instruction Words

SET-Freq	Variable name	Range of values	Default	Unit	Data size	Attribute
	Output frequency setting	0 to 40000	0	0.01 Hz	Unsigned 1-word data	Readable and writable

- Explanation

This variable can be used to read and write the frequency specified by the output frequency setting (F001) in the inverter. (See Notes 1 and 2.) The setting of this variable corresponds to inverter parameter "F001". The data written to this variable is not stored in the inverter's EEPROM. This variable will restore the initial setting when the inverter power is turned off. When the inverter receives an operation command (FW = 1 or RV = 1), it accelerates the motor up to the frequency that was set last.

Note 1: To reflect the frequency written in this variable as the set frequency, you must change the setting of frequency source setting (A001) to "07" (PRG).

Note 2: This variable can be read regardless of the setting of "A001". The currently applied set frequency is read from this variable.

- **Sample program:** Program to alternately repeat forward rotation of the motor at 60 Hz and reverse rotation at 10 Hz

(Code area [Code Window])

```

entry
LOOP   SET-Freq=      6000      : Set the output frequency to 60 Hz.
      FW=            1          : Start forward rotation of the motor.
      wait          FA1   =    1      : Wait until the output frequency reaches the set
frequency.
      wait          10.00        : Operate the motor for 10 seconds.
      FW=            0          : Decelerate and stop the motor.
      wait          RUN    =    0      : Wait until the motor stops.
      SET-Freq=     1000        : Change the set frequency to 10 Hz.
      RV=            1          : Start reverse rotation of the motor.
      wait          FA1   =    1      : Wait until the output frequency reaches the set
frequency.
      wait          10.00        : Operate the motor for 10 seconds.
      RV=            0          : Decelerate and stop the motor.
      wait          RUN    =    0      : Wait until the motor stops.
      goto          LOOP
end

```

(Parameter)

A001 = 07

ACCEL	Variable name	Range of values	Default	Unit	Data size	Attribute
	Acceleration time setting	0 to 360000	0	0.01 second	Unsigned 2-word data	Readable and writable

- Explanation

This variable can be used to read and write the motor acceleration time in the inverter. The acceleration time setting using this variable is enabled only when the setting of accel/decel time input selection (P031) is "03" (PRG). (The setting of this variable does not correspond to the setting of inverter parameter "F002".) The data written to this variable is not stored in the inverter's EEPROM. This variable will restore the initial setting when the inverter power is turned off.

Note 1: When "0" is set in this variable, the acceleration time follows the setting of inverter parameter "F002", "F202", or "F302".

Note 2: When a program writes a value to this variable, the value is reflected in the inverter in a 40-ms cycle, which conforms to the standard inverter specifications.

- Sample program: Program to change the acceleration time according to output frequency (Code area [Code Window])

```

:
SET-Freq=      6000      : Set the output frequency to 60 Hz.
ACCEL=        1000      : Set the acceleration time to 10 seconds.
FW=          1          : Start forward rotation of the motor.
LOOP  if      FW      <  U(00)  then  LBL1  : When the output frequency is less than 5 Hz,
:                                     set the acceleration time to 10 seconds.
      if      FM      <  U(01)  then  LBL2  : When the output frequency is less than 10 Hz,
:                                     change the acceleration time to 5 seconds.
      if      FM      <  U(02)  then  LBL3  : When the output frequency is less than 30 Hz,
:                                     change the acceleration time to 1 second.
      if      FM      <  U(03)  then  LBL4  : When the output frequency is less than 50 Hz,
:                                     change the acceleration time to 5 seconds.
      if      FM      <  U(04)  then  LBL5  : When the output frequency is less than 55 Hz,
:                                     change the acceleration time to 10 seconds.
      if      FM      <  U(05)  then  LBL6  : When the output frequency is less than 60 Hz,
:                                     change the acceleration time to 20 seconds.
      if      FM      >= U(05)  then  LBL8  : When the output frequency reaches or
:                                     exceeds 60 Hz, end acceleration.

LBL1  goto      LBL7
      ACCEL=    1000
LBL2  goto      LBL7
      ACCEL=    500
LBL3  goto      LBL7
      ACCEL=    100
LBL4  goto      LBL7
      ACCEL=    500
LBL5  goto      LBL7
      ACCEL=    1000
LBL6  goto      LBL7
      ACCEL=    2000
LBL7  goto      LOOP
LBL8  Y(00)=    1          : Turn Y (00) on when acceleration ends.
:

```

(Data area [Data Window])

```

U(00) = 500      : Set the frequency of 5 Hz in variable "U (00)".
U(01) = 1000    : Set the frequency of 10 Hz in variable "U (01)".
U(02) = 3000    : Set the frequency of 30 Hz in variable "U (02)".
U(03) = 5000    : Set the frequency of 50 Hz in variable "U (03)".
U(04) = 5500    : Set the frequency of 55 Hz in variable "U (04)".
U(05) = 6000    : Set the frequency of 60 Hz in variable "U (05)".

```

(Parameters)

```

A001 = 07
P031 = 03

```

Chapter 5 Instruction Words

DECCEL	Variable name	Range of values	Default	Unit	Data size	Attribute
	Deceleration time setting	0 to 360000	0	0.01 second	Unsigned 2-word data	Readable and writable

- Explanation

This variable can be used to read and write the motor deceleration time in the inverter. The deceleration time setting using this variable is enabled only when the setting of accel/decel time input selection (P031) is "03" (PRG). (The setting of this variable does not correspond to the setting of inverter parameter "F003".) The data written to this variable is not stored in the inverter's EEPROM. This variable will restore the initial setting when the inverter power is turned off.

Note 1: When "0" is set in this variable, the acceleration time follows the deceleration (1) time setting "F003", "F203", or "F304".

Note 2: When a program writes a value to this variable, the value is reflected in the inverter in a 40-ms cycle, which conforms to the standard inverter specifications.

- Sample program: Program to change the deceleration time according to output frequency

(Code area [Code Window])

```

:
DECCEL=          3000          : Set the deceleration time to 30 seconds.
FW=              0            : Start deceleration of the motor.
wait    FM      <    U(04)    : Wait until the output frequency falls below 55
Hz.
DECCEL=          1000          : Change the deceleration time to 10 seconds.
wait    FM      <    U(03)    : Wait until the output frequency falls below 50
Hz.
DECCEL=          500           : Change the deceleration time to 5 seconds.
wait    FM      <    U(02)    : Wait until the output frequency falls below 30
Hz.
DECCEL=          1000          : Change the deceleration time to 10 seconds.
wait    FM      <    U(01)    : Wait until the output frequency falls below 10
Hz.
DECCEL=          3000          : Change the deceleration time to 30 seconds.
wait    FM      <    U(00)    : Wait until the output frequency falls below 5 Hz.
DECCEL=          6000          : Change the deceleration time to 60 seconds.
wait    RUN     =    0         : Wait until the motor stops.
Y(01)=          1             : Turn Y (00) on when acceleration ends.
:

```

(Data area [Data Window])

```

U(00)    500          : Set the frequency of 5 Hz in variable "U (00)".
U(01)    1000         : Set the frequency of 10 Hz in variable "U (01)".
U(02)    3000         : Set the frequency of 30 Hz in variable "U (02)".
U(03)    5000         : Set the frequency of 50 Hz in variable "U (03)".
U(04)    5500         : Set the frequency of 55 Hz in variable "U (04)".
U(05)    6000         : Set the frequency of 60 Hz in variable "U (05)".

```

(Parameter)

P031 = 03

	Variable name	Range of values	Default	Unit	Data size	Attribute
XA (0)	General-purpose analog input (O terminal)	0 to 10000	0	0.01 %	Unsigned 1-word data	Readable
XA (1)	General-purpose analog input (OI terminal)	0 to 10000				
XA (2)	General-purpose analog input (O2 terminal)	-10000 to 10000				

- Explanation

These variables can be used to monitor the data input to the O, OI, and O2 terminals (among the analog input terminals of the inverter) in a data range from -100.00 to +100.00. The analog inputs monitored with these variables correspond to the data set by the [O]-[L], [OI]-[L], and [O2]-[L] input functions (A011 to A015, A101 to A105, and A111 to A114).

- Sample program: Program to configure output frequencies in steps of 10 Hz with general-purpose analog inputs
(Code area [Code Window])

```

:
FW=      1
LOOP    UL(00)=      XA(0)      : Fetch analog input data.
        UL(00)=      UL(00) *    60      : Convert the scale.
        UL(00)=      UL(00) /    100
        if      U(00) <      U(00) then  LBL1      : When data is less than 16.67%, set the
                                                output frequency to 10 Hz.
        if      U(00) <      U(01) then  LBL2      : When data is less than 33.33%, set the
                                                output frequency to 20 Hz.
        if      U(00) <      U(02) then  LBL3      : When data is less than 50.00%, set the
                                                output frequency to 30 Hz.
        if      U(00) <      U(03) then  LBL4      : When data is less than 66.67%, set the
                                                output frequency to 40 Hz.
        if      U(00) <      U(04) then  LBL5      : When data is less than 83.33%, set the
                                                output frequency to 50 Hz.
        if      U(00) <      U(05) then  LBL6      : When data is less than 100.00%, set the
                                                output frequency to 60 Hz.

        goto      LBL7
LBL1    SET-Freq=      1000
        goto      LBL7
LBL2    SET-Freq=      2000
        goto      LBL7
LBL3    SET-Freq=      3000
        goto      LBL7
LBL4    SET-Freq=      4000
        goto      LBL7
LBL5    SET-Freq=      5000
        goto      LBL7
LBL6    SET-Freq=      6000
LBL7    goto      LOOP
:

```

(Data area [Data Window])

```

U(00) = 1000
U(01) = 2000
U(02) = 3000
U(03) = 4000
U(04) = 5000
U(05) = 6000

```

Chapter 5 Instruction Words

	Variable name	Range of values	Default	Unit	Data size	Attribute
YA (0)	General-purpose analog output (FM terminal)	0 to 10000	0	0.01 %	Unsigned 1-word data	Readable and writable
YA (1)	General-purpose analog output (AM terminal)					
YA (2)	General-purpose analog output (AMI terminal)					

- Explanation

These variables can be used to monitor the data output to the FM, AM, and AMI terminals (analog output terminals of the inverter) in a data range from 0% to 100.00%. To obtain the analog outputs, you must assign general-purpose output functions to the FM, AM, and AMI terminals with inverter parameters "C027", "C028", and "C029".

YA (0): FM output terminal (C027 = 12 [YA0])

YA (1): AM output terminal (C028 = 13 [YA1])

YA (2): AMI output terminal (C029 = 14 [YA2])

- Sample program: Program to output inverter output frequency data to a general-purpose analog output as data that is one-half of the full-scale data

```

:
FW=      1
LOOP    UL(00)=      XA(0)      : Fetch analog input data.
        UL(00)=      UL(00) *    60      : Convert the scale.
        UL(00)=      UL(00) /    100
        SET-Freq=      UL(00)      : Set the output frequency.
        UL(01)=      FM      *    100
        UL(01)=      UL(01) /    60      : Convert the scale.
        YA(1)=      UL(01) /    2      : Output data that is one-half of the full-scale data.
        goto      LOOP
:

```


TC (0) to TC (7)	Variable name	Range of values	Default	Unit	Data size	Attribute
	Timer counters	0 to 2 ³¹ -1	0	10 ms	Unsigned 2-word data	Readable and writable

- Explanation

These variables can be used to monitor the counts of the timer counters. The timer counters "TC (0)" to "TC (7)" usually operate as 31-bit free-running timer counters that start simultaneously with user program startup and are incremented in a 10-ms cycle.

When a timer-start instruction (timer set) or delay operation instruction (delay on or delay off) is executed, the timer counter corresponding to the instruction operates as the counter for output to a specified timer contact. In this case, the counter is cleared to zero when the instruction is executed, starts counting, and then stops counting upon reaching the specified count.

When a timer-stop instruction (timer off) is executed, the timer counter corresponding to the instruction is cleared to zero and operates as a 31-bit free-running timer counter that is incremented in a 10-ms cycle.

- Sample program: Program to accelerate the motor step-by-step by using a free-running timer (Code area [Code Window])

```

:
ACCEL=          1000          : Set the acceleration time to 10 seconds.
DECEL=          1000          : Set the deceleration time to 10 seconds.
SET-Freq=       0            : Set the output frequency to 0 Hz.
LOOP  FW=       1            : Start forward rotation of the motor.
      if TC(5) < U(00) then LBL1 : When TC (5) indicates less than 10 seconds
      if TC(5) < U(01) then LBL2 : When TC (5) indicates less than 20 seconds
      if TC(5) < U(02) then LBL3 : When TC (5) indicates less than 30 seconds
      if TC(5) >= U(03) then LBL4 : When TC (5) indicates 40 seconds or more
LBL1  SET-Freq=  1000          : When TC (5) is less than 10 seconds,
                                increase the output frequency to 10 Hz.
      goto      LBL5
LBL2  SET-Freq=  3000          : When TC (5) is less than 20 seconds,
                                increase the output frequency to 30 Hz.
      goto      LBL5
LBL3  SET-Freq=  6000          : When TC (5) is less than 30 seconds,
                                increase the output frequency to 60 Hz.
      goto      LBL5
LBL4  FW=       0            : When TC (5) is 40 seconds or more,
                                decelerate and stop the motor.
      wait      RUN    =      0 : Wait until the motor stops.
      SET-Freq=  0            : Set the output frequency to 0 Hz.
      TC(5)=    0            : Clear TC (5) to zero.
LBL5  goto      LOOP
:

```

(Data area [Data Window])

```

U(00) = 1000          : Set 10 seconds in variable "U (00)".
U(01) = 2000          : Set 20 seconds in variable "U (01)".
U(02) = 3000          : Set 30 seconds in variable "U (02)".
U(03) = 4000          : Set 40 seconds in variable "U (03)".

