ID78K3
INTEGRATED DEBUGGER

REFERENCE

For PC-9801 and 9821 Series (Windows™)
and IBM-PC/AT Series (Windows™)
Ver. V1.20

Document No. U10441EJ1V0UM00 (1st edition)
Date Published October 1995 P
Printed in Japan

IBM-PC/AT is a trademark of IBM Corporation.
386 and 486 are trademarks of Intel Corporation.
MS-DOS and Windows are trademarks of Microsoft Corporation.
Windows is an abbreviation of Microsoft® Windows™ Operating System.

The export of this product from Japan is prohibited without governmental license. To export or re-export this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

Preface

Thank you for purchasing the ID78K3 integrated debugger.

Conventional debuggers are used by entering commands directly. The ID78K3 integrated debugger, on the other hand, runs under Windows to provide a friendly, easy-to-use GUI (Graphical User Interface). Its operation is mouse-based, and operation is possible without having to refer to the manual. Also, frequently used commands are represented as buttons, allowing their activation simply by clicking the button with the mouse.

«Purpose»

The purpose of this manual is to provide the user with details of the capabilities of the integrated debugger.

This manual assumes that the reader is reasonably familiar with the operation of in-circuit emulators and Windows.

«Files supplied with the integrated debugger»

Files used with the integrated debugger

File name	Explanation
ID78K3.EXE	Debugger main section. The debugger is started by executing this file.
DB78K3.DLL	Contains libraries for file and symbol processing.
AS78K3.DLL	Contains libraries for assembly and disassembly.
EX78K3.DLL	Contains libraries for communication with the in-circuit emulator.
EX78K3.OM0	Downloaded into the in-circuit emulator when the debugger starts.
ID78K3.HLP	Help file.
EXPC.INI	Initial file. Used to specify a set point and an interrupt address for the PC interface board.

Sample programs

File name	Explanation
SAMPLE.C	Sample program written in C.
SUB.C	Sample program written in C. Contains the subroutines of SAMPLE.C.
SAMPLE.LNK	Load module file for sample programs SAMPLE.C and SUB.C. Compiled by μPD78366.

«Target device»

The device which is to be the target of debugging by the integrated debugger is called a target device. The table below lists target devices, their associated device files, microprograms, and the names of the CPUs which select the target devices.

Target device	CPU name	Device file
μPD78350	78350	D350.78K
μPD78350A	78350A	D350A.78K
μPD78350B	78350B	D350B.78K
μPD78352A	78352A	D352A.78K
μPD78352B	78352B	D352B.78K
μPD78P352	78P352	DP352.78K

Note: For details of other devices, contact your NEC sales representative or authorized dealer.

«In-circuit emulator»

An in-circuit emulator and dedicated interface board are required to use the integrated debugger. The table below lists the in-circuit emulator boards and interface boards that can be connected to host machines.

In-circuit emulator

Product name	Explanation
IE-784000-R	In-circuit emulator main board
IE-78350-R-EM-A	78K/III series common board
IE-783xx-R-EM1(Note 1)	Product type dependent board

Note 1. For details, contact your NEC sales representative or authorized dealer.

Interface boards

Product name	Explanation
IE-70000-98-IF-A	Interface board for PC-9801 and 9821 Series
IE-70000-98-IF-B	Interface board for PC-9801 and 9821 Series
IE-70000-98N-IF(Note 2)	Interface board for 98NOTE
IE-70000-PC-IF-B(Note 3)	Interface board for IBM-PC/AT Series

Note 2. The IE-70000-98N-IF is corrected to the expansion bus (110-pin type) of 98NOTE.
Note 3. The IE-70000-PC-IF-A cannot be used.

«Host machine»

The integrated debugger runs under Windows. The table below lists the requirements for the machine to be used.

Item	Requirement
Host machine	PC-9801, 9821 or IBM-PC/AT Series
CPU	i80386 or above (i80486, 33 MHz or above recommended)
Main memory	4M bytes or more (8M bytes or more recommended)
OS	Windows 3.1 or later
Screen size	640 x 400 dots or larger (800 x 600 dots or larger recommended)

«Configuration»

Chapter 1 Overview

Explains the input conventions for the character set and file names used with the integrated debugger.

Chapter 2 Starting and Exiting the Debugger

Explains how to install, start, and exit the integrated debugger.

Chapter 3 Explanation of Terms

Describes the terms used in the explanation of the integrated debugger.

Chapter 4 Functions of the Debug Windows

Explains the basic operating instructions for the debug windows.

Chapter 5 Debug Windows

Explains the debug windows of the integrated debugger.

Chapter 6 Explanation of Debugger Functions

Explains the functions of the integrated debugger in detail.

«Conventions»

The following explains the conventions used throughout this manual.

[ ]	:	Indicates a key to be pressed.
+	:	Indicates keys which must be pressed at the same time.
" "	:	Indicates a character string.
' '	:	Indicates a character.
[ ]	:	Indicates an optional parameter.
[GRPH] key	:	Representation of a key featured by the PC-9801 and 9821 Series.
		The [Alt] key of the IBM-PC/AT Series has the same function.

All representations of keys in this manual are for the PC-9801 and 9821 Series. When using an IBM-PC/AT Series computer as a host machine, see **Appendix B**.

«Cautions»

- To perform source debugging, add options for creating debug information whenever compiling, assembly, or linking is performed. Otherwise, source debugging may not be possible.

- When creating your own startup routine in C, add the symbols given below. Failing to do so may result in part of the step execution not being performed correctly.

Where to add	Symbol to be added
Start of startup routine	_@cstart
End of startup routine	_@cend

«Related Documents»

The documents (user's manuals) related to this manual are listed below:

Document name	Document number
RA78K Series Assembler Package, Language	EEU-1404
RA78K Series Assembler Package, Operation	EEU-1399
RA78K Series Structured Assembler Preprocessor	EEU-1402
CC78K Series C Compiler, Language	EEU-1284
CC78K Series C Compiler, Operation	EEU-1280

Contents

List of Figures

List of Tables

Chapter 1 Overview

1.1 Overview of Debugger

To run ID78K3, the host machine (PC-9801, 9821, or IBM-PC/AT running Windows) and the in-circuit emulator IE-784000-R must be connected using a dedicated parallel interface board (IE-70000-98-IF-A, IE-70000-98-IF-B, or IE-70000-PC-IF-B).

1.2 Overview of Functions

The functions and features of ID78K3 are explained below.

(1) GUI

ID78K3 runs in the Windows environment, allowing the user to debug programs simply by manipulating the mouse. Buttons and menus are displayed in each window, allowing easy switching of the currently displayed information to other related information.

(2) Debugging at the source level

Operations such as the following can be performed effectively by specifying function names and line numbers at the source text level: referencing/setting of variables and structures, display of programs, and setting of breakpoints.

(3) Debugging at the instruction level

Operations such as the following can be performed effectively by specifying labels and addresses at the instruction level: referencing/setting of symbols and register values, display of programs, and setting of break points.

(4) The in-circuit emulator functions are available.

The fine event setting functions of the in-circuit emulator can be used when setting breakpoints and tracing programs.

(5) Monitoring (automatic display update at execution stop)

The values in the currently displayed windows (view windows and view/setting windows) are automatically updated when execution of a user program terminates.

(6) Saving and restoring the debugging environment

The state existing at any point during debugging can be saved. If necessary, that saved state can subsequently be restored.

(7) Display of source text from a function

Upon selecting a function from a function list, the source text containing that function is displayed.

1.3 Input Conventions for the Integrated Debugger

Character set

The following character set can be used with the integrated debugger:

- **Alphabet** **Upper case**: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
  Lower case: a b c d e f g h i j k l m n o p q r s t u v w x y z
- **Digits**: 0 1 2 3 4 5 6 7 8 9
- **Other characters**: @ ? _
- **Special characters**: . , : ; * / + - ' < > ( ) $ = ! # [ ]

Character	Name	Main use
.	Period	Bit position specifier
,	Comma	Operand delimiter
:	Colon	Label delimiter
;	Semicolon	Comment start symbol
*	Asterisk	Multiplication operator
/	Slash	Division operator
+	Plus	Addition operator
-	Minus	Sign of inequality or operator for subtraction
'	Quotation	Character constant or string start/end symbol
<	Less-than sign	Relational operator
>	Greater-than sign	Relational operator
(	Opening parenthesis	Used to change the order in which operations are performed.
)	Closing parenthesis	Used to change the order in which operations are performed.
$	Dollar sign	Relative addressing start symbol
=	Equal sign	Relational operator
!	Exclamation mark	Absolute addressing start symbol
#	Sharp	Indicates an immediate value.
[	Opening bracket	Indirect display symbol
]	Closing bracket	Indirect display symbol
↵	Carriage return	Only one ↵ before each LF is permitted (0DH).

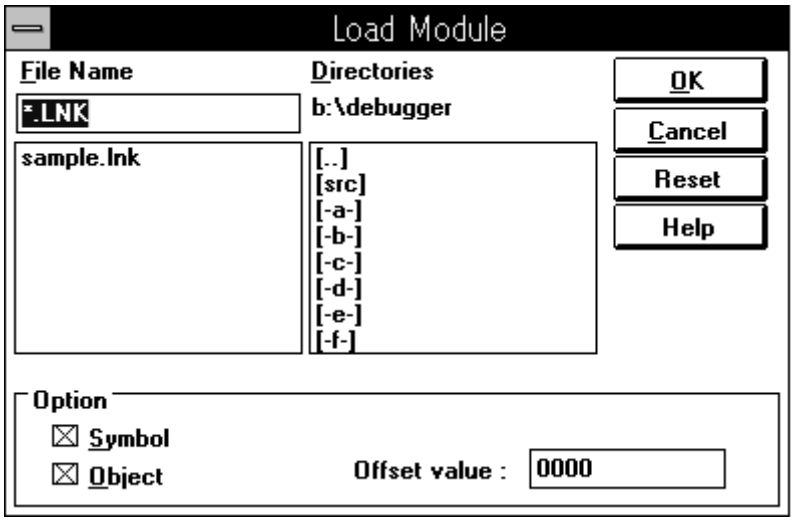
File specification

A file is specified in the format shown below:

File Name: **primary-name [.file-type]]**

Directories: **[drive-name:][[\directory-name]...]**

primary-name:	String of up to eight characters
file-type:	String of up to three characters
drive-name:	One character only
directory-name:	Has the same format as a file name.



Wild cards

- * and ?, in a path name and file name, are handled as wild cards.
- * is replaced by a string of any characters.
- ? is replaced by any one character. (In this case, a blank is also considered as being one character.)
- When a wild card is specified, all corresponding directory and file names in the directory are displayed.
- When a specific file name is to be specified, the use of wild cards results in an error.

Example: For the directory containing the eight files listed below, wild cards can be specified as shown, resulting in the display of the file names in the right-hand column.

AAAAA.HEX, ABC.C ABC.HEX, ABC.SYM, ABCDEFGH.HEX, AXYZ, BCDEFG.HEX, XYZ

Specification with wild card(s)	Corresponding file name(s)
A*.*	AAAAA.HEX, ABC.C, ABC.HEX, ABC.SYM, ABCDEFGH.HEX, AXYZ
A*	AXYZ
A*.HEX	AAAAA.HEX, ABC.HEX, ABCDEFGH.HEX
*.HEX	AAAAA.HEX, ABC.HEX, ABCDEFGH.HEX, BCDEFG.HEX
A??.HEX	ABC.HEX
A??.*	ABC.C ABC.HEX, ABC.SYM
???	XYZ
???.	XYZ
ABC.?	ABC.C
ABC.???	ABC.C ABC.HEX, ABC.SYM

Operands

There are five types of operands:

Numeric value
Address
Register
Symbol
Expression and operator

Numeric value

Four types of numeric values are supported:

- Binary
  Input formats nY(Note 2)
  n...nY(Note 2) (where n=0, 1)

- Octal
  Input formats nO(Note 2)
  n...nO(Note 2) (where n=0, 1, 2, 3, 4, 5, 6, 7)

- Decimal
  Input formats n
  n...n
  nT(Note 2)
  n...nT(Note 2) (where n=0, 1, 2, 3, 4, 5, 6, 7, 8, 9)

- Hexadecimal
  Input formats nH(Note 2)(Note 1)
  n...nH(Note 2)(Note 1)
  0xn(Note 2)
  0xn...n(Note 2) (where n=0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F(Note 2))

Notes 1. If a number begins with A through F, it must be preceded by 0.
Example: FFH → 0FFH

2. Suffixes (Y, O, T, H, 0x) and the alphabetic characters in a hexadecimal number can be either upper or lower case.

Address

- An address can be specified simply by specifying a numeric value.
- An address can also be specified by using a symbol or expression.
- When specifying an address by using a numeric value, hexadecimal, decimal, octal, or binary numbers can be used.

Register

- A general-purpose register has two names: Absolute and functional.
- Each bit of PSW is assigned a name.
- The following register types are supported:

Register type	Register name
Control register	PC
	SP
	PSW

Register type	Register name
PSW	UF
	RBS2
	RBS1
	RBS0
	S
	Z
	RSS
	AC
	IE
	P/V
	CY

Register type	Register name		
	Absolute name	Functional name	
		If RSS=0	If RSS=1
General-purpose register	R0	X	
	R1	A	
	R2	C	
	R3	B	
	R4		X
	R5		A
	R6		B
	R7		C
	R8		
	R9		
	R10		
	R11		
	R12	E	E
	R13	D	D
	R14	L	L
	R15	H	H
	RP0	AX	
	RP1	BC	
	RP2		AX
	RP3		BC
	RP4	VP	VP
	RP5	UP	UP
	RP6	DE	DE
	RP7	HL	HL
	RG4	VVP	VVP
	RG5	UUP	UUP
	RG6	TDE	TDE
	RG7	WHL	WHL

Symbol

- A symbol consists of any of A-Z, a-z, @, ?, _ (underscore), and 0-9.
- A symbol must begin with a character other than a digit (0-9).
- Symbols are case-sensitive (upper case A-Z and lower case a-z characters are regarded as being different.)
- A symbol can consist of up to 31 characters.
- If a symbol consists of more than 31 characters, only the first 31 characters are effective.
- A symbol is defined by loading a load module file.

- The following types of symbols are supported. Each has its own application range:
  Public symbol (assembler, structured assembler, and C)
  Local symbol
    In-module local symbol (assembler and structured assembler)
    In-file local symbol (C)
    In-function local symbol (C)

- The following symbols are supported for the corresponding language:
  Assembler/structured assembler:
    Label name, constant name, and bit symbol name
  C:
    Variable name (including point variable name, enumeration variable name, array name, structure name, and union name)
    Function name and label name
    Array element, structure element, union element, and bit field (if the symbol is an array, structure, or union)

- If a variable name in C is the same as a register name, flag name, SFR name, or SFR bit name, it must be explicitly distinguished by being prefixed with "_".

- A symbol can be specified instead of an address and a numeric value.

- The application range of a symbol is based on the source debugging information obtained at assemble or compile time.

- A public symbol is described by a symbol name only.

- A local symbol is represented by a pair of a file and module name.

Expression and operator

Expression

- An expression is a combination of constants, register names, SFR names, and symbols, joined by operators.

- When an SFR, label, function, or variable name is used as a symbol, an address is operated upon as the value of the symbol.

- Elements other than the operators that make up an expression are called terms (constants or labels). They are called the primary term, secondary term, and so on, starting from the left.

Operator

The following types of operators are supported:

Arithmetic operators

Symbol	Meaning	Explanation
+	Addition	Returns the sum of the values of the primary and secondary terms.
-	Subtraction	Returns the difference between the values of the primary and secondary terms.
*	Multiplication	Returns the product of the values of the primary and secondary terms.
/	Division	Divides the value of the primary term by the value of the secondary term and returns the integer part of the result.
MOD	Remainder	Divides the value of the primary term by the value of the secondary term and returns the remainder of the result.
- sign	Unary operation (negative)	Returns the 2's complement of the value of a term.
+ sign	Unary operation (positive)	Returns the 2's complement of the value of a term.

Logical operators

Symbol	Meaning	Explanation
NOT	Negation	Takes the logical NOT of each bit of a term and returns it.
AND	Logical product	Takes the logical product of each of the bits of the primary and secondary terms and returns it.
OR	Logical sum	Takes the logical sum of each of the bits of the primary and secondary terms and returns it.
XOR	Exclusive OR	Takes the exclusive OR of each of the bits of the primary and secondary terms and returns it.

Others

Symbol	Meaning	Explanation
(	Opening parenthesis	Operations within parentheses () are performed prior to those outside parentheses.
)	Closing parenthesis	

Cautions:

- (and) are always used as a pair.
- In a comparison operation, character strings can be given in terms.
- Operations are performed according to the following rules:
  The order in which operations are performed conforms to the operator precedence.
  Where the operators have equal precedence, operations are performed in order, from left to right.
  Operations enclosed in parentheses are performed prior to those outside the parentheses.
  Each term to be operated upon is handled as unsigned 32-bit data.
  All operation results are handled as unsigned 32-bit data.
  If an overflow occurs during an operation, the lower 32 bits are regarded as being valid; the overflow is not detected.

- The operator precedence is as follows:

Highest	↑	(,)
		- sign, NOT
		*, /, MOD
		+, -
		AND
Lowest	↓	OR, XOR

Term

When describing a constant in a term, the following numeric values can be used.

- Binary number
  0Y ≤ Numeric value ≤ 11111111111111111111111111111111Y (32 digits)

- Octal number
  0O ≤ Numeric value ≤ 37777777777O

- Decimal number
  -2147483648 ≤ Numeric value ≤ 4294967295
  A negative decimal number is internally converted to a 2's complement.

- Hexadecimal number
  0H ≤ Numeric value ≤ 0FFFFFFFFH

Chapter 2 Starting and Exiting the Debugger

This chapter explains how to install, start, and exit ID78K3.

2.1 Installation

2.1.1 Connecting the Devices

This section explains how to connect the devices.

(1) Confirming the environment

The following environment must have been established prior to starting the integrated debugger (referred to simply as the **debugger** throughout the remainder of this manual).

	PC-9801 and 9821 Series	IBM-PC/AT Series
Host machine	CPU: i80386 or above; memory: 4M bytes or more (Recommendation: CPU: i80486, 33 MHz or above; Memory: 8M bytes or more)	
OS	Windows Ver. 3.1 or later (386 enhanced mode)	
Interface board	For an expansion slot (C bus): IE-70000-98-IF-A, IE-70000-98-IF-B For 98NOTE (110-pin expansion bus): IE-70000-98N-IF	(ISA bus) IE-70000-PC-IF-B
In-circuit emulator	IE-784000-R IE-78350-R-EM-A IE-783xx-R-EM1 (emulation board) EP-78xxx-R (emulation probe)	

(2) Setting and connecting the devices

a. PC-9801 and 9821 (for an expansion slot)

①Turn off all the devices.

②Set the dip switches and jumpers of the PC interface board (IE-70000-98-IF-A or IE-70000-98-IF-B) as shown in Figure 2-1.

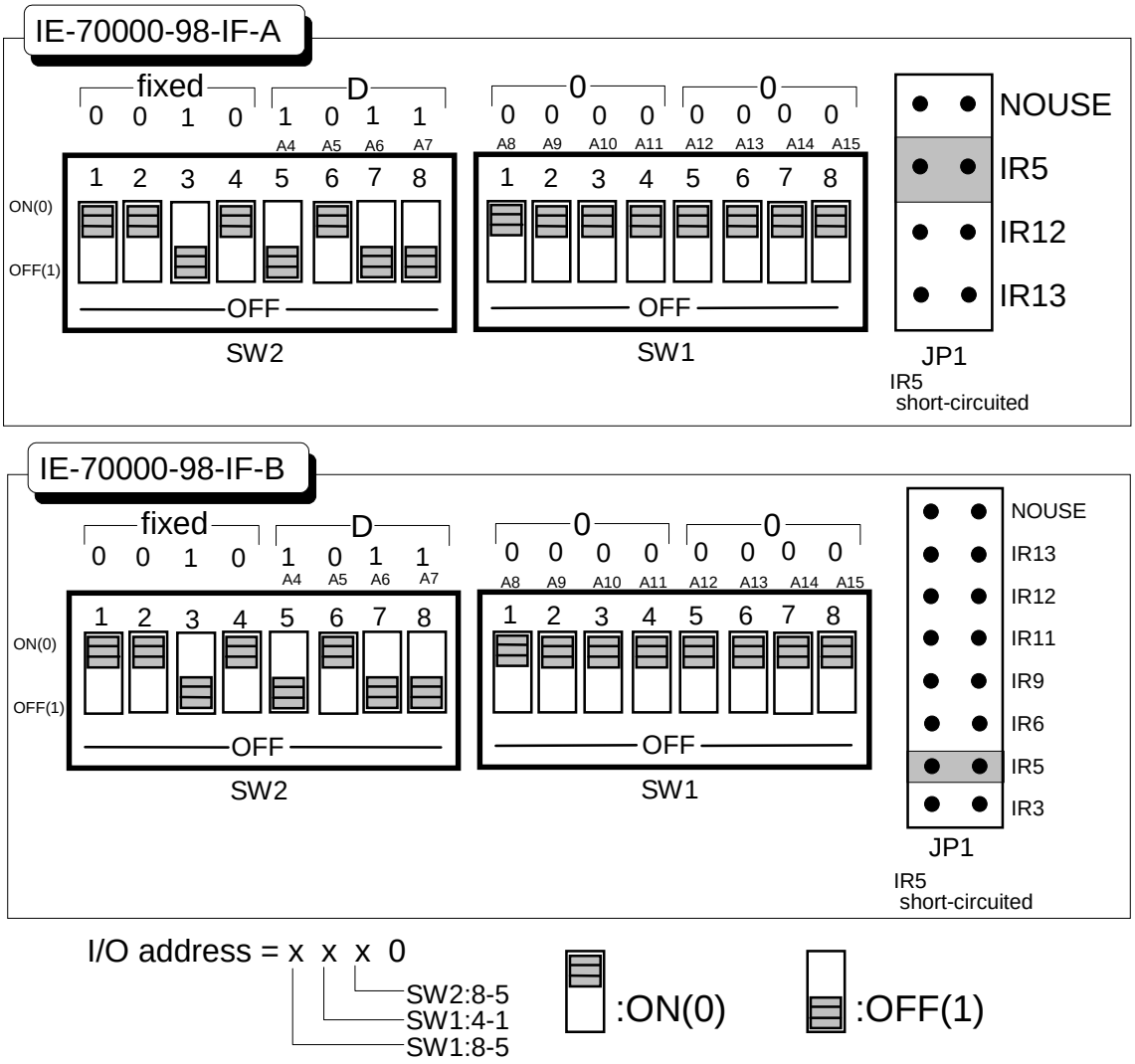


Fig. 2-1 Dip Switch Settings (PC-9801 and 9821)

Jumper set point	Vector number	IE-70000-98-IF-A	IE-70000-98-IF-B
IR3 (INT0)	0x0B	Not applied	Applied
IR5 (INT1)	0x0D	Applied	Applied
IR6 (INT2)	0x0E	Not applied	Applied
IR9 (INT3)	0x11	Not applied	Applied
IR11 (INT42)	0x13	Not applied	Applied
IR12 (INT5)	0x14	Applied	Applied
IR13 (INT6)	0x15	Applied	Applied

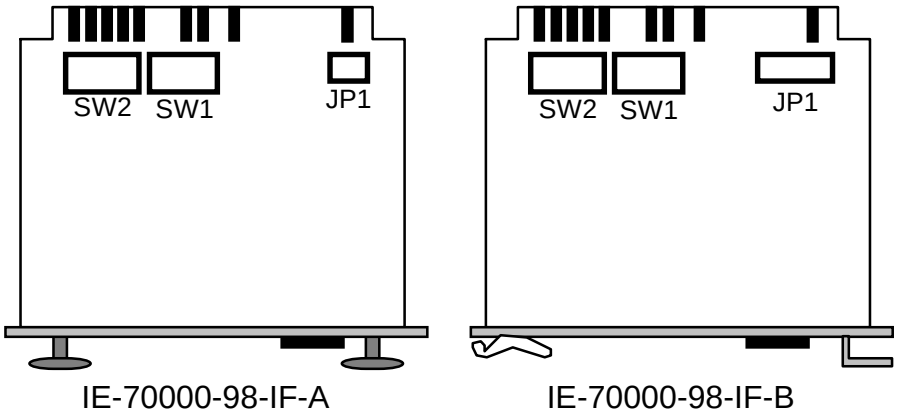


Fig. 2-2 PC Interface Boards (PC-9801 and 9821 Expansion Slots)

Note: These settings correspond to the following I/O address and interrupt:
00DxH (addresses 00D0 through 00DFH are used)
IR5 (INT1)
If these I/O address and interrupt are already being used for another device, the I/O address for the board must be set to an address between 000x and 0FFFxH (the lower four bits cannot be set) and the interrupt to one of IR3, IR5, IR6, IR9, IR11, IR12, or IR13, such that they do not cause a conflict with the other device.
Note that if the settings of the PC interface board are changed, the settings of the EXPC.INI file must also be changed.

③Mount the PC interface board in an expansion slot of the PC-9801 or 9821.
④Connect the parallel interface port of the PC interface board mounted in the PC-9801 to CH3 of the in-circuit emulator, using the parallel interface cable supplied with the in-circuit emulator.
⑤When using a target, connect the emulation probe to the in-circuit emulator and connect the other end of the probe to the target.
⑥This completes the setting and connection of the devices.

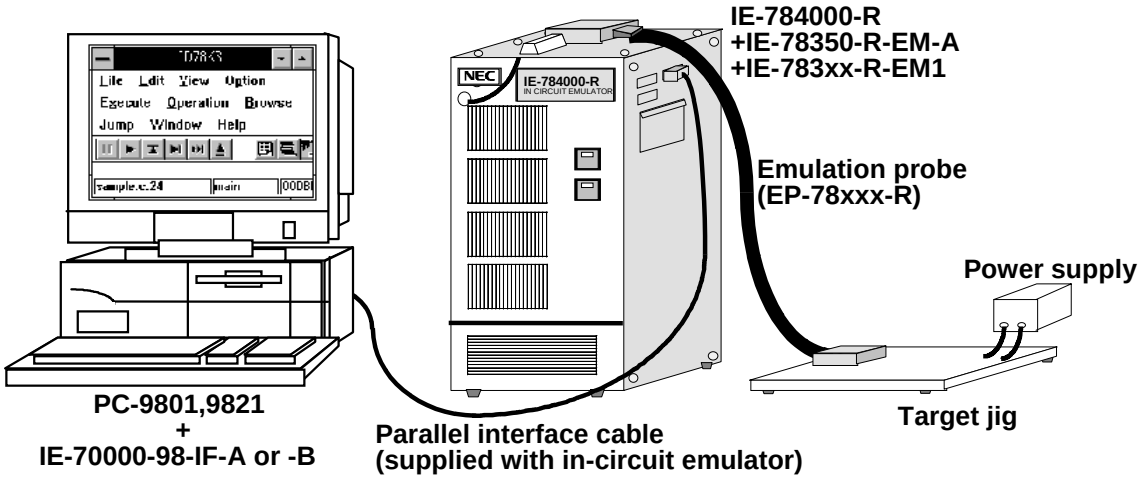


Fig. 2-3 Connection Diagram (PC-9801 and 9821 Expansion Slots)

b. PC-9801 and 9821 (for 98NOTE)

①Turn off all the devices. (Set the resume function of the 98NOTE to OFF.)

