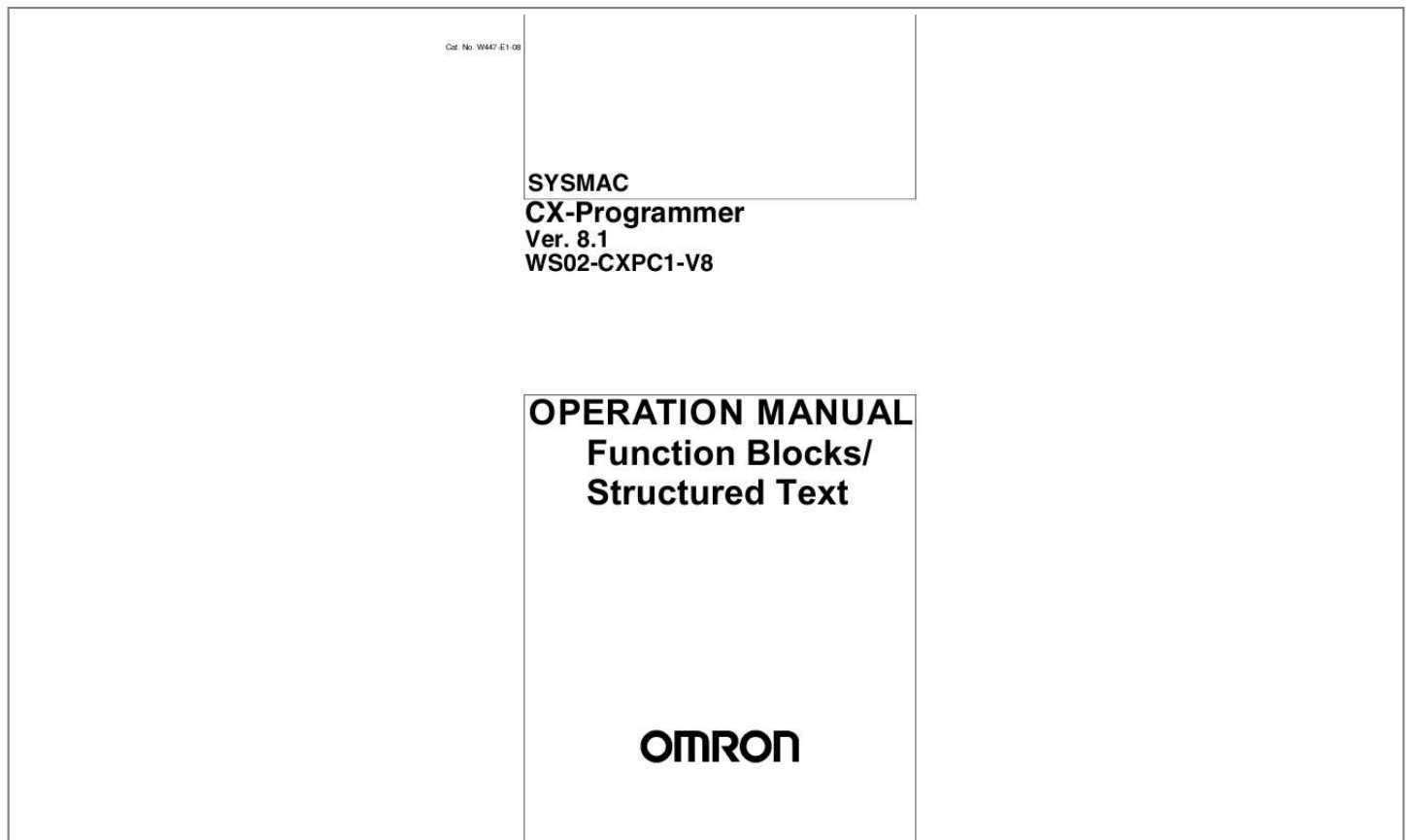




# Your PDF Guides

You can read the recommendations in the user guide, the technical guide or the installation guide for OMRON CX-PROGRAMMER V8.1. You'll find the answers to all your questions on the OMRON CX-PROGRAMMER V8.1 in the user manual (information, specifications, safety advice, size, accessories, etc.). Detailed instructions for use are in the User's Guide.

**User manual OMRON CX-PROGRAMMER V8.1**  
**User guide OMRON CX-PROGRAMMER V8.1**  
**Operating instructions OMRON CX-PROGRAMMER V8.1**  
**Instructions for use OMRON CX-PROGRAMMER V8.1**  
**Instruction manual OMRON CX-PROGRAMMER V8.1**



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>

**Manual abstract:**

No. W447-E1-08 SYSMAC CX-Programmer Ver. @@@@Always heed the information provided with them. @@Additionally, there may be severe property damage. Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury. Additionally, there may be severe property damage. Indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury, or property damage. !Caution OMRON Product References All OMRON products are capitalized in this manual. The word "Unit" is also capitalized when it refers to an OMRON product, regardless of whether or not it appears in the proper name of the product. The abbreviation "Ch," which appears in some displays and on some OMRON products, often means "word" and is abbreviated "Wd" in documentation in this sense.

The abbreviation "PLC" means Programmable Controller. "PC" is used, however, in some Programming Device displays to mean Programmable Controller.

Visual Aids The following headings appear in the left column of the manual to help you locate different types of information. Note Indicates information of particular interest for efficient and convenient operation of the product. 1,2,3.

.. 1. Indicates lists of one sort or another, such as procedures, checklists, etc. OMRON, 2008 All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, mechanical, electronic, photocopying, recording, or otherwise, without the prior written permission of OMRON. No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice. Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained in this publication. v vi

Part 1: Function Block SECTION 1 Introduction to Function Blocks SECTION 2 Function Block Specifications SECTION 3 Creating Function Blocks Part 2: Structured Text SECTION 4 Introduction to Structured Text SECTION 5 Structured Text (ST) Language Specifications SECTION 6 Creating ST Programs Appendices TABLE OF CONTENTS PRECAUTIONS . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .

. xxi 1 2 3 4 Intended Audience . . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .

. . . . . General Precautions . .

. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .

. . . . . Safety Precautions . . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .

. Application Precautions. . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .

. . . . . xxii xxiii Part 1: Function Blocks SECTION 1 Introduction to Function Blocks .

. . . . .  
. . . . .

. . . . . 1-1 1-2 1-3 1-4 1-5 1-6 Introducing the Function Blocks . . . . .

.....  
.....  
.....

*.. Function Blocks .....*

.....  
.....  
.....  
.....

*. Variables .....*

.....  
.....  
.....  
.....

*..... Converting Function Block Definitions to Library Files .....*

.....  
.....  
.....

*... Usage Procedures. ....*

.....  
.....  
.....  
.....

*. Version Upgrade Information .....*

.....  
.....  
.....  
.....

*..... 3 4 11 18 23 23 25 SECTION 2 Function Block Specifications .*

.....  
.....  
.....

*... 2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 Function Block Specifications .....*

.....  
.....  
.....

*. Data Types Supported in Function Blocks .....*

.....  
.....  
.....

..... Instance Specifications .

.....  
.....  
.....

.....  
.....  
.....

.. Programming Restrictions .....

.....  
.....  
.....

... Function Block Applications Guidelines .

.....  
.....

.....  
.....  
.....