```

Chapter 5 Instruction Words

	Variable name	Range of values	Default	Unit	Data size	Attribute
TD (0) to TD (7)	Timer contact output (bit access)	0: Off 1: On	0	-	Unsigned 1-word data	Readable
TDw	Timer contact output (word access)	0 to 255	0	-	Unsigned 1-word data	Readable

- Explanation

The data in timer contact output variables "TD (0)" to "TD (7)" is changed only when these variables are specified in the timer-start instruction (timer set) or delay operation instruction (delay on or delay off). A timer contact output variable is set to "0" (off) when the counter corresponding to the contact output is cleared to zero; the variable is set to "1" (on) when the counter stops counting.

While a timer counter variable "TC (k)" is being used for a free-running timer counter, timer contact output variable "TD (k)" corresponding to the timer counter variable retains its status.

- Sample program: Program to accelerate the motor step-by-step by using a timer contact

```

:
  TDw=      0
  ACCEL=           1000      : Set the acceleration time to 10 seconds.
  DECEL=           1000      : Set the deceleration time to 10 seconds.
  SET-Freq=           0      : Set the output frequency to 0 Hz.
LOOP  FW=      1      : Start forward rotation of the motor.
      timer set  TD(5)  10.00      : Start the 10-second timer counter.
      SET-Freq=           1000      : Keep the output frequency at 10 Hz for 10 seconds.
      wait      TD(5)  =      1      : Wait until the timer contact is turned on.
      timer set  TD(5)  10.00      : Start the 10-second timer counter.
      SET-Freq=           3000      : Keep the output frequency at 30 Hz for 10 seconds.
      wait      TD(5)  =      1      : Wait until the timer contact is turned on.
      timer set  TD(5)  10.00      : Start the 10-second timer counter.
      SET-Freq=           6000      : Keep the output frequency at 60 Hz for 10 seconds.
      wait      TD(5)  =      1      : Wait until the timer contact is turned on.
      FW=      0
      wait      RUN    =      0      : Wait until the motor stops.
      SET-Freq=           0      : Set the output frequency to 0 Hz.
      goto     LOOP
:

```

5.9 Inverter Montor Variables

FM	Variable name	Range of values	Default	Unit	Data size	Attribute
	Output frequency monitoring	0 to 40000	-	0.01 Hz	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor the inverter output frequency. The data monitored with this variable corresponds to the data monitored by the output frequency monitoring function (d001). This variable is read-only.

- Sample program: Program to turn a contact output on when output frequency exceeds 50 Hz and turn the contact output off when output frequency falls below 10 Hz

(Code area [Code Window])

```

:
LOOP  if      FM      <   U(00)  then  LBL1   : When the output frequency is less than 10 Hz
      if      FM      >   U(01)  then  LBL2   : When the output frequency is more than 50 Hz
      goto    LBL3
LBL1  UB(02)=  0                : Turn UB (02) off.
      goto    LBL3
LBL2  UB(02)=  1                : Turn UB (02) on.
LBL3  Y(02)=  UB(02)           : Set the data of UB (02) in Y (02).
      goto    LOOP
:
    
```

(Data area [Data Window])

```

U(00) = 1000           : Set the frequency of 10 Hz in variable "U (00)".
U(01) = 5000          : Set the frequency of 50 Hz in variable "U (01)".
    
```

Chapter 5 Instruction Words

	Variable name	Range of values	Default	Unit	Data size	Attribute
lout	Output current monitoring	0 to 9999	-	0.01 %	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor the inverter output current. The data monitored with this variable corresponds to the data monitored by the output current monitoring function (d002). The monitored data indicates the ratio of present output current to rated current of the inverter. This variable is read-only.

- **Sample program:** Program to accelerate the motor while increasing the acceleration time when output current is high

(Code area [Code Window])

```

:
LOOP  U(10)=          lout          : Fetch the output current data.
      if      U(10) <= U(05) then  LBL1 : When output current is 50% or less, change
                                          the acceleration time to 1 second.
      if      U(10) <= U(04) then  LBL2 : When output current is 80% or less, change
                                          the acceleration time to 2 seconds.
      if      U(10) <= U(03) then  LBL3 : When output current is 100% or less, change
                                          the acceleration time to 5 seconds.
      if      U(10) <= U(02) then  LBL4 : When output current is 150% or less, change
                                          the acceleration time to 10 seconds.
      if      U(10) <= U(01) then  LBL5 : When output current is 180% or less, change
                                          the acceleration time to 20 seconds.
      if      U(10) <= U(00) then  LBL6 : When output current is 200% or less, change
                                          the acceleration time to 50 seconds.
      if      U(10) >  U(00) then  LBL7 : When output current exceeds 200%, change
                                          the acceleration time to 100 seconds.

      goto    LBL8
LBL1  ACCEL=          100
      goto    LBL8
LBL2  ACCEL=          200
      goto    LBL8
LBL3  ACCEL=          500
      goto    LBL8
LBL4  ACCEL=         1000
      goto    LBL8
LBL5  ACCEL=         2000
      goto    LBL8
LBL6  ACCEL=         5000
      goto    LBL8
LBL7  ACCEL=        10000
LBL8  if      FA1 =    1      then  LBL9
      goto    LOOP
LBL9  Y(00)=    1          : Turn Y (00) on when acceleration ends.
:

```

(Data area [Data Window])

```

U(00) = 2000          : Set output current of 200% in variable "U (00)".
U(01) = 1800          : Set output current of 180% in variable "U (01)".
U(02) = 1500          : Set output current of 120% in variable "U (02)".
U(03) = 1000          : Set output current of 100% in variable "U (03)".
U(04) = 800           : Set output current of 80% in variable "U (04)".
U(05) = 500           : Set output current of 50% in variable "U (05)".

```

(Parameter)

P031 = 03

Dir	Variable name	Range of values	Default	Unit	Data size	Attribute
	Rotation direction monitoring	0: Stop 1: Forward rotation 2: Reverse rotation	-	-	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor the direction of motor operation by the inverter. The data monitored with this variable corresponds to the data monitored by the rotation direction monitoring function (d003). This variable is read-only.

- Sample program:** Program to output the output frequency data to a general-purpose analog output while operating the motor for reverse rotation at 60 Hz and forward rotation at 60 Hz

```

:
U(00)=          5000          : Set the offset data.
LOOP  U(01)=          Dir          : Fetch the operation direction data.
      UL(01)=          FM          : Fetch the output frequency data.
      UL(01)=  UL(01) * 5000      : Convert the scale to 0% to 50%.
      UL(01)=  UL(01) / 6000
      select  U(01)
      case   1          : When the inverter operates the motor for
                          forward rotation
          UL(00)=  U(00) +  UL(01)
          case   2          : When the inverter operates the motor for
                          reverse rotation
          UL(00)=  U(00) -  UL(01)
          case else          : When the motor is stopped
          UL(00)=          U(00)
      end select
      YA(1)=          UL(00)      : Output the data to an analog output (AM
                                  terminal).
goto  LOOP
:

```

Chapter 5 Instruction Words

	Variable name	Range of values	Default	Unit	Data size	Attribute
PID-FB	Process variable (PV), PID feedback monitoring	0 to 9990000	0	0.01 %	Unsigned 2-word data	Readable

- Explanation

This variable can be used to monitor PID feedback data in the inverter. The data monitored with this variable corresponds to the data monitored by the process variable (PV), PID feedback monitoring function (d004). This variable is read-only.

- **Sample program:** Program to stop inverter output when PID feedback data falls below the sleep level (to manage sleep status)
(Code area [Code Window])

```

:
LOOP   if      PID-FB  >=  U(20)   then  LABEL1  : Compare the monitored data with the
                                         sleep level.
      stop                                           : Stop inverter output.
      goto    LABEL2
LABEL1  FW=    1
LABEL2  goto    LOOP
:

```

(Data area [Data Window])

U(20) = 2000 : Set the PID sleep level of 20% in variable "U (20)".

	Variable name	Range of values	Default	Unit	Data size	Attribute
F-CNV	Scaled output frequency monitoring	0 to 3996000	-	0.01	Unsigned 2-word data	Readable

- Explanation

This variable can be used to monitor the converted output frequency of the inverter. The data monitored with this variable corresponds to the data monitored by the scaled output frequency monitoring function (d007). This variable is read-only.

- **Sample program:** Program to output the motor speed data to a general-purpose analog output.
(Code area [Code Window])

```

:
UL(00)=      F-CNV           : Fetch the converted frequency data.
UL(00)=  UL(00) * 10000
UL(00)=  UL(00) / 1800
YA(0)=      UL(00)           : Output the data to an analog output.
:

```

(Parameter)

b086 = 30.0 : Assign the motor speed in Hz to frequency conversion factor variable "b086".

Tmon	Variable name	Range of values	Default	Unit	Data size	Attribute
	Torque monitoring	-300 to 300	-	%	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor output torque of the motor operated by the inverter. The data monitored with this variable corresponds to the data monitored by the torque monitoring function (d012). This variable is read-only.

- Sample program: Program to increase the inverter output frequency when motor output torque is low (to automatically accelerate the motor)

(Code area [Code Window])

```

:
LOOP  UL(00)=          Tmon          : Fetch the motor output torque data.
      U(10)=          abs  UL(00)     : Convert the data to an absolute value.
      if      U(10) <= U(04) then LBL1 : When torque is 50% or less, change the
                                          output frequency to 100 Hz.
      if      U(10) <= U(03) then LBL2 : When torque is 60% or less, change the
                                          output frequency to 80 Hz.
      if      U(10) <= U(02) then LBL3 : When torque is 70% or less, change the
                                          output frequency to 70 Hz.
      if      U(10) <= U(01) then LBL4 : When torque is 80% or less, change the
                                          output frequency to 65 Hz.
      if      U(10) <= U(00) then LBL5 : When torque is 100% or less, change the
                                          output frequency to 60 Hz.
      if      U(10) >  U(00) then LBL6 : When torque exceeds 100%, change the
                                          output frequency to 60 Hz.

      goto    LBL7
LBL1  SET-Freq=      10000
      goto    LBL7
LBL2  SET-Freq=      8000
      goto    LBL7
LBL3  SET-Freq=      7000
      goto    LBL7
LBL4  SET-Freq=      6500
      goto    LBL7
LBL5  SET-Freq=      6000
      goto    LBL7
LBL6  SET-Freq=      6000
      goto    LBL7
LBL7  goto    LOOP
:

```

(Data area [Data Window])

```

U(00) = 100          : Set torque of 100% in variable "U (00)".
U(01) = 80           : Set torque of 80% in variable "U (01)".
U(02) = 70           : Set torque of 70% in variable "U (02)".
U(03) = 60           : Set torque of 60% in variable "U (03)".
U(04) = 50           : Set torque of 50% in variable "U (04)".

```

(Parameters)

```

A001 = 07
A004 = 100(Hz)

```

Chapter 5 Instruction Words

Vout	Variable name	Range of values	Default	Unit	Data size	Attribute
	Output voltage monitoring	0 to 6000	-	0.1 V	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor the inverter output voltage. The data monitored with this variable corresponds to the data monitored by the output voltage monitoring function (d013). This variable is read-only.

- **Sample program:** Program to turn a contact output on when output voltage exceeds 200 V
(Code area [Code Window])

```

:
LOOP   if      Vout  >=   U(00)   then   SKIP
        Y(00)=   1
SKIP   goto    LOOP
:

```

(Data area [Data Window])

```
U(00) = 2000           : 200V
```

Power	Variable name	Range of values	Default	Unit	Data size	Attribute
	Power monitoring	0 to 9999	-	0.1 kW	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor power input to the inverter. The data monitored with this variable corresponds to the data monitored by the power monitoring function (d014). This variable is read-only.

- **Sample program:** Program to output a signal when input power is lower than the specified minimum limit or higher than the specified maximum limit
(Code area [Code Window])

```

:
LOOP   if      Power <   U(10)   then   LBL1
        if      Power >   U(11)   then   LBL2
        goto    LBL3
LBL1   Y(00)=   1
        goto    LBL3
LBL2   Y(02)=   1
LBL3   goto    LOOP
:

```

(Data area [Data Window])

```
U(10) = 55           : U (10) = 55: Set input power of 5.5 kW in variable "U (10)".
U(11) = 110         : U (11) = 110: Set input power of 11 kW in variable "U (11)".
```


PlsCnt	Variable name	Range of values	Default	Unit	Data size	Attribute
	Pulse count monitoring	0 to 32767	-	1	Unsigned 2-word data	Readable

- Explanation

This variable can be used to reference the pulse count when the pulse counter function is selected. The data referenced with this variable corresponds to the data monitored by the pulse counter monitoring function (d028). This variable is read-only..

- Sample program: This program turns on contact output when the pulse count exceeds 2000 (times). (Code area [Code Window])

```

:
LOOP   if      PlsCnt >=      U(00)   then   SKIP
        Y(00)=   1
SKIP   goto    LOOP
:

```

(Data area [Data Window])

U(00) = 2000 : 2000(pulse)

POS	Variable name	Range of values	Default	Unit	Data size	Attribute
	current position monitoring	268435455 ~ -268435455 (1073741823 ~ -1073741823)	-	1	Signed 2-word data	Readable

- Explanation

This variable can be used to reference current position information when the absolute position control function is selected.

The data referenced with this variable corresponds to the data monitored by the current position monitoring function (d030). This variable is read-only.

When "03" (high-resolution absolute position control) has been selected for control pulse setting (P012), the parenthesized range of values applies.

- Sample program: This program turns on contact output when the current position data exceeds 100,000 (pulses).