②Attach the connecting cable, supplied with the PC interface board (IE-70000-98N-IF), referring to the instruction manual for the PC interface board. For this PC interface board:

1. The interrupt and I/O address are fixed and cannot be changed.
2. The extension bus of the 98NOTE must have a 110-pin connector. The interrupt and I/O address are as follows:

Interrupt and vector Nos.	INT0, 0x0B
I/O address	0x00D0

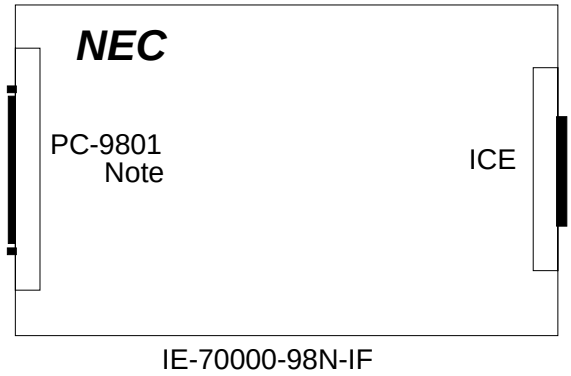


Fig. 2-4 PC Interface Board (98NOTE)

③Connect the parallel interface port of the PC interface board, connected to the 98NOTE, to CH3 of the in-circuit emulator, using the parallel interface cable supplied with the in-circuit emulator.

④When using a target, connect the emulation probe to the in-circuit emulator and connect the other end of the probe to the target.

⑤This completes the setting and connection of the devices.

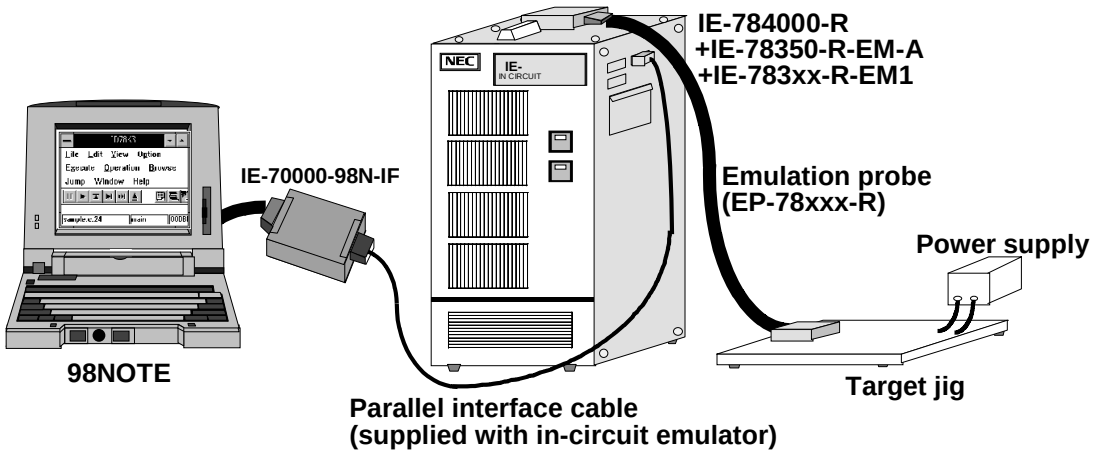


Fig. 2-5 Connection Diagram (98NOTE)

c. IBM-PC/AT

①Turn off all the devices.

②Set the dip switches and jumpers of the PC interface board (IE-70000-PC-IF-B), as shown below.

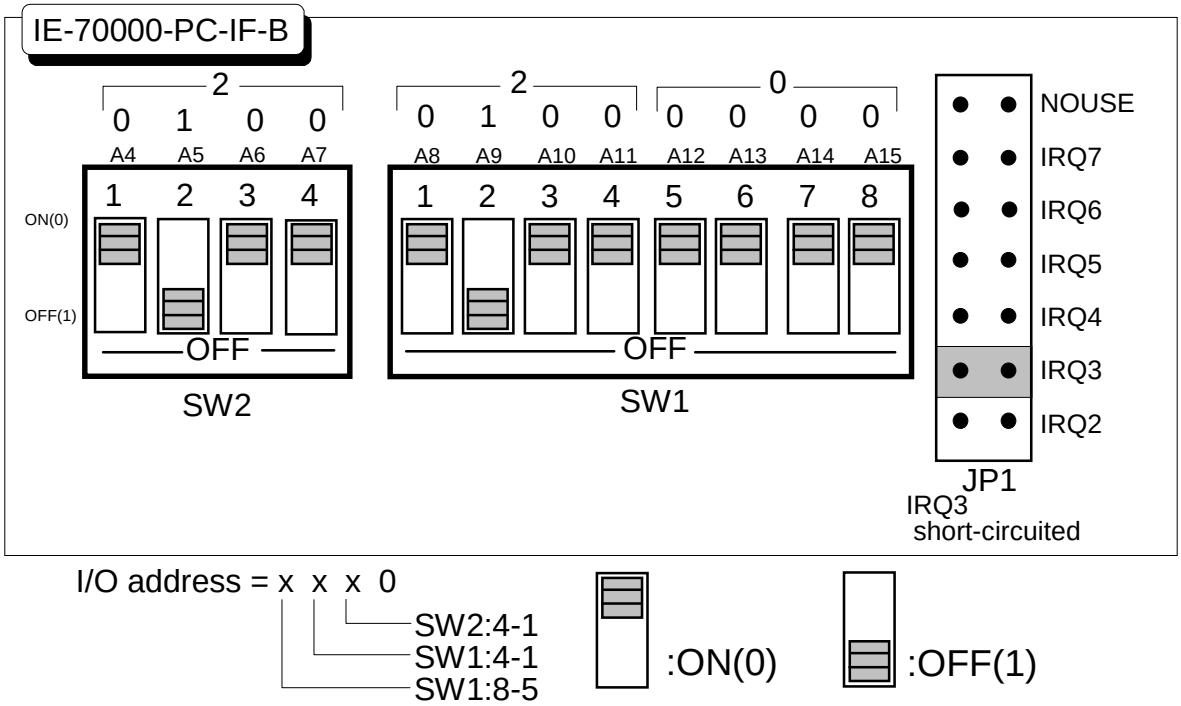


Fig. 2-6 Dip Switch Settings (for IBM-PC/AT)

Jumper set point	Vector number
IRQ2	0x0A
IRQ3	0x0B
IRQ4	0x0C
IRQ5	0x0D
IRQ6	0x0E
IRQ7	0x0F

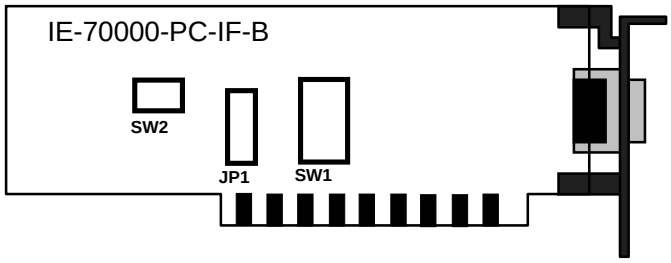


Fig. 2-7 PC Interface Board (for IBM-PC/AT)

Note: These settings correspond to the following I/O address and interrupt:
022xH (addresses 0220 through 022FH are used)
IRQ3
If these I/O address and interrupt are already being used for another interface, the I/O address for the board must be set to an address between 000x and 0FFFxH (the lower four bits cannot be set) and the interrupt to one of IRQ2, IRQ3, IRQ4, IRQ5, IRQ6, or IRQ7, such that they do not cause a conflict with the other interface.
If the settings of the PC interface board are changed, the settings of the EXPC.INI file must also be changed.

③Mount the interface board in an expansion slot of the IBM-PC/AT.
④Connect the parallel interface port of the interface board mounted in the IBM-PC/AT to CH3 of the in-circuit emulator, using the parallel interface cable supplied with the in-circuit emulator.
⑤When using a target, connect the emulation probe to the in-circuit emulator and connect the other end of the probe to the target.
⑥This completes the setting and connection of the devices.

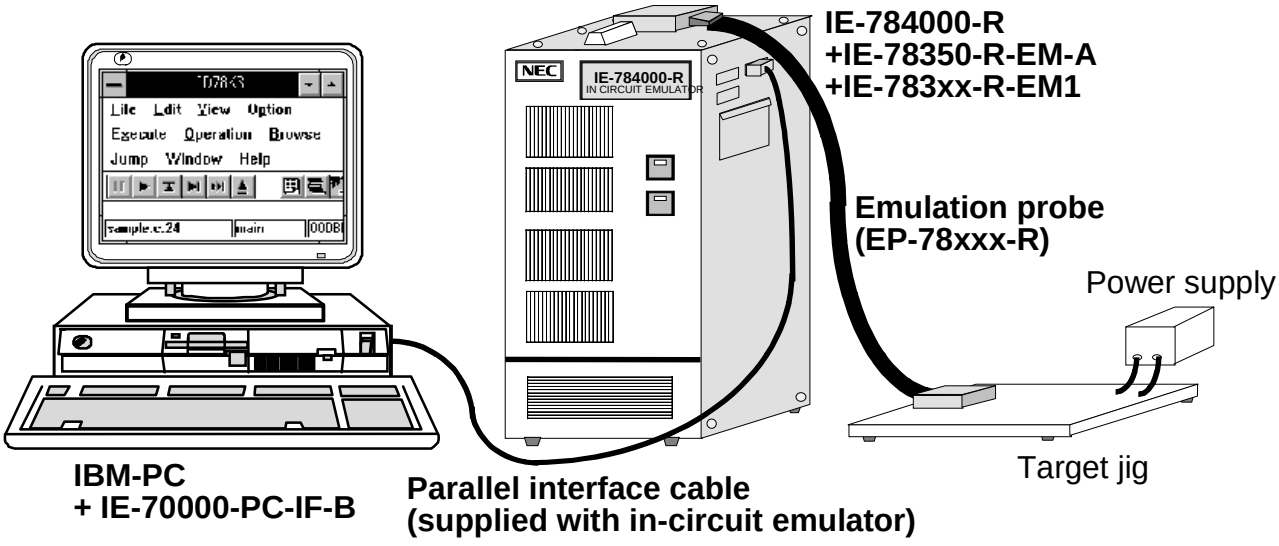


Fig. 2-8 Connection Diagram (IBM-PC/AT)

2.1.2 Installing the Debugger

This section explains how to install the debugger. (The explanation is common to the PC-9801, 9821, and IBM-PC/AT.)

(1) Confirming the environment and the files

The following lists the files used with the debugger. No device files are supplied with the debugger. Therefore, the necessary device file(s) must be purchased separately.

Table 2-1 Debugger Files

File name	Explanation
ID78K3.EXE	Debugger main section. The debugger is started by executing this file.
DB78K3.DLL	Contains libraries for file and symbol processing.
AS78K3.DLL	Contains libraries for assembly and disassembly.
EX78K3.DLL	Contains libraries for communication with the in-circuit emulator.
EX78K3.OM0	Downloaded into the in-circuit emulator when the debugger starts.
ID78K3.HLP	Help file.
EXPC.INI	Initial file. Used to specify a set point and an interrupt address for the PC interface board.
D3xxx.78K (device file) (sold separately)	Contains device-specific information. A device file corresponding to the target device is required.

(2) Installing the debugger

The following explains the procedure for installing the debugger. The explanation assumes that the debugger will be used from a fixed disk. Following the procedure will result in the debugger being installed in the \DEBUGGER directory on drive B of the fixed disk. It is assumed that Windows has already been installed in the \WINDOWS directory on drive B.

①Turn on the host machine (PC-9801, 9821, or IBM-PC/AT). Start MS-DOS from the fixed disk.

②Wait for the MS-DOS prompt to appear.

③Insert the debugger system disk into the floppy disk drive (C drive).

④Enter the following:

MD DEBUGGER↵

CD DEBUGGER↵

COPY C:*.* /V↵

This copies the contents of the system disk to the fixed disk.

⑤If the settings of the PC interface board have been changed, use the editor to change the EXPC.INI file as follows. If no changes have been made, go to ⑥.

The initial settings are as shown below. Change the **underlined** portions as necessary.

PC-9801 and 9821

```
[PCIF]
HOST=0                  ; PC-9801
PC_BASE=0x00D0          ; I/O base address
IE_INT=0x0D             ; interrupt vector
```

98NOTE (Only the following settings are possible)

```
[PCIF]
HOST=0                  ; PC-9801
PC_BASE=0x00D0          ; I/O base address
IE_INT=0x0B             ; interrupt vector
```

IBM-PC/AT

```
[PCIF]
HOST=1                  ; IBM-PC/AT
PC_BASE=0x0220          ; I/O base address
IE_INT=0x0B             ; interrupt vector
```

HOST= This switch is used to select the host machine.
0 for the PC-9801 or 9821
0 for 98NOTE
1 for the IBM-PC/AT

PC_BASE= Must be changed if the dip switch settings are changed.
This item must be set to exactly correspond to the dip switch settings.
(The lower four bits are fixed to 0.)

IE_INT= Must be changed if the jumper settings are changed.

PC-9801 and 9821

Jumper set point	IE_INT set point	Remarks
IR3	0x0B	Extended bus INT0
IR5	0x0D	Extended bus INT1
IR6	0x0E	Extended bus INT2
IR9	0x11	Extended bus INT3 (HD)
IR11	0x13	Extended bus INT42 (1M-byte FD)
IR12	0x14	Extended bus INT5 (sound)
IR13	0x15	Extended bus INT6 (mouse)

IBM-PC/AT

Jumper set point	IE_INT set point	Remarks
IRQ2	0x0A	
IRQ3	0x0B	
IRQ4	0x0C	
IRQ5	0x0D	
IRQ6	0x0E	
IRQ7	0x0F	

⑥Remove the system disk from the floppy disk drive (C drive) and insert the device file disk.

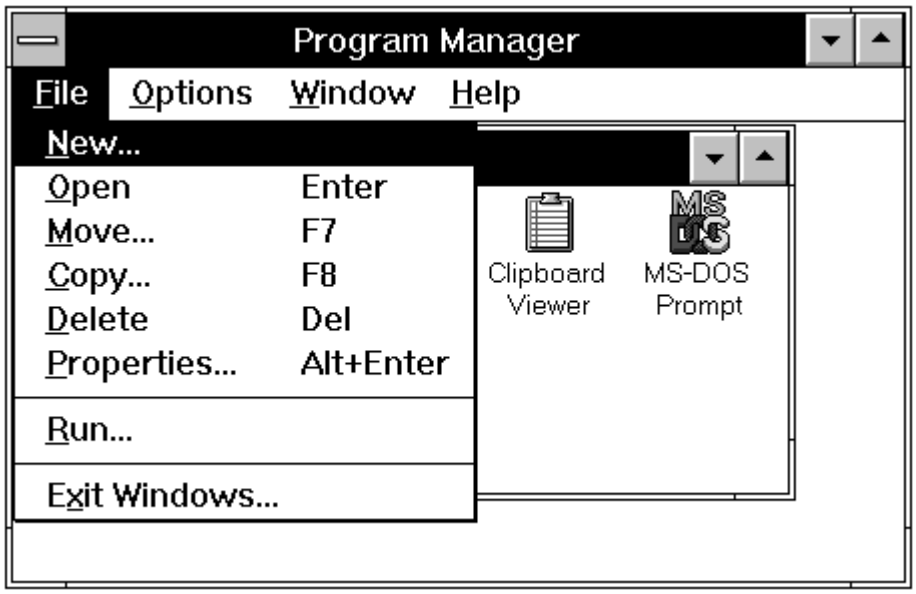
⑦Copy the following files from the device file disk to the B: \DEBUGGER directory, in which the debugger has been installed.

File name	Contents
D3xxx.78K	Device file for the 78K3 Series

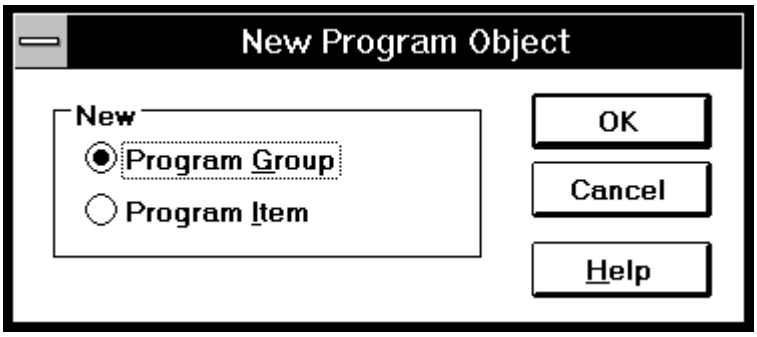
This copies the contents of the device file disk to the fixed disk.

⑧Start Windows.

⑨Select **File** from the Program Manager.

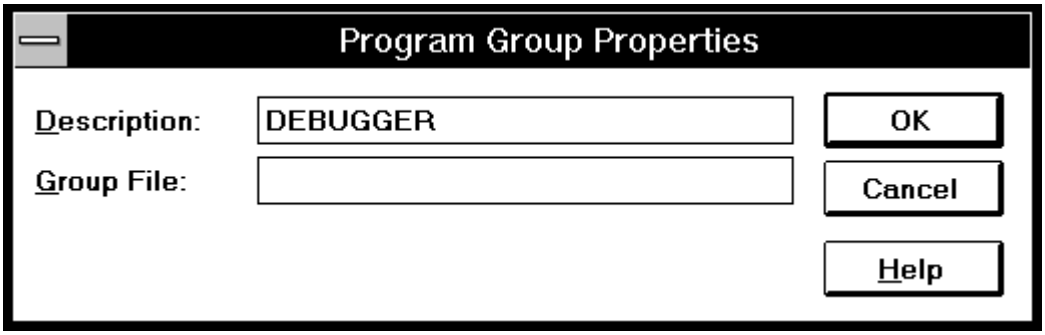


⑩Select **New...** from the pull-down menu.



⑪Select **Program Group** then click the OK button.

⑫Enter group name "DEBUGGER" in the **Description** field, then click the OK button.



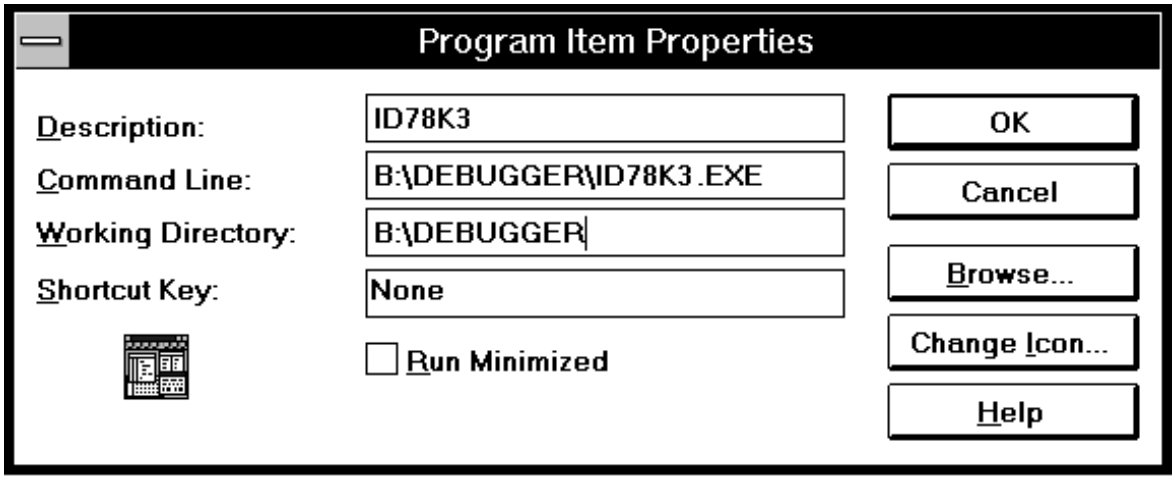
The group is now registered.

⑬Select **File→New** from the Program Manager.

⑭Select **Program Item**, then click the OK button.

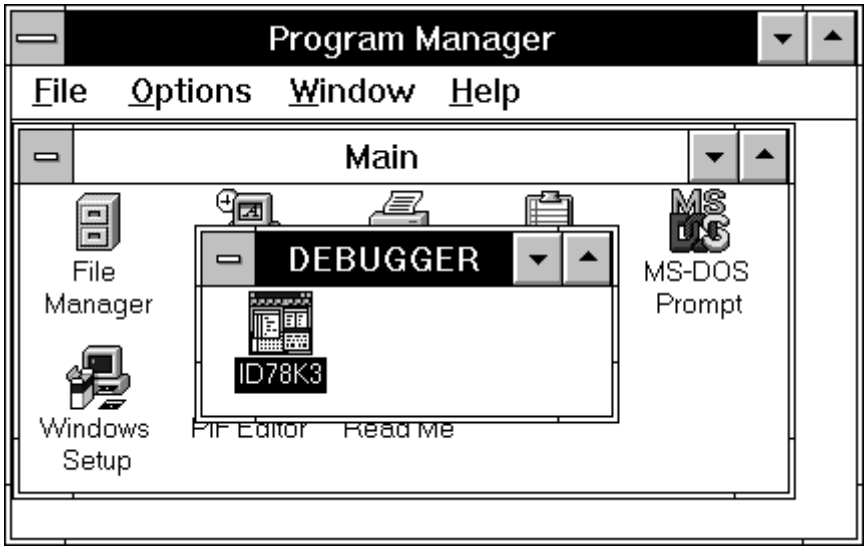
⑮ Enter the following, then click the OK button.

Description: **ID78K3**
Command Line: **B:\DEBUGGER\ID78K3.EXE**
Working Directory: **B:\DEBUGGER**



The icon is now registered. The debugger is now installed on the fixed disk.

Once the debugger has been installed, the debugger can be started by double-clicking the icon registered as described above.



2.2 Starting and Exiting the Debugger

2.2.1 Starting

①Start Windows.
②Turn on the in-circuit emulator.
③When using a target, turn on the target.
④Double-click the icon registered when the debugger was installed.

⑤When the debugger starts, the configuration dialog box opens first.

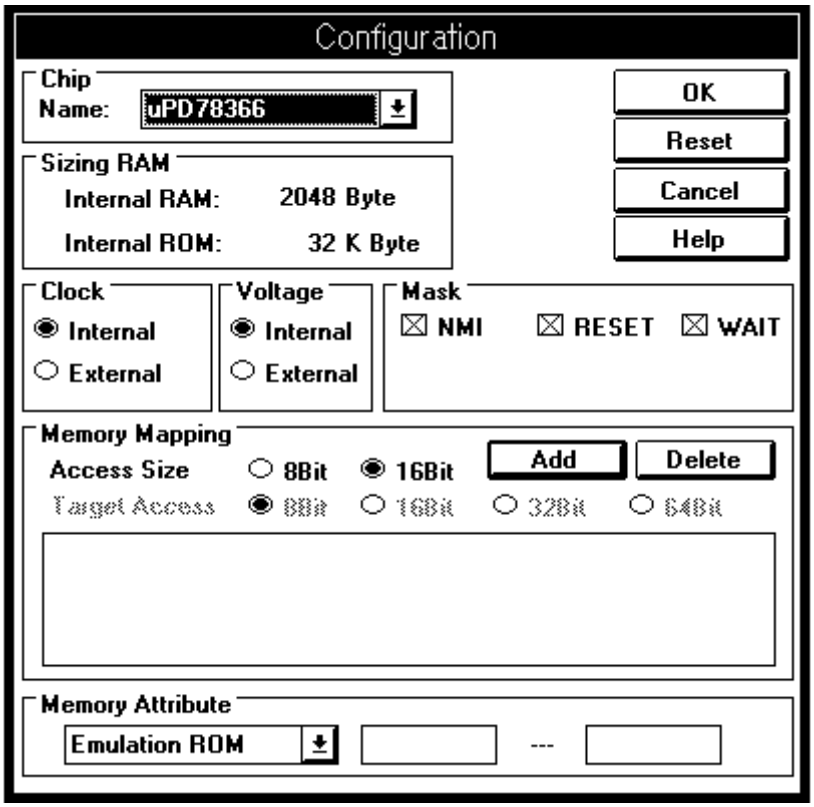
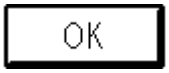


Fig. 2-9 Configuration Dialog Box Displayed when Debugger First Starts

⑥Select a debug target device.
(The debug target device can be selected only when the debugger first starts.)

⑦Set the clock source, pin mask, location, and memory mapping.

⑧Once all settings have been completed, click the 

button. This completes device initialization and downloads the necessary data into the in-circuit emulator.

⑨Once downloading has been completed, the main window of the debugger opens. This window is used the most during debugging.

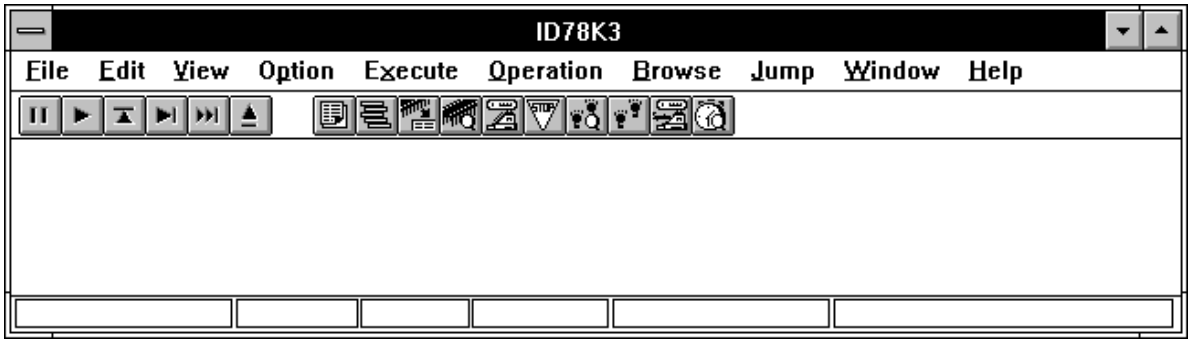


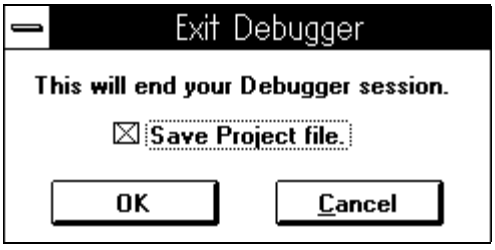
Fig. 2-10 Debugger Startup Screen

2.2.2 Exiting

①Select **File** from the **menu bar** of the main window.

②Select **Exit** from the **File** pull-down menu.

③The Exit Debugger dialog box opens.



④Click the 

button to exit the debugger.

Chapter 3 Explanation of Terms

This chapter describes the terms used in explaining how to use ID78K3.

(1) Debug mode

(2) File

(3) Current file

(4) Function

(5) Current function

(6) Structure

(7) Stack frame number

(8) Line

(9) Real-time RAM sampling

3.1 Debug Mode

When the window interface is being used, two debug modes are supported: Source mode and instruction mode.

- In source mode
  Step execution is performed in units of source text lines.

- In instruction mode
  Step execution is performed in units of instructions.

The debug mode is switched from the main window. The debugger is placed in source mode when it first starts.