... Precautions for Instructions with Operands Specifying the First or Last of Multiple Words Instruction Support and Operand Restrictions. ....

.....  
.....  
.....

..... CPU Unit Function Block Specifications .

.....  
.....  
.....

..... Number of Function Block Program Steps and Instance Execution Time ..

.....  
.....

.. 29 30 41 42 51 56 65 68 70 76 SECTION 3 Creating Function Blocks. ....

.....  
.....  
.....

3-1 3-2 Procedural Flow. ....

.....  
.....  
.....

... Procedures .

.....  
.....  
.....  
.....

.....  
.....  
.....

.. 79 80 82 ix TABLE OF CONTENTS Part 2: Structured Text (ST) SECTION 4 Introduction to Structured Text .....

.....  
.... 131 4-1 4-2 *ST Language* .

.....  
.....

.....  
.....

.....  
.....

.....  
.....

*CX-Programmer Specifications* . . . . .

.....  
.....

.....  
.....

.....  
.....

.. 132 133 *SECTION 5 Structured Text (ST) Language Specifications* . . . . .

.. 135 5-1 5-2 5-3 5-4 5-5 5-6 5-7 *Structured Text Language Specifications* . . . . .

.....  
.....

.....  
.....

.....  
.....

. *Data Types Used in ST Programs* . . . . .

.....  
.....

.....  
.....

..... *Inputting ST Programs* . . . . .

.....  
.....

.....  
.....

.....  
.....

*ST Language Configuration* . . . . .

.....  
.....

.....  
.....

.... *Statement Descriptions* .

.....  
.....

.....  
.....

.....  
.....

.. *ST-language Program Example* . . . . .

.....  
.....

.....  
.....

.....  
..... *Restrictions* .

.....  
.....  
.....

.....  
.....  
.....

. 136 137 138 141 148 166 167 SECTION 6 *Creating ST Programs* .....

.....  
.....  
.....

. 169 6-1 *Procedures* .....

.....  
.....  
.....

.....  
.....  
.....

.... 170 *Appendices A B C System-defined external variables supported in function blocks* .....

.....  
.....

..... *Structured Text Errors* .....

.....  
.....  
.....

.....  
.....

.... *Function Descriptions* .

.....  
.....  
.....  
.....

... 181 183 187 *Index* .

.....  
.....

.....  
.....  
.....

..... 201 *Revision History* ..

.....  
.....  
.....

... 203 x *About this Manual: This manual describes the CX-Programmer operations that are related to the function block functions and Structured Text (ST) functions. The function block and structure text functionality of CX-Programmer Ver. 8.1 is supported by CJ2H CPU Units, by CS1-H, CJI-H, and CJIM CPU Units with unit version 3.0 or later, by CP-series CPU Units, and by NSJ-series and FQMI-series Controllers. Some function block and structure text functionality, however, is supported only by CJ2H CPU Units, by CS1-H, CJI-H, and CJIM CPU Units with unit version 4.0 or later.*

*For details, refer to 1-6 Version Upgrade Information. For information on functionality other than function blocks and structure text, refer to the following manuals. · CX-Programmer : CX-Programmer Operation Manual (W446) and CX-Programmer Operation Manual: SFC (W469) · CPU Unit : The operation manuals for the CS-series, CJ-series, CP-series, and NSJ-series Controllers CX-Programmer Ver.*



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)

<http://yourpdfguides.com/dref/2890444>

8.1 Manuals Name SYSMAC WS02-CXPC1-V8 CX-Programmer Ver.

8.1 Operation Manual Function Blocks/Structured Text SYSMAC WS02-CXPC1-V8 CX-Programmer Operation Manual SYSMAC WS02-CXPC1-V8 CX-Programmer Operation Manual: SFC CX-Net Operation Manual SYSMAC CXONE-AL@@C-V3 SYSMAC CXONE-AL@@D-V3 CX-Integrator Operation Manual Cat. No. Contents Explains how to use the CX-Programmer Ver. 8.

I software's W447 function block and structured text functions. For explanations (this manual) of other shared CX-Programmer functions, refer to the CXProgrammer Operation Manual (W446). W446 Provides information on how to use the CX-Programmer for all functionality except for function blocks.

W469 Explains how to use the SFC programming functions. For explanations of other shared CX-Programmer functions, refer to the CX-Programmer Operation Manual (W446). W362 Information on setting up networks, such as setting data links, routing tables, and unit settings. W445 Describes the operating procedures for the CX-Integrator. xi CS1-H, CJ1-H, and CJ1M CPU Unit Manuals Name SYSMAC CJ Series CJ2H-CPU6@-EIP, CJ2H-CPU6@ Programmable Controllers Hardware User's Manual Cat. No. Contents W472 Provides an outline of and describes the design, installation, maintenance, and other basic operations for the CJ-series CJ2 CPU Units.

The following information is included: Overview and features System configuration Installation and wiring Troubleshooting W473 Use this manual together with the W473. Describes programming and other methods to use the functions of the CJ2 CPU Units. The following information is included: CPU Unit operation Internal memory areas Programming Tasks CPU Unit built-in functions W474 Use this manual together with the W472. Describes the ladder diagram programming instructions supported by CS/CJ-series or NSJ-series PLCs. When programming, use this manual together with the Operation Manual or Hardware User's Manual (CS1: W339, CJ1: W393, or CJ2:W472) and Programming Manual or Software User's Manual (CS1/CJ1:W394 or CJ2:W473). SYSMAC CJ Series CJ2H-CPU6@-EIP, CJ2H-CPU6@ Programmable Controllers Software User's Manual SYSMAC CS/CJ Series CS1G/H-CPU@@-EVI, CS1G/H-CPU@@H, CS1D-CPU@@H, CS1D-CPU@@S, CJ2H-CPU6@-EIP, CJ2H-CPU6@, CJ1H-CPU@@H-R CJ1G-CPU@@, CJ1G/H-CPU@@H, CJ1G-CPU@@P, CJ1M-CPU@@ SYSMAC One NSJ Series NSJ@-@@@ (B)-G5D NSJ@-@@@ (B)-M3D Programmable Controllers Instructions Reference Manual SYSMAC CS Series CS1G/H-CPU@@-EVI, CS1G/H-CPU@@H Programmable Controllers Operation Manual W339 Provides an outline of and describes the design, installation, maintenance, and other basic operations for the CS-series PLCs. The following information is included: An overview and features The system configuration Installation and wiring I/O memory allocation Troubleshooting SYSMAC CJ Series CJ1G-CPU@@, CJ1G/H-CPU@@H, CJ1H-CPU@@H-R, CJ1G-CPU@@P, CJ1M-CPU@@ Programmable Controllers Operation Manual W393 Use this manual together with the W394. Provides an outline of and describes the design, installation, maintenance, and other basic operations for the CJ-series PLCs. The following information is included: An overview and features The system configuration Installation and wiring I/O memory allocation Troubleshooting Use this manual together with the W394. xii Name SYSMAC CS/CJ Series CS1G/H-CPU@@-EVI, CS1G/H-CPU@@H, CJ1G-CPU@@, CJ1G/H-CPU@@H, CJ1H-CPU@@H-R, CJ1G-Clers Programming Manual (W451).