(Code area [Code Window])

```

:
UL(00)= 100000
LOOP   if      POS  >=      U(00)   then   SKIP
        Y(00)=   1
SKIP   goto    LOOP
:

```

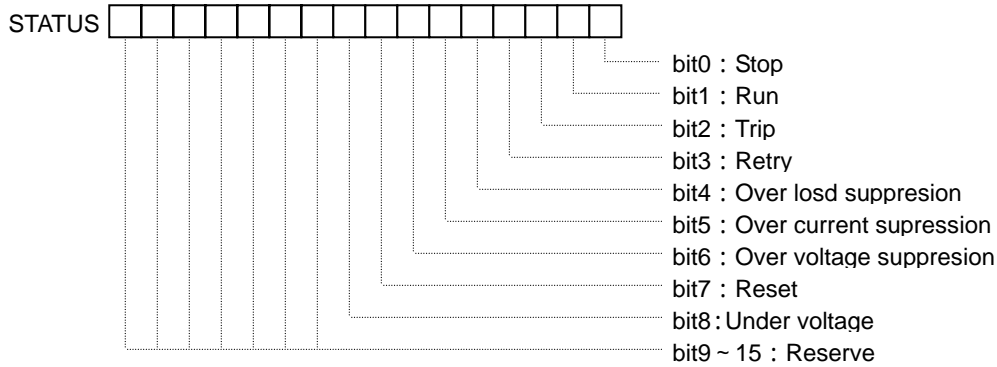
Chapter 5 Instruction Words

STATUS	Variable name	Range of values	Default	Unit	Data size	Attribute
	Inverter status monitoring	-	-	-	Unsigned 1-word data	Readable

- Explanation

This variable can be used to reference inverter status information.

The information to be referenced is defined as follows:



- **Sample program:** This program keeps turning on contact output while overvoltage restraint is applied.

(Code area [Code Window])

```

:
entry
LOOP  U(00) = STATUS and U(01) ;64 = b' 0100 0000(overvoltage restraint)
      ifs  U(00) = U(01)
      then
      Y(00)= 1 ;Y(00) ON
      else
      Y(00)= 0 ;Y(00) OFF
      end if
      goto LOOP
      end
:

```

(Data area [Data Window])

```
U(01) = 64
```

DCV	Variable name	Range of values	Default	Unit	Data size	Attribute
	DC voltage monitoring	0 to 9999	-	0.1V	Unsigned 1-word data	Readable

- Explanation

This variable can be used to reference the inverter DC voltage.

The data referenced with this variable corresponds to the data monitored by the DC voltage monitoring function (d102). This variable is read-only.

- **Sample program:** This program turns on contact output when the DC voltage exceeds 350 V.

```

:
LOOP  if Vout >= U(00) then SKIP
      Y(00)= 1
SKIP  goto LOOP
:

```

(Data area [Data Window])

```
U(00) = 3500 ; 350.0V
```

RUN-Time	Variable name	Range of values	Default	Unit	Data size	Attribute
	Cumulative operation RUN time monitoring	0 to 999999	-	Hour	Unsigned 2-word data	Readable

- Explanation

This variable can be used to monitor the accumulated running time of the inverter. The data monitored with this variable corresponds to the data monitored by the cumulative operation RUN time monitoring function (d016). This variable is read-only.

- Sample program:** Program to output a one-second pulse signal indicating the running time of the inverter to a contact output every hour

```

:
LOOP   UL(01)=          UL(00)          : Set the previous data in variable "UL
(01)".
       UL(00)=          RUN-Time        : Fetch the running-time data.
       if      UL(00) =  UL(01)  then  LBL1
       Y(00)=  1          : Turn Y (00) on.
       delay off Y(00)  TD(2)  100      : Turn Y (00) off after 1 second elapses.
LBL1   goto      LOOP
:

```

ON-Time	Variable name	Range of values	Default	Unit	Data size	Attribute
	Cumulative power-on time monitoring	0 to 999999	-	Hour	Unsigned 2-word data	Readable

- Explanation

This variable can be used to monitor the accumulated power-on time of the inverter. The data monitored with this variable corresponds to the data monitored by the cumulative power-on time monitoring function (d017). This variable is read-only.

- Sample program:** Program to convert the power-on time into a number of days and output the converted data as word data to Y (00) to Y (05)

```

UL(00)=          ON-Time        : Fetch the power-on time data.
UL(00)=  UL(00) /  24          : Convert the data into a number of days.
Yw=      UL(00)  and  31       : Output the data to Yw.
:

```

Chapter 5 Instruction Words

ERR CNT	Variable name	Range of values	Default	Unit	Data size	Attribute
	Trip counter	0 to 65535	-	Number of times	Unsigned 1-word data	Readable

- Explanation

This variable can be used to monitor the number of times the inverter has tripped. The data monitored with this variable corresponds to the data monitored by the trip counter function (d080). This variable is read-only.

- Sample program: Program to check whether the inverter has tripped more than 10,000 times

```

:
Yw=      0                               : Turn Y (00) to Y (05) off.
UL(00)=      ERR CNT                     : Fetch the trip count data.
UL(01)=      100
If      UL(00) < UL(01) then SKIP      : When the trip count exceeds 100 (times).
Y(00)=      1                             : turn Y (00) on.
goto    SKIP
SKIP    Y(01)= 1                           : Turn Y (01) on when the process ends.
:

```

ERR (1) to ERR (6)	Variable name	Range of values	Default	Unit	Data size	Attribute
	Trip monitoring 1 to 6	0 to 127	-	-	Unsigned 1-word data	Readable

- Explanation

These variables can be used to monitor the causes of the last six trips made by the inverter. The data monitored with this variable corresponds to the data monitored by trip monitoring functions 1 to 6 (d081 to d086). These variables are read-only.

- Sample program: Program to check whether the last six trips include one caused by overcurrent (Code area [Code Window])

```

:
entry
Yw=      0
if      ERR(1) = U(00) then MATCH      : Check the factor of the latest trip.
if      ERR(2) = U(00) then MATCH      : Check the factor of the trip preceding the
                                         latest.
if      ERR(3) = U(00) then MATCH      : Check the factor of the trip two trips before the
                                         latest.
if      ERR(4) = U(00) then MATCH      : Check the factor of the trip three trips before
                                         the latest.
if      ERR(5) = U(00) then MATCH      : Check the factor of the trip four trips before the
                                         latest.
if      ERR(6) = U(00) then MATCH      : Check the factor of the trip five trips before the
                                         latest.
Y(00)=      0                             : Turn Y (00) off.
goto    SKIP
MATCH    Y(00)= 1                           : Turn Y (00) on.
SKIP    Y(01)= 1                           : Turn Y (01) on when the process ends.
:

```

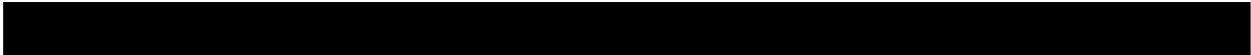
(Data area [Data Window])

```

U(00) = 3                               : Set "3" (E03) in variable "U (00)". (Error code "E03" indicates a trip due to
                                         overcurrent.)

```

Chapter 6 Interface with the Inverter



This chapter explains the inverter settings to use the easy sequence function.

- 6.1 Inverter Settings Related to the Easy Sequence Function 6 - 1
- 6.2 Switching of Operation 6 - 2
- 6.3 Switching of Input/Output Terminals..... 6 - 2
- 6.4 Switching of Command Input Device 6 - 6
- 6.5 Others..... 6 - 7

6.1 Inverter Settings Related to the Easy Sequence Function

The following table lists the inverter settings related to the easy sequence function.

Category	Item		Variable notation in program	Related function code	Variable/terminal use condition
		Terminal name			
Operation switching	Selection of easy sequence function		-	A017	-
Input/output switching	Program run signal	PRG (FW terminal)	-	A017	The PRG terminal is enabled when A017 = 01. (The FW function is disabled.)
	General-purpose input contacts (8 contacts)	Intelligent input terminals 1 to 8	X (00) to X (07) Xw	C001 to C008	Function code settings are required.
	General-purpose output contacts (6 contacts)	Intelligent output terminals 11 to 15	Y (00) to Y (05) Yw	C021 to C025	
		Intelligent relay output terminals AL0 to AL2		C026	
	General-purpose analog inputs (3 terminals)	O terminal	XA (0)	-	No setting is required.
		OI terminal	XA (1)	-	
		O2 terminal	XA (2)	-	
	General-purpose analog outputs (3 terminals)	FM terminal	YA (0)	C027	Function code settings are required.
		AM terminal	YA (1)	C028	
		AMI terminal	YA (2)	C029	
Command switching	Frequency command selection		SET-Freq	A001	Related variables are valid only when A001 = 07.
	Operation command selection		FW, RV STA, STP, F/R	A002	Related variables are valid only when A002 = 01.
	Acceleration/deceleration input selection		ACCEL DECEL	P031	Related variables are valid only when P031 = 03.
Other	User-defined variables (32 variables)		U (00) to U (31)	P100 to P131	The variables can be redefined by using the digital operator or EzSQ.

6.2 Switching of Operation

6.2.1 Easy sequence function selection (A017)

To enable the easy sequence function, specify "01" (enabling) for the easy sequence function selection (A017). When the easy sequence function is enabled, the FW terminal is switched to the PRG terminal, which is used to run the sequence program downloaded to the inverter. (The FW terminal does not function as the terminal to input the forward-rotation command while the easy sequence function is operating.)

Function code	Function name	Setting	Remarks
A017	Easy sequence function selection	00: Off (disabling) 01: On (enabling)	When A017 = 01, the FW terminal is switched to the PRG terminal.

6.3 Switching of Input/Output Terminals

6.3.1 Program run signal input terminal (PRG terminal)

Turning on the PRG (FW) terminal runs the sequence program downloaded to the inverter. When the PRG terminal is off, the inverter does not accept the operation command input via the RV terminal, but waits until the sequence program runs. If the PRG terminal is turned off while the sequence program is running, the program stops, and all operation command input terminals are turned off.

6.3.2 General-purpose contact input terminals

You can assign functions "56" (MI1) to "63" (MI8) to terminals 1 to 8 (C001 to C008) to use these terminals as general-purpose input terminals for the easy sequence function. The table below lists the inverter terminal functions and program variables corresponding to the terminal functions.

When a general-purpose input function is assigned to an intelligent input terminal, the status of the terminal is reflected in the corresponding program variables (X (00) to X (07) or Xw). If the easy sequence function is disabled or a program that does not use variables "X (00)" to "X (07)" and "Xw" runs, intelligent input terminals 1 to 8 will be ineffective even when functions MI1 to MI8 are assigned to the terminals.

You can also assign functions other than MI1 to MI8 to the intelligent input terminals and operate the terminals for those functions even while a sequence program is running. If both the easy sequence input and intelligent input functions have been assigned to an intelligent input terminal, the terminal functions when either input is effective (i.e., both inputs are ORed).

Function code	Intelligent terminal function	Program variable	Remarks
Terminal [1] to [8] functions (C001 to C008)	56 : MI1	X(00) or Xw	Each terminal can operate for easy sequence input and intelligent input. (Both inputs are ORed.)
	57 : MI2	X(01) or Xw	
	58 : MI3	X(02) or Xw	
	59 : MI4	X(03) or Xw	
	60 : MI5	X(04) or Xw	
	61 : MI6	X(05) or Xw	
	62 : MI7	X(06) or Xw	
	63 : MI8	X(07) or Xw	

6.3.3 General-purpose contact output terminals

You can assign functions "44" (MO1) to "49" (MO6) to terminals 11 to 15 (C021 to C025) and the alarm relay terminal (C026) to use these terminals as general-purpose output terminals for the easy sequence function. The table below lists the inverter terminal functions and program variables corresponding to the terminal functions.

When a general-purpose output function is assigned to one of these output terminals, the data stored in variables "Y (00)" to "Y (05)" or "Yw" can be output to the terminal. If the easy sequence function is disabled or a program that does not use variables "Y (00)" to "Y (05)" and "Yw" runs, the output terminals will be ineffective even when functions MO1 to MO6 are assigned to the terminals.

You can also assign functions other than MO1 to MO6 to the output terminals and operate the terminals for those functions even while a sequence program is running.

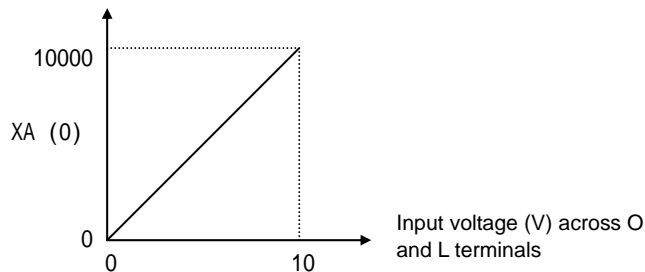
Function code	Intelligent terminal function	Program variable	Remarks
Terminal [11] to [15] functions (C021 to C025) Alarm relay terminal function (C026)	44 : MO1	Y(00) or Yw	-
	45 : MO2	Y(01) or Yw	
	46 : MO3	Y(02) or Yw	
	47 : MO4	Y(03) or Yw	
	48 : MO5	Y(04) or Yw	
	49 : MO6	Y(05) or Yw	

6.3.4 General-purpose analog input terminal (O terminal)

You can use the O terminal as a general-purpose analog input terminal. By referencing the data stored in variable "XA (0)", the data (ranging from 0 to 10000) input via the O terminal can be fetched .

Switching the O terminal to a general-purpose analog input terminal does not require any special setting. Even when the O terminal is used to input frequency commands, the O terminal can also function as a general-purpose analog input terminal. Note that the handling of data fetched via the O terminal depends on the settings made by the [O]-[L] input functions (A011 to A015).

The figure below shows the relationship between the input voltage and the value to be fetched (when the settings of functions "A011" to "A015" are the defaults).

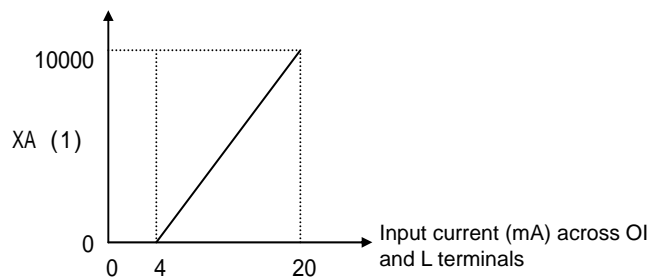


6.3.5 General-purpose analog input terminal (OI terminal)

You can use the OI terminal as a general-purpose analog input terminal. By referencing the data stored in variable "XA (1)", the data (ranging from 0 to 10000) input via the O terminal can be fetched.