3.2 File

ID78K3 supports the following file types:

- Source files
- Load module files
- Hexadecimal files
- Debugging environment files
- View files

3.3 Current File

The current file is the source file containing the instruction pointed to by the program counter (PC). When lines and functions in the current file are specified by commands, the name of the file can be omitted.

The file specification format is as follows:

a. path\file b. file

(path: Path name file: File name)

- For a (a path is specified)
  The file is written and read to and from the directory in the path.

- For b (no path is specified)
  The file is written and read to and from the current directory.

3.4 Function

A function refers to those functions constituting a C source program.
The function view and specification format is:

a. file#_func b. _func

(file: File name func: Function name)

- For a (a file is specified)
  func is interpreted as a static function that is effective within the specified file.

- For b (no file is specified)
  A corresponding function is retrieved from the current file in the order of effective static functions and global functions.

Function specification examples

test.c#_calc_data	Static function "calc_data" in the "test.c" file
_main	"main" which can be retrieved from the current file

3.5 Current Function

The current function is that function which contains the instruction pointed to by the program counter (PC). When a local variable in the current function is to be accessed, the specification of the function name can be omitted.

3.6 Structure

In C, structures and unions are generically referred to as structures. The term structure is used when a structure or a union variable is used without explicitly specifying a member.

3.7 Stack Frame Number

A stack frame number is a decimal integer beginning from 1. It indicates a level in a stack that corresponds to a certain function. The function having the largest stack frame number is the current function.

3.8 Line

A line is specified to identify a specific line in a source file.
The line view and specification format is:

file:line

(file: File name line: Line number)

This identifies the line-th line in the specified file.

Line specification example

test.c:100	100th line in the "test.c" file

3.9 Real-Time RAM Sampling

When it displays the variables assigned to a space, the contents of which can be read, ID78K3 reads the contents in real time to update the display even during the execution of a user program. This capability is called real-time RAM sampling. This memory address space is called a real-time RAM space.

Real-time RAM space	0xfe00 - 0x00feff

Chapter 4 Functions of the Debug Windows

4.1 Basic Window Operations

The basic operations in a debugging process using the window interface are specified as noun + verb. In other words, the user first selects the object to be debugged (a variable, line, or task) then selects the desired debug function by clicking the corresponding function button.

A window provides a menu containing the same functions as the function buttons, allowing the user to carry out debugging by using the keyboard shortcut keys.

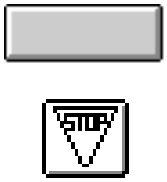
The following explains the objects that the user can manipulate when using ID78K3.

(1) Mouse

Mouse operations with the integrated debugger are mainly instigated with the left mouse button. In the following explanations, references to a mouse button refer to the left mouse button, unless otherwise specified. There are three basic mouse operations:

Click:	Press the mouse button once then release it.
Double-click:	Press the mouse button twice, in rapid succession, then release it.
Drag & drop:	Drag refers to repositioning an object by pressing and holding down the mouse button, then moving the mouse. Drop refers to releasing the mouse button once an object has been moved to a desired location.

(2) Pushbutton and function button



A pushbutton is a rectangular, sculptured button that contains a bit map (icon) or character string. Clicking a pushbutton starts the corresponding action.

A function button is a button which starts a debug function.

(3) Check box

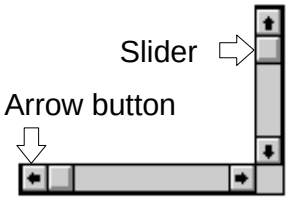
☒ NMI ☐ RESET

A check box consists of two parts: A square box and selectable text. Clicking a check box causes the square box to change from '☐' to '☒'. More than one check box can be selected at any one time.

(4) Radio button

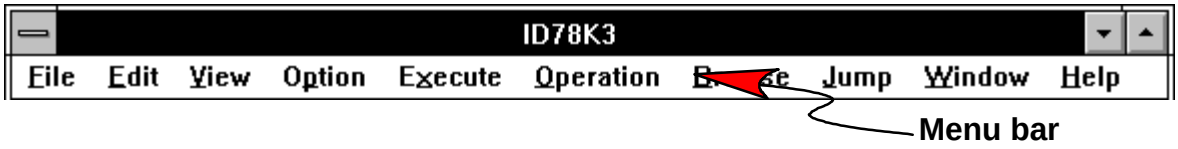
A radio button consists of two parts: A circle and selectable text.
Clicking a radio button causes the circle to change from '○' to '◉'.
Usually, a group of two or more radio buttons are displayed, from which only one can be selected.

(5) Scroll bar



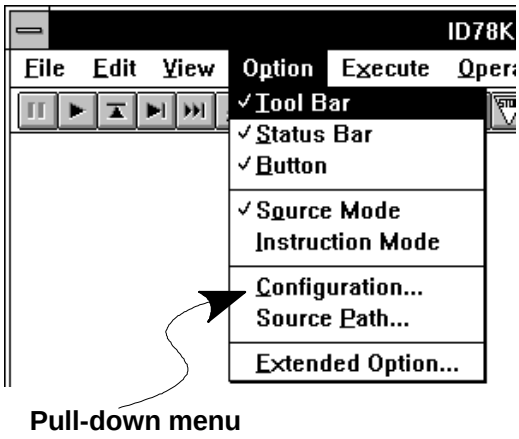
Scroll bars allow the user to scroll up and down on the screen (vertical scroll bar) and to the right and left (horizontal scroll bar).
The slider in a scroll bar indicates the position of the displayed portion relative to the entire scrollable information.
Clicking an arrow button scrolls by one line in that direction.
Dragging and dropping the slider box displays the corresponding part of the information.

(6) Menu bar



A menu bar is displayed at the top of each window.
Clicking an item in a menu bar causes the pull-down menu for that item to be displayed.
The same effect is obtained by pressing the **GRPH** key (PC-9801 and 9821 Series) or **Alt** key (IBM-PC/AT Series) then entering the corresponding underlined alphabetic character.

(7) Pull-down menu



A pull-down menu is an extension of a menu bar. Clicking an item in a menu causes the corresponding action to be activated. The same effect is obtained by entering the corresponding underlined alphabetic character for a desired item.
Some items in a pull-down menu can be activated directly by pressing **CTRL** + alphabetic character without first clicking the menu bar. Such items are indicated by "CTRL+alphabetic character", to the right of the items in the pull-down menus.

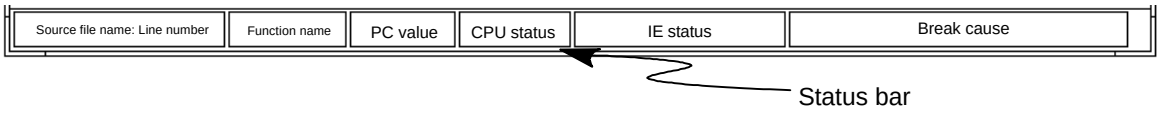
Menu items are indicated as follows, such that the effect of selecting the menu items can easily be recognized.

1. **For "item"**
   When this item is selected, the indicated action is activated.
2. **For "item ..."**
   When this item is selected, a dialog box requiring a user response appears.
3. **For "item ▸"**
   When this item is selected, the corresponding cascade menu appears.

(8) Tool bar

A tool bar contains a group of buttons corresponding to frequently used commands, allowing those commands to be executed simply by clicking the corresponding button. Each button is identified by an easy-to-understand graphical.

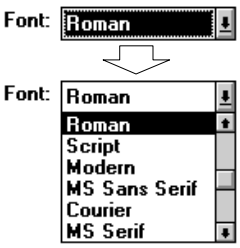
(9) Status bar



A status bar is an area showing the states of the debugger and the in-circuit emulator.
Starting from the left, the status bar indicates the following items:

Source file name and line number pointed to by the PC
Function name pointed to by the PC
PC value
CPU (μPD784xxx) status
In-circuit emulator status
Break cause

(10) Drop-down list



Before clicking the arrow

After clicking the arrow

A drop-down list is a single-item field in which only the currently selected item is displayed. Other items become selectable by clicking the arrow.

4.2 Active State and Hold State

The active state is that state in which the values displayed in a window are automatically updated as a user program or a command is executed. The hold state is that state in which the values are retained irrespective of whether a user program or a command is executed. A window whose contents change as a user program is executed (view window and view/setting window) can be switched between the active state and the hold state.

With ID78K3, only one window of each type can be set to the active state. More than one view window of the same type can be displayed at any one time if placed in the hold state. If view windows of the same type are displayed, one in the active state and the others in the hold state, and an attempt is made to switch a hold-state window to the active state, the message "Other view mode windows exist." appears and the hold-state window is closed. A window in the hold state has a highlighted background, with **[HOLD xx]** displayed in its title bar.

To switch a window between the active and hold states, follow the procedure below.

To switch from the active to the hold state:

1. Select **Operation→Hold** from the **menu bar**.
2. Press the **CTRL** + **H** keys.

To switch from the hold to the active state:

1. Select **Operation→Active** from the **menu bar**.
2. Press the **CTRL** + **I** keys.

4.3 View Mode and Modify Mode

Some windows of ID78K3 are provided with two window modes: View and modify. The windows are:

- Variable window
- Local Variable window
- Memory window
- Register window
- SFR window
- Disassemble window

These windows are usually set to view mode. By switching these windows to modify mode, variable values in a user program and memory contents can be modified during debugging.

The values modified in modify mode are reflected by clicking the Write in button, displayed in modify mode. Clicking the Restore button causes all the values modified in modify mode to be restored to their previous states. However, if the Write in button has already been clicked, the values are restored to those existing immediately after the Write in button was clicked.

Only an active-state window can be switched to modify mode.

To switch a window between view and modify modes, follow the procedure below.

To switch from view mode to modify mode:

1. Select **Operation→ToModify** from the **menu bar**.
2. Press the **CTRL** + **F** keys.
3. Click the ToModify button.

To switch from modify mode to view mode:

1. Select **Operation→ToView** from the **menu bar**.
2. Press the **CTRL** + **W** keys.
3. Click the ToView button.

4.4 Errors/Warnings

ID78K3 handles errors and warnings separately.
All errors are generated from the debugger.

4.4.1 GUI Operational Errors/Warnings

All GUI operational errors are handled as warnings.
If a warning occurs, an alarm is generated or an error/warning dialog box appears.

4.4.2 Errors/Warnings Generated by the Debugger

An error/warning dialog box appears when an error occurs.

Chapter 5 Debug Windows

5.1 Window Types and Configurations

ID78K3 uses several windows and dialog boxes. Dialog boxes are displayed only temporarily to enable a specified operation. Both windows and dialog boxes can be modified by using the Window Manager. In principle, however, while windows can be reduced to an icon, dialog boxes cannot.

5.1.1 Windows

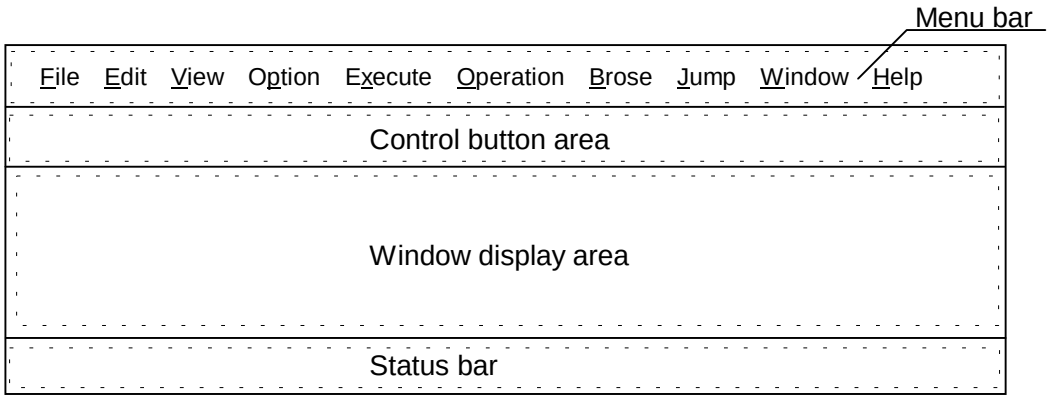
Windows are classified into the following four types according to their function:

- **Execution windows**
- **View windows**
- **View/setting windows**
- **Management windows**

Each of these window types is described below.

(1) Execution windows

Execution windows are used to control windows or program execution. An execution window consists of a menu bar, window display area, status bar, and control buttons.

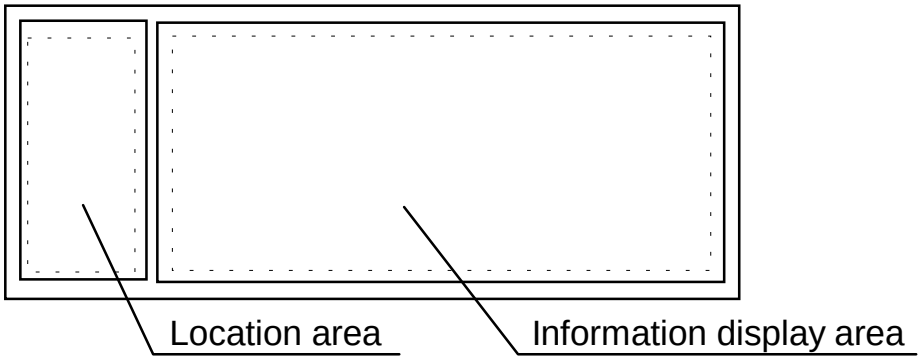


Windows of this type:

Main window

(2) View windows

View windows are used to display the contents of a program or memory related to the target system. View windows display data but do not enable data modification. A view window consists of a location area and information display area.



Windows of this type:

Source window	**Trace View window**
Coverage window	**Stack window**

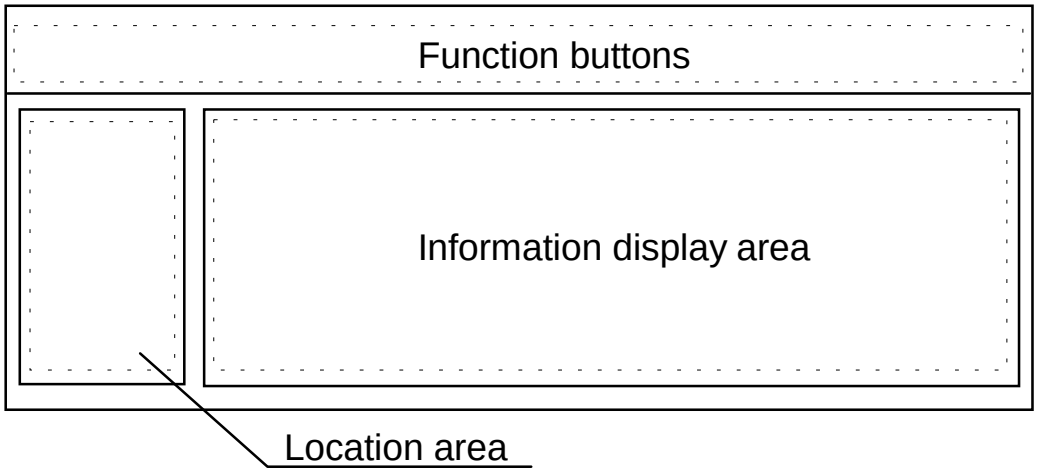
(3) View/setting windows

View/setting windows are used to display or modify the contents of a program or memory related to the target system. View/setting windows usually allow only the display of data. Data modification is possible only in modify mode.

There are two types of view/setting windows: Those opened from within the main window and those opened outside the main window.

a. View/setting window opened from within the main window

This type of window consists of function buttons, a location area, and information display area.

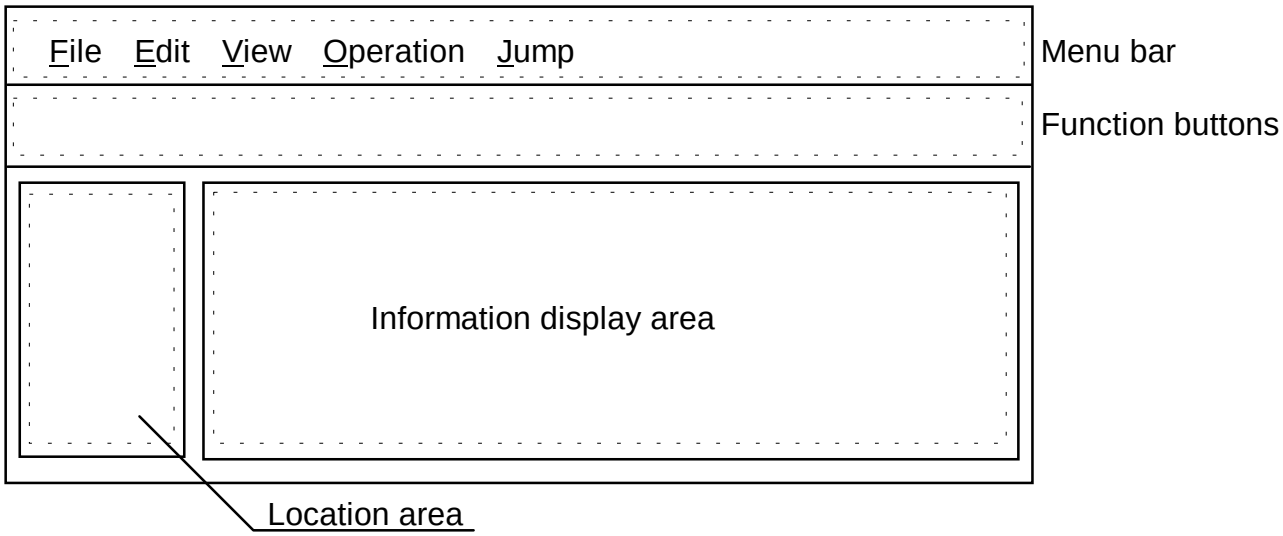


Windows of this type:

Local Variable window	**SFR window**
Memory window	**Assemble window**

b. View/setting window opened outside the main window

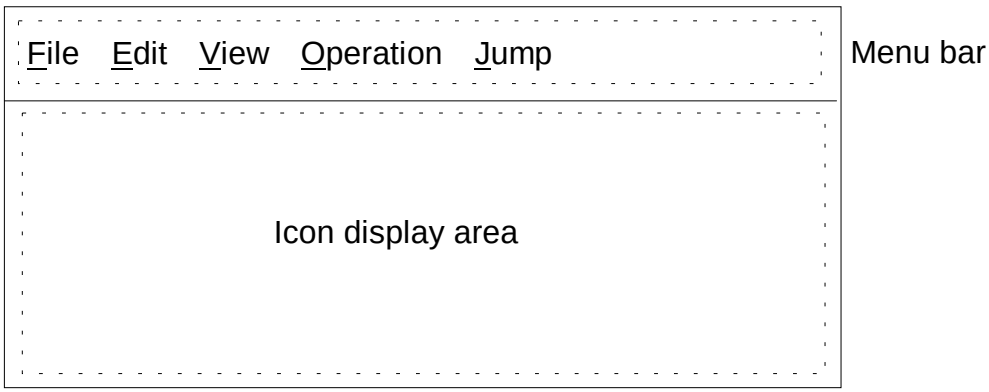
This type of window can be placed at any location outside the main window. It is, however, always displayed in front of the main window and cannot be reduced to an icon.
This type of window consists of a menu bar, function buttons, a location area, and information display area.



Windows of this type:
Register window **Variable window**

(4) Management windows

Management windows are used to manage the settings used for debugging. A management window consists of a menu bar and information display area.



Windows of this type:
Event Manager

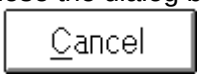
5.1.2 Dialog Boxes

Dialog boxes are classified into the following two types:

- **Modal dialog boxes**
- **Modeless dialog boxes**

(1) Modal dialog boxes

When you are working with this type of dialog box, you cannot access other debugger windows or dialog boxes, until you exit from this dialog box.

To access other windows or dialog boxes, close the dialog box by completing the operation enabled by the dialog box or by clicking the 

button in the dialog box

(2) Modeless dialog boxes

When you are working with this type of dialog box, you can access other debugger windows or dialog boxes, without first having to exit from this dialog box.

Dialog boxes are also classified into the following six types according to their function:

- **Selection dialog boxes**
- **Specification dialog boxes**
- **Setting dialog boxes**
- **Confirmation dialog boxes**
- **Auxiliary dialog boxes**
- **View dialog boxes**

(a) Selection dialog boxes

Selection dialog boxes are used to select a chip name or file name.

Dialog boxes of this type:

Configuration dialog box	**Load Module dialog box**
Project file save dialog box	**Project file load dialog box**
Upload dialog box	**View file load dialog box**
View file save dialog box	**Source file select dialog box**

(b) Specification dialog boxes

Specification dialog boxes provide text areas for specifying an address or path.

Dialog boxes of this type:

Addressing dialog box	**Trace pick-up dialog box**
Source Path dialog box	

(c) Setting dialog boxes

Setting dialog boxes are used to set conditions.

Dialog boxes of this type:

Extended Option dialog box
Event Set dialog box
Event Link dialog box
Break dialog box
Trace dialog box
Snap-Shot dialog box
Stub dialog box
Coverage Condition Setting dialog box
Coverage Memory Clear dialog box

(d) Confirmation dialog boxes

Confirmation dialog boxes are used to confirm a selected operation.

Dialog boxes of this type:

Reset Debugger dialog box
Exit Debugger dialog box
Error/Warning dialog box

(e) Auxiliary dialog boxes

Auxiliary dialog boxes are used to perform auxiliary operations for a window.

Dialog boxes of this type:

Variable dialog box
Find dialog box
Memory Copy dialog box
Memory Fill dialog box
Memory Compare dialog box
Add Variable dialog box

(f) View dialog boxes

View dialog boxes are used to temporarily display information.

Dialog boxes of this type:

Memory Compare result dialog box
About dialog box

(g) View/setting dialog boxes

View/setting dialog boxes provide areas for setting conditions and displaying information.

Dialog boxes of this type:

Timer dialog box

5.2 Debug Windows

Debug windows are listed below.

Table 5-1 Debug Windows (1/2)

Name	Outline	Page
Main window	Appears immediately after the debugger is activated.	41
Configuration dialog box	Sets the debugger operating environment.	56
Extended Option dialog box	Sets extended options.	63
Project file load dialog box	Loads the debugging environment.	66
Project file save dialog box	Saves the debugging environment.	69
Load Module dialog box	Loads an object file or symbol file.	72
Upload dialog box	Uploads memory contents into a file.	75
Source Path dialog box	Specifies source path information.	78
Source file select dialog box	Selects the source file to be displayed in the Source window.	80
Source window	Displays the source text.	83
Find dialog box	Searches for a character string in the current window.	88
Symbol to Address dialog box	Displays the address assigned to a symbol.	91
Variable View dialog box	Temporarily displays the value of a variable.	93
Variable window	Displays and modifies variables.	95
Add Variable dialog box	Adds variables to be displayed in the Variable window.	100
Local Variable window	Displays and modifies local variables in the current function.	102
Addressing dialog box	Specifies the display start address.	105
Assemble window	Displays a disassembled program and performs online assembly.	108
Memory window	Displays and modifies the memory contents.	115
Memory Fill dialog box	Initializes memory.	120
Memory Copy dialog box	Copies the memory contents.	122
Memory Compare dialog box	Compares the memory contents.	124
Memory Compare result dialog box	Displays the results of memory comparison.	126
Stack window	Displays the stack contents for functions.	128
Event Set dialog box	Registers event conditions.	131
Event Manager	Manages each registered event condition.	137
Event Link dialog box	Registers event link conditions.	146
Break dialog box	Registers and sets break event conditions.	154
Trace dialog box	Registers and sets trace event conditions.	158
Snap-Shot dialog box	Registers and sets snapshot event conditions.	165
Stub dialog box	Registers and sets stub event conditions.	172

Table 5-1 Debug Windows (2/2)

Name	Outline	Page
Timer dialog box	Displays the results of execution time measurement, and registers and sets timer event conditions.	176
Trace View window	Displays the results of trace.	182
Trace pick-up dialog box	Sets trace display conditions.	188
Register window	Displays and modifies the register contents.	192
SFR window	Displays and modifies the SFR contents.	198
Coverage window	Displays the results of coverage.	202
Coverage Efficiency View dialog box	Displays the results of coverage as a percentage.	205
Coverage Condition Setting dialog box	Sets the measurement range for coverage efficiency.	208
Coverage Memory Clear dialog box	Initializes the results of coverage.	212
View file load dialog box	Opens a reference window for the current window.	214
View file save dialog box	Saves the contents of the current window into a file.	218
Error/Warning dialog box	Displays an error or warning.	223
Reset Debugger dialog box	Resets the debugger and target CPU.	224
About dialog box	Displays the version of the debugger.	226
Exit Debugger dialog box	Exits from the debugger.	228

5.3 Details of Debug Windows

This section details each debug window, using the following format:

window-name	**window-type** **(mode-type-of-dialog-box)**

The window name and window type (and mode type, in the case of a dialog box) are indicated in this frame.

Outline

Provides a brief outline of the window.

Window

Displays the configuration of the window as it appears on the screen.

Description

Describes the contents of the window.

Buttons

Describes the function of each button in the window.

Menu bar

Lists menus that can be pulled down from the titles in the menu bar, and describes the functions of each menu item.

Main window

Execution window

Outline

The main window is automatically opened immediately after you have activated the debugger and completed the initial settings. The window remains displayed until the debugger is terminated. Other windows are opened from within this window.

The execution of a user program is controlled by using this window at the following two levels:

- Source level
  When a program is debugged in units of source text lines.
- Instruction level
  When a program is debugged in units of instructions.

When the debugger is started, source level debugging is selected by default.

Window

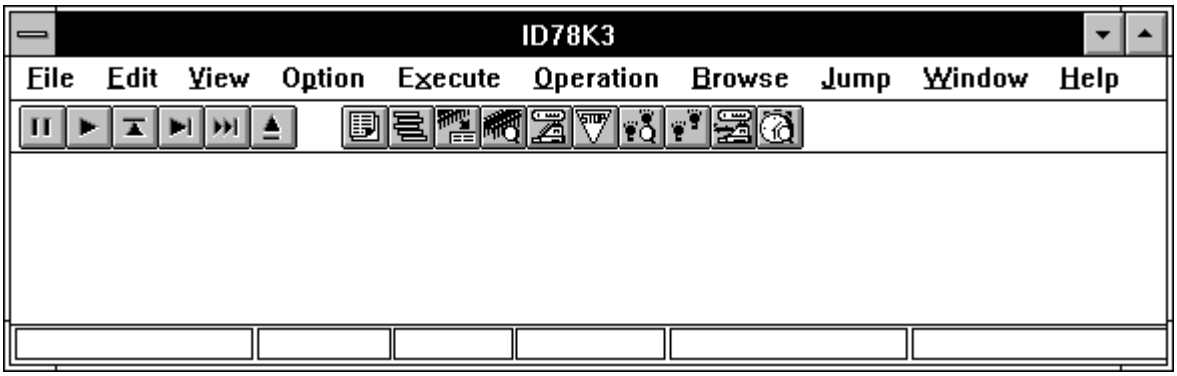


Fig. 5-1 Main Window

Description

The main window consists of the following components:

- Menu bar
- Tool bar
- Window display area
- Status bar

The function of each component is described below.