Provides the following information on the CP-series CP1H and CP1L PLCs: · Programming instructions · Programming methods · Tasks Use this manual together with the CP1H/CP1L Programmable Controllers Operation Manual (W450). W462 CP1L-M@@@-@ CP1L-L@@@-@ W451 CP1H-X@@@-@ CP1H-XA@@@-@ CP1H-Y@@@-@ CP1L-M@@@-@ CP1L-L@@@-@ SYSMAC CP Series CP1H/CP1L CPU Unit Programming Manual Installation from CX-One For details on procedures for installing the CX-Programmer from CX-One FA Integrated Tool Package, refer to the CX-One Ver. 3.0 Setup Manual provided with CX-One. Cat.

No. W463 Model CXONE-AL@@C-V3/AL@@D-V3 Manual name CX-One Setup Manual Contents Installation and overview of CX-One FA Integrated Tool Package. xiv Overview of Contents Precautions provides general precautions for using the CX-Programmer Ver. 8.1.

Part 1 Part 1 contains the following sections. Section 1 introduces the function block functionality of the CX-Programmer and explains the features that are not contained in the non-function block version of CX-Programmer. Section 2 provides specifications for reference when using function blocks, including specifications on function blocks, instances, and compatible PLCs, as well as usage precautions and guidelines. Section 3 describes the procedures for creating function blocks on the CX-Programmer. Part 2 Part 2 contains the following sections. Section 4 introduces the structure text programming functionality of the CX-Programmer and explains the features that are not contained in the non-structured text version of CX-Programmer. Section 5 provides specifications for reference when using structured text programming, as well as programming examples and restrictions. Section 6 explains how to create ST programs. Appendices provide information on structured text errors and ST function descriptions. !WARNING Failure to read and understand the information provided in this manual may result in personal injury or death, damage to the product, or product failure.

Please read each section in its entirety and be sure you understand the information provided in the section and related sections before attempting any of the procedures or operations given. xv xvi Read and Understand this Manual Please read and understand this manual before using the product. Please consult your OMRON representative if you have any questions or comments. Warranty and Limitations of Liability WARRANTY (1) The warranty period for the Software is one year from either the date of purchase or the date on which the Software is delivered to the specified location. (2) If the User discovers a defect in the Software (i.e., substantial non-conformity with the manual), and returns it to OMRON within the above warranty period, OMRON will replace the Software without charge by offering media or downloading services from the Internet. And if the User discovers a defect in the media which is attributable to OMRON and returns the Software to OMRON within the above warranty period, OMRON will replace the defective media without charge. If OMRON is unable to replace the defective media or correct the Software, the liability of OMRON and the User's remedy shall be limited to a refund of the license fee paid to OMRON for the Software. LIMITATIONS OF LIABILITY (1) THE ABOVE WARRANTY SHALL CONSTITUTE THE USER'S SOLE AND EXCLUSIVE REMEDIES AGAINST OMRON AND THERE ARE NO OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>



IN NO EVENT WILL OMRON BE LIABLE FOR ANY LOST PROFITS OR OTHER INDIRECT, INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF USE OF THE SOFTWARE. (2) OMRON SHALL ASSUME NO LIABILITY FOR DEFECTS IN THE SOFTWARE BASED ON MODIFICATION OR ALTERATION OF THE SOFTWARE BY THE USER OR ANY THIRD PARTY. (3) OMRON SHALL ASSUME NO LIABILITY FOR SOFTWARE DEVELOPED BY THE USER OR ANY THIRD PARTY BASED ON THE SOFTWARE OR ANY CONSEQUENCE THEREOF. xvii Application

Considerations **SUITABILITY FOR USE** THE USER SHALL NOT USE THE SOFTWARE FOR A PURPOSE THAT IS NOT DESCRIBED IN THE ATTACHED USER MANUAL. xviii Disclaimers **CHANGE IN SPECIFICATIONS** The software specifications and accessories may be changed at any time based on improvements or for other reasons.

**EXTENT OF SERVICE** The license fee of the Software does not include service costs, such as dispatching technical staff. **ERRORS AND OMISSIONS** The information in this manual has been carefully checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical, or proofreading errors, or omissions. xix xx **PRECAUTIONS** This section provides general precautions for using the CX-Programmer and the Programmable

Logic Controller. The information contained in this section is important for the safe and reliable application of the CX-Programmer and Programmable Controller. You must read this section and understand the information contained before attempting to set up or operate the CX-Programmer and Programmable Controller.

1 2 3 4 Intended Audience . . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .

. . . . . General Precautions . . . . .

. . . . .  
. . . . .  
. . . . .

. . . . . Safety Precautions.

. . . . .  
. . . . .  
. . . . .  
. . . . .

. . . . . Application Precautions . . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .

. xxii xxii xxii xxiii xxi Intended Audience 1 1 Intended Audience This manual is intended for the following personnel, who must also have knowledge of electrical systems (an electrical engineer or the equivalent). · Personnel in charge of installing FA systems. · Personnel in charge of designing FA systems. · Personnel in charge of managing FA systems and facilities.

2 General Precautions The user must operate the product according to the performance specifications described in the operation manuals. Before using the product under conditions which are not described in the manual or applying the product to nuclear control systems, railroad systems, aviation systems, vehicles, combustion systems, medical equipment, amusement machines, safety equipment, and other systems, machines, and equipment that may have a serious influence on lives and property if used improperly, consult your OMRON representative. Make sure that the ratings and performance characteristics of the product are sufficient for the systems, machines, and equipment, and be sure to provide the systems, machines, and equipment with double safety mechanisms. This manual provides information for programming and operating the product. Be sure to read this manual before attempting to use the product and keep this manual close at hand for reference during operation. !WARNING It is extremely important that a PLC and all PLC Units be used for the specified purpose and under the specified conditions, especially in applications that can directly or indirectly affect human life. You must consult with your OMRON representative before applying a PLC System to the above-mentioned applications. 3 Safety Precautions !WARNING Confirm safety sufficiently before transferring I/O memory area status from the CX-Programmer to the actual CPU Unit. The devices connected to Output Units may malfunction, regardless of the operating mode of the CPU Unit. Caution is required in respect to the following functions.

· Transferring from the CX-Programmer to real I/O (CIO Area) in the CPU Unit using the PLC Memory Window. · Transferring from file memory to real I/O (CIO Area) in the CPU Unit using the Memory Card Window. !Caution Variables must be specified either with AT settings (or external variables), or the variables must be the same size as the data size to be processed by the instruction when specifying the first or last address of multiple words in the instruction operand. 1. If a non-array variable with a different data size and without an AT setting is specified, the CX-Programmer will output an error when compiling.

2. Array Variable Specifications xxii Application Precautions 4 · When the size to be processed by the instruction operand is fixed: The number of array elements must be the same as the number of elements to be processed by the instruction. Otherwise, the CX-Programmer will output an error when compiling. · When the size to be processed by the instruction operand is not fixed: The number of array elements must be greater than or the same as the size specified in the other operands. · If the other operand specifying a size is a constant, the CX-Programmer will output an error when compiling.