Switching the OI terminal to a general-purpose analog input terminal does not require any special setting. Even when the OI terminal is used to input frequency commands, the OI terminal can also function as a general-purpose analog input terminal. Note that the handling of data fetched via the OI terminal depends on the settings made by the [OI]-[L] input functions (A101 to A105).

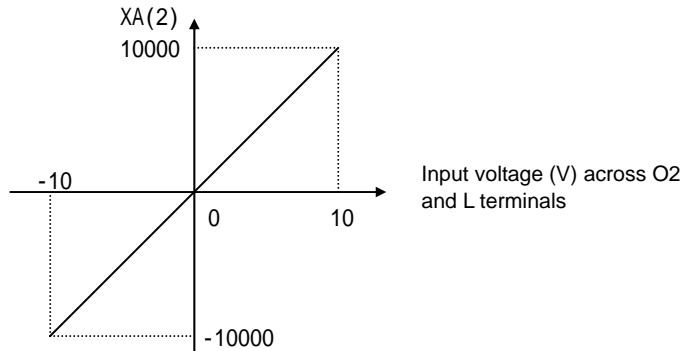
The figure below shows the relationship between the input current and the value to be fetched (when the settings of functions "A101" to "A105" are the defaults).



Chapter 6 Interface with the Inverter

6.3.6 General-purpose analog input terminal (O2 terminal)

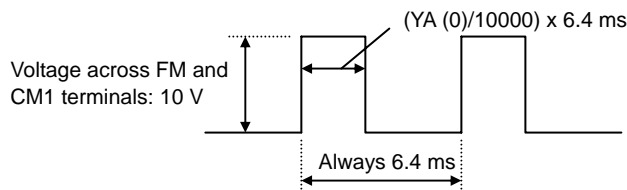
You can use the O2 terminal as a general-purpose analog input terminal. By referencing the data stored in variable "XA (2)", the data (ranging from -10000 to 10000) input via the O2 terminal can be fetched. Switching the O2 terminal to a general-purpose analog input terminal does not require any special setting. Even when the O2 terminal is used to input frequency commands, the O2 terminal can also function as a general-purpose analog input terminal. Note that the handling of data fetched via the O2 terminal depends on the settings made by the [O2]-[L] input functions (A111 to A115). The figure below shows the relationship between the input voltage and the value to be fetched (when the settings of functions "A111" to "A115" are the defaults).



6.3.7 General-purpose analog output terminal (FM terminal)

You can use the FM terminal as a general-purpose analog output terminal for the easy sequence function. For this purpose, specify "12" (YA0: general-purpose output 0) for the [FM] signal selection (C027). When used as a general-purpose analog output terminal, the FM terminal can output the pulse signal that corresponds to the data (0 to 10000) stored in variable "YA (0)". The FM output characteristics follow the FM gain adjustment (C105). The figure below shows the output waveform (with "C105" set to 100%).

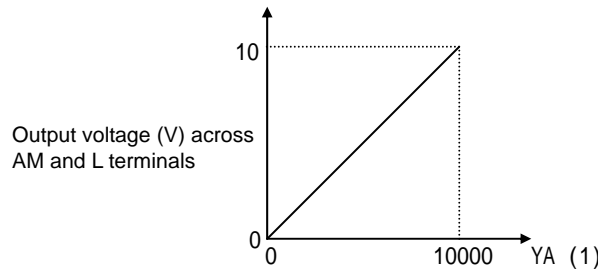
Function code	Function name	Setting	Remarks
C027	[FM] signal selection	00: Output frequency 01: Output current 02: Output torque 03: Digital output frequency 04: Output voltage 05: Input power 06: Electronic thermal overload 07: LAD frequency 08: Digital current monitoring 09: Motor temperature 10: Heat sink temperature 12: General-purpose output 0	The analog output of program variable (YA (0)) data is enabled only when C027 = 12.



6.3.8 General-purpose analog output terminal (AM terminal)

You can use the AM terminal as a general-purpose analog output terminal for the easy sequence function. For this purpose, specify "13" (YA1: general-purpose output 1) for the [AM] signal selection (C028). When used as a general-purpose analog output terminal, the AM terminal can output the data (0 to 10000) stored in variable "YA (1)". The AM output characteristics follow the AM gain adjustment (C106) and AM bias adjustment (C109). The figure below shows the relationship between the value of variable "YA (1)" and AM output voltage (with "C106" set to 100% and "C109" set to 0%).

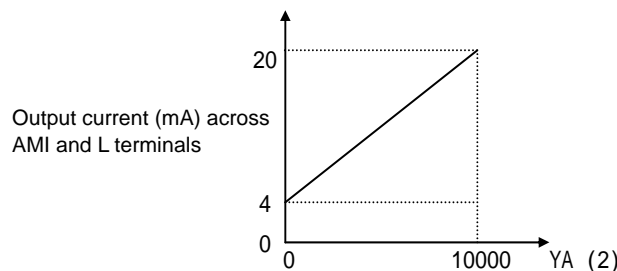
Function code	Function name	Setting	Remarks
C028	[AM] signal selection	00: Output frequency 01: Output current 02: Output torque 04: Output voltage 05: Input power 06: Electronic thermal overload 07: LAD frequency 09: Motor temperature 10: Heat sink temperature 11: Output torque (signed value) 13: General-purpose output 1	The analog output of program variable (YA (1)) data is enabled only when C028 = 13.



6.3.9 General-purpose analog output terminal (AMI terminal)

You can use the AMI terminal as a general-purpose analog output terminal for the easy sequence function. For this purpose, specify "14" (YA2: general-purpose output 2) for the [AMI] signal selection (C029). When used as a general-purpose analog output terminal, the AMI terminal can output the data (0 to 10000) stored in variable "YA (2)". The AMI output characteristics follow the AMI gain adjustment (C107) and AMI bias adjustment (C110). The figure below shows the relationship between the value of variable "YA (2)" and AMI output voltage (with "C107" set to 100% and "C110" set to 0%).

Function code	Function name	Setting	Remarks
C029	[AMI] signal selection	00: Output frequency 01: Output current 02: Output torque 04: Output voltage 05: Input power 06: Electronic thermal overload 07: LAD frequency 09: Motor temperature 10: Heat sink temperature 14: General-purpose output 2	The analog output of program variable (YA (2)) data is enabled only when C029 = 14.



6.4 Switching of Command Input Device

6.4.1 Frequency source setting (A001)

Selection of the device used to input frequency commands follows the frequency source setting (A001), regardless of whether the easy sequence function is enabled.

EzSQ provides variable "SET-Freq" for setting the inverter output frequency. To enable the use of this variable, specify "07" (PRG) for the frequency source setting (A001). Otherwise, the frequency setting in "SET-Freq" will not be reflected in the inverter.

Function code	Function name	Setting	Remarks
A001	Frequency source setting	00: VR (keypad potentiometer) 01: TRM (control circuit terminal block) 02: OPE (digital operator) 03: RS485 (RS485) 04: OP1 (option 1) 05: OP2 (option 2) 06: PLS (pulse-string input) 07: PRG (easy sequence) 10: MATH (operation function result)	The use of program variable (SET-Freq) data is enabled only when A001 = 07.

6.4.2 Run command source setting (A002)

Selection of the device used to input operation commands follows the run command source setting (A002), regardless of whether the easy sequence function is enabled.

EzSQ provides variables "FW", "RV", "STA", "STP", and "F/R" for the inverter control related to operation commands. Since these variables are handled as terminal input data, specify "01" (TRM) for the run command source setting (A002) to enable the use of the variables.

Function code	Function name	Setting	Remarks
A002	Run command source setting	01: TRM (control circuit terminal block) 02: OPE (digital operator) 03: RS485 (RS485) 04: OP1 (option 1) 05: OP2 (option 2)	The use of program variables "FW", "RV", "STA", "STP", and "F/R" is enabled only when A002 = 01.

6.4.3 Accel/decel time input selection (P031)

Selection of the device used to input acceleration/deceleration time settings follows the setting of accel/decel time input selection (P031), regardless of whether the easy sequence function is enabled.

EzSQ provides variables "ACCEL" and "DECEL" for the inverter control related to acceleration and deceleration time. To enable the use of these variable, specify "03" (PRG) for the accel/decel time input selection (P031). Otherwise, the acceleration/deceleration time settings in "ACCEL" and "DECEL" will not be reflected in the inverter.

Function code	Function name	Setting	Remarks
P031	Accel/decel time input selection	00: OPE (digital operator) 01: OP1 (option 1) 02: OP2 (option 2) 03: PRG (easy sequence)	The use of program variables "ACCEL" and "DECEL" is enabled only when P031 = 03.

6.5 Others

6.5.1 User-defined variables "U (00)" to "U (31)" (P100 to P131)

The easy sequence function provides 32 user-defined variables "U (00)" to "U (31)", which correspond to inverter parameters "P100" to "P131". You can use the "Data Window" of EzSQ to set data in these variables, and store them as inverter parameters "P100" to "P131" by downloading the program containing the variables to the inverter. After downloading the program, you can update the parameter data by accessing parameters "P100" to "P131" from the digital operator connected to the inverter without using EzSQ.

Function code	Function name	Setting	Remarks
P100 to P131	User-defined variables U (00) to U (31)	0 to 65535 (to be defined by user)	Updateable via digital operator or EzSQ

Chapter 7 Errors and Troubleshooting

This chapter explains the errors that may occur when using the easy sequence function and the methods of handling the errors.

7.1	Errors Specific to the Easy Sequence Function.....	7 - 1
7.2	Troubleshooting.....	7 - 2

7.1 Errors Specific to the Easy Sequence Function

The table below lists the errors that are specific to the easy sequence function. For other errors in the inverter, refer to the SJ700 Series Inverter Instruction Manual.

No.	Error (causing inverter trip)	Factor code (*1)	Description
1	Invalid instruction	E43.*	The inverter assumes an error if the downloaded program includes an invalid instruction code. This error is detected if the PRG terminal is turned on when the downloaded program has been destroyed or no program has been downloaded.
2	Nesting count error	E44.*	The inverter assumes an error if subroutines, for statements, and/or next statements are nested in more than eight layers.
3	Instruction error 1	E45.*	<ul style="list-style-type: none"> - The inverter assumes an error if the jump destination of a goto statement is not the beginning of a nested for statement but preceded by the end of a nested next statement. - The inverter assumes an error if the variable "U (xx)" referenced via another variable is not found. - The inverter assumes an error if an arithmetic instruction results in overflow or underflow, or causes a division by zero. - The inverter assumes an error if a chg param instruction causes reference to a nonexistent parameter, change of a parameter value outside the setting limits, or updating of a parameter that cannot be updated during inverter operation.
4	User trip 0 to 9	E50.* to E59.*	

1 The asterisk () in the factor code represents an inverter status code. For details, refer to the SJ700-2 Series Inverter Instruction Manual.

Chapter 7 Errors and Troubleshooting

7.2 Troubleshooting

The table below shows how to handle the errors specific to the easy sequence function. For how to handle other errors in the inverter, refer to the SJ700 Series Inverter Instruction Manual.

Factor code	Error (causing inverter trip)	Possible cause	Checking method	Corrective action
E43	Invalid instruction	The PRG terminal was turned on without a program downloaded to the inverter.	Upload the program from the inverter to the personal computer, and check whether the uploaded program matches one of the programs stored on the personal computer.	Recreate the program, and then download it to the inverter.
		The program stored in inverter memory has been destroyed.		
E44	Nesting count error	Subroutines are nested in more than eight layers.	Read the program to check the number of nesting layers.	Correct the program so that the number of layers will be eight or less.
		for-next loop statements are nested in more than eight layers.		
		if statements are nested in more than eight layers.		
E45	Instruction error 1	The jump destination of a goto instruction is a next instruction to end a for or other loop.	Check whether each goto instruction jumps to an instruction that ends a loop.	Correct the jump destinations of goto instructions.
		The variable "U (ii)" referenced via another variable is not found.	Check the numerical value specified in "U (ii)".	Correct the value of variable "U (ii)" or limit the range of values of variable "U (ii)".
		An arithmetic instruction caused: - overflow, - underflow, or - division by zero.	Check the program for the instruction causing overflow, underflow, or division by zero.	Correct the program so that no arithmetic instruction causes overflow, underflow, or division by zero.
		A chg param instruction caused: - reference to a nonexistent parameter, - writing of a value out of the setting range, - change of a parameter value (during inverter operation) that cannot be updated during inverter operation, or - change of a parameter value of which updating is restricted by software lock (when software lock is enabled).	- Check the parameters and the values to be written. - If the error has occurred during inverter operation, check whether the parameter in question is the one that can be updated during inverter operation. (*1) - Check the setting of software lock selection (b031). (*1)	- Correct the parameters or the values to be written to parameters so that they will be within the setting range. - Disable software lock. (*2) - If the parameter to be updated is the one that cannot be updated during inverter operation, change the setting of software lock selection (b031) to "10" to switch to the mode enabling parameter updating during inverter operation. (*2)

*1 For details, refer to the SJ700-2 Series Inverter Instruction Manual.

*2 The settings of some parameters affect inverter output and the functions of input/output terminals. Changing the settings of said parameters during inverter operation may entail the risk of abnormal operation of the motor or machine driven by the inverter. If you change the setting of a parameter after disabling the software lock or switching to the mode enabling parameter updating during inverter operation, check the influence of the update beforehand to ensure the safety of system operation.

Chapter 8 Appendix

8.1	Inverter Parameters and Available Settings.....	8 - 1
-----	---	-------

8.1 Inverter Parameters and Available Settings

The tables below list the parameters and ranges of settings available for updating with the chg param instruction.