(1) Tool bar

The tool bar contains buttons which allow you to execute frequently used commands by a single action. On each button, the corresponding command is indicated graphically, as an icon. Commands assigned to the buttons can also be executed from the menu bar. You can hide the tool bar by selecting **Option→ToolBar** from the menu bar and removing the check mark.

Stops the user program.

Executes the user program.
While the program is being executed, this button appears to remain pressed. Once program execution stops, the button is released.

Executes the program in real time, until the execution returns to the calling function.

Executes the program, step by step.
This button enables the execution of as many steps as the number of times the button is clicked. For source level debugging, the program is executed in units of lines. For instruction level debugging, the program is executed in units of instructions.

Performs Next step execution of the program.
This button enables the execution of as many steps as the number of times the button is clicked, by means of Next step execution. For source level debugging, the program is executed in units of lines. For instruction level debugging, the program is executed in units of instructions.

Initializes the debugger or emulation CPU.
Opens the Reset Debugger dialog box.

Displays the source text.
Opens the Source window.

Displays the stack contents.
Opens the Stack window.

Displays a disassembled program.
Opens the Assemble window.

Displays the memory contents.
Opens the Memory window.

Displays the register contents.
Opens the Register window.

Registers and sets break events.
Opens the Break dialog box.

Displays the results of trace.
Opens the Trace View window.

Registers and sets trace events.
Opens the Trace dialog box.

Displays the SFR contents.
Opens the SFR window.

Registers and sets timer events, and displays the results of timer measurement.
Opens the Timer dialog box.

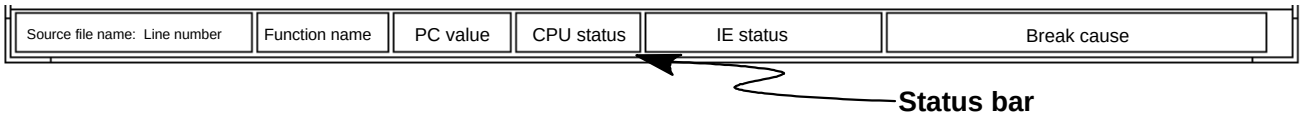
(2) Window display area

The window display area displays debug windows. Within this area, displayed windows can be expanded, contracted, or reduced to an icon.

Windows displayed in this area:

Source window	Assemble window
Local Variable window	Trace View window
Memory window	Coverage window
SFR window	Stack window

(3) Status bar



The status bar displays the status of the debugger and in-circuit emulator.

Source file name Displays the source file name and source line number corresponding to the indicated PC value. If no file information is provided, --- is displayed.

Function name Displays the function name corresponding to the indicated PC value. If no function information is provided, --- is displayed.

PC value Displays the current value of the program counter (PC).

CPU status Displays the status (such as Bus Hold, Stop, or Hold mode) of the CPU (μPD784xxx: Target device).

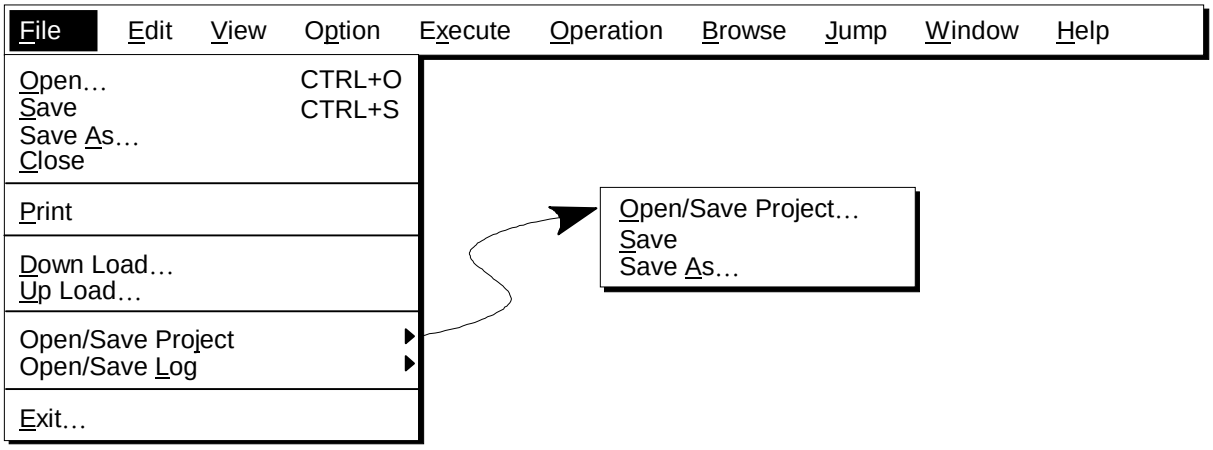
IE status Displays the status (such as RUN or Break mode) of the in-circuit emulator.

Break cause Displays the cause of a break. The table below lists the possible causes of break.

Cause	Description
Compulsory Break	Normal break
Event Break	Break triggered by an event
Non Map Break	An access has been attempted to a non-map area.
Relocation Break	A relocation instruction contradicting the initial setting has been executed.
SFR Illegal	An illegal access has been attempted to an SFR.
Software Break	Break triggered by a software break event
Stack Overflow	Break due to stack overflow
Temporary Break	Temporary break
Trace Full Break	Break due to trace full
Uninitialize Mem read	Reading from uninitialized memory has been attempted.
Write Protect	Writing to a write-protected area has been attempted.

Menu bar

(a) File



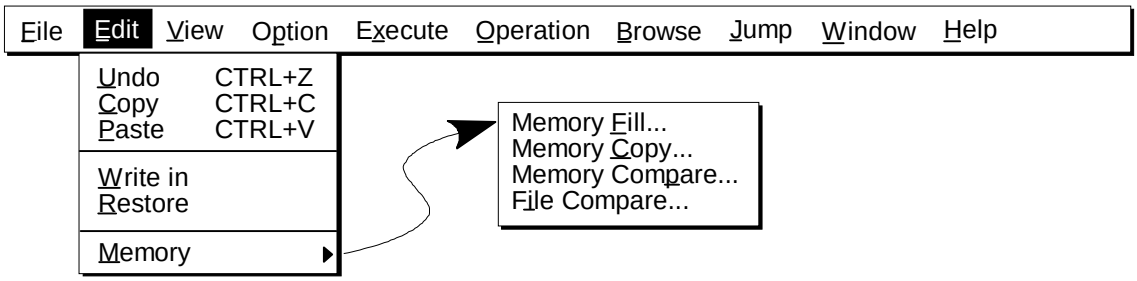
Open... The operation performed depends on the current window.
When the current window is the Source window:
Opens the source file select dialog box to enable selection of a source text file.
When the current window is other than the Source window:
Opens the view file load dialog box to enable display of a view file in the current window.

Save Saves the contents of the current window into the view file.

Save As... Saves the contents of the current window into a view file having a different name. The view file save dialog box is opened.

Close	Closes the current window.
Print	Prints the contents of the current window.
Down Load...	Downloads a program: The Load Module dialog box is opened.
Up Load...	Uploads a program: The Upload dialog box is opened.
Open/Save Project ▸	
Open Project...	Opens a project file: The project file load dialog box is opened.
Save	Overwrites the project file with the current operating environment. The file selected using **Open Project...** or **Save As...** of **File→ Open/Save Project** of the **menu bar** is overwritten.
Save As…	Saves the current operating environment into a project file: The project file save dialog box is opened.
Exit	Exits from the debugger: The Exit Debugger dialog box is opened.

(b) Edit



Undo Cancels the most recent editing.

Copy Copies a selected character string into the clipboard buffer.

Paste Pastes the contents of the clipboard buffer at the point to which the text cursor is positioned.

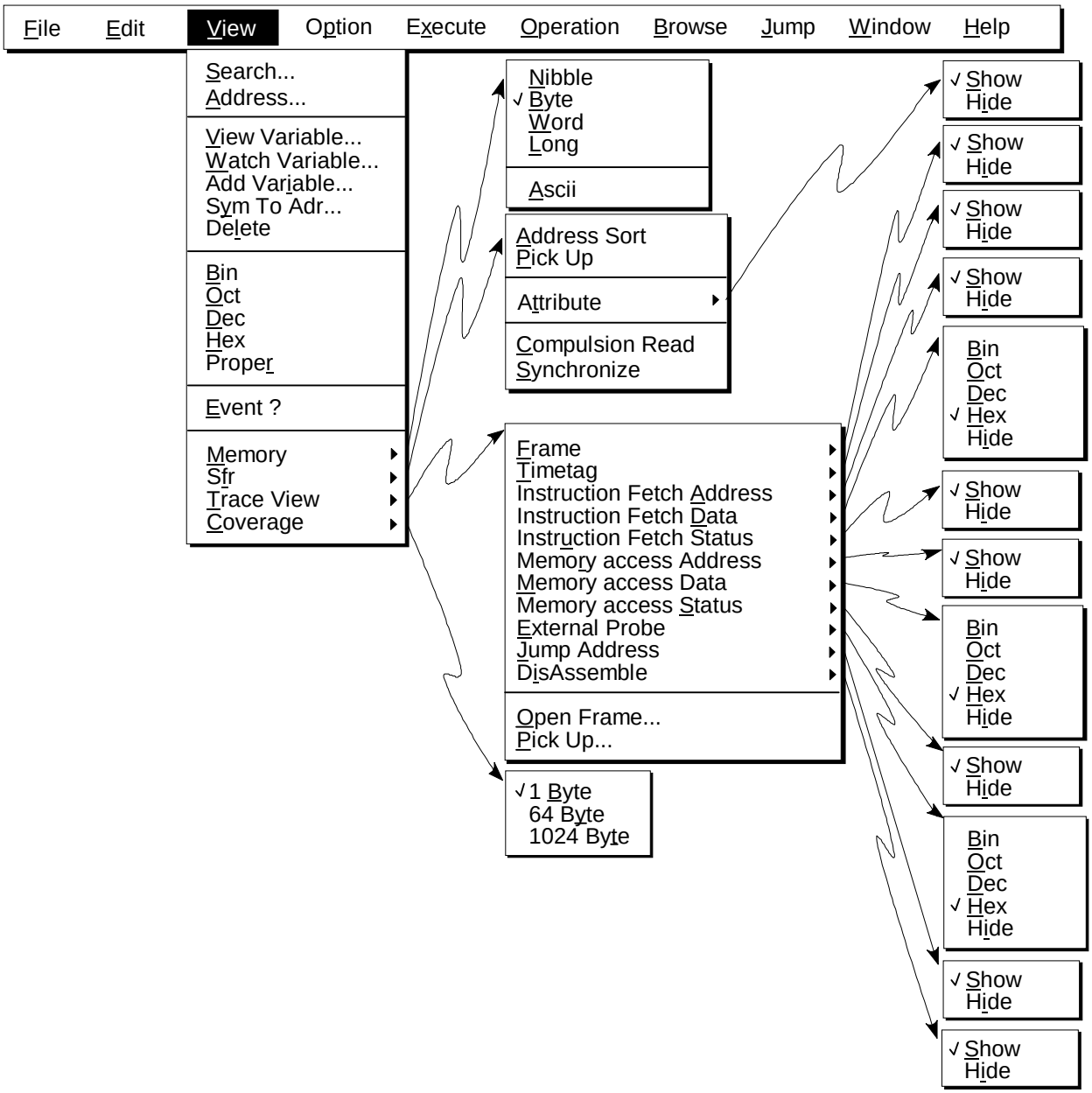
Write in Writes the modified contents into the target device. Executing this command has the same effect as clicking the Write in button.

Restore Cancels the modified contents. Executing this command has the same effect as clicking the Restore button.

Memory ▸

- **Memory Fill...** Initializes memory: The Memory Fill dialog box is opened.
- **Memory Copy...** Copies the contents of memory: The Memory Copy dialog box is opened.
- **Memory Compare...** Compares the contents of memory: The Memory Compare dialog box is opened.
- **File Compare...** Compares the view file with the contents of memory: The file select dialog box is opened.

(c) View



Search… Searches for a character string or numerical value: The Find dialog box is opened. Executing this command has the same effect as clicking the Search button.

Address… Displays the contents of memory at a specified address: The addressing dialog box is opened.

View Variable... Displays variables: The Variable View dialog box is opened. The value of a specified symbol is displayed.

Watch Variable... Displays specified variables: The Variable window is opened.

Add Variable... Adds specified variables to the Variable window: The Add Variable dialog box is opened.

Sym To Adr... Displays the address of a specified variable.

Delete Deletes a specified variable.

Bin Sets binary display format.

Oct Sets octal display format.

Dec Sets decimal display format.

Hex Sets hexadecimal display format.

Proper Sets default display format for each variable (default).

Event ? Displays event information: The Event Manager is opened.

Memory ▸
- **Nibble** Displays data in units of 4 bits.
- **Byte** Displays data in units of 8 bits (default).
- **Word** Displays data in units of 16 bits.
- **Long** Displays data in units of 32 bits.
- **Ascii** Turns on/off ASCII view mode.

Sfr ▸
- **Address Sort** Specifies the SFR display order.
  Without check mark: Alphabetical order
  With check mark (✓): In order of addresses
- **Pick Up** Displays only modified SFRs.
- **Attribute** ▸
  - **Show** Displays the SFR attribute (default).
  - **Hide** Hides the SFR attribute.
- **Compulsion Read** Performs forced reading of a read-protected SFR.
- **Synchronize** Writes the modified SFRs to the target device.

Trace View ▸
- **Frame** ▸ Displays or hides the frame number field.
  - **Show** Displays the frame number (default).
  - **Hide** Hides the frame number.
- **Timetag** ▸ Displays or hides the time tag field.
  - **Show** Displays the time tag (default).
  - **Hide** Hides the time tag.
- **Instruction Fetch Address** ▸ Displays or hides the fetch address field.
  - **Show** Displays the address (default).
  - **Hide** Hides the address.
- **Instruction Fetch Data** ▸ Displays or hides the fetch data field.
  - **Bin** Displays data in binary format.
  - **Oct** Displays data in octal format.
  - **Dec** Displays data in decimal format.
  - **Hex** Displays data in hexadecimal format (default).
  - **Hide** Hides data.

Instruction Fetch Status ▸ Displays or hides the fetch status field.
- **Show** Displays the status (default).
- **Hide** Hides the status.

Memory access Address ▸ Displays or hides the access address field.
- **Show** Displays the address (default).
- **Hide** Hides the address.

Memory access Data ▸ Displays or hides the access data field.
- **Bin** Displays data in binary format.
- **Oct** Displays data in octal format.
- **Dec** Displays data in decimal format.
- **Hex** Displays data in hexadecimal format (default).
- **Hide** Hides data.

Memory access Status ▸ Displays or hides the access status field.
- **Show** Displays the status (default).
- **Hide** Hides the status.

External Probe ▸ Displays or hides the external sense data field.
- **Bin** Displays external sense data in binary format.
- **Oct** Displays external sense data in octal format.
- **Dec** Displays external sense data in decimal format.
- **Hex** Displays external sense data in hexadecimal format (default).
- **Hide** Hides external sense data.

Jump Address ▸ Displays or hides the jump address field.
- **Show** Displays the jump address (default).
- **Hide** Hides the jump address.

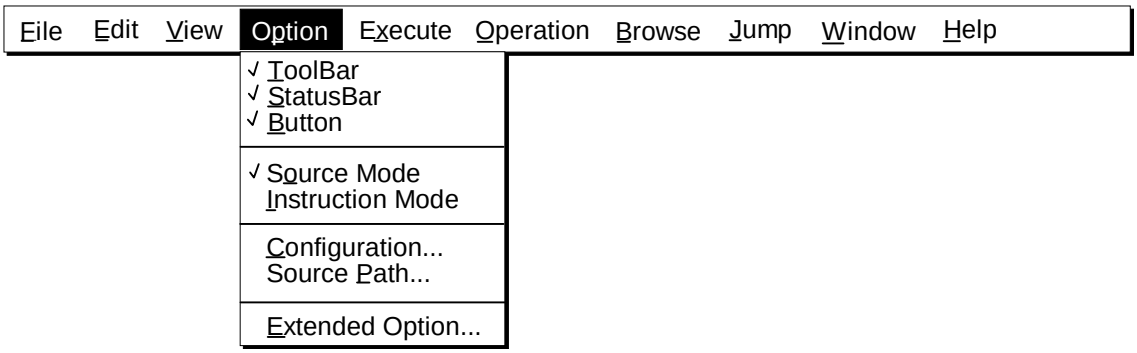
DisAssemble ▸ Displays or hides the disassembled data field.
- **Show** Displays the disassembled data (default).
- **Hide** Hides the disassembled data.

Open Frame... Specifies the view start frame: The start frame specification dialog box is opened.

Pick Up... Sets trace view conditions: The trace pick-up dialog box is opened.

Coverage ▸
- **1 Byte** Displays data in 1-byte units.
- **16 Byte** Displays data in 16-byte units.
- **1024 Byte** Displays data in 1024-byte units.

(d) Option



ToolBar Displays or hides the tool bar.

StatusBar Displays or hides the status bar.

Button Displays or hides the buttons in windows.

Source Mode Performs step execution at the source level.

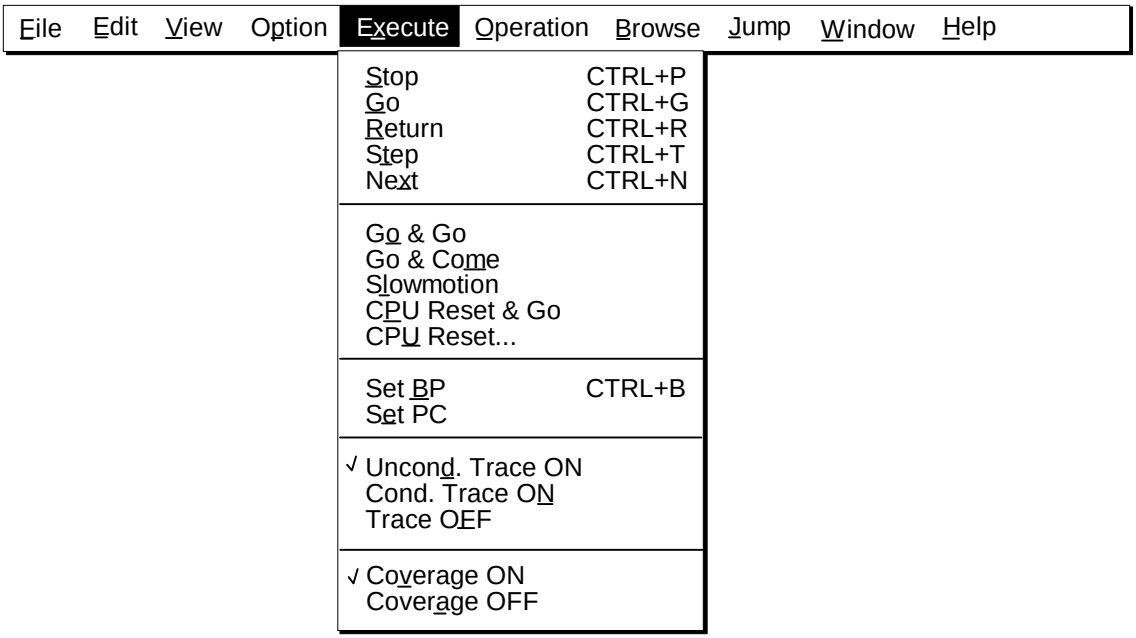
Instruction Mode Performs step execution at the instruction level.

Configuration... Sets the environment: The Configuration dialog box is opened.

Source Path... Sets source path information: The Source Path dialog box is opened.

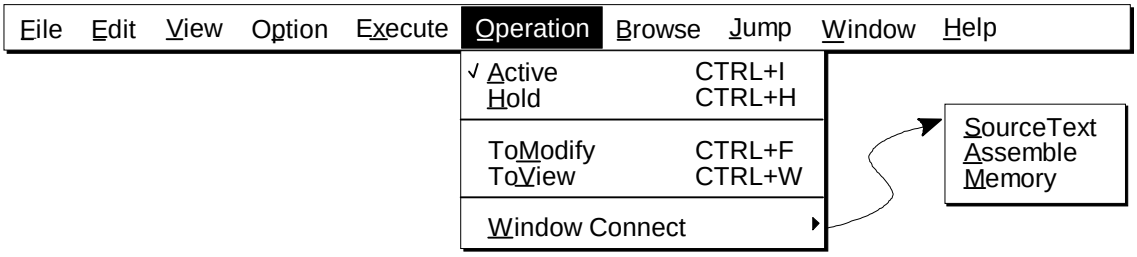
Extended Option... Sets extended options.

(e) Execute



Stop	Stops the execution of a program: Executing this command has the same effect as clicking the  button.
Go	Executes a program: Executing this command has the same effect as clicking the  button.
Return	Executes a program in real time, until the execution is returned to the calling function: Executing this command has the same effect as clicking the  button.
Step	Executes a program, step by step: Executing this command has the same effect as clicking the  button.
Next	Performs Next step execution of a program: Executing this command has the same effect as clicking the  button.
Go & Go	Repeatedly executes a program. When a break condition is satisfied, the window is updated and the program is executed again. Executing this command has the same effect as clicking the  button each time a break occurs.
Go & Come	Executes a program in real time, up to a specified address.
Slowmotion	Continues step-by-step execution of a program.
CPU Reset & Go	Resets the target device, then executes a program.
CPU Reset...	Resets either the target device alone or the entire debugger system: The Reset Debugger dialog box is opened. Executing this command has the same effect as clicking the  button.
Set BP	Sets a breakpoint on a selected line.
Set PC	Sets the address of a selected line in the program counter (PC).
Uncond. Trace ON	Sets unconditional tracing of program execution by the tracer.
Cond. Trace ON	Sets tracing of program execution by the tracer only when preset conditions are satisfied.
Trace OFF	Disables the tracer (does not trace the execution of a program).
Coverage ON	Enables coverage measurement.
Coverage OFF	Disables coverage measurement.

(f) Operation



Active Sets the window to the active state.

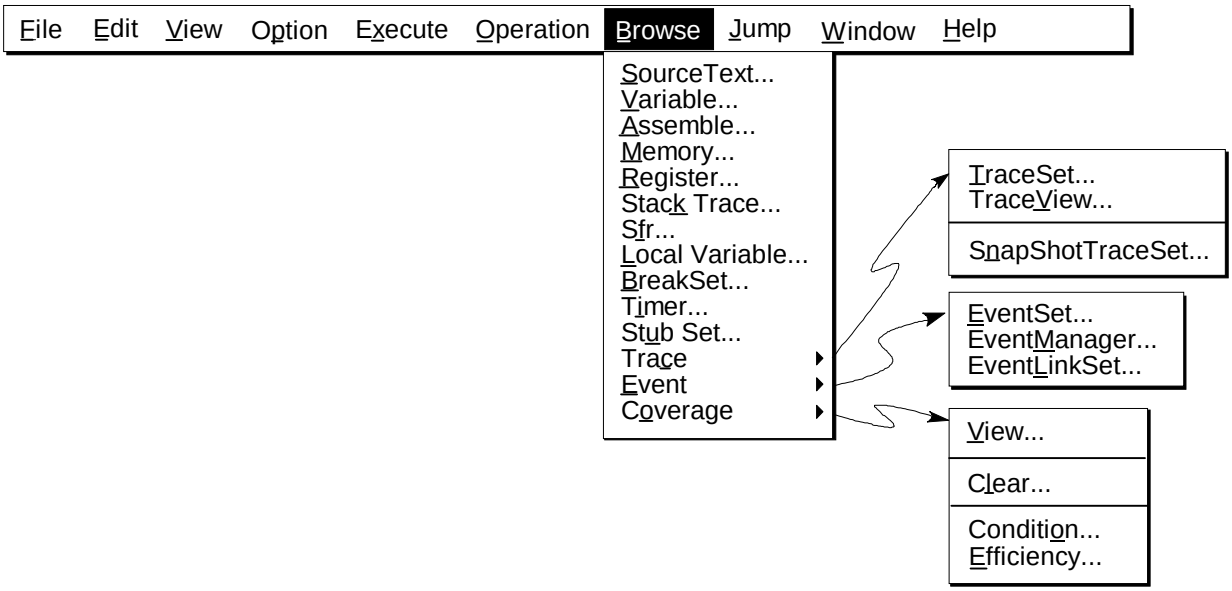
Hold Sets the window to the hold state.

ToModify Sets the window to modify mode: Executing this command has the same effect as clicking the ToModify button.

ToView Sets the window to view mode: Executing this command has the same effect as clicking the ToView button.

Window Connect ▸ Links the Trace View window to a specified window.
- **SourceText** The Trace View window is linked to the Source window.
- **Assemble** The Trace View window is linked to the Assemble window.
- **Memory** The Trace View window is linked to the Memory window.

(g) Browse



Source Text... Displays the source text: The Source window is opened. Executing this command has the same effect as clicking the  button.

Variable... Displays specified variables: The Variable window is opened.

Assemble... Displays disassembled text: The Assemble window is opened. Executing this command has the same effect as clicking the  button.

Memory... Displays the memory contents: The Memory window is opened. Executing this command has the same effect as clicking the  button.

Register... Displays the register contents: The Register window is opened. Executing this command has the same effect as clicking the  button.

Stack Trace... Displays the stack contents: The Stack window is opened. Executing this command has the same effect as clicking the  button.

Sfr... Displays the SFR contents: The SFR window is opened. Executing this command has the same effect as clicking the  button.

Local Variable... Displays local variables: The Local Variable window is opened.

Break Set... Registers and sets break event conditions: The Break dialog box is opened. Executing this command has the same effect as clicking the  button.

Timer... Registers and sets timer event conditions, and displays the results of timer measurement: The Timer dialog box is opened. Executing this command has the same effect as clicking the  button.

Stub Set... Registers and sets stub event conditions: The Stub dialog box is opened.

Trace ▸ Opens a window for a trace.

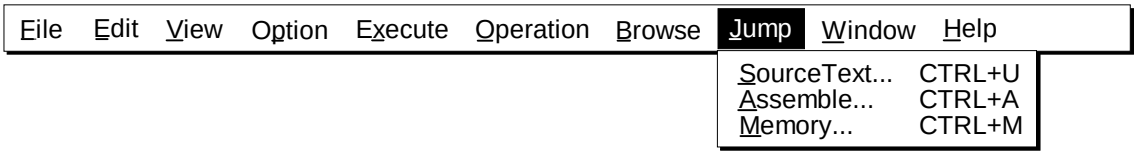
- **TraceSet...** Registers and sets trace event conditions: The Trace dialog box is opened. Executing this command has the same effect as clicking the  button.
- **TraceView...** Displays the results of trace: The Trace View window is opened. Executing this command has the same effect as clicking the  button.
- **SnapShotTraceSet...** Registers and sets snapshot event conditions: The Snap-Shot dialog box is opened.

Event ▸ Opens a window for an event.

EventSet... Registers event conditions: The Event Set dialog box is opened.
EventManager... Manages event conditions: The Event Manager is opened.
EventLinkSet Set... Registers event link conditions: The Event Link dialog box is opened.

Coverage ▸ Opens a window for coverage.
View... Displays the results of coverage measurement: The Coverage window is opened.
Clear... Initializes the results of coverage: The Coverage Memory Clear dialog box is opened.
Condition... Sets the conditions for measuring coverage efficiency: The Coverage Condition Setting dialog box is opened.
Efficiency... Displays the coverage efficiency: The Coverage Efficiency View dialog box is opened.