· If the other operand specifying a size is a variable, the CX-Programmer will not output an error when compiling, even if the size of the array variable is not the same as that specified by the other operand (variable). A warning message, however, will be displayed. In particular, if the number of array elements is

*less than the size specified by the other operand (e.g., the size of the instruction operand is 16, and the number of elements registered in the actual variable table is 10), the instruction will execute read/write processing for the area that exceeds the number of elements.*

*For example, read/write processing will be executed for the 6 words following those for the number of elements registered in the actual variable table. If these words are used for other instructions (including internal variable allocations), unexpected operation will occur, which may result in serious accidents. Check that the system will not be adversely affected if the size of the variable specified in the operand is less than the size in the operand definition before starting PLC operations. !Caution Confirm safety at the destination node before transferring a program to another node or changing contents of the I/O memory area.*

*Doing either of these without confirming safety may result in injury.*

*!Caution Execute online editing only after confirming that no adverse effects will be caused by extending the cycle time. Otherwise, the input signals may not be readable. !Caution If synchronous unit operation is being used, perform online editing only after confirming that an increased synchronous processing time will not affect the operation of the main and slave axes. !Caution Confirm safety sufficiently before monitoring power flow and present value status in the Ladder Section Window or when monitoring present values in the Watch Window. If force-set/reset or set/reset operations are inadvertently performed by pressing short-cut keys, the devices connected to Output Units may malfunction, regardless of the operating mode of the CPU Unit.*



**[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)**

**<http://yourpdfguides.com/dref/2890444>**

4 Application Precautions Observe the following precautions when using the CX-Programmer. · User programs cannot be uploaded to the CX-Programmer. · Observe the following precautions before starting the CX-Programmer. · Exit all applications not directly related to the CX-Programmer. Particularly exit any software such as screen savers, virus checkers, E-mail or other communications software, and schedulers or other applications that start up periodically or automatically.

xxiii Application Precautions 4 · Disable sharing hard disks, printers, or other devices with other computers on any network. · With some notebook computers, the RS-232C port is allocated to a modem or an infrared line by default. Following the instructions in documentation for your computer and enable using the RS-232C port as a normal serial port. · With some notebook computers, the default settings for saving energy do not supply the rated power to the RS-232C port. There may be both Windows settings for saving energy, as well as setting for specific computer utilities and the BIOS. Following the instructions in documentation for your computer, disable all energy saving settings. · Do not turn OFF the power supply to the PLC or disconnect the connecting cable while the CX-Programmer is online with the PLC. The computer may malfunction. · Confirm that no adverse effects will occur in the system before attempting any of the following. Not doing so may result in an unexpected operation.

· Changing the operating mode of the PLC. · Force-setting/force-resetting any bit in memory. · Changing the present value of any word or any set value in memory. · Check the user program for proper execution before actually running it on the Unit. Not checking the program may result in an unexpected operation.

· When online editing is performed, the user program and parameter area data in CJ2, CS1-H, CJI-H, CJ1M, and CPH CPU Units is backed up in the built-in flash memory. The BKUP indicator will light on the front of the CPU Unit when the backup operation is in progress. Do not turn OFF the power supply to the CPU Unit when the BKUP indicator is lit. The data will not be backed up if power is turned OFF. To display the status of writing to flash memory on the CX-Programmer, select Display dialog to show PLC Memory Backup Status in the PLC properties and then select Windows PLC Memory Backup Status from the View Menu.

· Programs including function blocks (ladder programming language or structured text (ST) language) can be downloaded or uploaded in the same way as standard programs that do not contain function blocks. Tasks including function blocks, however, cannot be downloaded in task units (uploading is possible).

· If a user program containing function blocks created on the CX-Programmer Ver. 5.0 or later is downloaded to a CPU Unit that does not support function blocks (CS/CJ-series CPU Units with unit version 2.0 or earlier), all instances will be treated as illegal commands and it will not be possible to edit or execute the user program. · If the input variable data is not in boolean format, and numerical values only (e.g., 20) are input in the parameters, the actual value in the CIO Area address (e.g.

, 0020) will be passed. Therefore, be sure to include an &, #, or +, - prefix before inputting the numerical value. · Addresses can be set in input parameters, but an address itself cannot be passed as an input variable. (Even if an address is set as an input parameter, the value passed to the function block will be that for the size of data of the input variable.) Therefore, an input variable cannot be used as the operand of an instruction in the function block when the operand specifies the first or last of multiple words. With CX-Programmer version 7.0, use xxiv Application Precautions 4 an input-output variable specified as an array variable (with the first address set for the input parameter) and specify the first or last element of the array variable, or, with any version of CX-

Programmer, use an internal variable with an AT setting. Alternatively, specify the first or last element in an internal variable specified as an array variable. Values are passed in a batch from the input parameters to the input variables or input-output variables before algorithm execution (not at the same time as the instructions in the algorithm are executed). Therefore, to pass the value from a parameter to an input variable or input-output variable when an instruction in the function block algorithm is executed, use an internal variable or external variable instead of an input variable or input-output variable. The same applies to the timing for writing values to the parameters from output variables. · Always use internal variables with AT settings in the following cases. · The addresses allocated to Basic I/O Units, Special I/O Units, and CPU Bus Units cannot be registered to global symbols, and these variables cannot be specified as external variables (e.g., the data set for global variables may not be stable).

· Use internal variables when Auxiliary Area bits other than those preregistered to external variables are registered to global symbols and these variables are not specified as external variables. · Use internal variables when specifying PLC addresses for another node on the network: For example, the first destination word at the remote node for SEND(090) and the first source word at the remote node for RECV(098). · Use internal variables when the first or last of multiple words is specified by an instruction operand and the operand cannot be specified as an array variable (e.g., the number of array elements cannot be specified).

xxv Application Precautions 4 xxvi Part 1: Function Blocks SECTION 1 Introduction to Function Blocks This section introduces the function block functionality of the CX-Programmer and explains the features that are not contained in the non-function block version of CX-Programmer. 1-1 Introducing the Function Blocks. . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .  
1-1-1 1-1-2 1-1-3 1-1-4 1-2 Overview and Features . . . . .

. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .  
. . . . .  
· Function Block Specifications . . . . .  
. . . . .  
. . . . .  
. . . . .

Files Created with CX-Programmer Ver. 6.0 or Later . . . . .  
. . . Function Block Menus in CX-Programmer Ver. 5.  
0 (and later Versions) . . . . .  
. . . . .

.....  
.....  
.....  
... *Outline* .....

.....  
.....  
.....  
.....

.....  
*Advantages of Function Blocks* .....

.....  
.....

..... *Function Block Structure* ..

.....  
.....  
.....

.....  
*Introduction*.....

.....  
.....

.....  
.....

..... *Variable Usage and Properties* .....

.....  
.....  
.....

..... *Variable Properties* .....

.....  
.....  
.....

.....  
.....

.. *Variable Properties and Variable Usage* .....

.....  
.....

.... *Internal Allocation of Variable Addresses* .....

.....  
.... 4 4 5 8 8 11 11 12 13 18 18 19 19 20 21 23 23 23 24 25 *Function Blocks* .



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>

.....  
.....  
.....  
.....  
.....

. 1-2-1 1-2-2 1-2-3 1-3 Variables .....

.....  
.....  
.....  
.....  
.....

.... 1-3-1 1-3-2 1-3-3 1-3-4 1-3-5 1-4 1-5 Converting Function Block Definitions to Library Files .....

Usage Procedures .....

.....  
.....  
.....  
.....

..... 1-5-1 1-5-2 Creating Function Blocks and Executing Instances ....

..... Reusing Function Blocks .