(1) F parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
F001	Output frequency setting	Do not use	-	-	-
F002	Acceleration (1) time setting	1 to 360000	0.01 (s)		
F202	Acceleration (1) time setting, 2nd motor	1 to 360000	0.01 (s)		
F302	Acceleration (1) time setting, 3rd motor	1 to 360000	0.01 (s)		
F003	Deceleration (1) time setting	1 to 360000	0.01 (s)		
F203	Deceleration (1) time setting, 2nd motor	1 to 360000	0.01 (s)		
F303	Deceleration (1) time setting, 3rd motor	1 to 360000	0.01 (s)		
F004	Keypad Run key routing	0 (forward rotation), 1 (reverse rotation)	-	x	x

(2) A parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
A001	Frequency source setting	0 (keypad potentiometer), 1 (control circuit terminal block), 2 (digital operator), 3 (RS485), 4 (option 1), 5 (option 2), 6 (pulse-string input), 7 (easy sequence), 10 (operation function result)	-	x	x
A002	Run command source setting	1 (control circuit terminal block), 2 (digital operator), 3 (RS485), 4 (option 1), 5 (option 2)	-	x	x
A003	Base frequency setting	30 to "maximum frequency"	1 (Hz)	x	x
A203	Base frequency setting, 2nd motor	30 to "maximum frequency, 2nd motor"	1 (Hz)	x	x
A303	Base frequency setting, 3rd motor	30 to "maximum frequency, 3rd motor"	1 (Hz)	x	x
A004	Maximum frequency setting	30 to 400	1 (Hz)	x	x
A204	Maximum frequency setting, 2nd motor	30 to 400	1 (Hz)	x	x
A304	Maximum frequency setting, 3rd motor	30 to 400	1 (Hz)	x	x
A005	[AT] selection	0 (switching between O and OI terminals), 1 (switching between O and O2 terminals), 2 (switching between O terminal and keypad potentiometer), 3 (switching between OI terminal and keypad potentiometer), 4 (switching between O2 and keypad potentiometer)	-	x	x
A006	[O2] selection	0 (single), 1 (auxiliary frequency input via O and OI terminals) (nonreversible), 2 (auxiliary frequency input via O and OI terminals) (reversible), 3 (disabling O2 terminal)	-	x	x
A011	[O]-[L] input active range start frequency	0 to 40000	0.01 (Hz)	x	
A012	[O]-[L] input active range end frequency	0 to 40000	0.01 (Hz)	x	
A013	[O]-[L] input active range start voltage	0. to "[O]-[L] input active range end voltage"	1 (%)	x	
A014	[O]-[L] input active range end voltage	"[O]-[L] input active range start voltage" to 100	1 (%)	x	
A015	[O]-[L] input start frequency enable	00 (external start frequency), 01 (0 Hz)	-	x	
A016	External frequency filter time const.	1. to 30. or 31 (500 ms filter ± 0.1 Hz with hysteresis)	-	x	
A017	Easy-sequence function selection	0 (disabling), 1 (enabling)	-	x	x
A019	Multispeed operation selection	0 (binary: 16 speeds selectable with 4 terminals), 1 (bit: 8 speeds selectable with 7 terminals)	-	x	x
A020	Multistage frequency setting	0 to "maximum frequency"	0.01 (Hz)		
A220	Multistage frequency setting, 2nd motor	0 to "maximum frequency, 2nd motor"	0.01 (Hz)		
A320	Multistage frequency setting, 3rd motor	0 to "maximum frequency, 3rd motor"	0.01 (Hz)		

Chapter 8 Appendix

(2) A parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
A021	Multispeed 1 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A022	Multispeed 2 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A023	Multispeed 3 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A024	Multispeed 4 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A025	Multispeed 5 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A026	Multispeed 6 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A027	Multispeed 7 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A028	Multispeed 8 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A029	Multispeed 9 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A030	Multispeed 10 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A031	Multispeed 11 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A032	Multispeed 12 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A033	Multispeed 13 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A034	Multispeed 14 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A035	Multispeed 15 setting	0 to "n-th maximum frequency"	0.01 (Hz)		
A038	Jog frequency setting	"Start frequency" to 999	0.01 (Hz)		
A039	Jog stop mode	0 (free-running after jogging stops [disabled during operation]), 1 (deceleration and stop after jogging stops [disabled during operation]), 2 (DC braking after jogging stops [disabled during operation]), 3 (free-running after jogging stops [enabled during operation]), 4 (deceleration and stop after jogging stops [enabled during operation]), 5 (DC braking after jogging stops [enabled during operation])	-	×	
A041	Torque boost method selection	0 (manual torque boost), 1 (automatic torque boost)	-	×	×
A241	Torque boost method selection, 2nd motor	0 (manual torque boost), 1 (automatic torque boost)	-	×	×
A042	Manual torque boost value	0 to 200	0.1 (%)		
A242	Manual torque boost value, 2nd motor	0 to 200	0.1 (%)		
A342	Manual torque boost value, 3rd motor	0 to 200	0.1 (%)		
A043	Manual torque boost frequency adjustment	0 to 500	0.1 (%)		
A243	Manual torque boost frequency adjustment, 2nd motor	0 to 500	0.1 (%)		
A343	Manual torque boost frequency adjustment, 3rd motor	0 to 500	0.1 (%)		
A044	V/F characteristic curve selection, 1st motor	0 (VC), 1 (VP), 2 (free V/f), 3 (sensorless vector control), 4 (0Hz-range sensorless vector), 5 (vector with sensor)	-	×	×
A244	V/F characteristic curve selection, 2nd motor	0 (VC), 1 (VP), 2 (free V/f), 3 (sensorless vector control), 4 (0Hz-range sensorless vector)	-	×	×
A344	V/F characteristic curve selection, 3rd motor	0(VC), 1(VP)	-	×	×
A045	V/f gain setting	20 to 100	1 (%)		
A046	Voltage compensation gain setting for automatic torque boost, 1st motor	0 to 255	1 (%)		
A246	Voltage compensation gain setting for automatic torque boost, 2nd motor	0 to 255	1 (%)		
A047	Slippage compensation gain setting for automatic torque boost, 1st motor	0 to 255	1 (%)		
A247	Slippage compensation gain setting for automatic torque boost, 2nd motor	0 to 255	1 (%)		
A051	DC braking enable	0 (disabling), 1 (enabling), 2 (set frequency only)	-	×	
A052	DC braking frequency setting	0 to 40000	0.01 (Hz)	×	
A053	DC braking wait time	0 to 50	0.1 (s)	×	
A054	DC braking force during deceleration	*1	1 (%)	×	
A055	DC braking time for deceleration	0 to 600	0.1 (s)	×	
A056	DC braking/edge or level detection for [DB] input	0 (edge operation), 1 (level operation)	-	×	
A057	DC braking force for starting	*1	1 (%)	×	
A058	DC braking time for starting	0 to 600	0.1 (s)	×	

(2) A parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
A059	DC braking carrier frequency setting	*2	0.1 (kHz)	×	×
A061	Frequency upper limit setting	0 or "1st minimum frequency limit" to "maximum frequency"	0.01 (Hz)	×	
A261	Frequency upper limit setting, 2nd motor	0 or "2nd minimum frequency limit" to "maximum frequency, 2nd motor"	0.01 (Hz)	×	
A062	Frequency lower limit setting	0 or "start frequency" to "maximum frequency limit"	0.01 (Hz)	×	
A262	Frequency lower limit setting, 2nd motor	0 or "start frequency" to "maximum frequency, 2nd motor limit"	0.01 (Hz)	×	
A063	Jump (center) frequency setting 1	0 to 40000	0.01 (Hz)	×	
A064	Jump (hysteresis) frequency width setting 1	0 to 1000	0.01 (Hz)	×	
A065	Jump (center) frequency setting 2	0 to 40000	0.01 (Hz)	×	
A066	Jump (hysteresis) frequency width setting 2	0 to 1000	0.01 (Hz)	×	
A067	Jump (center) frequency setting 3	0 to 40000	0.01 (Hz)	×	
A068	Jump (hysteresis) frequency width setting 3	0 to 1000	0.01 (Hz)	×	
A069	Acceleration stop frequency setting	0 to 40000	0.01 (Hz)	×	
A070	Acceleration stop time frequency setting	0 to 600	0.1 (s)	×	
A071	PID Function Enable	0 (disabling), 1 (enabling), 2 (enabling inverted-data output)	-	×	
A072	PID proportional gain	2 to 50	0.1		
A073	PID integral time constant	0 to 36000	0.1 (s)		
A074	PID derivative gain	0 to 10000	0.01 (s)		
A075	PV scale conversion	1 to 9999	0.01	×	
A076	PV source setting	0 (input via OI), 1 (input via O), 2 (external communication), 3 (pulse-string frequency input), 10 (operation result output)	-	×	
A077	Output of inverted PID deviation	0(OFF), 1 (ON)	-	×	
A079	PID feed forward selection	0 (disabled), 1 (O input), 2 (OI input), 3 (O2 input)	-	×	
A078	PID variation range	0 to 1000	0.1 (%)	×	
A081	AVR function select	0 (always on), 1 (always off), 02 (off during deceleration)	-	×	×
A082	AVR voltage select	200 V class: 0(200),1(215),2(220),3(230),4(240) 400 V class: 5(380),6(400),7(415),8(440),9(460),10(480)	-	×	×
A085	Operation mode selection	0 (normal operation), 1 (energy-saving operation), 2 (fuzzy operation)	-	×	×
A086	Energy saving mode tuning	1 to 1000	0.1 (%)		
A092	Acceleration (2) time setting	1 to 360000	0.01 (s)		
A292	Acceleration (2) time setting, 2nd motor	1 to 360000	0.01 (s)		
A392	Acceleration (2) time setting, 3rd motor	1 to 360000	0.01 (s)		
A093	Deceleration (2) time setting	1 to 360000	0.01 (s)		
A293	Deceleration (2) time setting, 2nd motor	1 to 360000	0.01 (s)		
A393	Deceleration (2) time setting, 3rd motor	1 to 360000	0.01 (s)		
A094	Select method to switch to Acc2/Dec2 profile	0 (switching by 2CH terminal), 1 (switching by setting), 2 (switching only when rotation is reversed)	-	×	×
A294	Select method to switch to Acc2/Dec2, 2nd motor	0 (switching by 2CH terminal), 1 (switching by setting), 2 (switching only when rotation is reversed)	-	×	×
A095	Acc1 to Acc2 frequency transition point	0 to 40000	0.01 (Hz)	×	×
A295	Acc1 to Acc2 frequency transition point, 2nd motor	0 to 40000	0.01 (Hz)	×	×
A096	Dec1 to Dec2 frequency transition point	0 to 40000	0.01 (Hz)	×	×
A296	Dec1 to Dec2 frequency transition point, 2nd motor	0 to 40000	0.01 (Hz)	×	×
A097	Acceleration curve selection	0 (linear), 1 (S curve), 2 (U curve), 3 (inverted-U curve), 4 (EL-S curve)	-	×	×

Chapter 8 Appendix

(2) A parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
A098	Deceleration curve setting	0 (linear), 1 (S curve), 2 (U curve), 3 (inverted-U curve), 4 (EL-S curve)	-	×	×
A101	[OI]-[L] input active range start frequency	0 to 40000	0.01 (Hz)	×	
A102	[OI]-[L] input active range end frequency	0 to 40000	0.01 (Hz)	×	
A103	[OI]-[L] input active range start current	0 to "[OI]-[L] input active range end current"	1 (%)	×	
A104	[OI]-[L] input active range end current	"[OI]-[L] input active range start current" to 100	1 (%)	×	
A105	[OI]-[L] input start frequency enable	0 (external start frequency), 1 (0 Hz)	-	×	
A111	[O2]-[L] input active range start frequency	-40000 to 40000	0.01 (Hz)	×	
A112	[O2]-[L] input active range end frequency	-40000 to 40000	0.01 (Hz)	×	
A113	[O2]-[L] input active range start voltage	-100 to O2 end-frequency rate	1 (%)	×	
A114	[O2]-[L] input active range end voltage	"O2 start-frequency rate" to 100	1 (%)	×	
A131	Acceleration curve constants setting	1 (smallest swelling) to 10 (largest swelling)	-	×	
A132	Deceleration curve constants setting	1 (smallest swelling) to 10 (largest swelling)	-	×	
A141	Operation-target frequency selection 1	0 (digital operator), 1 (keypad potentiometer), 2 (input via O), 3 (input via OI), 4 (external communication), 5 (option 1), 6 (option 2), 7 (pulse-string frequency input)	-	×	
A142	Operation-target frequency selection 2	0 (digital operator), 1 (keypad potentiometer), 2 (input via O), 3 (input via OI), 4 (external communication), 5 (option 1), 6 (option 2), 7 (pulse-string frequency input)	-	×	
A143	Operator selection	0 (addition: A141 + A142), 1 (subtraction: A141 - A142), 2 (multiplication: A141 x A142)	-	×	
A145	Frequency to be added	0 to 40000	0.01 (Hz)	×	
A146	Sign of the frequency to be added	0 (frequency command + A145), 1 (frequency command - A145)	-	×	
A150	EL-S-curve acceleration ratio 1	0 to 50	1 (%)	×	×
A151	EL-S-curve acceleration ratio 2	0 to 50	1 (%)	×	×
A152	EL-S-curve deceleration ratio 1	0 to 50	1 (%)	×	×
A153	EL-S-curve deceleration ratio 2	0 to 50	1 (%)	×	×

*1

Display code	Inverter capacity	Setting range
A054/A057	Under 55(kW)	0 to 100
	75 to 132(kW)	0 to 80
	Over 160(kW)	0 to 35

*2

Display code	Inverter capacity	Setting range
A059	Under 55(kW)	5 to 150
	75 to 132(kW)	5 to 100
	Over 160(kW)	5 to 30