(h) Jump

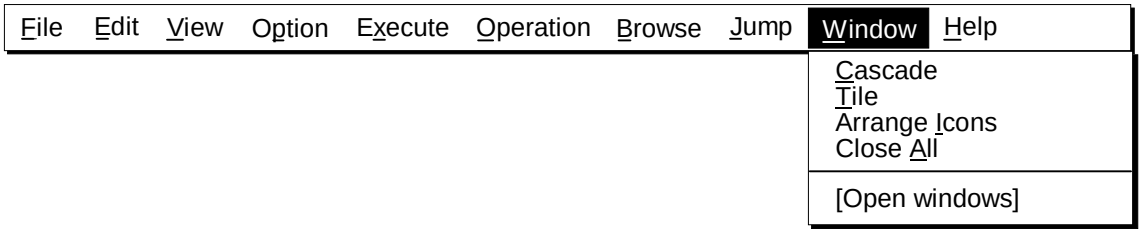


SourceText... Sets the data selected in the current window as the jump address, and displays the source text and source line starting from that address: The Source window is opened. No jump can be performed if the jump address contains no line information.

Assemble... Sets the data selected in the current window as the jump address, and displays the disassembled text starting from that address: The Assemble window is opened.

Memory... Sets the data selected in the current window as the jump address, and displays the memory contents starting from that address: The Memory window is opened.

(i) Window



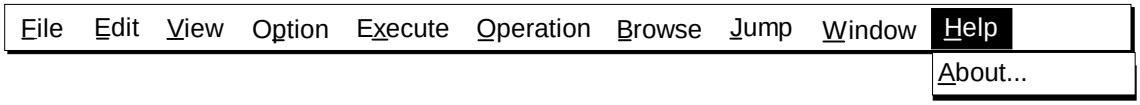
Cascade Displays the windows in cascade style within the main window.

Tile Displays the windows in tile style within the main window.

Arrange Icons Re-arranges the icons within the main window.

Close All	Closes all windows except the main window.
[Open windows]	Lists the names of all currently open windows. Selecting a window name from the list causes that window to become the current window.

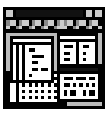
(j) Help



About...	Displays the version of the debugger.

Icon

The main window can be reduced to the following icon by clicking the ▼ button on the title bar. The user can freely create or modify the graphic or title of the icon.



ID78K3

Configuration dialog box

Selection dialog box (Modal)

Outline

The Configuration dialog box is used to display and set the operating environment of the in-circuit emulator.

This dialog box is displayed when the debugger is started. To use the debugger, first set the operating environment of the in-circuit emulator using this dialog box. To load a project file, however, there is no need to set the operating environment. In such a case, the contents of the project file are reflected in the Configuration dialog box.

The settings for pin mask, location, and memory mapping in this dialog box can also be added or modified during debugging.

This dialog box can be opened in any of the following ways:

- When the debugger is started, the dialog box is automatically opened.
- In the main window, select **Option→Configuration...** from the **menu bar**.
- In the main window, press the **GRPH**, **P**, **and** **C** **keys**, in this order.

Window

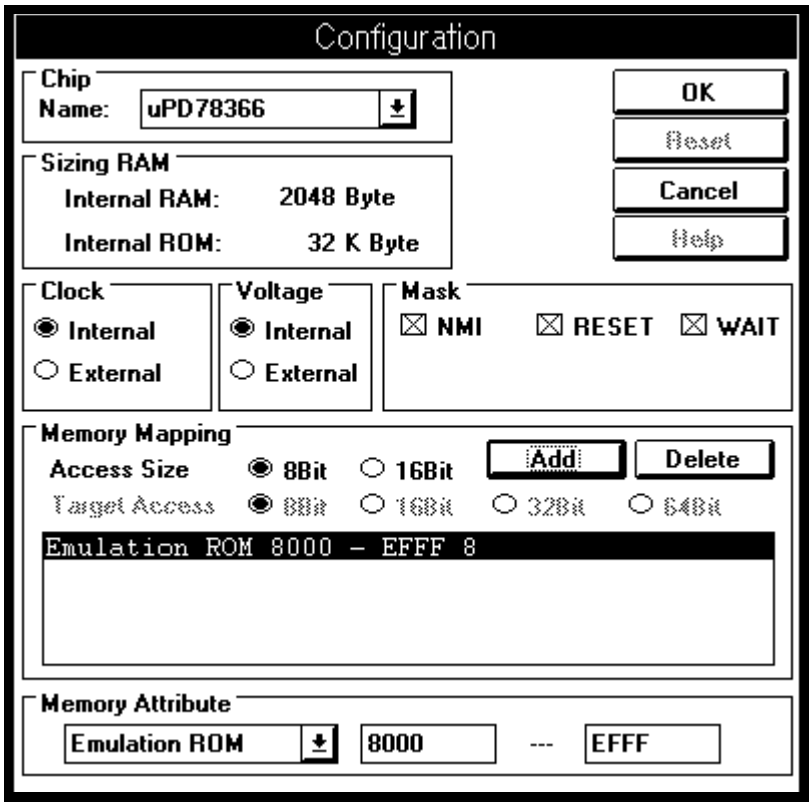


Fig. 5-2 Configuration Dialog Box

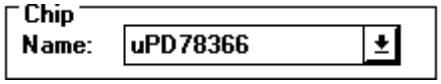
Description

The Configuration dialog box consists of the following components:

- Emulation CPU selection area
- Internal ROM/RAM view area
- CPU clock source selection area
- Power supply selection area
- Mask setting area
- Mapping setting area
- Mapping attribute specification area
- Function buttons

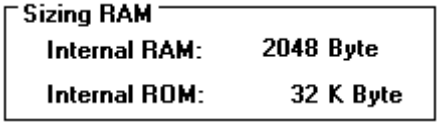
The function of each component is described below.

(1) Emulation CPU selection area



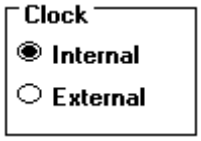
Select the CPU to be emulated. The emulation CPU can be selected only when this dialog box is opened upon activation of the debugger.

(2) Internal ROM/RAM view area



This area displays the internal ROM and RAM sizes of the emulation CPU. These sizes are displayed automatically once the emulation CPU has been selected.

(3) CPU clock source selection area

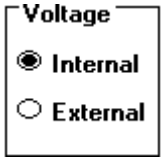


Select the CPU clock source. The following three clock sources are supported (Internal is the default):

- **Internal**: The clock of the in-circuit emulator is used as the CPU clock (32 MHz).
- **External**: The clock of the target device is used as the CPU clock.

The CPU clock can be selected when the debugger is started or a project file is loaded. You cannot select the CPU clock once debugger operation has been started.

(4) Power supply selection area



Select how power is to be supplied to the emulation CPU. The following two power supplies are supported (Internal is the default):

- **Internal:** The power supply of the in-circuit emulator is used as the power supply. (The operating voltage is fixed to 5 V.)
- **External:** The power supply of the target device is used as the power supply. (The operating voltage can be varied within the specifications of the device.)

Caution: Just setting the power supply selection area is not enough to select the power supply. Also set emulation board 1 according to the selected power supply.

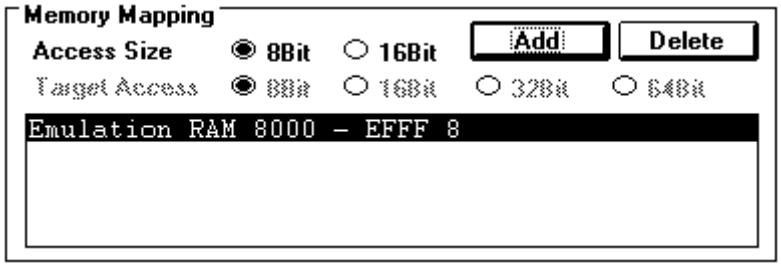
(5) Mask setting area

You can mask specified signals from the target device using this area. Masked signals are not input to the in-circuit emulator. (Signals should be masked only when the operation of the target device becomes unstable during debugging.)

The signals of the following three pins can be masked:

- NMI pin
- RESET pin
- WAIT pin

(6) Mapping setting area



Select the memory bus width. This area also allows you to add or delete memory mapping.

a. Selecting the bus width

Select a bus width from the displayed choices:

- 8 bits
- 16 bits

(The choices vary with the target chip.)

b. Setting memory mapping

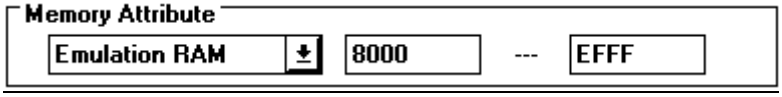
- To add memory mapping

  Click the Add button.
  Memory mapping is added according to the data specified in the Memory Attribute area and the selected bus width.

- To delete memory mapping

  Select the memory mapping to be deleted, then click the Delete button.
  The selected memory mapping is deleted.

(7) Mapping attribute specification area



Specify the type and range of memory mapping.

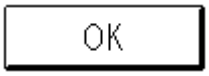
Select one of the following three memory mapping types according to the application:

- **Emulation ROM:** Selects alternate ROM for the in-circuit emulator.
- **Emulation RAM:** Selects alternate RAM for the in-circuit emulator.
- **Target:** Selects target memory.

The mapping units depend on the mapping addresses, as listed below.

Mapping addresses	Mapping units
0x0000 - 0x0ffff	8K bytes

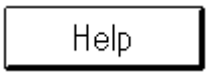
Buttons



Completes the environment setting. Closes the dialog box and sets the environment as specified in the dialog box.

Restores the environment existing before the dialog box was opened.

Ignores any selections and closes the dialog box.

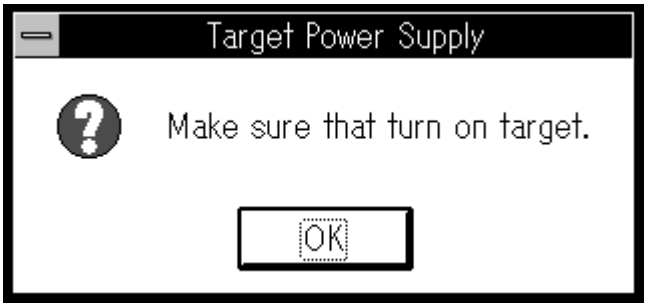


Opens the help window, which provides a detailed explanation of the Configuration dialog box.

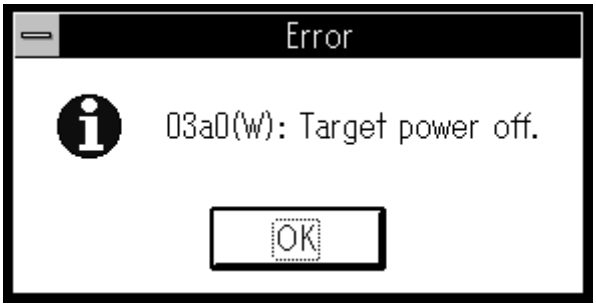
Caution

When any of the following settings is made using the Configuration dialog box, the power of the target device must be turned on. Otherwise, a message prompting power-on of the target device is displayed. Confirm that the power of the target device is turned on before clicking the OK button.

Item	Setting
Power supply selection area	External
Mapping attribute specification area	Target



If the OK button is clicked even though the target device power is not turned on, the following message appears, and the debugger is terminated:



Extended Option dialog box

Setting dialog box (Modal)

Outline

The Extended Option dialog box is used to display and set the extended debugger options.

This dialog box can be opened in either of the following ways:

- In the main window, select **Option→Extended Option...** from the **menu bar**.
- In the main window, press the **GRPH**, **P**, **and E keys**, in this order.

Window

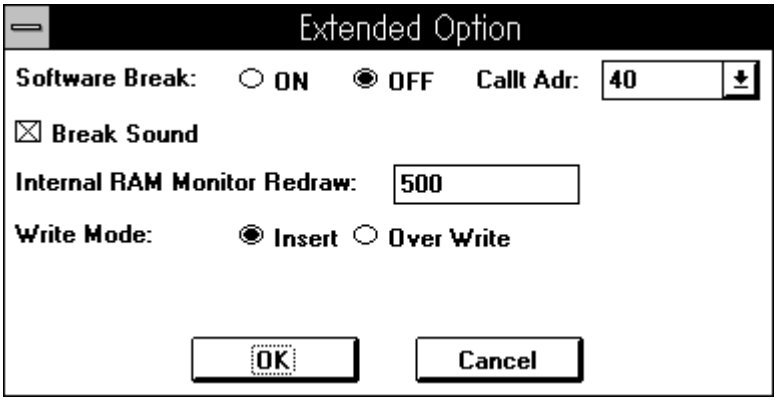


Fig. 5-3 Extended Option Dialog Box

Description

The Extended Option dialog box consists of the following components:

- Software break setting area
- Break mode setting area
- Real-time internal RAM sampling time setting area
- Write mode selection area
- Function buttons

The function of each component is described below.

(1) Software break setting area

Turn software break on or off.

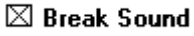
ON: Software break is used.

OFF: Software break is not used.

Callt Adr: Vector address for CALLT instruction to be freed for debugger

When software break is used, a vector address for the CALLT instruction must be freed so that the debugger can use the address.

(2) Break mode setting area



Turn the break sound on or off.

☒ **Break Sound** : Sound is output.
☐ **Break Sound** : Sound is not output.

(3) Real-time internal RAM sampling time setting area

Set the sampling time for real-time RAM.

Real-time sampling can be performed within the following range:

256 bytes from address 0xfe00 to 0xfeff

Variables and other data stored within this range can be displayed, in real time, within the Variable window or Memory window. The sampling time can be specified in units of 1 ms.

(4) Write mode selection area

Select write mode for windows in modify mode. The following two write modes are supported:

- **Insert**: Insert mode
- **Over Write:** Overwrite mode

Buttons

Stores the changes which have been made and closes the dialog box.

Cancels the changes which have been made and closes the dialog box.

Project file load dialog box

Selection dialog box (Modal)

Outline

The project file load dialog box is used to restore a previously saved debugger environment. Once a project file has been loaded, the size and position of each displayed window are set to the state saved in the file (except for analyzer functions).

This dialog box can be opened in either of the following ways:

- In the main window, select **File→Open/Save Project→Open Project...** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **J**, **and** **O** **keys**, in this order.

Window

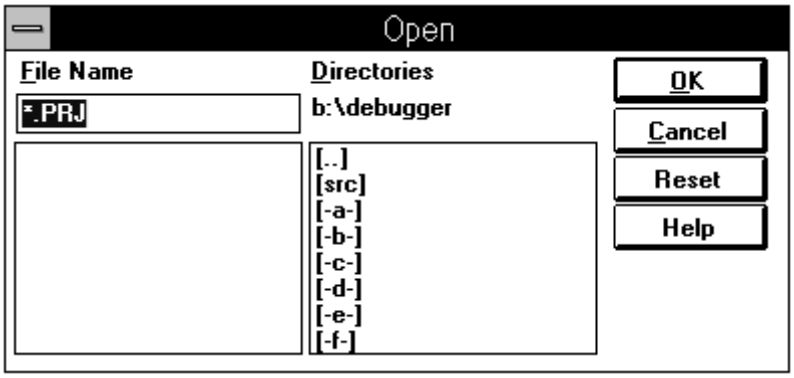


Fig. 5-4 Project File Load Dialog Box

Description

The project file load dialog box consists of the following components:

- File selection area
- Path setting area
- Function buttons

The function of each component is described below.

(1) File selection area

Specify the name of the project file to be loaded.

You can select a project file from the list by clicking it. The selected file name is highlighted and displayed in the area above the list. The default extension for a project file name is **.PRJ**.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the OK button.

(2) Path setting area

Specify the path of the project file to be loaded.•

Double-clicking a path name in the list displays the project files under the path, in the project file list.

Directories and drives are distinguished in the list as follows:
[xxx]: Directory name
[-x-]:Drive name

Buttons

OK	Loads the specified project file and sets the environment accordingly.
Cancel	Closes the project file load dialog box.
Reset	Ignores any selections and resets the initial state.
Help	Opens the help window.

Loaded data

The following data is loaded from a project file. If a project file for a different target device is loaded after the debugger has been started, data related to the target device is ignored.

Window	Data
Configuration dialog box	All items
Main window	Display position, display information about tool bar/status bar/buttons, execution mode, and trace on/off information
Load Module dialog box	Download file information
Extended Option dialog box	Set information
Source Path dialog box	Source path information
Source window	Window display information and font information
Assemble window	Window display information and display start address
Memory window	Window display information and display start address
Stack window	Window display information
SFR window	Window display information
Local Variable window	Window display information
Trace View window	Window display information
Event Manager	Window display information and all event information
Event Link dialog box	Window display information
Break dialog box	Window display information
Trace dialog box	Window display information
Snap-Shot dialog box	Window display information
Stub dialog box	Window display information
Event Set dialog box	Window display information
Register window	Window display information and display banks
Variable window	Window display information and displayed variable information
Coverage window	Window display information

Project file save dialog box

Selection dialog box (Modal)

Outline

The project file save dialog box is used to save the current debugger environment into a file called a project file. The size and position of each displayed window are also stored. Only data for active windows is stored.

This dialog box can be opened in either of the following ways:

- In the main window, select **File→Open/Save Project→Save As...** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **J**, **and** **A** **keys**, in this order.

When a project file has already been loaded or saved, you can save the environment into that project file in either of the following ways:

- In the main window, select **File→Open/Save Project→Save** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **J**, **and** **S** **keys**, in this order.

In this case, the project file save dialog box does not appear but the current environment is saved to an existing project file.

Window

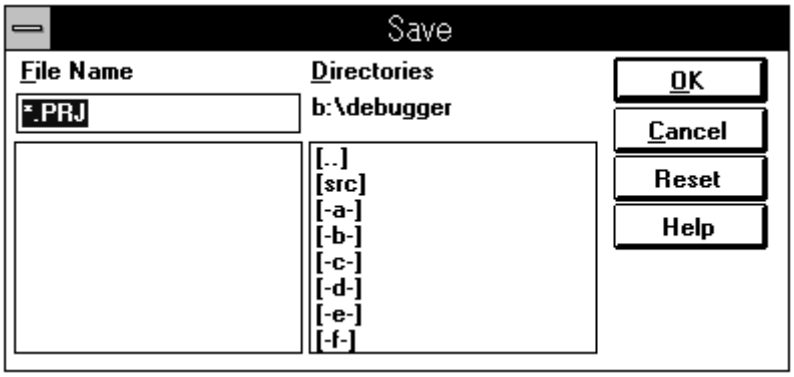


Fig. 5-5 Project File Save Dialog Box

Description

The project file save dialog box consists of the following components:

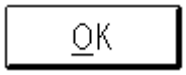
- File selection area
- Path setting area
- Function buttons

The function of each component is described below.

(1) File selection area

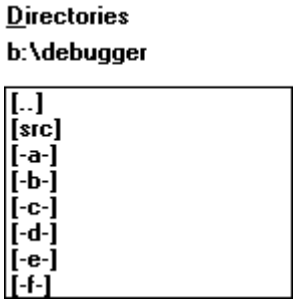
Specify the name of the project file into which the current environment is to be saved.

Enter the file name from the keyboard. You can also select a project file from the list by clicking it, when that file is to be overwritten. The selected file name is highlighted and displayed in the area above the list. The default extension for a project file name is **.PRJ**.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the 

button.

(2) Path setting area



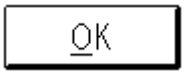
Specify the path under which the project file will be stored.•

Double-clicking a path name in the list displays the project files under that path, in the project file list.

Directories and drives are distinguished in the list as follows:

[xxx]: Directory name
[-x-]:Drive name

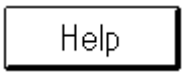
Buttons



Saves the current environment into the specified project file.

Closes the project file save dialog box.

Ignores any selections and resets the initial state.



Opens the help window.

Saved data

The following data is saved into a project file:

Window	Data
Configuration dialog box	All items (Target device, clock setting, pin mask setting, and mapping information)
Main window	Display position, display information about tool bar/status bar/buttons, execution mode, and trace on/off information
Load Module dialog box	Download file information
Extended Option dialog box	Set information
Source Path dialog box	Source path information
Source window	Window display information and font information
Assemble window	Window display information and display start address
Memory window	Window display information and display start address
Stack window	Window display information
SFR window	Window display information
Local Variable window	Window display information
Trace View window	Window display information
Event Manager	Window display information and all event information
Event Link dialog box	Window display information
Break dialog box	Window display information
Trace dialog box	Window display information
Snap-Shot dialog box	Window display information
Stub dialog box	Window display information
Event Set dialog box	Window display information
Register window	Window display information and display banks
Variable window	Window display information and displayed variable information
Coverage window	Window display information

Load Module dialog box

Selection dialog box (Modal)

Outline

The Load Module dialog box is used to download a file to the in-circuit emulator or target device by specifying the name and type of the file.

Files of the following types can be downloaded:

- Load module object files (.LNK)
- Intel extended hexadecimal format
- S type of Motorola hexadecimal format (standard address)
- Extended Tektronix hexadecimal format

Source debugging can be performed only for load module object files.

This dialog box can be opened in either of the following ways:

- In the main window, select **File→Down Load....** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **and** **D** **keys**, in this order.

Window

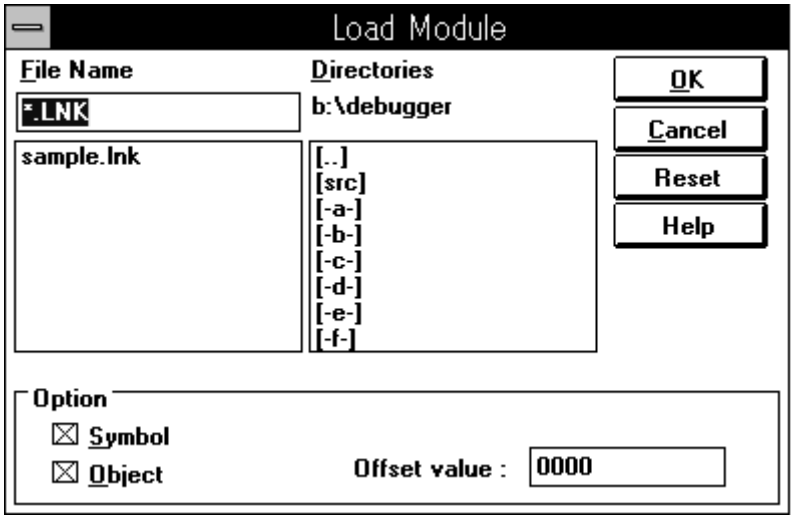


Fig. 5-6 Load Module Dialog Box

Description

The Load Module dialog box consists of the following components:

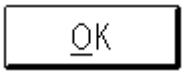
- File selection area
- Path setting area
- Load condition specification area
- Function buttons

The function of each component is described below.

(1) File selection area

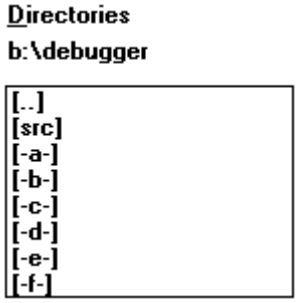
Specify the name of the load module file to be downloaded.

You can select a load module file from the list by clicking it. The selected file name is highlighted and displayed in the area above the list. The default extension for a load module file name is **.LNK**.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the 

button.

(2) Path setting area



Specify the path of the load module file to be downloaded.•

Double-clicking a path name in the list displays the load module files under that path, in the load module file list.

Directories and drives are distinguished in the list as follows:

[xxx]: Directory name
[-x-]: Drive name

(3) Load condition specification area

Specify the load conditions.

Symbol	Specifies whether to read symbol information.
Object	Specifies whether to read object information.
Offset value	Specifies the offset address.

Buttons

OK	Downloads the selected file.
Cancel	Ignores any selections and closes the dialog box.
Reset	Ignores any selections and resets the initial state.
Help	Opens the help window, which provides a detailed explanation of the Load Module dialog box.

Upload dialog box

Selection dialog box (Modal)

Outline

The Upload dialog box is used to save the memory contents to a file by specifying the name and type of the file.

The memory contents can be saved to files of the following three types:

- Intel extended hexadecimal format
- S type of Motorola hexadecimal format (standard address)
- Extended Tektronix hexadecimal format

This dialog box can be opened in either of the following ways:

- In the main window, select **File→Up Load...** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **and** **U** **keys**, in this order.

Window

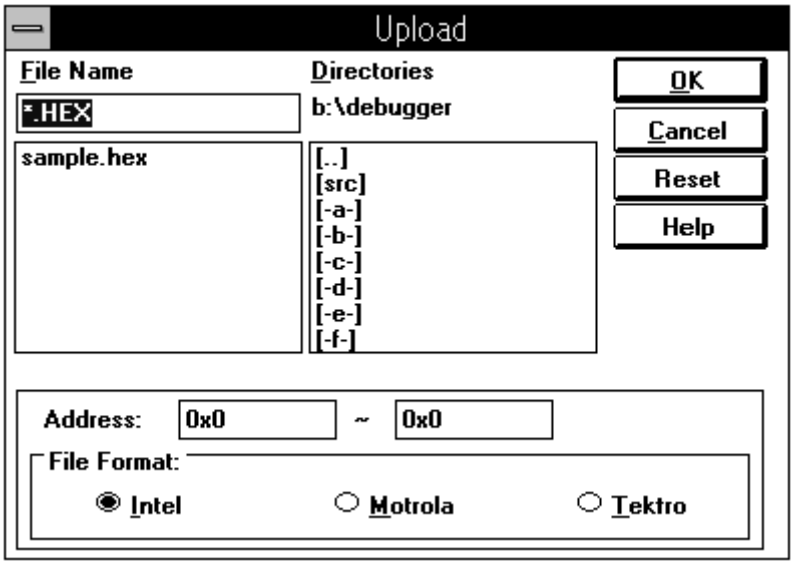


Fig. 5-7 Upload Dialog Box

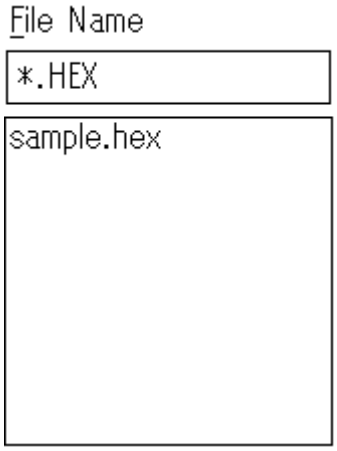
Description

The Upload dialog box consists of the following components:

- File selection area
- Path setting area
- Upload condition specification area
- Function buttons

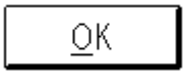
The function of each component is described below.

(1) File selection area



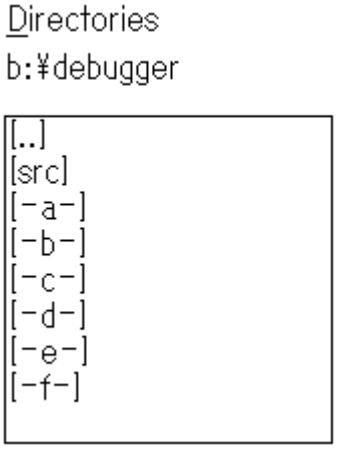
Specify the name of the object file for uploading.