.....  
.....  
.....

.. 1-6 Version Upgrade Information ...

.....  
.....  
.....

... 3 Introducing the Function Blocks Section 1-1 1-1-1 1-1-1 Introducing the Function Blocks Overview and Features The CX-Programmer Ver.

5.0 (and later versions) is a Programming Device that can use standard IEC 61131-3 function blocks. The CX-Programmer function block function is supported for CJ2H CPU Units, CPHH CPU Units, NSJ-series NSJ Controllers, and FQMI Flexible Motion Controllers as well as CS/CJ-series CPU Units with unit version 3.0 or later and has the following features. · User-defined processes can be converted to block format by using function blocks. · Function block algorithms can be written in the ladder programming language or in the structured text (ST) language. (See note.) · When ladder programming is used, ladder programs created with nonCX-Programmer Ver. 4.0 or earlier can be reused by copying and pasting.

· When ST language is used, it is easy to program mathematical processes that would be difficult to enter with ladder programming. Note The ST language is an advanced language for industrial control (primarily Programmable Logic Controllers) that is described in IEC 61131-3. The ST language supported by CX-Programmer conforms to the IEC 61131-3 standard. · Function blocks can be created easily because variables do not have to be declared in text. They are registered in variable tables. A variable can be registered automatically when it is entered in a ladder or ST program. Registered variables can also be entered in ladder programs after they have been registered in the variable table. · A single function block can be converted to a library function as a single file, making it easy to reuse function blocks for standard processing. · A program check can be performed on a single function block to easily confirm the function block's reliability as a library function. · Programs containing function blocks (ladder programming language or structured text (ST) language) can be downloaded or uploaded in the same way as standard programs that do not contain function blocks.

Tasks containing function blocks, however, cannot be downloaded in task units (uploading is possible). · One-dimensional array variables are supported, so data handling is easier for many applications. Note The IEC 61131 standard was defined by the International Electrotechnical Commission (IEC) as an international programmable logic controller (PLC) standard. The standard is divided into 7 parts. Specifications related to PLC programming are defined in Part 3 Textual Languages (IEC 61131-3).

· A function block (ladder programming language or structured text (ST) language) can be called from another function block (ladder programming language or structured text (ST) language). Function blocks can be nested up to 8 levels and ladder/ST language function blocks can be combined freely. 4 Introducing the Function Blocks Section 1-1 1-1-2 Function Block Specifications For specifications that are not listed in the following table, refer to the CX-Programmer Operation Manual (W446). Specifications WS02-CXPC1-E-V7 CD-ROM CS/CJ-series CS1-H, CJ1-H, and CJ1M CPU Units with unit version 3.0 or later Note The function block and structured are compatible.

Device Type CPU Type text functions supported by CS/ CJ-series CPU Units with unit ver- · CJ2H CJ2H-CPU68/67/66/65/64/68-EIP/67-EIP/66-EIP/65-EIP sion 4.0 or later can not be used /64-EIP in CS/CJ-series CPU Units with · CS1G-H CS1G-CPU42H/43H/44H/45H unit version 3.0 or earlier, CPCS1H-CPU63H/64H/65H/66H/67H series PLCs, NSJ-series PLCs, or · CS1H-H · CJ1G-H CJ1G-CPU42H/43H/44H/45H FQMI-series PLCs. For details, refer to

1-6 Version · CJ1H-H CJ1H-CPU65H/66H/67H/64H-R/65H-R/66H-R/67H-R Upgrade Information. · CJ1M CJ1M-CPU11/12/13/21/22/23 The following CP-series CPU Units are compatible. · CP1H CP1H-X/XA/Y · CP1L CP1L-M/L Note If a user program containing function blocks created on the CX-Programmer Ver. 5.0 or later is downloaded to a CPU Unit that does not support function blocks (CS/CJ-series CPU Units with unit version 2.0 or earlier), all instances will be treated as illegal commands and it will not be possible to edit or execute the user program. G5D (Used for the NSJ5-TQ0@-G5D, NSJ5-SQ0@-G5D, NSJ8-TV0@-G5D, NSJ10-TV0@-G5D, and NSJ12-TS0@-G5D) M3D (Used for the NSJ5-TQ0@-M3D, NSJ5-SQ0@-M3D, and NSJ8-TV0@-M3D) · FQM1-CM FQM1-CM002 · FQM1-MMA FQM1-MMA22 · FQM1-MMP FQM1-MMP22 CS/CJ/CP Series Function Restrictions · Instructions Not Supported in Function Block Definitions Block Program Instructions (BPRG and BEND), Subroutine Instructions (SBS, GSBS, RET, MCRO, and SBN), Jump Instructions (JMP, CJP, and CJPN), Step Ladder Instructions (STEP and SNXT), Immediate Refresh Instructions (!), I/O REFRESH (IORF), ONE-MS TIMER (TMHH and TMHHX) (These timers can be used with CJ1-H-R CPU Units.

) Note For details and other restrictions, refer to 2-4 Programming Restrictions. · NSJ Item Model number Setup disk Compatible CPU Units (PLC models) 5 Introducing the Function Blocks Item Defining and creating function blocks Section 1-1 Functions not supported by CX-Programmer Ver. 4.0 or earlier.

Specifications Number of CJ2H Units: function · CJ2H-CPU6@(-EIP): 2,048 max. per CPU Unit block definiCS1-H/CJ1-H CPU Units: tions · Suffix -CPU44H/45H/64H/65H/66H/67H/64H-R/65H-R/66H-R/67H-R: 1,024 max. per CPU Unit · Suffix -CPU42H/43H/63H: 128 max. per CPU Unit CJ1M CPU Units: · CJ1M-CPU11/12/13/21/22/23: 128 max. per CPU Unit CP1H CPU Units: · All models: 128 max. per CPU Unit CP1L CPU Units: · CP1L-M/L: 128 max.

per CPU Unit NSJ Controllers: · NSJ@-@@@@-G5D: 1,024 max. per Controller; NSJ@-@@@@-M3D: 128 max. per Controller FQM1 Flexible Motion Controllers: · FQM1-CM002/MMA22/MMP22: 128 max. per Controller Function 64 characters max. block names 6 Introducing the Function Blocks Item Defining Variables and creating function blocks Section 1-1 Specifications 30,000 characters max.

Input variables (Inputs), output variables (Outputs), input-output variables (In Out), internal variables (Internals), and external variables (Externals) Functions not supported by CX-Programmer Ver. 4.0 or earlier. Variable names Variable types Number of variables used in a function block (not including internal variables, external variables, EN, and EN0) Language Creating instances Number of instances Maximum number of variables per function block definition · Input-output variables: 16 max.



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>

· Input variables + input-output variables: 64 max.