(3) B parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
b001	Selection of restart mode	0 (tripping), 1 (starting with 0 Hz), 2 (starting with matching frequency), 3 (tripping after deceleration and stopping with matching frequency), 4 (restarting with active matching frequency)	-	×	
b002	Allowable under-voltage power failure time	3 to 250	0.1 (s)	×	
b003	Retry wait time before motor restart	3 to 1000	0.1 (s)	×	
b004	Instantaneous power failure/under-voltage trip alarm enable	0 (disabling), 1 (enabling), 2 (disabling during stopping and decelerating to stop)	-	×	
b005	Number of restarts on power failure/under-voltage trip events	0 (16 times), 1 (unlimited)	-	×	
b006	Phase loss detection enable	0 (disabling), 1 (enabling)	-	×	
b007	Restart frequency threshold	0 to 40000	0.01 (Hz)	×	
b008	Selection of retry after tripping	0 (tripping), 1 (starting with 0 Hz), 2 (starting with matching frequency), 3 (tripping after deceleration and stopping with matching frequency), 4 (restarting with active matching frequency)	-	×	
b009	Selection of retry after undervoltage	0 (16 times), 1 (unlimited)	-	×	
b010	Selection of retry count after overvoltage or overcurrent	1 to 3	1 (time)	×	
b011	Retry wait time after tripping	3 to 1000	0.1 (s)	×	
b012	Electronic thermal setting (calculated within the inverter from current output)	200 to 1000	0.1 (%)	×	
b212	Electronic thermal setting (calculated within the inverter from current output), 2nd motor	200 to 1000	0.1 (%)	×	
b312	Electronic thermal setting (calculated within the inverter from current output), 3rd motor	200 to 1000	0.1 (%)	×	
b013	Electronic thermal characteristic	0 (reduced-torque characteristic), 1 (constant-torque characteristic), 2 (free setting)	-	×	
b213	Electronic thermal characteristic, 2nd motor	0 (reduced-torque characteristic), 1 (constant-torque characteristic), 2 (free setting)	-	×	
b313	Electronic thermal characteristic, 3rd motor	0 (reduced-torque characteristic), 1 (constant-torque characteristic), 2 (free setting)	-	×	
b015	Free setting, electronic thermal frequency (1)	0 to 400	1 (Hz)	×	
b016	Free setting, electronic thermal current (1)	0 to rated current * 10	0.1 (A)	×	
b017	Free setting, electronic thermal frequency (2)	0 to 400	1 (Hz)	×	
b018	Free setting, electronic thermal current (2)	0 to rated current * 10	0.1 (A)	×	
b019	Free setting, electronic thermal frequency (3)	0 to 400	1 (Hz)	×	
b020	Free setting, electronic thermal current (3)	0 to rated current * 10	0.1 (A)	×	
b021	Overload restriction operation mode	0 (disabling), 1 (enabling during acceleration and deceleration), 2 (enabling during constant speed), 3 (enabling during acceleration and deceleration (increasing the speed during regeneration))	-	×	
b022	Overload restriction setting	*3	0.1 (%)	×	
b023	Deceleration rate at overload restriction	10 to 3000	0.01 (s)	×	
b024	Overload restriction operation mode (2)	0 (disabling), 1 (enabling during acceleration and deceleration), 2 (enabling during constant speed), 03 (enabling during acceleration and deceleration (increasing the speed during regeneration))	-	×	
b025	Overload restriction setting (2)	*3	0.1 (%)	×	
b026	Deceleration rate at overload restriction (2)	10 to 3000	0.01 (s)	×	
b027	Overcurrent suppression enable	0 (disabling), 1 (enabling)	-	×	

Chapter 8 Appendix

(3) B parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
b028	Active frequency matching, scan start frequency	200 to 1000	0.1 (%)	×	
b029	Active frequency matching, scan-time constant	10 to 3000	0.01 (s)	×	
b030	Active frequency matching, restart frequency select	0 (frequency at the last shutoff), 1 (maximum frequency), 02 (set frequency)	-	×	
b031	Software lock mode selection	0 (disabling change of data other than "b031" when SFT is on), 1 (disabling change of data other than "b031" and frequency settings when SFT is on), 2 (disabling change of data other than "b031"), 3 (disabling change of data other than "b031" and frequency settings), 10 (enabling data changes during operation)	-	×	
b034	Run/power-on warning time	0 to 65535	1 (10 hours)	×	
b035	Rotational direction restriction	0 (enabling both forward and reverse rotations), 1 (enabling only forward rotation), 2 (enabling only reverse rotation)	-	×	×
b036	Reduced voltage start selection	0 (minimum reduced voltage start time) to 255 (maximum reduced voltage start time)	-	×	
b037	Function code display restriction	0 (full display), 1 (function-specific display), 2 (user setting), 3 (data comparison display), 4 (basic display)	-	×	
b038	Initial-screen selection	0 (screen displayed when the STR key was pressed last), 1 (d001), 2 (d002), 3 (d003), 4 (d007), 5 (F001)	-	×	
b039	Automatic user-parameter setting function enable	0 (disabling), 1 (enabling)	-	×	
b040	Torque limit selection	0 (quadrant-specific setting), 1 (switching by terminal), 2 (analog input), 3 (option 1), 4 (option 2)	—	×	
b041	Torque limit (1) (forward-driving in 4-quadrant mode)	*4	1 (%)	×	
b042	Torque limit (2) (reverse-regenerating in 4-quadrant mode)	*4	1 (%)	×	
b043	Torque limit (3) (reverse-driving in 4-quadrant mode)	*4	1 (%)	×	
b044	Torque limit (4) (forward-regenerating in 4-quadrant mode)	*4	1 (%)	×	
b045	Torque limit LADSTOP enable	0 (disabling), 1 (enabling)	-	×	
b046	Reverse Run protection enable	0 (disabling), 1 (enabling)	-	×	
b050	Controller deceleration and stop on power loss	0 (disabling), 1 (nonstop deceleration to stop), 2 (DC voltage constant control, with resume), 3 (without resume)	-	×	×
b051	DC bus voltage trigger level during power loss	0 to 10000	0.1 (V)	×	×
b052	Over-voltage threshold during power loss	0 to 10000	0.1 (V)	×	×
b053	Deceleration time setting during power loss	0 to 360000	0.01 (s)	×	×
b054	Initial output frequency decrease during power loss	0 to 1000	0.01 (Hz)	×	×
b055	Proportional gain setting for nonstop operation at power loss	0 to 255	-		
b056	Integral time setting for nonstop operation at power loss	0 to 65535	-		
b060	Maximum-limit level of window comparators O	0 to 100 (lower limit : b061 + b062 / 2)	1(%)		
b061	Minimum-limit level of window comparators O	0 to 100 (lower limit : b060 - b062 / 2)	1(%)		
b062	Hysteresis width of window comparators O	0 to 10 (lower limit : b061 - b062 / 2)	1(%)		
b063	Maximum-limit level of window comparators OI	0 to 100. (lower limit : b064 + b066 / 2)	1(%)		
b064	Minimum-limit level of window comparators OI	0 to 100. (lower limit : b063 - b066 / 2)	1(%)		
b065	Hysteresis width of window comparators OI	0 to 10. (lower limit : b063 - b064 / 2)	1(%)		

(3) B parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
b066	Maximum-limit level of window comparators O2	-100. to 100. (lower limit : b067 + b068 / 2)	1(%)		
b067	Minimum-limit level of window comparators O2	-100. to 100. (lower limit : b066 - b068 / 2)	1(%)		
b068	Hysteresis width of window comparators O2	0 to 10. (lower limit : b066 - b067 / 2)	1(%)		
b070	Operation level at O disconnection	0 to 100 or "no" (ignore)	1(%)	×	
b071	Operation level at OI disconnection	0 to 100 or "no" (ignore)	1(%)	×	
b072	Operation level at O2 disconnection	-100 to 100 or "no" (ignore)	1(%)	×	
b078	Cumulative input power data clearance	Clearance by setting "01" and pressing the STR key	-		
b079	Cumulative input power display gain setting	1 to 1000.	-	×	×
b082	Start frequency adjustment	10 to 999	0.01 (Hz)	×	
b083	Carrier frequency setting	*5	0.1 (kHz)	×	×
b084	Initialization mode (parameters or trip history)	0 (clearing the trip history), 1 (initializing the data), 2 (clearing the trip history and initializing the data)	-	×	×
b085	Country code for initialization	0 (Japan), 1 (EU), 2 (U.S.A.)	-	×	×
b086	Frequency scaling conversion factor	1 to 990	0.1		
b087	STOP key enable	0 (enabling), 1 (disabling), 2 (disabling only the function to stop)	-	×	
b088	Restart mode after FRS	0 (starting with 0 Hz), 1 (starting with matching frequency), 2 (starting with active matching frequency)	-	×	
b089	Automatic carrier frequency reduction	0 (invalid), 1 (valid)	-	×	×
b090	Dynamic braking usage ratio	0 to 1000	0.1 (%)	×	
b091	Stop mode selection	0 (deceleration until stop), 1 (free-run stop)	-	×	
b092	Cooling fan control	0 (always operating the fan), 1 (operating the fan only during inverter operation [including 5 minutes after power-on and power-off])	-	×	
b095	Dynamic braking control	0 (disabling), 1 (enabling [disabling while the motor is topped]), 2 (enabling [enabling also while the motor is topped])	-	×	
b096	Dynamic braking activation level	200V class: 330 to 380 400V class: 660 to 760	1 (V)	×	
b098	Thermistor for thermal protection control	0 (disabling the thermistor), 1 (enabling the thermistor with PTC), 2 (enabling the thermistor with NTC)	-	×	
b099	Thermal protection level setting	0 to 9999	1 (Ω)	×	
b100	Free-setting V/f frequency (1)	0 to "free-setting V/f frequency (2)"	1 (Hz)	×	×
b101	Free-setting V/f voltage (1)	0 to 8000	0.1 (V)	×	×
b102	Free-setting V/f frequency (2)	0 to "free-setting V/f frequency (3)"	1 (Hz)	×	×
b103	Free-setting V/f voltage (2)	0 to 8000	0.1 (V)	×	×
b104	Free-setting V/f frequency (3)	0 to "free-setting V/f frequency (4)"	1 (Hz)	×	×
b105	Free-setting V/f voltage (3)	0 to 8000	0.1 (V)	×	×
b106	Free-setting V/f frequency (4)	0 to "free-setting V/f frequency (5)"	1 (Hz)	×	×
b107	Free-setting V/f voltage (4)	0 to 8000	0.1 (V)	×	×
b108	Free-setting V/f frequency (5)	0 to "free-setting V/f frequency (6)"	1 (Hz)	×	×
b109	Free-setting V/f voltage (5)	0 to 8000	0.1 (V)	×	×
b110	Free-setting V/f frequency (6)	0 to "free-setting V/f frequency (7)"	1 (Hz)	×	×
b111	Free-setting V/f voltage (6)	0 to 8000	0.1 (V)	×	×
b112	Free-setting V/f frequency (7)	0. to "free-setting V/f frequency (8)"	1 (Hz)	×	×
b113	Free-setting V/f voltage (7)	0 to 8000	0.1 (V)	×	×
b120	Brake Control Enable	0 (disabling), 1 (enabling)	-	×	
b121	Brake Wait Time for Release	0 to 500	0.01 (s)	×	
b122	Brake Wait Time for Acceleration	0 to 500	0.01 (s)	×	
b123	Brake Wait Time for Stopping	0 to 500	0.01 (s)	×	
b124	Brake Wait Time for Confirmation	0 to 500	0.01 (s)	×	
b125	Brake Release Frequency Setting	0 to 40000	0.01 (Hz)	×	
b126	Brake Release Current Setting	*6	0.1 (%)	×	
b127	Braking frequency	0 to 40000	0.01 (Hz)	×	

Chapter 8 Appendix

(3) B parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
b130	Overvoltage suppression enable	0 (disabling the restraint), 1 (controlled deceleration), 2 (enabling acceleration)	-	×	
b131	Overvoltage suppression level	200V class: 330 to 390 400V class: 660 to 780	1 (V)	×	
b132	Acceleration and deceleration rate at overvoltage suppression	10 to 3000	0.01 (s)	×	
b133	Overvoltage suppression proportional gain	0 to 255	-		
b134	Overvoltage suppression Integral time	0 to 65535	0.01(s)		

*3

Display code	Inverter capacity	Setting range
b022/b025	Under 55(kW)	20 to 2000
	75 to 132(kW)	20 to 1800
	Over 160(kW)	20 to 1800

*4

Display code	Inverter capacity	Setting range
b041 to b044	Under 55(kW)	0 to 200/255(no)
	75 to 132(kW)	0 to 180/255(no)
	Over 160(kW)	0 to 180/255(no)

*5

Display code	Inverter capacity	Setting range
b083	Under 55(kW)	5 to 150
	75 to 132(kW)	5 to 100
	Over 160(kW)	5 to 30

*6

Display code	Inverter capacity	Setting range
b126	Under 55(kW)	0 to 2000
	75 to 132(kW)	0 to 1800
	Over 160(kW)	0 to 1800