Enter the file name from the keyboard. You can also select an object file from the list by clicking it, when that file is to be overwritten. The default extension for an object file name is **.HEX**.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the 

button.

(2) Path setting area



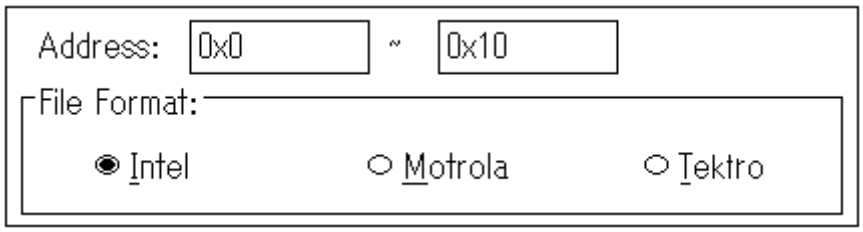
Specify the path under which the object file for uploading will be stored.

Double-clicking a path name in the list displays the object files under that path, in the object file list.

Directories and drives are distinguished in the list as follows:

[xxx]: Directory name
[-x-]: Drive name

(3) Upload condition specification area



Specify the upload conditions.

■ **Address:**
Specifies the address range in memory to be uploaded.

■ **File Format:**
Specifies the type of the object file to which the memory contents will be uploaded. The following three file types are supported:

Intel: Intel extended hexadecimal format
Motrola: S type of Motorola hexadecimal format (standard address)
Tektro: Extended Tektronix hexadecimal format

Buttons

OK	Saves the memory contents within the specified address range into a file according to the specified directory, file name, and file type.
Cancel	Closes the Upload dialog box.
Reset	Ignores any selections and resets the initial state.
Help	Opens the help window.

Source Path dialog box

Specification dialog box (Modal)

Outline

The Source Path dialog box is used to specify the source paths. When the source paths are specified, you can perform source debugging of the source text stored in two or more directories.

This dialog box can be opened in either of the following ways:

- In the main window, select **Option→Source Path...** from the **menu bar**.
- In the main window, press the **GRPH**, **P**, **and** **P** **keys**, in this order.

Window

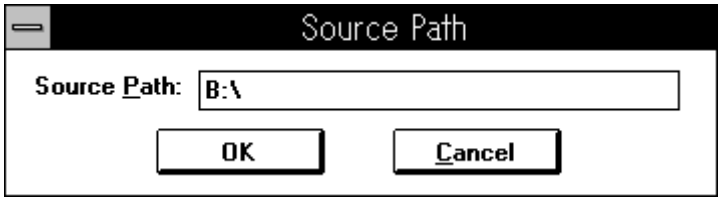


Fig. 5-8 Source Path Dialog Box

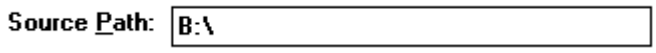
Description

The Source Path dialog box consists of the following components:

- Source path specification area
- Function buttons

The function of each component is described below.

(1) Source path specification area



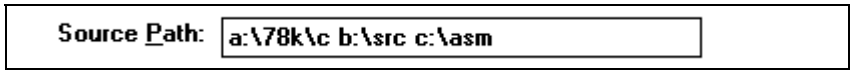
Specify the source paths, delimiting them with a blank.

Path information specifiable in this area is limited to 256 characters (including delimiters).

Example: When the source text is stored in the following directories:

a:\78k\c
b:\src
c:\asm

Specify the source paths as follows:



Buttons

OK	Sets the specified source paths and closes the dialog box.
Cancel	Cancels the specified source paths and closes the dialog box.

Source file select dialog box

Selection dialog box (Modal)

Outline

The source file select dialog box is used to select the source text to be displayed within the Source window. The source text can be selected in either of the following two ways:

- Specifying a source file name
- Specifying a function name

This dialog box can be opened in any of the following ways when the current window is the Source window:

- In the main window, select **File→Open...** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **and** **O** **keys**, in this order.
- Press the **CTRL**+**O** **short cut keys**.

Window

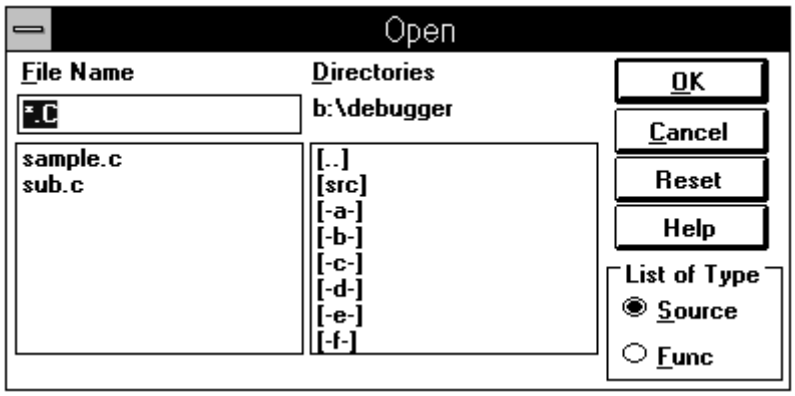


Fig. 5-9 Source File Select Dialog Box

Description

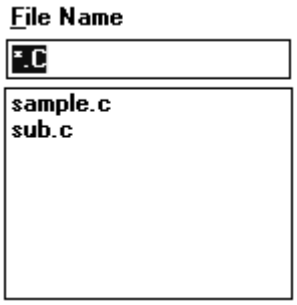
The components of the source file select dialog box vary with the selection mode, as follows:

When file selection mode is specified
- File selection area
- Path setting area
- Mode selection area
- Function buttons

When function selection mode is specified
- Function selection area
- Path setting area
- Mode selection area
- Function buttons

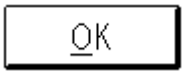
The function of each component is described below.

(1) File selection area



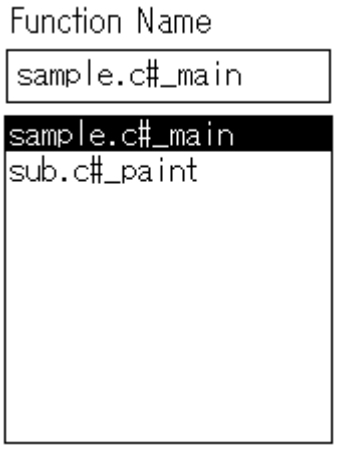
Specify the name of the source file to be displayed within the Source window.

You can select a source file from the list by clicking it. The selected file name is highlighted and displayed in the area above the list. The default extension for a source file name is **.C**.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the 

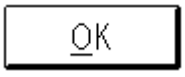
button.

(2) Function selection area



Specify the name of the function to be displayed within the Source window.

You can select a function name from the list by clicking it. The selected function name is highlighted and displayed in the area above the list.

Double-clicking a function name in the list has the same effect as selecting the function name and clicking the 

button.

(3) Path setting area

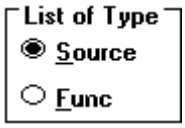
Specify the path of the source file to be displayed within the Source window.

Double-clicking a path name in the list displays the source files under that path, in the source file list.

Directories and drives are distinguished in the list as follows:

[xxx]: Directory name
[-x-]: Drive name

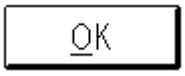
(4) Mode selection area



Select either of the following source selection modes:

Source Selects a source file.
Func Selects a function.

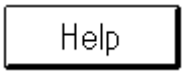
Buttons



Displays the selected source file or function within the Source window.

Closes the source file select dialog box.

Ignores any selections and resets the initial state.



Opens the help window.

Source window

View window

Outline

The Source window is used to display the source text.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Source Text...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **S** **keys**, in this order.
- In the tool bar, click the  button.

The jump function can be used to display the source text from other windows. Use of the jump function enables displaying of the target source and source lines quickly. To use the jump function, select the pointer and follow the steps below.

① Select **Jump→Source Text...** from the **menu bar**.

② Press the **GRPH**, **J**, **and** **S** **keys**, in this order.

③ Press the **CTRL**+**U** **keys**.

The following table lists jump functions.

Window	Pointer	Operation method		
		①	②	③
Assemble window	Address view area	○	○	○
Memory window	Address view area	○	○	○
Trace View window	Trace result view area	○	○	○
Coverage window	Address view area	○	○	○
Stack window	Stack frame number view area	○	○	○
Event manager	Event	○	○	--
Register window	Register	○	○	--

Window

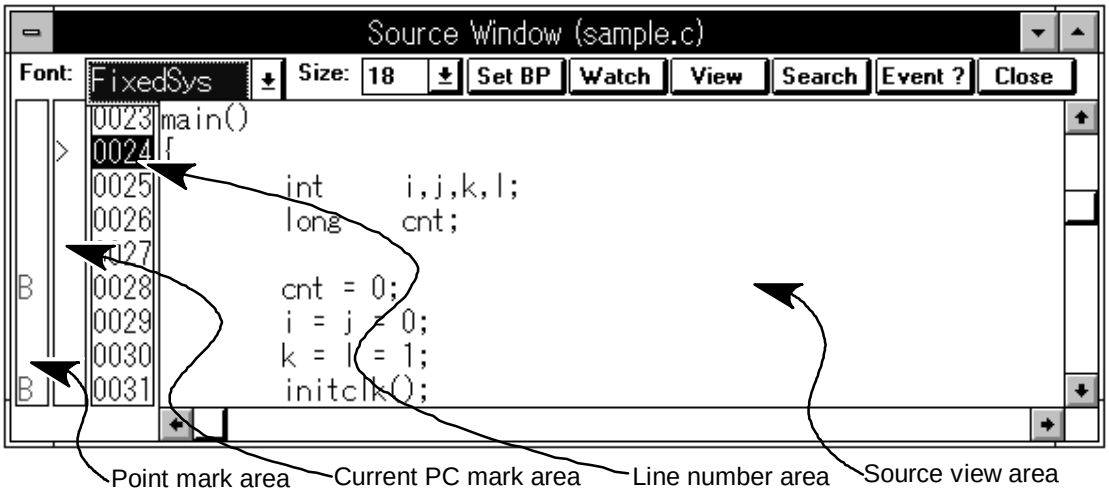


Fig. 5-10 Source Window

Description

The Source window consists of the following components:

- Font specification area
- Point mark area
- Current PC mark area
- Line number area
- Source view area
- Function buttons

The function of each component is described below.

(1) Font specification area

Specify the font and character size for the source text to be displayed in the source view area.

Font: Specify the font (Default: FixedSys)
Size: Specify the character size.

(2) Point mark area

The point mark area displays the event setting conditions. You can set or delete break points and software break points using this area.

a. Break point set/delete function

You can set or delete break points by clicking this area, as listed in the table below.

Clicked location	Color	Clicked button	Operation
On the B mark	Red or black	Left button	Deletes the break point.
	Blue	Right button	Deletes the software break point.
On a mark other than B or where no mark is indicated	---	Left button	Sets a break point.
	---	Right button	Sets a software break point.

b. Event display function

This area displays the setting condition of each type of events. When an execution event or access fetch event is set for a source line, the mark corresponding to the event type is displayed on the left of the source line.

Mark	Description
E	An event condition is set.
L	The final phase of an event link is set.
B	A break event is set.
T	A trace event is set.
Ti	A timer event is set.
S	A snapshot event is set.
U	A stub event is set.
A	Two or more events are set.

(3) Current PC mark area

The current PC mark area displays a mark (>) indicating the current value in the program counter (PC). Clicking this mark and holding the mouse button will display the content at the address indicated by the PC in a pop-up window.

(4) Line number area

The line number area displays the line numbers of the source text. You can also perform the following five functions using this area:

a. Go & Come function

This function executes the user program up to the selected line number. When the user program is being executed in this mode, currently set break events do not occur.

This function is used as follows:

1. Select the line number up to which the program will be executed.
2. In the main window, select **Execute→Go & Come** from the **menu bar** or press the **GRPH**, **X**, and **M** keys, in this order.

b. Break event set function

This function sets an execution break event at the first address corresponding to the selected line number.

This function is used as follows:

1. Select the line number for which a break event will be set.
2. In the main window, select **Execute→Set BP** from the **menu bar** or press the **GRPH**, **X**, and **B** keys, in this order. Or, press the **CTRL**+**B** short cut keys.

c. Program counter set function

This function sets the first address corresponding to the selected line number in the program counter (PC).

This function is used as follows:

1. Select a line number.
2. In the main window, select **Execute→Set PC** from the **menu bar** or press the **GRPH**, **X**, and **E** keys, in this order.

d. Jump function

The jump function causes a jump to the Assemble or Memory window, with the first address corresponding to the selected line number being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

This function is used as follows (when jumping to the Assemble window):

1. Select a line number.
2. In the main window, select **Jump→Assemble...** from the **menu bar** or press the **GRPH**, **J**, and **A** keys, in this order. Or, press the **CTRL**+**A** short cut keys.

e. Window linkage function

This function indicates the linkage between the source text and other windows (Assemble, Memory, and Trace View), using line numbers. The line numbers subject to linkage are displayed in reverse video.

(5) Source view area

The source view area displays the source text. Double-clicking or dragging a displayed symbol selects the symbol.

Buttons

Set BP	Sets a break point for the selected source text line.
Watch	Opens the Variable window to display the values of variables.
View	Opens the Variable View dialog box to display the value of a specified variable.
Search	Opens the Find dialog box to search the source text for a character string.
Event ?	When a mark is set for the selected source text line, opens the Event Manager to display details of the setting. When no mark is set, performs no operation.
Close	Closes the Source window.

Icon

The Source window can be reduced to the following icon by clicking the button on the title bar:

Source Window
(sample.c)

Find dialog box

Auxiliary dialog box (Modeless)

Outline

The Find dialog box is used to search for data. The results of search are reflected to the window that calls the dialog box.

When the Find dialog box is called from within the Source window, the file is searched.
When the Find dialog box is called from within the Assemble window, the disassembled text is searched.
When the Find dialog box is called from within the Memory window, memory is searched.
When the Find dialog box is called from within the Coverage window, the coverage view contents are searched.

This dialog box can be opened in any of the following ways:

- In the main window, select **View→Search...** from the **menu bar**.
- In the main window, press the **GRPH**, **V**, **and** **S** **keys**, in this order.
- In the Source window, click the Search button.
- In the Assemble window, click the Search button.
- In the Memory window, click the Search button.
- In the Coverage window, click the Search button.

Window

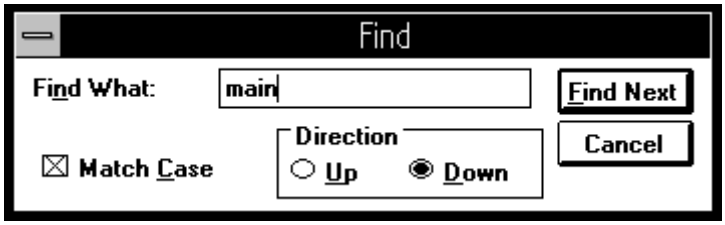


Fig. 5-11 Find Dialog Box

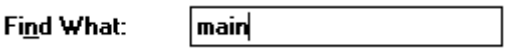
Description

The Find dialog box consists of the following components:

- Search data specification area
- Search condition specification area
- Search direction specification area
- Function buttons

The function of each component is described below.

(1) Search data specification area



Specify the data to be searched for. If you have selected a string within the calling window, that string is displayed in the area. You can modify the string, as required, using a keyboard.

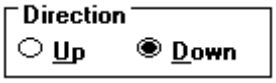
(2) Search condition specification area

Specify whether the search is case-sensitive, using the radio buttons. The default is case-sensitive search.

Not case-sensitive

Case-sensitive

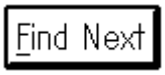
(3) Search direction specification area



Specify the direction of search, upward or downward.

Up: Upward
Down: Downward

Buttons



Searches for the specified data according to the specified conditions.

Stops search. The Cancel button is changed to the Stop button during search.

Closes the Find dialog box. This button is changed to the Stop button during search.

Symbol to Address dialog box

Auxiliary dialog box (Modeless)

Outline

The Symbol to Address dialog box is used to display the address assigned to a variable.

This dialog box can be opened in either of the following ways:

- In the main window, select **View→Sym To Adr...** from the **menu bar**.
- In the main window, press the **GRPH**, **V**, **and** **Y** **keys**, in this order.

Window

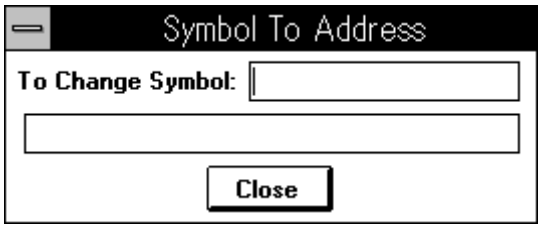


Fig. 5-12 Symbol to Address Dialog Box

Description

The Symbol to Address dialog box consists of the following components:

- Variable specification area
- Variable address view area
- Function buttons

The function of each component is described below.

(1) Variable specification area

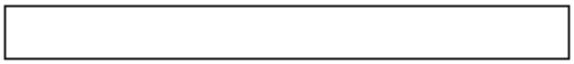
Specify a function name or variable name or line number to be converted to the address. After entering data, press the ↵ key to display the address in the variable address display area. The table below lists how to specify a variable or line number.

Function or variable	_fnc file#_fnc (for static function or variable)
SFR	sfrname
Line number in source text	file:no

fnc: Function or variable name sfrname: SFR name
file: File name no: Line number

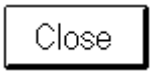
When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

(2) Variable address view area



This area displays address assigned to the variable specified in the variable specification area.

Buttons



Closes the dialog box.

Variable View dialog box

Auxiliary dialog box (Modal)

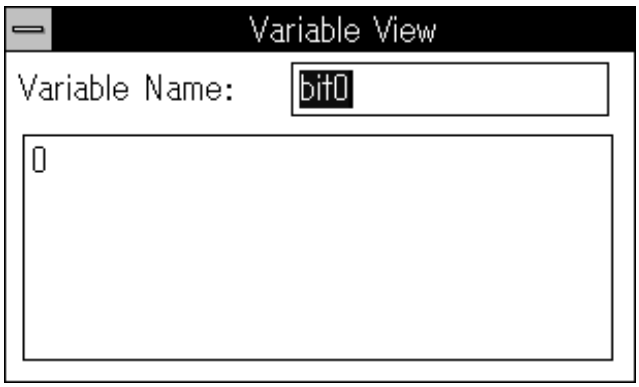
Outline

The Variable View dialog box is used to temporarily display the value of the variable which is specified in the Source window.

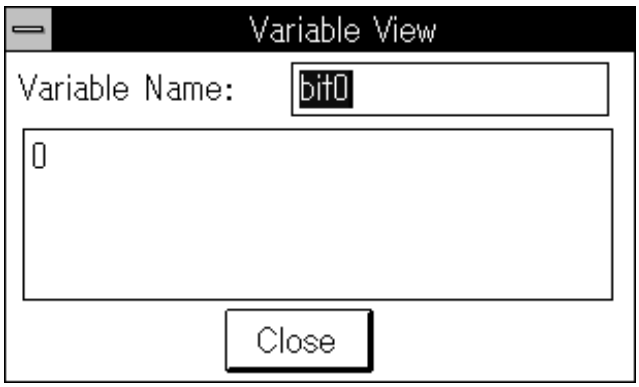
This dialog box can be opened in any of the following ways:

- Select a variable in the Source window, then select **View→View Variable...** from the **menu bar**.
- Select a variable in the Source window, then press the **GRPH**, **V**, **and** **V** **keys**, in this order.
- Select a variable in the Source window, then click the View button.

Window



When opened using button on Source window



When opened using menu bar

Fig. 5-13 Variable View Dialog Box

Description

The Variable View dialog box consists of the following components:

- Variable specification area
- Variable value view area
- Function buttons

The function of each component is described below.

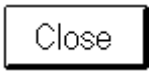
(1) Variable specification area

The name of the variable selected in the Source window is specified by default. To display another variable, enter the variable name using a keyboard.

(2) Variable value view area

This area displays the value of the variable specified in the variable specification area.

Buttons



Closes the dialog box.

Variable window

View/setting window

Outline

The Variable window is used to display and modify the values of variables specified in the Source window.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Variable...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **V** **keys**, in this order.
- Select a variable in the Source window, then select **View→Watch Variable...** from the **menu bar**.
- Select a variable in the Source window, then press the **GRPH**, **V**, **and** **W** **keys**, in this order.
- Select a variable in the Source window, then click the Watch button.

Window

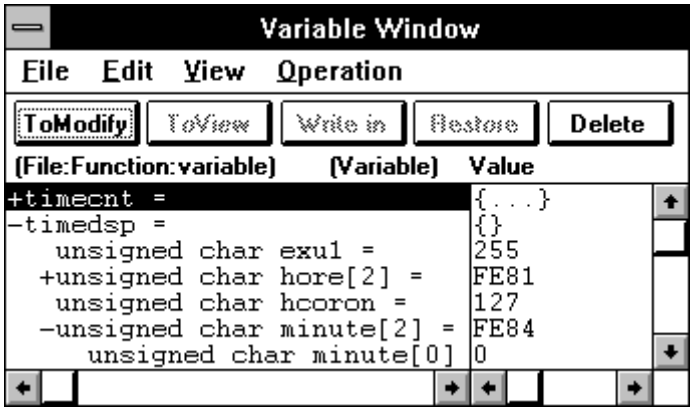


Fig. 5-14 Variable Window

Description

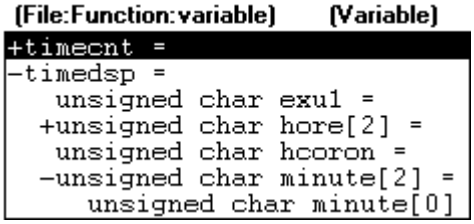
The Variable window is used to display and modify variables. Each time a variable is specified, the variable is added to the display area of the window. When a variable already being displayed is specified, however, the variable is not added. This window can be set to view mode or modify mode.

The Variable window consists of the following components:

- Menu bar
- Function buttons
- Variable name view area
- Variable value view/setting area

The function of each component is described below.

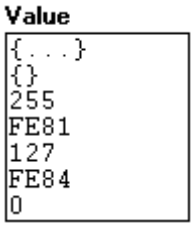
(1) Variable name view area



This area displays the names of variables.

A prefix "+" attached to a variable means that the variable is a pointer. Double-clicking it displays the data pointed to by the variable in the variable value view/setting area and toggles the prefix to "-".

(2) Variable value view/setting area



This area displays the values of variables.

For a pointer type variable, the displayed value is an address or data item.

Buttons

ToModify

Switches the window to modify mode. This button can be clicked only in view mode. Clicking this button enables variables to be modified. When the window is placed in modify mode, the window is highlighted and this button is disabled.

To modify the value of a variable, click the current value of the variable to display a text cursor, then enter a new value using a keyboard. Clicking the Write in button writes the new value into the target device.

ToView

Switches the window to view mode. This button can be clicked only in modify mode. When the window is placed in view mode, the window is not highlighted any more and this button is disabled.

Write in

Writes the new value of a variable into the target device.

Restore

Cancels modification. Clicking this button restores the initial values of variables which have been modified in modify mode. If the Write in button has been clicked, modification before clicking the button is not canceled.

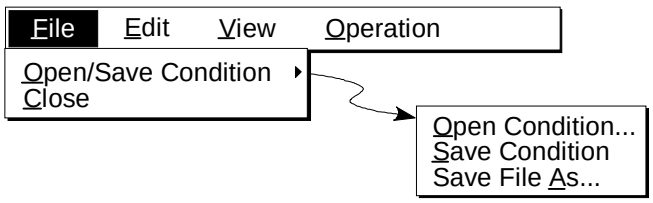
Delete

Removes a specified variable from the Variable window.

Menu bar

Double-clicking the mouse button on an item of the menu bar opens the corresponding pull-down menu.

(a) File

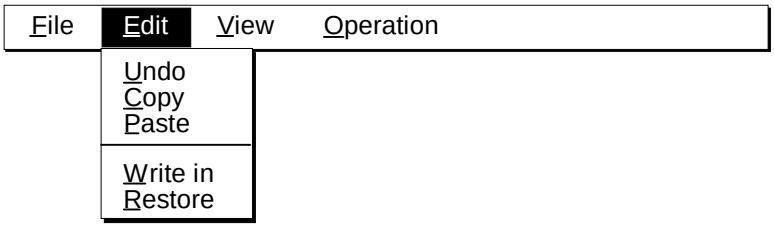


Open/Save Condition ▸ Loads or saves variable values.

- **Open Condition** — Opens the selected file for reference. The view file load dialog box is opened.
- **Save Condition** — Saves the contents of the window into a view file.
- **Save File As...** — Saves the contents of the window into a view file. The view file save dialog box is opened.

Close — Closes the Variable window.

(b) Edit



Undo Cancels the most recent editing.

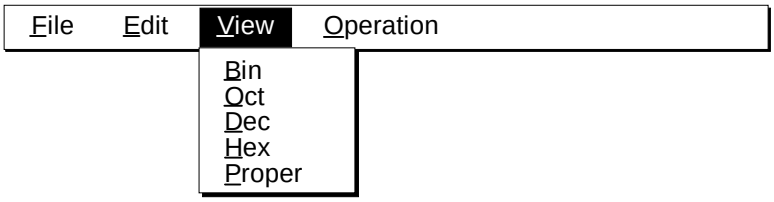
Copy Copies a selected character string into the clipboard buffer.

Paste Pastes the contents of the clipboard buffer at the point to which the text cursor is positioned.

Write in Writes the modified contents into the target device.

Restore Cancels the modified contents.

(c) View



Bin Displays variable values in binary format.

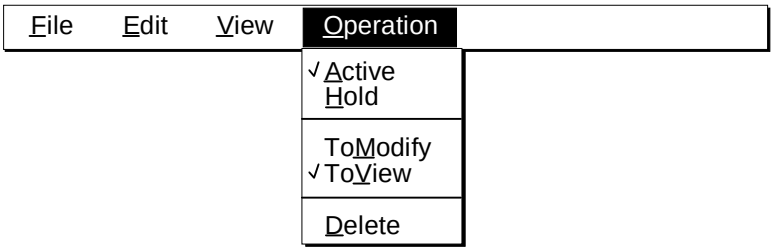
Oct Displays variable values in octal format.

Dec Displays variable values in decimal format.

Hex Displays variable values in hexadecimal format.

Proper Displays variable values in default format for each variable.

(d) Operation



Active Sets the Variable window to the active state.

Hold Sets the Variable window to the hold state.

ToModify Sets the Variable window to modify mode.

ToView Sets the Variable window to view mode.

Delete Removes a specified variable from the Variable window.

Add Variable dialog box

Auxiliary dialog box (Modeless)

Outline

The Add Variable dialog box is used to add variables to be displayed in the Variable window. This dialog box can be opened in either of the following ways:

- In the main window, select **View → Add Variable...** from the **menu bar**.
- In the main window, press the **GRPH**, **V**, **and** **I** **keys**, in this order.