· Output variables + input-output variables: 64 max. Allocation of addresses Automatic allocation (The allocation range can be used by variables set by the user.) Actual address specification Supported Array specifications Supported (one-dimensional arrays only and only for internal variables and input-output variables) Function blocks can be created in ladder programming language or structured text (ST, see note). CJ2H Units: · CJ2H-CPU6@(-EIP): 2,048 max. per CPU Unit CS1-H/CJ1-H CPU Units: · Suffix -CPU44H/45H/64H/65H/66H/67H/64H-R/65H-R/66H-R/67H-R: 2,048 max. per CPU Unit · Suffix -CPU42H/43H/63H: 256 max. per CPU Unit CJ1M CPU Units: · CJ1M-CPU11/12/13/21/22/23: 256 max. per CPU Unit CP1H CPU Units: · All models: 256 max. per CPU Unit CP1L CPU Units: · CP1L-M/L: 256 max. per CPU Unit NSJ Controllers: · SJ@-@@@-G5D: 2,048 max.

per Controller; NSJ@-@@@-M3D: 256 max. per Controller FQM1 Flexible Motion Controllers: · FQM1-CM002/MMA22/MMP22: 256 max. per Controller 15,000 characters max. The project file (.xcp/ct) Includes function block definitions and instances. The file memory program file (\*.obj) includes function block definitions and instances. Each function block definition can be stored as a single file (.cxf) for reuse in other projects. Storing function blocks as files Instance names Project files Program files Function block library files Note The structured text (ST language) conforms to the IEC 61131-3 standard, but CX-Programmer Ver.

5.0 supports only assignment statements, selection statements (CASE and IF statements), iteration statements (FOR, WHILE, REPEAT, and EXIT statements), RETURN statements, arithmetic operators, logical operators, comparison functions, numeric functions, standard string functions, numeric string functions, OMRON expansion functions, and comments. For details, refer to SECTION 5 Structured Text (ST) Language Specifications in Part 2: Structured Text (ST). 7 Introducing the Function Blocks Section 1-1 1-1-3 Files Created with CX-Programmer Ver. 6.

0 or Later Projects created using CX-Programmer that contain function block definitions and projects with instances are saved in the same standard project files (\*.xcp) and file memory program files (\*.obj). The following diagram shows the contents of a project. The function block definitions are created at the same directory level as the program within the relevant PLC directory.

Project file (.xcp) PLC1 Global symbol table I/O table PLC Setup PLC memory table Program (with rung comments) Local symbol table Section 1 (with instances) Section 2 (with instances) END section (with instances) Function block definitions FunctionBlock1 FunctionBlock2 Project Files (\*.xcp) and File Memory Program Files (\*.obj) PLC2 Instances created in program sections. Each function block can be stored in a separate definition file (.cxf). Function Block Library Files (\*.cxf) Note Project Text Files Containing Function Blocks (\*.cxt) A function block definition created in a project with CX-Programmer Ver. 6.

0 can be saved as a file (1 definition = 1 file), enabling definitions to be loaded into other programs and reused. When function blocks are nested, all of the nested (destination) function block definitions are included in this function block library file (.cxf). Data equivalent to that in project files created with CX-Programmer Ver. 6.0 (\*.xcp) can be saved as CXT text files (\*.cxt). 1-1-4 Function Block Menu in CX-Programmer Ver. 5.

0 (and later Versions) The following tables list menus related to function blocks in CX-Programmer Ver. 5.0 and later versions. For details on all menus, refer to the CX-Programmer Operation Manual (W446). Main Menu Main menu File Submenu Function Block Shortcut Function Reads the saved function block library files (\*.

cxf). Saves the created function block definitions to a file ([function block library file]\*.cxf). Load Function --Block from File Save Function --Block to File 8 Introducing the Function Blocks Main menu Edit Submenu Update Function Block Shortcut --Function Section 1-1 To Lower Layer Function Block (ladder) generation ----- View Monitor FB Instance --- To Lower Layer --- To Upper Layer Window FB Instance Viewer ----- Insert Function Block Invocation Function Block Parameter F P PLC Memory Function Allocation/ tion SFC Memory Program Online Edit Function Block /SFC Memory Allocation Function Block /SFC Memory Statistics Function Block Instance Address Optimize Function Block/SFC Memory Begin Send Change Cancel Transfer FB Source Release FB Online Edit Access Rights --- When a function block definition's input variables, output variables, or input-output variables have been changed after the instance was created, an error will be indicated by displaying the instance's left bus bar in red. This command updates the instance with the new information and clears the error.

Jumps to the function block definition for the selected instance. Generates a ladder-programmed function block for the selected program section while automatically determining address application conditions. When monitoring the program online, monitors ST variable status as well as I/O bit and word status (I/O bit monitor) of the ladder diagram in the instance. (Supported by CX-Programmer Ver. 6.1 and later only). Displays on the right side the contents of the function block definition of the selected instance. (Supported by CX-Programmer Ver. 6.0 and later only).

) Returns to the calling instance (ladder diagram or ST). (Supported by CX-Programmer Ver. 6.0 and later only.) Displays the FB Instance Viewer. (When nesting, the display shows details such as the relationship between instance nesting levels and allocated variable addresses in the instances.) Creates an instance in the program (section) at the present cursor location. When the cursor is located to the left of an input variable or the right of an output variable, sets the variable's input or output parameter. Sets the range of addresses (function block instance areas) internally allocated to the selected instance's variables. Checks the status of the addresses internally allocated to the selected instance's variables.

Checks the addresses internally allocated to each variable in the selected instance. Optimizes the allocation of addresses internally allocated to variables. Starts online editing of a function block. Transfers changes made during online editing of a function block. Cancels changes made to a function block being edited online.

Transfers only the function block source. Forcefully releases the access rights for function block, SFC, and ST online editing held by another user. --- --- --- ----- 9 Introducing the Function Blocks Main menu Tools Simulation Submenu Break Point \ Set/Clear Break Point Break Point \ Clear All Break Point Mode \ Run (Monitor Mode) Mode \ Stop (Program Mode) Mode \ Pause Step Run Step Run \ Step In Step Run \ Step Out Step Run \ Continuous Step Run Step Run \ Scan Run Shortcut --Function Sets or clears a break point.



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)

<http://yourpdfguides.com/dref/2890444>



Section 1-1 --- Clears all break points. ----- Executes continuous scanning.

(Sets the ladder execution engine's run mode to MONITOR mode.) Sets the simulator's operation mode to PROGRAM mode. ----- --- Pauses simulator operation. Executes just one step of the simulator's program. When there is a function block call instruction, this command moves to execution of the internal program step. When a function block's internal program step is being executed, this command returns to the next higher level (call source) and pauses execution. Executes steps continuously for a fixed length of time. --- Executes for one cycle and pauses execution. Used with the Step Run or Continuous Step Run commands to automatically scroll the display and always show the pause point. Displays a list of the break points that have been set.

(Operation can be jumped to a specified point.) Always Display --Current Execution Point Break Point List --- Main Pop-up Menus Pop-up Menu for Function Block Definitions Pop-up menu Insert Function Block Ladder Structured Text From file Function Creates a function block definition with a ladder programming language algorithm. Creates a function block definition with an ST language algorithm. Reads a function block definition from a function block library file (\*.cxf). Pop-up Menu for Inserted Function Blocks Pop-up menu Open Save Function Block File Compile FB online Edit Begin Send Change Cancel Transfer FB Source Release FB Online Edit Access Rights Function Displays the contents of the selected function block definition on the right side of the window. Saves the selected function block definition in a file. Compiles the selected function block definition. Starts online editing of a function block.

Transfers changes made during online editing of a function block.

Cancel Transfer FB Source Release FB Online Edit Access Rights Function Displays the contents of the selected function block definition on the right side of the window. Saves the selected function block definition in a file. Compiles the selected function block definition. Starts online editing of a function block.