(4) C parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
C001	Terminal [1] function	1 (RV: Reverse RUN), 2 (CF1: Multispeed 1 setting), 3 (CF2: Multispeed 2 setting), 4 (CF3: Multispeed 3 setting), 5 (CF4: Multispeed 4 setting), 6 (JG: Jogging), 7 (DB: external DC braking), 8 (SET: Set 2nd motor data), 9 (2CH: 2-stage acceleration/deceleration), 11 (FRS: free-run stop), 12 (EXT: external trip), 13 (USP: unattended start protection), 14: (CS: commercial power source enable), 15 (SFT: software lock), 16 (AT: analog input voltage/current select), 17 (SET3: 3rd motor control), 18 (RS: reset), 20 (STA: starting by 3-wire input), 21 (STP: stopping by 3-wire input), 22 (F/R: forward/reverse switching by 3-wire input), 23 (PID: PID disable), 24 (PIDC: PID reset), 26 (CAS: control gain setting), 27 (UP: remote control UP function), 28 (DWN: remote control DOWN function), 29 (DWN: remote control data clearing), 31 (OPE: forcible operation), 32 (SF1: multispeed bit 1), 33 (SF2: multispeed bit 2), 34 (SF3: multispeed bit 3), 35 (SF4: multispeed bit 4), 36 (SF5: multispeed bit 5), 37 (SF6: multispeed bit 6), 38 (SF7: multispeed bit 7), 39 (OLR: overload restriction selection), 40 (TL: torque limit enable), 41 (TRQ1: torque limit selection bit 1), 42 (TRQ2: torque limit selection bit 2), 43 (PPI: P/PI mode selection), 44 (BOK: braking confirmation), 45 (ORT: orientation), 46 (LAC: LAD cancellation), 47 (PCLR: clearance of position deviation), 48 (STAT: pulse train position command input enable), 50 (ADD: trigger for frequency addition [A145]), 51 (F-TM: forcible-terminal operation), 52 (ATR: permission of torque command input), 53 (KHC: cumulative power clearance), 54 (SON: servo-on), 55 (FOC: pre-excitation), 56 (MI1: general-purpose input 1), 57 (MI2: general-purpose input 2), 58 (MI3: general-purpose input 3), 59 (MI4: general-purpose input 4), 60 (MI5: general-purpose input 5), 61 (MI6: general-purpose input 6), 62 (MI7: general-purpose input 7), 63 (MI8: general-purpose input 8), 65 (AHD: analog command holding), 66 (CP1: multistage position settings selection 1), 67 (CP2: multistage position settings selection 2), 68 (CP3: multistage position settings selection 3), 69 (ORL: Zero-return limit function), 70 (ORG: Zero-return trigger function), 71 (FOT: forward drive stop), 72 (ROT: reverse drive stop), 73 (SPD: speed / position switching), 74 (PCNT: pulse counter), 75 (PCC: pulse counter clear), no (NO: no assignment)	-	×	
C002	Terminal [2] function		-	×	
C003	Terminal [3] function		-	×	
C004	Terminal [4] function		-	×	
C005	Terminal [5] function		-	×	
C006	Terminal [6] function		-	×	
C007	Terminal [7] function		-	×	
C008	Terminal [8] function		-	×	
C011	Terminal [1] active state	00 (NO) / 01 (NC)	-	×	
C012	Terminal [2] active state	00 (NO) / 01 (NC)	-	×	
C013	Terminal [3] active state	00 (NO) / 01 (NC)	-	×	
C014	Terminal [4] active state	00 (NO) / 01 (NC)	-	×	
C015	Terminal [5] active state	00 (NO) / 01 (NC)	-	×	
C016	Terminal [6] active state	00 (NO) / 01 (NC)	-	×	
C017	Terminal [7] active state	00 (NO) / 01 (NC)	-	×	
C018	Terminal [8] active state	00 (NO) / 01 (NC)	-	×	
C019	Terminal [FW] active state	00 (NO) / 01 (NC)	-	×	

Chapter 8 Appendix

(4) C parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
C021	Terminal [11] function	0 (RUN: running), 1 (FA1: constant-speed reached), 2 (FA2: set frequency overreached), 3 (OL: overload notice advance signal (1)), 4 (OD: output deviation for PID control), 5 (AL: alarm signal), 6 (FA3: set frequency reached), 7 (OTQ: over-torque), 8 (IP: instantaneous power failure), 9 (UV: undervoltage), 10 (TRQ: torque limited), 11 (RNT: operation time over), 12 (ONT: plug-in time over), 13 (THM: thermal alarm signal), 19 (BRK: brake release), 20 (BER: braking error), 21 (ZS: 0 Hz detection signal), 22 (DSE: speed deviation maximum), 23 (POK: positioning completed), 24 (FA4: set frequency overreached 2), 25 (FA5: set frequency reached 2), 26 (OL2: overload notice advance signal (2)), 27 (Odc: Analog O disconnection detection), 28 (OIDc: Analog OI disconnection detection), 29 (O2Dc: Analog O2 disconnection detection), 31 (FBV: PID feedback comparison), 32 (NDC: communication line disconnection), 33 (LOG1: logical operation result 1), 34 (LOG2: logical operation result 2), 35 (LOG3: logical operation result 3), 36 (LOG4: logical operation result 4), 37 (LOG5: logical operation result 5), 38 (LOG6: logical operation result 6), 39 (WAC: capacitor life warning), 40 (WAF: cooling-fan speed drop), 41 (FR: starting contact signal), 42 (OHF: heat sink overheat warning), 43 (LOC: low-current indication signal), 44 (M01: general-purpose output 1), 45 (M02: general-purpose output 2), 46 (M03: general-purpose output 3), 47 (M04: general-purpose output 4), 48 (M05: general-purpose output 5), 49 (M06: general-purpose output 6), 50 (IRDY: inverter ready), 51 (FWR: forward rotation), 52 (RVR: reverse rotation), 53 (MJA: major failure), 54(WCO: window comparator O), 55(WCOI: window comparator OI), 56 (WCO2: window comparator O2)	-	×	
C022	Terminal [12] function		-	×	
C023	Terminal [13] function		-	×	
C024	Terminal [14] function		-	×	
C025	Terminal [15] function		-	×	
C026	Alarm relay terminal function	(When alarm code output is selected for "C062", functions "AC0" to "AC2" or "AC0" to "AC3" [ACn: alarm code output] are forcibly assigned to intelligent output terminals 11 to 13 or 11 to 14, respectively.)	-	×	
C027	[FM] signal selection	0 (output frequency), 1 (output current), 2 (output torque), 3 (digital output frequency), 4 (output voltage), 5 (input power), 6 (electronic thermal overload), 7 (LAD frequency), 8 (digital current monitoring), 9 (motor temperature), 10 (heat sink temperature), 12 (general-purpose output YA0)	-	×	
C028	[AM] signal selection	0 (output frequency), 1 (output current), 2 (output torque), 4 (output voltage), 5 (input power), 6 (electronic thermal overload), 7 (LAD frequency), 9 (motor temperature), 10 (heat sink temperature), 11 (output torque [signed value]), 13 (general-purpose output YA1)	-	×	
C029	[AMI] signal selection	0 (output frequency), 1 (output current), 2 (output torque), 4 (output voltage), 5 (input power), 6 (electronic thermal overload), 7 (LAD frequency), 9 (motor temperature), 10 (heat sink temperature), 14 (general-purpose output YA2)	-	×	
C030	Digital current monitor reference value	200 to 2000	0.1 (%)		
C031	Terminal [11] active state	0 (NO) / 1 (NC)		×	
C032	Terminal [12] active state	0 (NO) / 1 (NC)		×	
C033	Terminal [13] active state	0 (NO) / 1 (NC)		×	
C034	Terminal [14] active state	0 (NO) / 1 (NC)		×	
C035	Terminal [15] active state	0 (NO) / 1 (NC)	-	×	
C036	Alarm relay active state	0 (NO) / 1 (NC)	-	×	
C038	Low-current indication signal output mode selection	0 (output during acceleration/deceleration and constant-speed operation), 1 (output only during constant-speed operation)	-	×	
C039	Low-current indication signal detection level	0 to 2000	0.1 (%)	×	
C040	Overload signal output mode	0 (output during acceleration/deceleration and constant-speed operation), 1 (output only during constant-speed operation)	-	×	

(4) C parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
C041	Overload level setting	0 to 2000	0.1 (%)	×	
C042	Frequency arrival setting for accel.	0 to 40000	0.01 (Hz)	×	
C043	Frequency arrival setting for decel.	0 to 40000	0.01 (Hz)	×	
C044	PID deviation level setting	0 to 1000	0.1 (%)	×	
C045	Frequency arrival setting for acceleration (2)	0 to 40000	0.01 (Hz)	×	
C046	Frequency arrival setting for deceleration (2)	0 to 40000	0.01 (Hz)	×	
C052	Maximum PID feedback data	0 to 1000	0.1 (%)	×	
C053	Minimum PID feedback data	0 to 1000	0.1 (%)	×	
C055	Over-torque (forward-driving) level setting	*7	1 (%)	×	
C056	Over-torque (reverse regenerating) level setting	*7	1 (%)	×	
C057	Over-torque (reverse driving) level setting	*7	1 (%)	×	
C058	Over-torque (forward regenerating) level setting	*7	1 (%)	×	
C061	Electronic thermal warning level setting	0 to 100	1 (%)	×	
C062	Alarm code output	0 (disabling), 1 (3 bits), 2 (4 bits)	-	×	
C063	Zero speed detection level	0 to 1000	0.1 (Hz)	×	
C064	Heat sink overheat warning level	0 to 200	1 ()	×	
C071	Communication speed selection	2 (loopback test), 3 (2,400 bps), 4 (4,800 bps), 5 (9,600 bps), 6 (19,200 bps)	-	×	
C072	Node allocation	1 to 32	-	×	
C073	Communication data length selection	7 (7 bits), 8 (8 bits)	-	×	
C074	Communication parity selection	0 (no parity), 1 (even parity), 2 (odd parity)	-	×	
C075	Communication stop bit selection	1 (1 bit), 2 (2 bits)	-	×	
C076	Selection of the operation after communication error	0 (tripping), 1 (tripping after decelerating and stopping the motor), 2 (ignoring errors), 3 (stopping the motor after free-running), 4 (decelerating and stopping the motor)	-	×	
C077	Communication timeout limit before tripping	0 to 9999	0.01 (s)	×	
C078	Communication wait time	0 to 1000	1 (ms)	×	
C079	Communication mode selection	0(ASCII), 1(Modbus-RTU)	-	×	
C081	[O] input span calibration	0 to 65530	-		
C082	[O1] input span calibration	0 to 65530	-		
C083	[O2] input span calibration	0 to 65530	-		
C085	Thermistor input tuning	0 to 10000	-		
C091	Debug mode enable	(Do not change this parameter, which is intended for factory adjustment.)	-	×	×
C101	Up/Down memory mode selection	0 (not storing the frequency data), 1 (storing the frequency data)	-	×	
C102	Reset mode selection	0 (resetting the trip when RS is on), 1 (resetting the trip when RS is off), 2 (enabling resetting only upon tripping [resetting when RS is on]),3 (resetting only trip)	-		
C103	Restart mode after reset	0 (starting with 0 Hz), 1 (starting with matching frequency), 2 (restarting with active matching frequency)	-	×	
C105	FM gain adjustment	50 to 200	1 (%)		
C106	AM gain adjustment	50 to 200	1 (%)		
C107	AMI gain adjustment	50 to 200	1 (%)		
C109	AM bias adjustment	0. to 100	1 (%)		
C110	AMI bias adjustment	0. to 100	1 (%)		
C111	Overload setting (2)	0 to 2000	0.1 (%)	×	
C121	[O] input zero calibration	0 to 65530	-		
C122	[O1] input zero calibration	0 to 65530	-		
C123	[O2] input zero calibration	0 to 65530	-		
C130	Output 11 on-delay time	0 to 1000	0.1 (s)	×	
C131	Output 11 off-delay time	0 to 1000	0.1 (s)	×	
C132	Output 12 on-delay time	0 to 1000	0.1 (s)	×	
C133	Output 12 off-delay time	0 to 1000	0.1 (s)	×	

Chapter 8 Appendix

(4) C parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
C134	Output 13 on-delay time	0 to 1000	0.1 (s)	×	
C135	Output 13 off-delay time	0 to 1000	0.1 (s)	×	
C136	Output 14 on-delay time	0 to 1000	0.1 (s)	×	
C137	Output 14 off-delay time	0 to 1000	0.1 (s)	×	
C138	Output 15 on-delay time	0 to 1000	0.1 (s)	×	
C139	Output 15 off-delay time	0 to 1000	0.1 (s)	×	
C140	Output RY on-delay time	0 to 1000	0.1 (s)	×	
C141	Output RY off-delay time	0 to 1000	0.1 (s)	×	
C142	Logical output signal 1 selection 1	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C143	Logical output signal 1 selection 2	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C144	Logical output signal 1 operator selection	0 (AND), 1 (OR), 2 (XOR)	-	×	
C145	Logical output signal 2 selection 1	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C146	Logical output signal 2 selection 2	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C147	Logical output signal 2 operator selection	0 (AND), 1 (OR), 2 (XOR)	-	×	
C148	Logical output signal 3 selection 1	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C149	Logical output signal 3 selection 2	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C150	Logical output signal 3 operator selection	0 (AND), 1 (OR), 2 (XOR)	-	×	
C151	Logical output signal 4 selection 1	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C152	Logical output signal 4 selection 2	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C153	Logical output signal 4 operator selection	0 (AND), 1 (OR), 2 (XOR)	-	×	
C154	Logical output signal 5 selection 1	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C155	Logical output signal 5 selection 2	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C156	Logical output signal 5 operator selection	0 (AND), 1 (OR), 2 (XOR)	-	×	
C157	Logical output signal 6 selection 1	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C158	Logical output signal 6 selection 2	Same as the settings of C021 to C026 (except those of LOG1 to LOG6)	-	×	
C159	Logical output signal 6 operator selection	0 (AND), 1 (OR), 2 (XOR)	-	×	
C160	Input terminal response time setting 1	0 to 200	× 2(ms)	×	
C161	Input terminal response time setting 2	0 to 200	× 2(ms)	×	
C162	Input terminal response time setting 3	0 to 200	× 2(ms)	×	
C163	Input terminal response time setting 4	0 to 200	× 2(ms)	×	
C164	Input terminal response time setting 5	0 to 200	× 2(ms)	×	
C165	Input terminal response time setting 6	0 to 200	× 2(ms)	×	
C166	Input terminal response time setting 7	0 to 200	× 2(ms)	×	
C167	Input terminal response time setting 8	0 to 200	× 2(ms)	×	
C168	Input terminal response time setting FW	0 to 200.	× 2(ms)	×	
C169	Multistage speed/position determination time	0 to 200	× 10(ms)	×	

*7

Display code	Inverter capacity	Setting range
C055 to C058	Under 55(kW)	20 to 2000
	75 to 132(kW)	20 to 1800
	Over 160(kW)	20 to 1800