Window

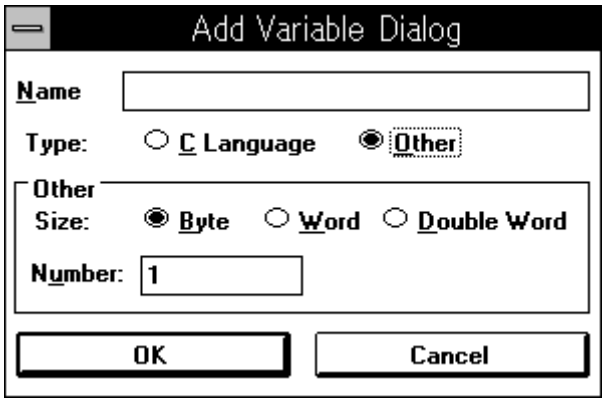


Fig. 5-15 Add Variable Window

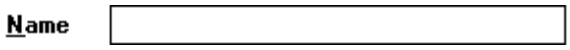
Description

The Add Variable dialog box consists of the following components:

- Variable name specification area
- Variable type selection area
- Variable size specification area
- Function buttons

The function of each component is described below.

(1) Variable name specification area



This area is used to specify the name of a variable to be added.

Variable	_fnc file#_fnc
SFR	sfrname

fnc: Function, variable name sfrname: SFR name
file: File name

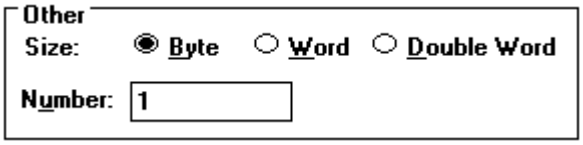
Prefix the variable name with an underscore (_).
Separate the file and variable names using a sharp (#).

(2) Variable type selection area

Select the language type of a variable specified in the variable name specification area.

C Language	Variable defined in C
Other	Variable defined in a language other than C (such as SFR or assembly variable)

(3) Variable size specification area



Specify the size and quantities of variables to be added. This area cannot be used if C is selected from the variable type selection area.

a. Size

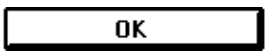
Specify the size of a variable by selecting one of the following:

Byte
Word
Double Word

b. Number

Specify the quantity of variables to be specified.

Buttons



Adds variables in the Variable window.

Closes the Add Variable dialog box.

Local Variable window

View/setting window

Outline

The Local Variable window is used to display and modify the values of local variables in the current function.

This window can be opened in either of the following ways:

- In the main window, select **Browse→Local Variable...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **L** **keys**, in this order.

Window

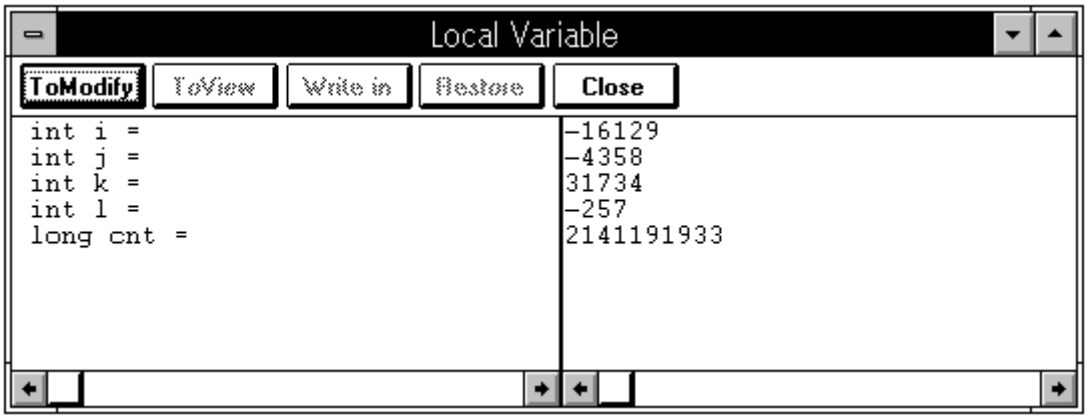


Fig. 5-16 Local Variable Window

Description

The Local Variable window is used to display and modify local variables. This window automatically displays the local variables in the current function. You cannot additionally display a variable.

You can move, using the mouse, the boundary between the local variable name view area and local variable value view/setting area. Positioning the mouse pointer to the boundary changes the pointer from ↖ to ⇔. Then, drag & drop the boundary to the desired position.

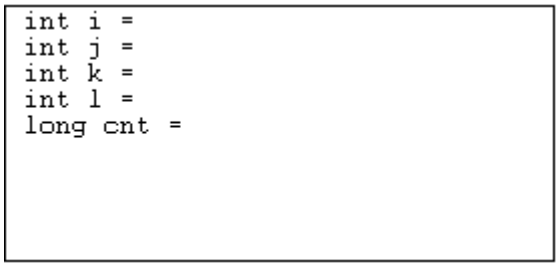
This window can be set to view mode or modify mode.

The Local Variable window consists of the following components:

- Local variable name view area
- Local variable value view/setting area
- Function buttons

The function of each component is described below.

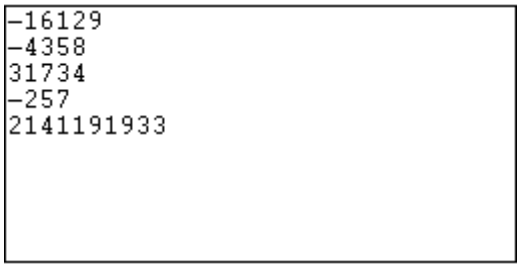
(1) Local variable name view area



This area displays the names of local variables.

A prefix "+" attached to a variable means that the variable is a pointer. Double-clicking it displays the data pointed to by the variable in the local variable value view/setting area and toggles the prefix to "-".

(2) Local variable value view/setting area



This area displays the values of local variables.

For a pointer type variable, the displayed value is an address or data item.

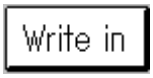
Buttons

Switches the window to modify mode. This button can be clicked only in view mode. Clicking this button enables variables to be modified. When the window is placed in modify mode, the window is highlighted and this button is disabled.

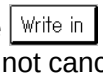
To modify the value of a variable, click the current value of the variable to display a text cursor, then enter a new value using a keyboard. Clicking the 

button writes the new value into the target device.

Switches the window to view mode. This button can be clicked only in modify mode. When the window is placed in view mode, the window is not highlighted any more and this button is disabled.



Writes the new value of a variable into the target device.

Cancels modification. Clicking this button restores the initial values of variables which have been modified in modify mode. If the 

button has been clicked, modification before clicking the button is not canceled.

Delete Removes a specified variable from the Local Variable window.

Icon

The Local Variable window can be reduced to the following icon by clicking the button on the title bar:

Local Variable

Addressing dialog box

Specification dialog box (Modal)

Outline

The addressing dialog box is used to specify the start address for memory view, disassemble view, or coverage view.

This dialog box can be opened in any of the following ways:

When the Assemble window is the current window

- In the main window, select **Browse→Assemble...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **A** **keys**, in this order.
- In the tool bar, click the  button.

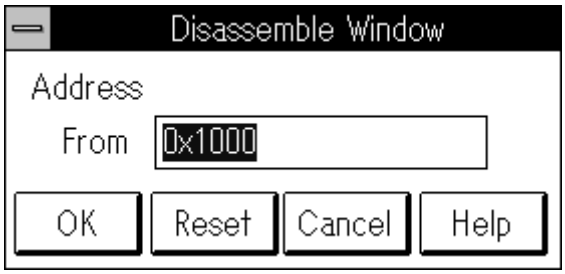
When the Memory window is the current window

- In the main window, select **Browse→Memory...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **M** **keys**, in this order.
- In the tool bar, click the  button.

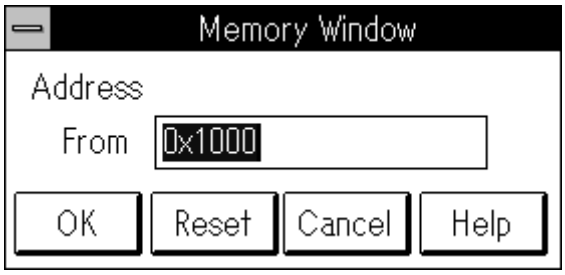
When the Coverage window is the current window

- In the main window, select **Browse→Coverage→View...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **C**, **and** **V** **keys** in this order.

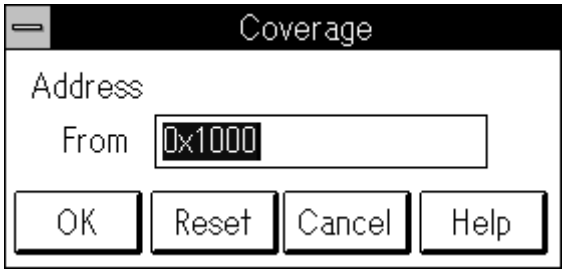
Window



When the Assemble window is a target of operation



When the Memory window is a target of operation



When the Coverage window is a target of operation

Fig. 5-17 Addressing Dialog Box

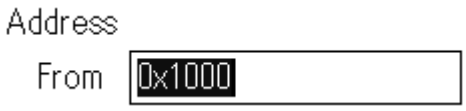
Description

The addressing dialog box is used to specify the view start address. The dialog box consists of the following components:

- Address specification area
- Function buttons

The function of each component is described below.

(1) Address specification area



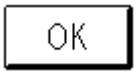
Specify the address. The current value of the PC is displayed by default. You can enter an address using a keyboard, as required. Symbols can also be used to specify the address, as follows:

Function or variable	_fnc file#_fnc (for static function or variable)
Line number in source text	file:no

fnc: Function or variable name
file: File name no: Line number

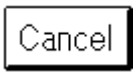
When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

Buttons



Starts memory view, disassemble view, or coverage view, from the specified address.

Resets the address to the default value.



Closes the addressing dialog box.

Opens the help window.

Assemble window

View/setting window

Outline

The Assemble window is used to display the disassembled text of a program. You can also perform on-line assembly using this window.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Assemble...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **A** **keys**, in this order.
- In the tool bar, click the  button.

The jump function can be used to display the assembly line from other windows. Use of the jump function enables displaying of the target assembly line quickly. To use the jump function, select the pointer and follow the steps below.

① Select **Jump→Assemble...** from the **menu bar**.

② Press the **GRPH**, **J**, **and** **A** **keys**, in this order.

③ Press the **CTRL**+**A** **keys**.

The following table lists jump functions.

Window	Pointer	Operation method		
		①	②	③
Source window	Line number area	○	○	○
Memory window	Address view area	○	○	○
Trace View window	Trace result view area	○	○	○
Coverage window	Address view area	○	○	○
Stack window	Stack frame number view area	○	○	○
Event manager	Event	○	○	--
Register window	Register	○	○	--

Window

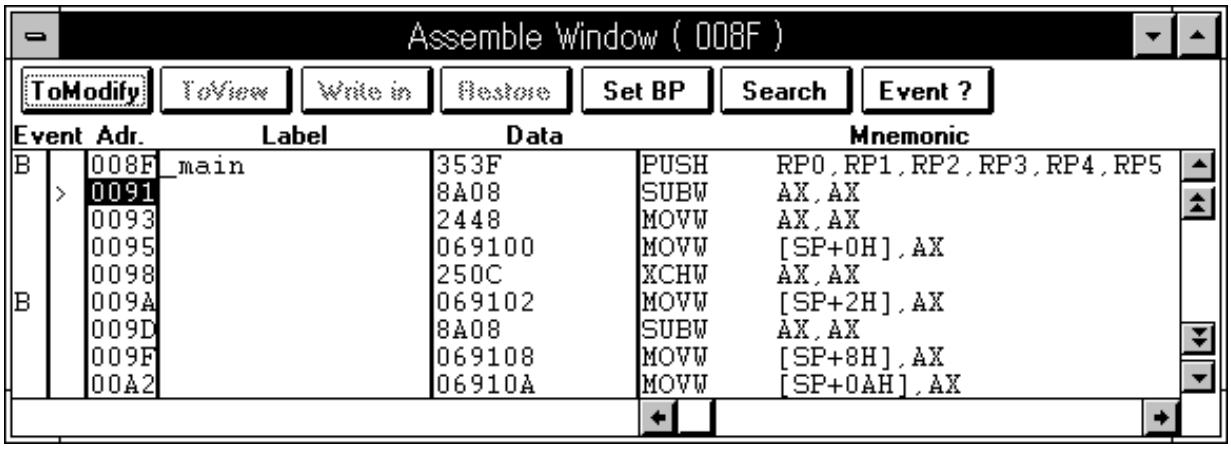


Fig. 5-18 Assemble Window

Description

The Assemble window displays the disassembled text of a program and enables on-line assembly. This window can be set to view mode or modify mode.

The Assemble window consists of the following components:

- Point mark area
- Current PC mark area
- Address view area
- Label view area
- Data view area
- Mnemonic view/modification area
- Function buttons

The function of each component is described below.

(1) Point mark area

The point mark area displays the event setting conditions. You can set or delete break points using this area.

a. Break point set/delete function

You can set or delete break points by clicking this area, as listed in the table below.

Clicked location	Color	Clicked button	Operation
On the B mark	Red or black	Left button	Deletes the break point.
	Blue	Right button	Deletes the software break point.
On a mark other than B or where no mark is indicated	---	Left button	Sets a break point.
	---	Right button	Sets a software break point.

b. Event display function

This area displays the setting condition of each type of events. When an execution event or access fetch event is set for an assembly line, the mark corresponding to the event type is displayed on the left of the assembly line.

Mark	Description
E	An event condition is set.
L	The final phase of an event link is set.
B	A break event is set.
T	A trace event is set.
Ti	A timer event is set.
S	A snapshot event is set.
U	A stub event is set.
A	Two or more events are set.

(2) Current PC mark area

The current PC mark area displays a mark (>) indicating the current value in the program counter (PC). Clicking this mark and holding the mouse button will display the content at the address indicated by the PC in a pop-up window.

(3) Address view area

This area displays the disassembly start address. You can also perform the following five functions using this area:

a. Go & Come function

This function executes the user program up to the selected address. When the user program is being executed in this mode, currently set break events do not occur.

This function is used as follows:

1. Select the address up to which the program will be executed.
2. In the main window, select **Execute→Go & Come** from the **menu bar** or press the **GRPH**, **X**, and **M** keys, in this order.

b. Break event set function

This function sets an execution break event at the selected address.

This function is used as follows:

1. Select the address for which a break event will be set.
2. In the main window, select **Execute→Set BP** from the **menu bar** or press the **GRPH**, **X**, and **B** keys, in this order. Or, press the **CTRL** + **B** short cut keys.

c. Program counter set function

This function sets the selected address in the program counter (PC).

This function is used as follows:

1. Select an address.
2. In the main window, select **Execute→Set PC** from the **menu bar** or press the **GRPH**, **X**, and **E** keys, in this order.

d. Jump function

The jump function causes a jump to the Source or Memory window, with the selected address being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

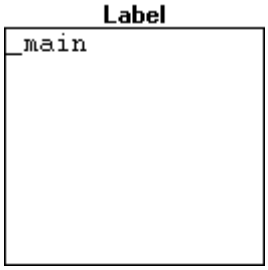
This function is used as follows (when jumping to the Source window):

1. Select an address.
2. In the main window, select **Jump→SourceText...** from the **menu bar** or press the **GRPH**, **J**, and **S** keys, in this order. Or, press the **CTRL**+**U** shortcut keys.

e. Window linkage function

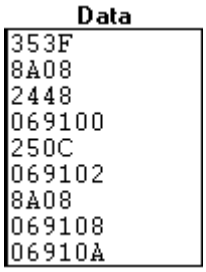
This function indicates the linkage between the disassembled text and other windows (Source, Memory, and Trace View), using addresses. The addresses subject to linkage are displayed in reverse video.

(4) Label view area



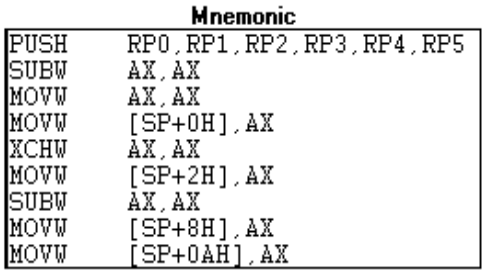
This area displays labels.

(5) Data view area



This area displays mnemonic data.

(6) Mnemonic view/modification area



This area displays the disassembled text. In modify mode, data in this area can be directly modified.

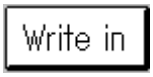
Note the following: If the mnemonic after modification is longer than the previous one, the mnemonic on the next line is corrupted. If the mnemonic after modification is shorter than the previous one, the mnemonic on the next line becomes invalid.

Buttons

Switches the window to modify mode. This button can be clicked only in view mode. Clicking this button enables mnemonics to be modified. When the window is placed in modify mode, the window is highlighted and this button is disabled.

To modify a mnemonic, click the mnemonic to display a text cursor, then enter a new mnemonic using a keyboard. Clicking the Write in button writes the new mnemonic into the target device.

Switches the window to view mode. This button can be clicked only in modify mode. When the window is placed in view mode, the window is not highlighted any more and this button is disabled.



Writes the new mnemonics into the target device.

Cancels modification. Clicking this button restores the initial states of mnemonics which have been modified in modify mode. If the Write in button has been clicked, modification before clicking the button is not canceled.

Sets a break point for the selected assembly line.

Opens the Find dialog box to search the disassembled text for a mnemonic.

When an event mark is set for the selected assembly line, opens the Event Manager to display details of the setting. When no mark is set, performs no operation.

Icon

The Assemble window can be reduced to the following icon by clicking the button on the title bar:

Assemble Window

Memory window

View/setting window

Outline

The Memory window is used to display and modify the contents of memory.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Memory...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **M** **keys**, in this order.
- In the tool bar, click the  button.

The jump function can be used to display the memory contents from other windows. Use of the jump function enables displaying of the target memory contents quickly. To use the jump function, select the pointer and follow the steps below.

① Select **Jump→Memory...** from the **menu bar**.

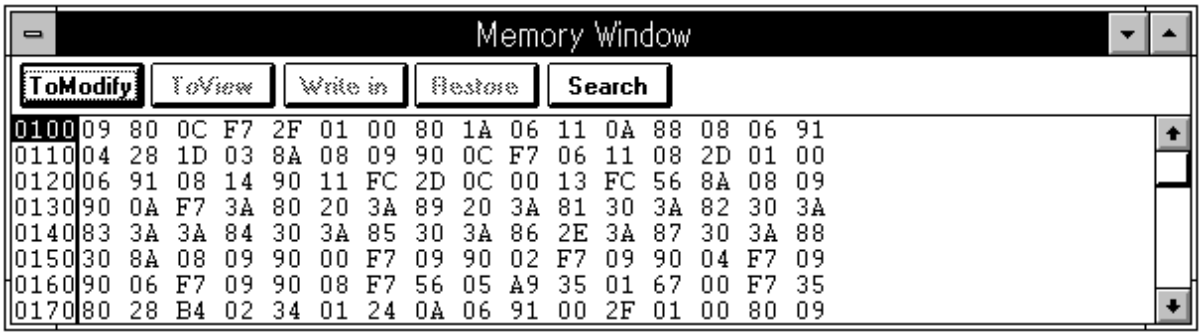
② Press the **GRPH**, **J**, **and** **M** **keys**, in this order.

③ Press the **CTRL**+**M** **keys**.

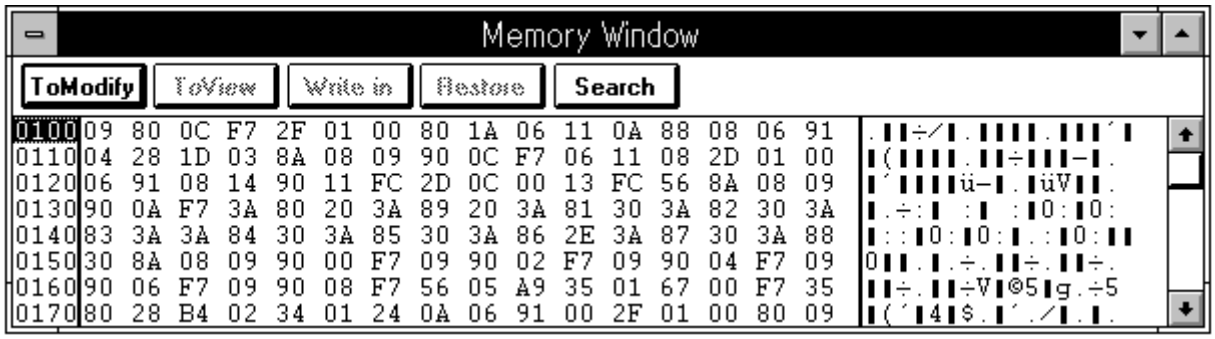
The following table lists jump functions.

Window	Pointer	Operation method		
		①	②	③
Source window	Line number area	○	○	○
Assemble window	Address view area	○	○	○
Trace View window	Trace result view area	○	○	○
Coverage window	Address view area	○	○	○
Event manager	Event	○	○	--
Register window	Register	○	○	--

Window



Without ASCII view



With ASCII view

Fig. 5-19 Memory Window

Description

The Memory window displays the contents of memory and enables the modification of the contents. This window can be set to view mode or modify mode.

The Memory window consists of the following components:

- Address view area
- Memory view area
- ASCII view area
- Function buttons

The function of each component is described below.

(1) Address view area

This area displays memory addresses. You can also perform the following two functions using this area:

a. Jump function

The jump function causes a jump to the Source or Assemble window, with the selected address being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

This function is used as follows (when jumping to the Source window):

1. Select an address.
2. In the main window, select **Jump→SourceText...** from the **menu bar** or press the **GRPH**, **J**, and **S** keys, in this order. Or, press the **CTRL**+**U** shortcut keys.

b. Window linkage function

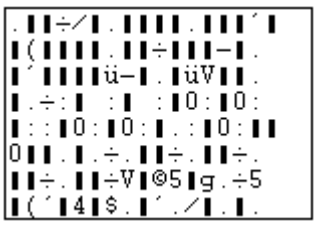
This function indicates the linkage between the memory contents and other windows (Source, Assemble, and Trace View), using addresses. The addresses subject to linkage are displayed in reverse video.

(2) Memory view area

```
09 80 0C F7 2F 01 00 80 1A 06 11 0A 88 08 06 91
04 28 1D 03 8A 08 09 90 0C F7 06 11 08 2D 01 00
06 91 08 14 90 11 FC 2D 0C 00 13 FC 56 8A 08 09
90 0A F7 3A 80 20 3A 89 20 3A 81 30 3A 82 30 3A
83 3A 3A 84 30 3A 85 30 3A 86 2E 3A 87 30 3A 88
30 8A 08 09 90 00 F7 09 90 02 F7 09 90 04 F7 09
90 06 F7 09 90 08 F7 56 05 A9 35 01 67 00 F7 35
80 28 B4 02 34 01 24 0A 06 91 00 2F 01 00 80 09
```

This area displays the contents of memory. You can modify the contents in modify mode.

(3) ASCII view area



This area displays the contents of memory using ASCII code. You can modify each ASCII character to modify the corresponding memory contents in modify mode.

Selecting **View→Memory→Ascii** from the **menu bar** toggles whether ASCII code is displayed.

Buttons

ToModify

Switches the window to modify mode. This button can be clicked only in view mode. Clicking this button enables memory contents to be modified. When the window is placed in modify mode, the window is highlighted and this button is disabled.

To modify the content at a memory location, click the memory location to display a text cursor, then enter a new content using a keyboard. Clicking the Write in button writes the new content into the target device.

When ASCII code is displayed in the ASCII view area, memory contents can be modified using ASCII code.

ToView

Switches the window to view mode. This button can be clicked only in modify mode. When the window is placed in view mode, the window is not highlighted any more and this button is disabled.

Write in

Writes the new memory contents into the target device.

Restore

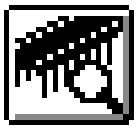
Cancels modification. Clicking this button restores the initial contents of the memory locations which have been modified in modify mode. If the Write in button has been clicked, modification before clicking the button is not canceled.

Search

Opens the Find dialog box to search memory for a string.

Icon

The Memory window can be reduced to the following icon by clicking the button on the title bar:



Memory Window

Memory Fill dialog box

Auxiliary dialog box (Modal)

Outline

The Memory Fill dialog box is used to initialize memory by filling it with specified code.

This dialog box can be opened in either of the following ways when the current window is Memory window:

- In the main window, select **Edit→Memory→Memory Fill...** from the **menu bar**.
- In the main window, press the **GRPH**, **E**, **M**, **and F** **keys**, in this order.

Window

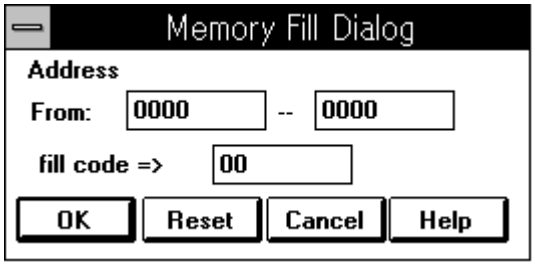


Fig. 5-20 Memory Fill Dialog Box

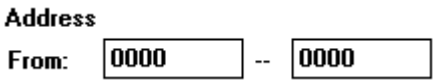
Description

The Memory Fill dialog box is used to initialize memory. The dialog box consists of the following components:

- Address range specification area
- Code specification area
- Function buttons

The function of each component is described below.

(1) Address range specification area

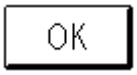


Specify the address range to be initialized in memory, as initialization start address -- initialization end address.

(2) Code specification area

Specify the code with which memory will be filled. A string of up to 16 bytes can be specified.

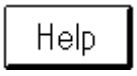
Buttons



Initializes memory.

Ignores any selections and resets the initial state.

Closes the Memory Fill dialog box.



Opens the help window.

Memory Copy dialog box

Auxiliary dialog box (Modal)

Outline

The Memory Copy dialog box is used to copy the contents of memory from one location to another.

This dialog box can be opened in either of the following ways when the current window is Memory window:

- In the main window, select **Edit→Memory→Memory Copy...** from the **menu bar**.
- In the main window, press the **GRPH**, **E**, **M**, **and C keys**, in this order.

Window

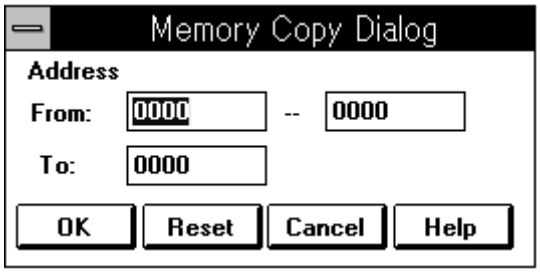


Fig. 5-21 Memory Copy Dialog Box

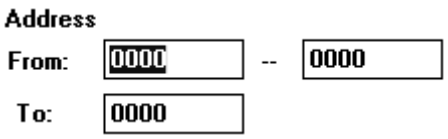
Description

The Memory Copy dialog box is used to copy the contents of memory from one location to another. The dialog box consists of the following components:

- Address range specification area
- Function buttons

The function of each component is described below.

(1) Address range specification area

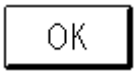


Specify the address range to be copied and the destination address to which the range will be copied.

From: Specify the address range to be copied, as copy start address -- copy end address .

To: Specify the destination address.

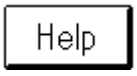
Buttons



Copies the specified contents of memory.

Ignores any selections and resets the initial state.

Closes the Memory Copy dialog box.



Opens the help window.

Memory Compare dialog box

Auxiliary dialog box (Modal)

Outline

The Memory Compare dialog box is used to compare the contents of memory between specified locations.