Transfers changes made during online editing of a function block.

Searches for the variable. Variable names, variable comments, or all (text strings) can be searched. Replaces the variable. Deletes the variable. Changes only the name of the variable. Pop-up Menu for Instances Pop-up menu Edit Update Invocation Changes the instance name. When a function block definition's input variables, output variables, or input-output variables have been changed after the instance was created, an error will be indicated by displaying the instance's left bus bar in red. This command updates the instance with the new information and clears the error. When monitoring the program online, monitors I/O bit and word status (I/O bit monitor) of the ladder diagram in the instance. (Supported by CX-Programmer Ver.

6.0 and later only). When monitoring the program online, monitors ST variable status as well as I/O bit and word status (I/O bit monitor) of the ladder diagram in the instance. (Supported by CX-Programmer Ver. 6.1 and later only). Displays the FB variables registration Dialog Box in order to register a variable from the selected instance to the Watch Window. Displays the selected instance's function block definition on the right side of the window. Function Monitor FB Ladder Instance Monitor FB Instance Register in Watch Window Function Block Definition Shortcut Keys F Key: Pasting Function Block Definitions in Program Enter Key: Inputting Parameters Move the cursor to the position at which to create the copied function block instance in the Ladder Section Window, and press the F Key. This operation is the same as selecting Insert - Function Block Invocation.

Position the cursor at the left of the input variable or input-output variable, or at the right of the output variable and press the Enter Key. This operation is the same as selecting Insert - Function Block Parameter. 1-2 1-2-1 Function Blocks Outline A function block is a basic program element containing a standard processing function that has been defined in advance. Once the function block has been defined, the user just has to insert the function block in the program and set the I/O in order to use the function. As a standard processing function, a function block does not contain actual addresses, but variables.

The user sets addresses or constants in those variables. These address or constants are called parameters. The addresses used by the variables themselves are allocated automatically by the CX-Programmer for each program. 11 Function Blocks Section 1-2 With the CX-Programmer, a single function block can be saved as a single file and reused in other PLC programs, so standard processing functions can be made into libraries. Program 2 Copy of function block A Function block A Standard program section written with variables Program 1 Copy of function block A Variable Output Input Define in advance.

Insert in program. Set Variable Variable Output Set Copy of function block A Save function block as a file. Convert to library function. Input Output Variable Variable Function block A Reuse. To another PLC program 1-2-2 Advantages of Function Blocks Function blocks allow complex programming units to be reused easily. Once standard programming is created in a function block and saved in a file, it can be reused just by placing the function block in a program and setting the parameters for the function block's I/O. The ability to reuse existing function blocks will save significant time when creating/debugging programs, reduce coding errors, and make the program easier to understand. Structured Programming Easy-to-read "Black Box" Design Use One Function Block for Multiple Processes Reduce Coding Errors Black-boxing Know-how Data Protection Improved Reusability with Variable Programming Structured programs created with function blocks have better design quality and require less development time. The I/O operands are displayed as variable names in the program, so the program is like a "black box" when entering or reading the program and no extra time is wasted trying to understand the internal algorithm. Many different processes can be created easily from a single function block by using the parameters in the standard process as input variables (such as timer SVs, control constants, speed settings, and travel distances).

Coding mistakes can be reduced because blocks that have already been debugged can be reused. Read-protection can be set for function blocks to prevent programming knowhow from being disclosed.



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>



The variables in the function block cannot be accessed directly from the outside, so the data can be protected. (Data cannot be changed unintentionally.) The function block's I/O is entered as variables, so it isn't necessary to change data addresses in a block when reusing it. Processes that are independent and reusable (such as processes for individual steps, machinery, equipment, or control systems) can be saved as function block definitions and converted to library functions. Creating Libraries 12 Function Blocks Section 1-2 The function blocks are created with variable names that are not tied to actual addresses, so new programs can be developed easily just by reading the definitions from the file and placing them in a new program. Supports Nesting and Multiple Languages Mathematical expressions can be entered in structured text (ST) language. With CX-Programmer Ver. 6.

0 and later versions, function blocks can be nested. The function block nesting function allows just special processing to be performed in a ST-language function block nested within a ladder-language function block. Function block (ladder language) Call (Nesting) Function block (ST language) 1-2-3 Function Block Structure Function blocks consist of function block definitions that are created in advance and function block instances that are inserted in the program. Function Block Definitions Function block definitions are the programs contained in function blocks. Each function block definition contains the algorithm and variable definitions, as shown in the following diagram.

Function Block Definition Example: CLOCK PULSE Example: CLOCK PULSE 1. Algorithm tim\_b TIMX tim\_a OFF\_TIME tim\_a Algorithm Variable definitions TIMX tim\_b ON\_TIME ENO 2. Variable Definitions Usage Name Internal Internal Input Input tim\_a tim\_b ON\_TIME OFF\_TIME Type TIMER TIMER INT INT 1. Algorithm Standardized programming is written with variable names rather than real I/O memory addresses. In the CX-Programmer, algorithms can be written in either ladder programming or structured text.

2. Variable Definitions The variable table lists each variable's usage (input, output, input-output, or internal) and properties (data type, etc.). For details, refer to 1-3 Variables. Number of Function Block Definitions The maximum number of function block definitions that can be created for one CPU Unit is either 128 or 1,024 depending on the CPU Unit model. 13 Function Blocks Instances Section 1-2 To use an actual function block definition in a program, create a copy of the function block diagram and insert it in the program. Each function block definition that is inserted in the program is called an "instance" or "function block instance." Each instance is assigned an identifier called an "instance name." By generating instances, a single function block definition can be used to process different I/O data with the same function. Not yet in program and memory not yet allocated (abstract).

Function Block Definition FBI 1. Algorithm Standard program unit with variable names a, b, c, etc. Block instance in program with memory allocated. (object) Program Instance Memory used Automatic allocation Memory for FBI\_1 Memory for FBI\_2 Instance FBI\_1 of function block definition FBI Insert in program. Input data a b c Insert in program. Instance FBI\_2 of function block definition FBI Output data Output data Automatic allocation 2. Parameters Table defining usage and properties of variables a, b, c, etc. Different I/O data can be processed with the same function. Input data a b c Output data Output data Note Instances are managed by names. More than one instance with the same name can also be inserted in the program.

If two or more instances have the same name, they will use the same internal variables. Instances with different names will have different internal variables. For example, consider multiple function blocks that use a timer as an internal variable. In this case all instances will have to be given different names. If more than one instance uses the same name, the same timer would be used in multiple locations, resulting in duplicated use of the timer.

If, however, internal variables are not used or they are used only temporarily and initialized the next time an instance is executed, the same instance name can be used to save memory. instance\_A TIMER\_FB Function Block Definition TIMER\_FB Variable Definitions Internal variable: WORK\_NUM Use same internal variables. instance\_A TIMER\_FB Use different internal variables. instance\_B TIMER\_FB Number of Instances Multiple instances can be created from a single function block definition. Up to either 256 or 2,048 instances can be created for a single CPU Unit depending on the CPU Unit model.