Chapter 8 Appendix

(5) H parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
H001	Auto-tuning Setting	0 (disabling auto-tuning), 1 (auto-tuning without rotation), 2 (auto-tuning with rotation)	-	×	×
H002	Motor data selection, 1st motor	0 (Hitachi standard data), 1 (auto-tuned data), 2 (auto-tuned data [with online auto-tuning function])	-	×	×
H202	Motor data selection, 2nd motor	0 (Hitachi standard data), 1 (auto-tuned data), 2 (auto-tuned data [with online auto-tuning function])	-	×	×
H003	Motor capacity, 1st motor	*8	-	×	×
H203	Motor capacity, 2nd motor	*8	-	×	×
H004	Motor poles setting, 1st motor	0(2P), 1(4P), 2(6P), 3(8P), 4(10P)	-	×	×
H204	Motor poles setting, 2nd motor	0(2P), 1(4P), 2(6P), 3(8P), 4(10P)	-	×	×
H005	Motor speed constant, 1st motor	1 to 80000	0.001 (Hz)		
H205	Motor speed constant, 2nd motor	1 to 80000	0.001 (Hz)		
H006	Motor stabilization constant, 1st motor	0 to 255	1 (%)		
H206	Motor stabilization constant, 2nd motor	0 to 255	1 (%)		
H306	Motor stabilization constant, 3rd motor	0 to 255	1 (%)		
H020	Motor constant R1, 1st motor	1 to 65530	*9	×	×
H220	Motor constant R1, 2nd motor	1 to 65530	*9	×	×
H021	Motor constant R2, 1st motor	1 to 65530	*9	×	×
H221	Motor constant R2, 2nd motor	1 to 65530	*9	×	×
H022	Motor constant L, 1st motor	1 to 65530	*10	×	×
H222	Motor constant L, 2nd motor	1 to 65530	*10	×	×
H023	Motor constant lo	1 to 65530	0.01(A)	×	×
H223	Motor constant lo, 2nd motor	1 to 65530	0.01(A)	×	×
H024	Motor constant J	1 to 9999000	0.001 (kgm2)	×	×
H224	Motor constant J, 2nd motor	1 to 9999000	0.001 (kgm2)	×	×
H030	Auto constant R1, 1st motor	1 to 65530	*9	×	×
H230	Auto constant R1, 2nd motor	1 to 65530	*9	×	×
H031	Auto constant R2, 1st motor	1 to 65530	*9	×	×
H231	Auto constant R2, 2nd motor	1 to 65530	*9	×	×
H032	Auto constant L, 1st motor	1 to 65530	*10	×	×
H232	Auto constant L, 2nd motor	1 to 65530	*10	×	×
H033	Auto constant lo, 1st motor	1 to 65530	0.01(A)	×	×
H233	Auto constant lo, 2nd motor	1 to 65530	0.01(A)	×	×
H034	Auto constant J, 1st motor	1 to 9999000	0.001 (kgm2)	×	×
H234	Auto constant J, 2nd motor	1 to 9999000	0.001 (kgm2)	×	×
H050	PI proportional gain for 1st motor	0 to 10000	0.1(%)		
H250	PI proportional gain for 2nd motor	0 to 10000	0.1(%)		
H051	PI integral gain for 1st motor	0 to 10000	0.1(%)		
H251	PI integral gain for 2nd motor	0 to 10000	0.1(%)		
H052	P proportional gain setting for 1st motor	1 to 1000	0.01		
H252	P proportional gain setting for 2nd motor	1 to 1000	0.01		
H060	Zero LV limit for 1st motor	0 to 1000	0.1		
H260	Zero LV limit for 2nd motor	0 to 1000	0.1		
H061	Zero LV starting boost current for 1st motor	0 to 50	1(%)		
H261	Zero LV starting boost current for 2nd motor	0 to 50	1(%)		
H070	Terminal selection PI proportional gain setting	0 to 10000	0.1		
H071	Terminal selection PI integral gain setting	0 to 10000	0.1		
H072	Terminal selection P proportional gain setting	0 to 1000	0.01		
H073	Gain switching time	0 to 9999	1(ms)		

*8

Display code	Inverter capacity	Setting range
H003 to H203	Under 55(kW)	0(0.2kW) to 21(75kW)
	75 to 132(kW)	0(0.2kW) to 26(160kW)
	Over 160(kW)	13(11kW) to 36(400kW)

Motor capacity code

Area code	00	01	02	03	04	05	06	07	08	09	10
JP,USA(b085=00,02)	0.2	-	0.4	-	0.75	-	1.5	2.2	-	3.7	-
EU(b085=01)	0.2	0.37	-	0.55	0.75	1.1	1.5	2.2	3.0	-	4.0
Area code	11	12	13	14	15	16	17	18	19	20	21
JP,USA(b085=00,02)	5.5	7.5	11	15	18.5	22	30	37	45	55	75
EU(b085=01)	5.5	7.5	11	15	18.5	22	30	37	45	55	75
Area code	22	23	24	25	26	27	28	29	30	31	32
JP,USA(b085=00,02)	90	110	132	150	160	185	200	220	250	280	300
EU(b085=01)	90	110	132	150	160	185	200	220	250	280	300
Area code	33	34	35	36	-	-	-	-	-	-	-
JP,USA(b085=00,02)	315	340	355	400	-	-	-	-	-	-	-
EU(b085=01)	315	340	355	400	-	-	-	-	-	-	-

*9

Display code	Inverter capacity	Minimum unit
H020 to H221 H030 to H231	Under 55(kW)	1(mOHM)
	75 to 132(kW)	1(mOHM)
	Over 160(kW)	0.1(mOHM)

*10

Display code	Inverter capacity	Minimum unit
H022,H222/ H032,H232	Under 55(kW)	0.01(mH)
	75 to 132(kW)	0.01(mH)
	Over 160(kW)	0.0011(mH)

Chapter 8 Appendix

(6) P parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
P001	Operation mode on expansion card 1 error	00 (tripping)/01(running)	-	×	
P002	Operation mode on expansion card 2 error	00 (tripping) /01(running)	-	×	
P011	Encoder pulse-per-revolution (PPR) setting	128 to 9999	1 (pulse)	×	×
P012	Control pulse setting	0 (ASR), 1 (APR), 2 (APR2), 3 (HAPR)	-	×	×
P013	Pulse line mode setting	0 (mode 0)/1 (mode 1)/2 (mode 1)	-	×	×
P014	Home search stop position setting	0 to 4095	-	×	
P015	Home search speed setting	"start frequency" to "maximum frequency" (up to 12000)	0.01 (Hz)	×	
P016	Home search direction setting	0 (forward), 1 (reverse)	-	×	×
P017	Home search completion range setting	0 to 10000	0.1 (pulse)	×	
P018	Home search completion delay time setting	0 to 999	0.01 (s)	×	
P019	Electronic gear set position selection	0 (feedback side), 1 (commanding side)	-	×	
P020	Electronic gear ratio numerator setting	0 to 9999	-	×	
P021	Electronic gear ratio denominator setting	0 to 9999	-	×	
P022	Feed-forward gain setting	0 to 65535	-	×	
P023	Position loop gain setting	0 to 10000	-	×	
P024	Position bias setting	-2048 to 2048	1(pulse)		
P025	Temperature compensation thermistor enable	0 (no compensation), 1 (compensation)	-	×	
P026	Over-speed error detection level setting	0 to 1500	0.1 (%)	×	
P027	Speed deviation error detection level setting	0 to 12000	0.01 (Hz)	×	
P028	Numerator of motor gear ratio	0 to 9999	-	×	
P029	Denominator of motor gear ratio	0 to 9999	-	×	
P031	Accel/decel time input selection	0 (digital operator), 1 (option 1), 2 (option 2), 3 (easy sequence)	-	×	×
P032	Positioning command input selection	0 (digital operator), 1 (option 1), 2 (option 2)	-	×	
P033	Torque command input selection	0 (O terminal), 1 (OI terminal), 2 (O2 terminal), 3 (digital operator)	-	×	×
P034	Torque command setting	*11	1 (%)		
P035	Polarity selection at the torque command input via O2 terminal	0 (as indicated by the sign), 1 (depending on the operation direction)	-	×	×
P036	Torque bias mode	0 (disabling the mode), 1 (digital operator), 2 (input via O2 terminal)	-	×	×
P037	Torque bias value	*12	1 (%)		
P038	Torque bias polarity selection	00 (as indicated by the sign), 01 (depending on the operation direction)	-	×	×
P039	Speed limit for torque-controlled operation (forward rotation)	0 to "maximum frequency"	0.01 (Hz)		
P040	Speed limit for torque-controlled operation (reverse rotation)	0 to "maximum frequency"	0.01 (Hz)		
P044	DeviceNet comm watchdog timer	0 to 9999	0.01 (S)	×	×
P045	Inverter action on DeviceNet comm error	0 (tripping), 1 (tripping after decelerating and stopping the motor), 2 (ignoring errors), 3 (stopping the motor after free-running), 4 (decelerating and stopping the motor)	-	×	×
P046	DeviceNet polled I/O: Output instance number	20, 21, 100	-	×	×
P047	DeviceNet polled I/O: Input instance number	70, 71, 101	-	×	×
P048	Inverter action on DeviceNet idle mode	0 (tripping), 1 (tripping after decelerating and stopping the motor), 2 (ignoring errors), 3 (stopping the motor after free-running), 4 (decelerating and stopping the motor)	-	×	×
P049	Motor poles setting for RPM	0(0P), 1(2P), 2(4P), 3(6P), 4(8P), 5(10P), 6(12P), 7(14P), 8(16P), 9(18P), 10(20P), 11(22P), 12(24P), 13(26P), 14(28P), 15(30P), 16(32P), 17(34P), 18(36P), 19(38P)	-	×	×

(6) P parameters

Display code	Function name	Range of setting	Minimum unit	Setting during operation (allowed or not)	Updating during operation (allowed or not)
P055	Pulse-string frequency scale	10 to 500	0.1 (kHz)	×	
P056	Time constant of pulse-string frequency filter	1 to 200	0.01 (s)	×	
P057	Pulse-string frequency bias	-100 to +100	1 (%)	×	
P058	Pulse-string frequency limit	0 to 100	1 (%)	×	
P060	Multistage position setting 0	Position setting range reverse side to forward side	1(pulse)		
P061	Multistage position setting 1	Position setting range reverse side to forward side	1(pulse)		
P062	Multistage position setting 2	Position setting range reverse side to forward side	1(pulse)		
P063	Multistage position setting 3	Position setting range reverse side to forward side	1(pulse)		
P064	Multistage position setting 4	Position setting range reverse side to forward side	1(pulse)		
P065	Multistage position setting 5	Position setting range reverse side to forward side	1(pulse)		
P066	Multistage position setting 6	Position setting range reverse side to forward side	1(pulse)		
P067	Multistage position setting 7	Position setting range reverse side to forward side	1(pulse)		
P068	Zero-return mode selection	0(Low) / 1 (Hi1) / 0 (Hi2)	-		
P069	Zero-return direction selection	0 (FW) / 1 (RV)	-		
P070	Low-speed zero-return frequency	0 to 1000	0.01Hz		
P071	High-speed zero-return frequency	0 to 40000	0.01(Hz)		
P072	Position range specification (forward)	0 to 268435455 (when P012 = 02) 0 to 1073741823 (when P012 = 03)	1(pulse)		
P073	Position range specification (reverse)	-268435455 to 0 (when P012 = 02) -1073741823 to 0 (when P012 = 03)	1(pulse)		
P074	Teaching selection	0 (X00) / 1 (X01) / 2 (X02) / 3 (X03) / 4 (X04) / 5 (X05) / 6 (X06) / 7 (X07) /	-		
P100	Easy sequence user parameter U (00)	0 to 65535	-		
P101	Easy sequence user parameter U (01)	0 to 65535	-		
P102	Easy sequence user parameter U (02)	0 to 65535	-		
P103	Easy sequence user parameter U (03)	0 to 65535	-		
P104	Easy sequence user parameter U (04)	0 to 65535	-		
P105	Easy sequence user parameter U (05)	0 to 65535	-		
P106	Easy sequence user parameter U (06)	0 to 65535	-		
P107	Easy sequence user parameter U (07)	0 to 65535	-		
P108	Easy sequence user parameter U (08)	0 to 65535	-		
P109	Easy sequence user parameter U (09)	0 to 65535	-		
P110	Easy sequence user parameter U (10)	0 to 65535	-		
P111	Easy sequence user parameter U (11)	0 to 65535	-		
P112	Easy sequence user parameter U (12)	0 to 65535	-		
P113	Easy sequence user parameter U (13)	0 to 65535	-		
P114	Easy sequence user parameter U (14)	0 to 65535	-		
P115	Easy sequence user parameter U (15)	0 to 65535	-		
P116	Easy sequence user parameter U (16)	0 to 65535	-		
P117	Easy sequence user parameter U (17)	0 to 65535	-		
P118	Easy sequence user parameter U (18)	0 to 65535	-		
P119	Easy sequence user parameter U (19)	0 to 65535	-		
P120	Easy sequence user parameter U (20)	0 to 65535	-		
P121	Easy sequence user parameter U (21)	0 to 65535	-		
P122	Easy sequence user parameter U (22)	0 to 65535	-		
P123	Easy sequence user parameter U (23)	0 to 65535	-		
P124	Easy sequence user parameter U (24)	0 to 65535	-		
P125	Easy sequence user parameter U (25)	0 to 65535	-		
P126	Easy sequence user parameter U (26)	0 to 65535	-		
P127	Easy sequence user parameter U (27)	0 to 65535	-		
P128	Easy sequence user parameter U (28)	0 to 65535	-		
P129	Easy sequence user parameter U (29)	0 to 65535	-		
P130	Easy sequence user parameter U (30)	0 to 65535	-		
P131	Easy sequence user parameter U (31)	0 to 65535	-		

Chapter 8 Appendix

*11

Display code	Inverter capacity	Minimum unit
P034	Under 55(kW)	0 to 200
	75 to 132(kW)	0 to 180
	Over 160(kW)	0 to 180

*12

Display code	Inverter capacity	Minimum unit
P037	Under 55(kW)	-200 to +200
	75 to 132(kW)	-180 to +180
	Over 160(kW)	-180 to +180