The allowed number of instances is not related to the number of function block definitions and the number of tasks in which the instances are inserted. 14

Function Blocks Parameters Section 1-2 Each time an instance is created, set the real I/O memory addresses or constants for input variables, output variables, and input-output variables used to pass input data values to instances and obtain output data values from instances. These addresses and constants are called parameters. Instance of Function Block Definition A Input 0.00 Input 3.00 a c b Output 2.00 Set the constants or input source addresses from which to pass data. Set the constant or output destination address to which to pass data. Using Input Variables and Output Variables With input variables and output variables, it is not the input source address itself, but the contents at the input address in the form and size specified by the variable data type that is passed to the function block. In a similar fashion, it is not the output destination address itself, but the contents for the output address in the form and size specified by the variable data type that is passed from the function block.

Even if an input source address (i.e., an input parameter) or an output destination address (i.e., an output parameter) is a word address, the data that is passed will be the data in the form and size specified by the variable data type starting from the specified word address. Program Instance of Function Block Definition A Input D100 Input D200 m k Output D300 n Examples: If m is type WORD, one word of data from D100 will be passed to the variable. If n is type DWORD, two words of data from D200 and D201 will be passed to the variable. If k is type LWORD, four words of data from the variable will be passed to the D300 to D303. Note (1) Only addresses in the following areas can be used as parameters: CIO Area, Auxiliary Area, DM Area, EM Area (banks 0 to C), Holding Area, and Work Area. The following cannot be used: Index and Data Registers (both direct and indirect specifications) and indirect addresses to the DM Area and EM Area (both in binary and BCD mode).



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>

(2) Local and global symbols in the user program can also be specified as parameters. To do so, however, the data size of the local or global symbol must be the same as the data size of the function block variable. (3) When an instance is executed, input values are passed from parameters to input variables before the algorithm is processed. Output values are 15 Function Blocks Section 1-2 passed from output variables to parameters just after processing the algorithm.

If it is necessary to read or write a value within the execution cycle of the algorithm, do not pass the value to or from a parameter.

Assign the value to an internal variable and use an AT setting (specified addresses). !Caution If an address is specified in an input parameter, the values in the address are passed to the input variable. The actual address data itself cannot be passed. !Caution Parameters cannot be used to read or write values within the execution cycle of the algorithm. Use an internal variable with an AT setting (specified addresses).

Alternatively, reference a global symbol as an external variable. Using Input-Output Variables (In Out) When using an input-output variable, set the address for the input parameter. A constant cannot be set. The address set for the input parameter will be passed to the function block. If processing is performed inside the function block using the input-output variable, the results will be written to I/O starting at the address set for the size of the variable. Program Instance of function block definition A Automatically set. Input D200 Address passed. a a Output D200 Address passed. "a" changed by function block processing. D200 D201 Variable "a" Processing is performed inside the function block using variable "a."

" The resulting value is written to I/O memory for the size of variable "a" starting at address D200. Note Input-output variables are specified in a CX-Programmer variable table by selecting "In Out" for the variable usage. 16 Function Blocks Reference Information Section 1-2 A variety of processes can be created easily from a single function block by using parameter-like elements (such as fixed values) as input variables and changing the values passed to the input variables for each instance. Example: Creating 3 Instances from 1 Function Block Definition Instance CASCADE\_01 Algorithm Internal and I/O variables Cyclic task 0 P\_On EN &20 &10 CASCADE\_01 CONTROL ENO 1.0 ON\_TIME OFF\_TIME Function Block Definition Example: CONTROL Algorithm Variables Example: There are 3 FB instances and each has its own I/O and internal variables. Instance CASCADE\_02 Algorithm Internal and I/O variables CASCADE\_02 P\_On EN &15 &10 CONTROL ENO 1.1 ON\_TIME OFF\_TIME Instance CASCADE\_03 Algorithm Internal and I/O variables Cyclic task 1 P\_On &7 &8 CASCADE\_03 EN CONTROL ENO 1.2 ON\_TIME OFF\_TIME If internal variables are not used, if processing will not be affected, or if the internal variables are used in other locations, the same instance name can be used at multiple locations in the program. Cyclic task CASCADE 0 P\_On &100 CONTROL EN ENO PARA\_1 PARA\_2 1.0 The same instance can be used at multiple locations.

Function block definition Example: CONTROL &130 Algorithm Variables Instance CASCADE Algorithm Internal and I/O variables P\_On &50 &150 CASCADE CONTROL EN ENO PARA\_1 PARA\_2 1.1 Cyclic task 1 CASCADE P\_On &100 &200 CONTROL EN ENO PARA\_1 PARA\_2 1.2 Some precautions are required when using the same memory area. For example, if the same instance containing a timer instruction is used in more than one program location, the same timer number will be used causing coil duplication, and the timer will not function properly if both instructions are executed. 17 Variables Registration of Instances Section 1-3 Each instance name is registered in the global symbol table as a file name.

Program Instance (sample) of function block definition A a b c The instance is registered in the global symbol table with the instance name as the symbol name. Name Data type Address/ value N/A[Auto] sample FB [FunctionBlock1] Instance name The function block definition name is registered after FB in square parentheses [ ]. 1-3 1-3-1 Variables Introduction In a function block, the addresses (see note) are not entered as real I/O memory addresses, they are all entered as variable names. Each time an instance is created, the actual addresses used by the variable are allocated automatically in the specified I/O memory areas by the CX-Programmer. Consequently, it isn't necessary for the user to know the real I/O memory addresses used in the function block, just as it isn't necessary to know the actual memory allocations in a computer.

A function block differs from a subroutine in this respect, i.e., the function block uses variables and the addresses are like "black boxes." Example: Program Function block definition A Standard program section with variable names a, b, c, etc. a b Insert in program. Input 0.00 Input 3.00 c MOV Specify inputs and outputs at the same time. a c b Output 2.00 Instance of function block definition A Table indicating usage and properties of variables a, b, c, etc.

Usage: Inputs Properties: Name Type a c BOOL BOOL AT Initial Value Retained Status of 0.00 (1 or 0) is passed to a. 0.00 1 a 1 Status of b (1 or 0) is passed to 2.00. b 1 2.00 1 Status of 3.00 (1 or 0) is passed to c. 3.00 0 c 0 Usage: Outputs Properties: Name Type BOOL b AT Initial Value Retained The system automatically allocates the addresses used by variables a, b, and c.

For example, when W100 to W120 is set as the system's non-retained memory area, bit addresses such as a = W10000, b = W10001, and c = W10002 will be allocated. Note Constants are not registered as variables. Enter constants directly in instruction operands. 18 Variables Section 1-3 · Ladder programming language: Enter hexadecimal numerical values after the # and decimal values after the &. · Structured text (ST language): Enter hexadecimal numerical values after 16# and enter decimal numerical values as is.

Exception: Enter directly or indirectly specified addresses for Index Registers IR0 to IR15 and Data Registers DR0 to DR15 directly into the instruction operand. 1-3-2 Variable Usage and Properties The following variable types (usages) are supported. Internals: Inputs: Internal variables are used only within an instance. They cannot be used pass data directly to or from I/O parameters. Input variables can input data from input parameters outside of the instance. The default input variable is an EN (Enable) variable, which passes input condition data. Output variables can output data to output parameters outside of the instance. The default output variable is an ENO (Enable Out) variable, which passes the instance's execution status.



[You're reading an excerpt. Click here to read official OMRON CX-PROGRAMMER V8.1 user guide](http://yourpdfguides.com/dref/2890444)  
<http://yourpdfguides.com/dref/2890444>