This dialog box can be opened in either of the following ways when the current window is Memory window:

- In the main window, select **Edit→Memory→Memory Compare...** from the **menu bar**.
- In the main window, press the **GRPH**, **E**, **M**, **and P** **keys**, in this order.

Window

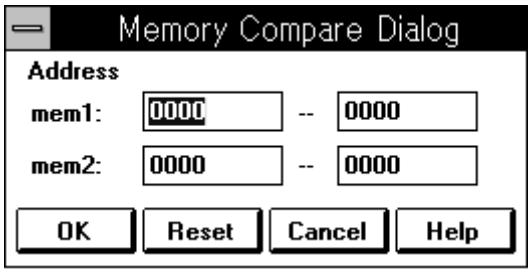


Fig. 5-22 Memory Compare Dialog Box

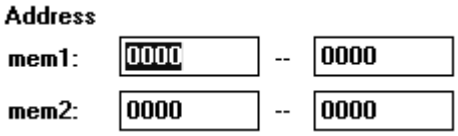
Description

The Memory Compare dialog box is used to compare the contents of memory between specified locations. The dialog box consists of the following components:

- Comparison range specification area
- Function buttons

The function of each component is described below.

(1) Comparison range specification area



Specify the address ranges to be compared.

mem1: Specify the source address range to be compared, as comparison start address --comparison end address.

mem2: Specify the destination address range to be compared, as comparison start address--comparison end address.

Buttons

OK — Compares the contents of memory in the specified ranges.

When no differences have been found during comparison, a confirmation dialog box appears.

When a difference has been found during comparison, the Memory Compare result dialog box appears.

The following is the confirmation dialog box which appears when no differences have been found:

Pressing the OK button ends memory comparison.

Reset — Ignores any selections and resets the initial state.

Cancel — Closes the Memory Compare dialog box.

Help — Opens the help window.

Memory Compare result dialog box

View dialog box (Modal)

Outline

The Memory Compare result dialog box is used to display the results of memory comparison.

This dialog box is displayed only when a difference has been found as a result of memory comparison performed in the Memory Compare dialog box. If no differences have been found, the confirmation dialog box is displayed, instead.

Window

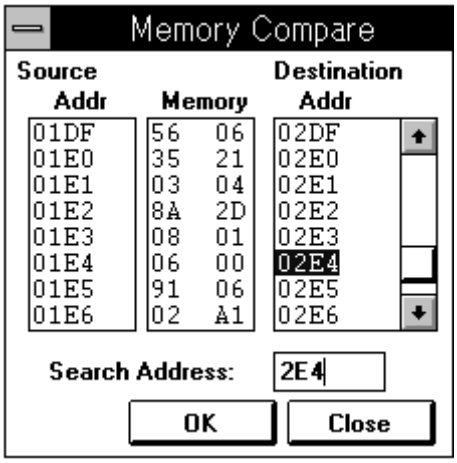


Fig. 5-23 Memory Compare Result Dialog Box

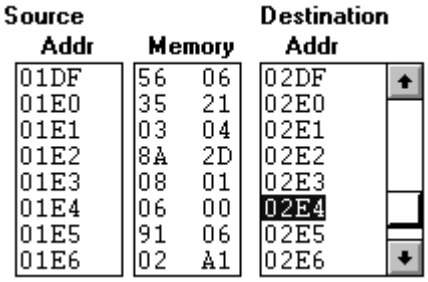
Description

The Memory Compare result dialog box is used to display the results of memory comparison. The dialog box consists of the following components:

- Comparison result view area
- Address search specification area
- Function buttons

The function of each component is described below.

(1) Comparison result view area



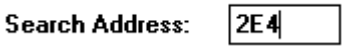
This area displays the results of memory comparison.

Source Addr
This area displays those addresses in the source range which contain different values from those at the corresponding addresses in the destination range. Double-clicking an address highlights the corresponding location in the Memory window.

Memory
This area displays the values which are different in the source and destination ranges. The values in the source range are displayed on the left. The values in the destination range are displayed on the right.

Destination Addr
This area displays those addresses in the destination range which contain different values from those at the corresponding addresses in the source range. Double-clicking an address highlights the corresponding location in the Memory window.

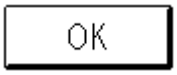
(2) Address search specification area



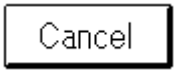
You can search for an address whose content you want to view. If the specified address is found, the address and its contents are displayed in the comparison result view area.

Entering an address using an keyboard starts searching for the address (pressing the ↵ (return) key is not necessary).

Buttons



Closes the Memory Compare result dialog box. The address which has been specified in the **Search Address:** area is highlighted in the Memory window.



Closes the Memory Compare result dialog box.

Stack window

View window

Outline

The Stack window displays the stack contents for the current user program.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Stack Trace...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **K** **keys**, in this order.
- In the tool bar, click the button.

Window

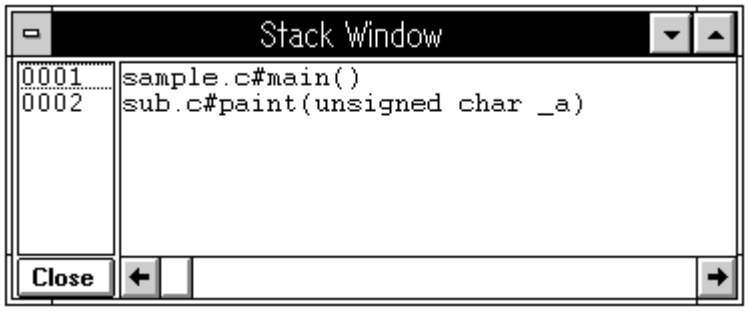


Fig. 5-24 Stack Window

Description

The Stack window consists of the following components:

- Stack frame number view area
- Stack contents view area
- Function buttons

The function of each component is described below.

(1) Stack frame number view area

The stack frame number view area displays the numbers assigned to the stack contents. Stack frame numbers are integers starting from one and increase as the nesting level increases. When a function calls another function, the stack frame number of the called function is larger than that of the calling function by one.

In addition to displaying stack frame numbers, this area can be used to perform the following function:

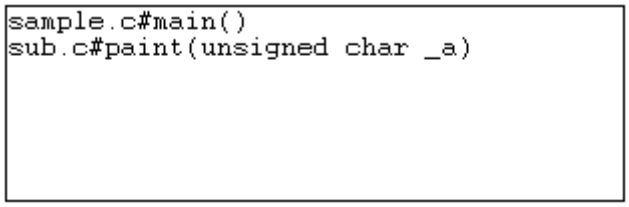
a. Jump function

The jump function causes a jump to the Source or Assemble window, with the first address of the function having the selected stack frame number being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

This function is used as follows (when jumping to the Source window):

1. Select a stack frame number.
2. In the Main window, select **Jump→SourceText...** from the **menu bar** or press the **GRPH**, **J**, and **S** keys, in this order. Or, press the **CTRL** + **U** shortcut keys.

(2) Stack contents view area



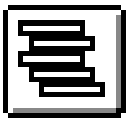
The stack contents view area displays the contents of the stack. Each stack content is represented as "file-name#function-name(parameter)". The file name is separated from the function name with a sharp (#).

Buttons

Closes the Stack window.

Icon

The Stack window can be reduced to the following icon by clicking the button on the title bar:



Stack Window

Caution

Stack contents may not be correctly displayed for functions which do not push a frame pointer (RP5) into the stack (such as noauto and norec) or when the -qf option has been specified to optimize compilation.

Event Set dialog box

Setting dialog box (Modeless)

Outline

The Event Set dialog box is used to register and display event conditions. Once the event conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→Event→EventSet...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **E**, **and** **E** **keys**, in this order.
- In the Event Manager, select **Operation→EventSet...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **E** **keys**, in this order.

Window

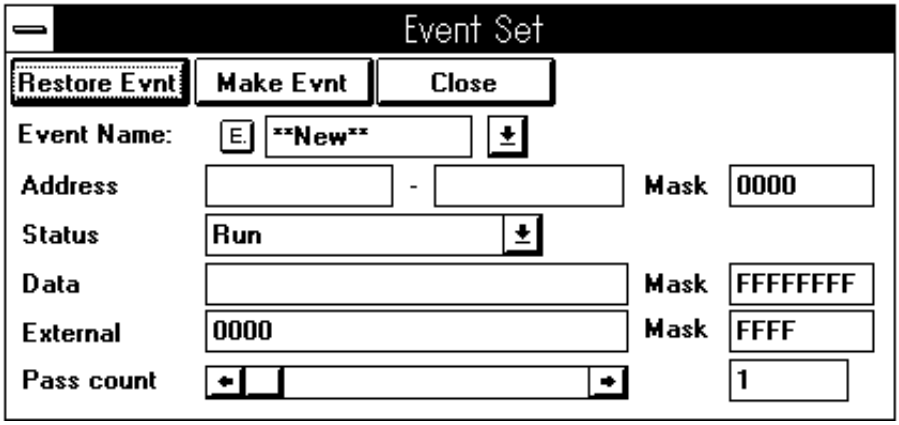


Fig. 5-25 Event Set Dialog Box

Description

The Event Set dialog box is used to register and display event conditions.

Up to 32,767 event conditions can be registered. Up to ten of these conditions can be simultaneously used as conditions for a break, timer, or tracer (three execution events and seven access events). A single event condition can be simultaneously used, for example, as a break, timer, trace, and event link condition.

The Event Set dialog box consists of the following components:

- Event name setting area
- Address setting area
- Status selection area
- Data setting area
- External sense data setting area
- Pass count setting area
- Function buttons

The function of each component is described below.

(1) Event name setting area

This area is used to specify an event name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the button displays a drop-down list, from which you can select an event name. You can also type in, using the keyboard, a new event name of up to eight characters.

(2) Address setting area

This area is used to specify address conditions.

Valid range: 0 ≤ Address value ≤ 0xffffff

Address conditions consist of an address and a mask value for that address. Set each value as follows:

a. Address

Enter addresses as [lower address] **-** [upper address]. Either a single address or address range can be set, as follows:

①Setting a single address
Specify an address in the lower address field only. Or, specify the same address in both the lower and upper address fields. A mask value can be specified.

②Setting an address range
Specify both the lower and upper addresses. No mask value can be specified.

Symbols can also be used to specify an address, as follows:

Function or variable	_fnc file**#**_fnc (for static function or variable)
SFR	sfrname
Line number in source text	file**:**no

fnc: Function or variable name sfrname: SFR name
file: File name no: Line number

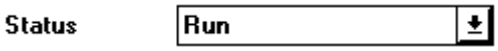
When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

b. Mask

When a single address is specified, a mask value can be set for the address. The default is 0x00000000 (no mask). **The logical OR of the specified address and mask value is used as the address condition.**

Example: When **Address** [0x4000] **-** [0x4000] and **Mask** [0xFF] are set
Addresses 0x4000 to 0x40FF satisfy the condition.
When **Address** [0x4000] **-** [0x4000] and **Mask** [0x101] are set
Addresses 0x4000, 0x4001, 0x4100, and 0x4101 satisfy the condition.

(3) Status selection area



This area is used to select the status of the event to be set. Selecting a status also determines whether the event is an execution or access event.

The table below lists the statuses.

Status	Event type	Description
Run	Execution event	Program execution
Fetch	Access event	Program fetch (including pre-fetch)
Program Read		Program data read
Program Write		Program data write
Program R/W		Program data read/write
Macro Read		Data read during a macro service
Macro Write		Data write during a macro service
Macro R/W		Data read/write during a macro service
Program/Macro Read		Data read
Program/Macro Write		Data write
Program/Macro R/W		Data read/write
VECT		Vector read by interrupts
ALL(No Condition)		All accesses

(4) Data setting area

This area is used to specify the data conditions.

Valid range: 0 ≤ Data ≤ 0xffff

Data conditions consist of a data value and mask value for that data. Set each value as follows:

a. Data

Enter a data value.

Symbols can also be used to specify a data value, as follows:

Function or variable	_fnc file#_fnc (for static function or variable)
SFR	sfrname
Line number in source text	file:no

fnc: Function or variable name sfrname: SFR name
file: File name no: Line number

When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

b. Mask

A mask value can be set for the data value. The default is 0xffff (the data condition is ignored; all data satisfies the condition). **The logical OR of the specified data value and mask value is used as the data condition.**

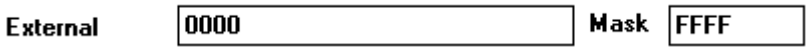
Example: When **Data** 0x4000 and **Mask** 0xFF are set

Data values 0x4000 to 0x40FF satisfy the condition.

When **Data** 0x4000 and **Mask** 0x101 are set

Data values 0x4000, 0x4001, 0x4100, and 0x4101 satisfy the condition.

(5) External sense data setting area



This area is used to specify the external sense data conditions.

Valid range: 0x0 ≤ External sense data ≤ 0xf

Each bit of the external sense data corresponds to the input pin level of an external sense clip of the emulation probe connected to the in-circuit emulator. The states of the bits can be used for an event condition. The table below lists the correspondence between the external sense clips and the bits of the external sense data.

External sense clip number	External sense data bit
8	Bit 3
7	Bit 2
6	Bit 1
5	Bit 0

To specify the high input level for an external sense clip for an event condition, set the corresponding bit to 1. To specify the low input level for an external sense clip for an event condition, set the corresponding bit to 0.

External sense data conditions consist of an external data value (External field) and mask value for the data. Set each value as follows:

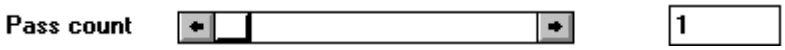
a. External

Enter an external sense data value.

b. Mask

A mask value can be set for the external sense data value. The default is 0xf (the external sense data condition is ignored; all data satisfies the condition). **The logical OR of the specified external sense data value and mask value is used as the data condition.**

(6) Pass count setting area



This area is used to specify the pass count condition.

Valid range: 1 ≤ Pass count ≤ 0xffff

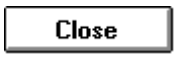
The pass count specifies the number of times the conditions for the event (address condition, status condition, data condition, and external sense data condition) must be satisfied to recognize the occurrence of the event.

When the pass count is set to 1, the occurrence of the event is recognized as soon as the conditions are satisfied. When the pass count is set to two or more, no more than two events can be enabled at the same time.

Buttons

Ignores any selections and restores the initial settings of a specified event condition.

Registers a specified event condition into the Event Manager (the registered conditions are displayed with a red E. mark).



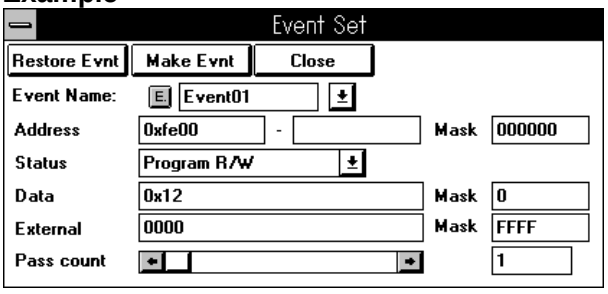
Closes the Event Set dialog box.

Notes

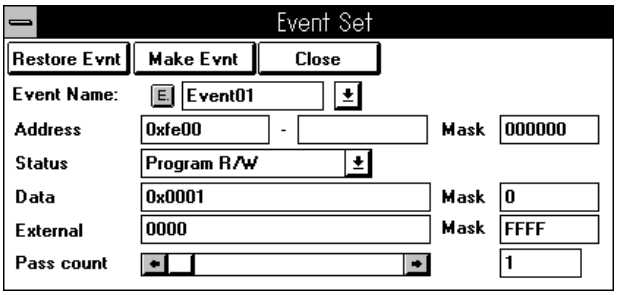
To set up an event for data access such as byte or word access, input the following data in the Data field of the data setting area.

Data access	Value
Byte	Byte-size data such as '0' or '0x00'
Word	Word-size data such as '000' or '0x0000'

Example



Address: 0xfe00
Status: Read/write by program
Data: 0x12
Access size: Byte access



Address: 0xfe00
Status: Read/write by program
Data: 0x1
Access size: Word access

Event Manager

Management window

Outline

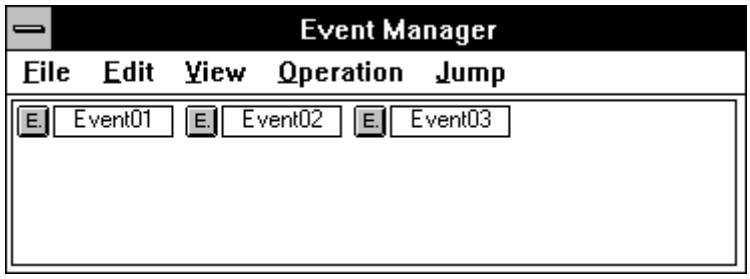
The Event Manager is used to display or delete events.

This window is also used to assign each event condition, registered using the Event Set or Event Link dialog box, to a break, trace, snapshot, stub, or timer event.

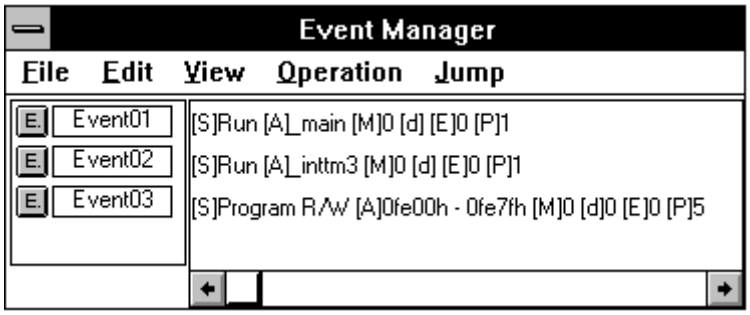
This window can be opened in any of the following ways:

- In the main window, select **Browse→Event→EventManager...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **E**, **and** **M** **keys**, in this order.
- Select a line number for which an event is set in the Source window, then select **View→ Event ?** from the **menu bar**.
- Select a line number for which an event is set in the Source window, then click the Event ? button.
- Select an address for which an event is set in the Assemble window, then select **View→ Event ?** from the **menu bar**.
- Select an address for which an event is set in the Assemble window, then click the Event ? button.

Window



Normal view



Detailed view

Fig. 5-26 Event Manager

Description

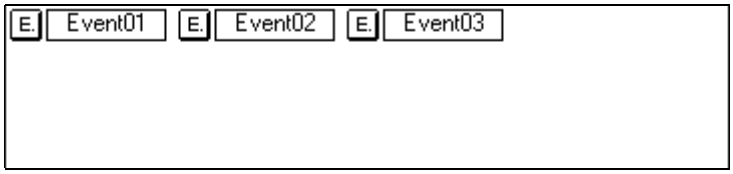
The Event Manager is used to display or delete events. This window manages event conditions to facilitate the registering and setting of event conditions (as event link, break, trace, snapshot, stub, or timer event conditions).

The Event Manager consists of the following components:

- Menu bar
- Event view area
- Event detail view area

The function of each component is described below.

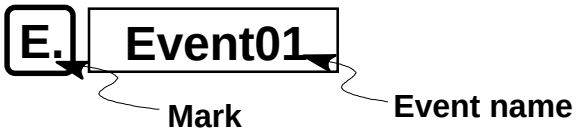
(1) Event view area



In normal view mode

In detailed view mode

The event view area displays the icons for the registered or set events. Each icon consists of a mark indicating the type of the event and the event name.



The table below lists these marks.

Mark	Description
E.	Event condition
L.	Event link condition
B.	Break event
T.	Trace event
Ti.	Timer event
U.	Stub event
S.	Snapshot event

The color of the character within each mark indicates the type of the event and whether the event is registered or set.

Character color	Relevant marks	Description
Red	E., L.	Registered event and event link conditions are always displayed in red.
	B., T., Ti., U., S.	The event is set. When the specified conditions are satisfied, the event is triggered.
Black	B., T., Ti., U., S.	The event is registered but not set. The event is not triggered even when the specified conditions are satisfied.
Blue	B.	Software break event

In addition to displaying event icons, this area can be used to perform the following two functions:

a. Jump function

The jump function causes a jump to the Source, Assemble, or Memory window, with the address condition for the selected icon being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

This function is used as follows (when jumping to the Source window):

1. Select an icon.
2. In the Event Manager, select **Jump→SourceText...** from the **menu bar** or press the **GRPH**, **J**, and **S** keys, in this order. Or, press the **CTRL**+**U** shortcut keys.

b. Deletion function

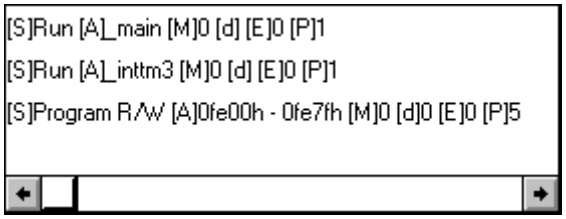
You can delete the settings for a specified event and cancel the registration of that event.

Event conditions (E.) and event link conditions (L.) can be deleted only when those conditions are not being used for other events (B., T., Ti., U., or S.). To delete an event or event link condition that is being used for another event, delete that event first.

This function is used as follows:

1. Select an icon.
2. In the Event Manager, select **Edit→Delete...** from the **menu bar** or press the **GRPH**, **E**, and **D** keys, in this order.

(2) Event detail view area



The event detail view area is displayed only in detailed view mode. This area displays detailed information for each event icon.

As the event conditions, the status condition, address condition, address mask condition, data condition, external data condition, and pass count condition are displayed, in this order, with the following headers:

For event conditions

Header	Condition
[S]	Status condition
[A]	Address condition
[M]	Address mask condition
[d]	Data condition
[E]	External sense data condition
[P]	Pass count condition

For event link conditions

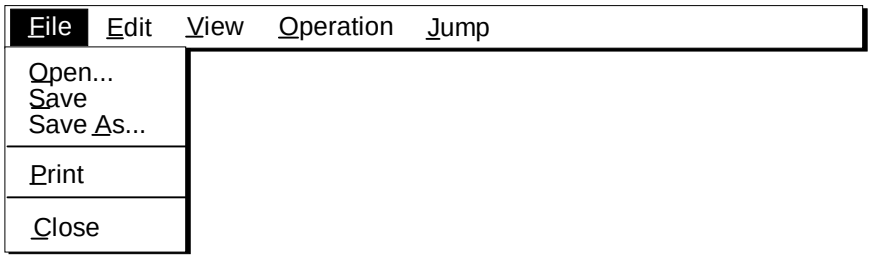
Header	Condition
[P1] - [P4]	Event link condition in the n-th phase
[D]	Disable condition

For break, trace, timer, snapshot, or stub event conditions

Header	Condition
[B]	Break condition
[SS]	Sectional trace start condition
[SE]	Sectional trace end condition
[Q]	Qualified trace condition
[S]	Timer start condition
[E]	Timer end condition
[Sn]	Snapshot condition
[Su]	Stub condition
[A]	Jump destination address upon the occurrence of a stub event

Menu bar

(a) File



Open... Loads an event setting file. The setting file select dialog box is opened (loading an event setting file deletes all previous registrations/settings.)

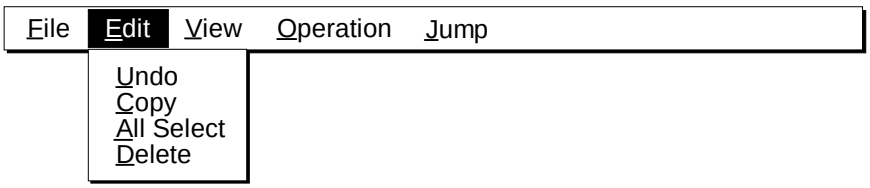
Save Saves the current event settings into the event setting file, overwriting the previously saved settings.

Save As... Saves the current event settings into a specified event setting file. The setting file select dialog box is opened.

Print Prints the event registration/setting information. The print dialog box is opened.

Close Closes the Event Manager.

(b) Edit



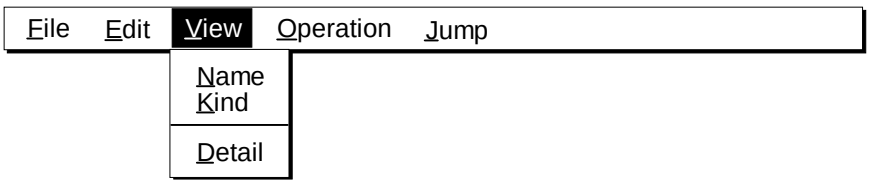
Undo Cancels the most recent editing.

Copy Copies a specified icon using a different name.

All Select Selects all icons.

Delete Deletes a specified icon.

(c) View

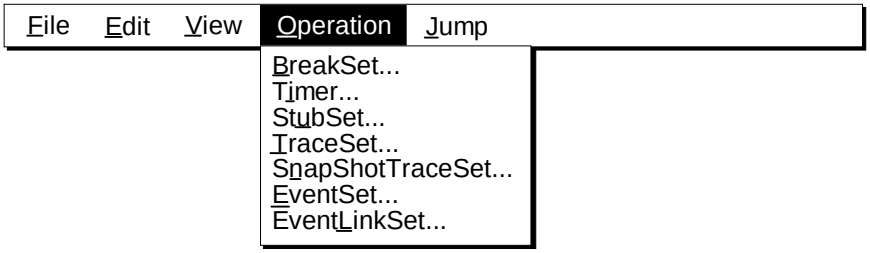


Name Sorts the icons into event name order.

Kind Sorts the icons into event type order.

Detail Switches between normal view and detailed view.

(d) Operation



BreakSet... Opens the Break dialog box.

Timer... Opens the Timer dialog box.

StubSet... Opens the Stub dialog box.

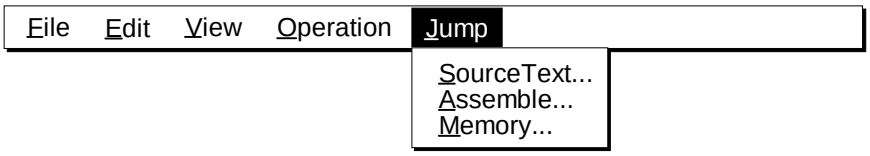
TraceSet... Opens the Trace dialog box.

SnapShotTraceSet... Opens the Snap-Shot dialog box.

EventSet... Opens the Event Set dialog box.

EventLinkSet... Opens the Event Link dialog box.

(e) Jump



SourceText... Sets the address set for the selected event as the jump address, and displays the source text and source line starting from that address: The Source window is opened.

Assemble... Sets the address set for the selected event as the jump address, and displays the disassembled text starting from that address: The Assemble window is opened.

Memory... Sets the address set for the selected event as the jump address, and displays the memory contents starting from that address: The Memory window is opened.

Notes

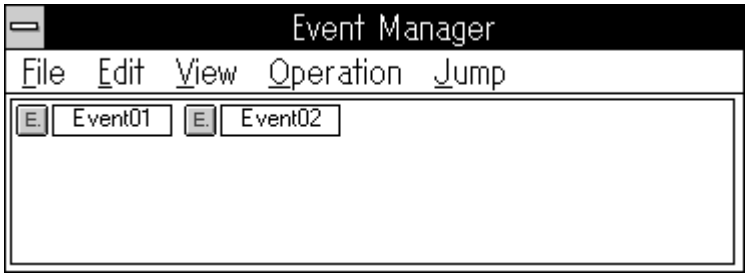
《How to set events》

You can easily set break, trace, and timer events by using the Event Manager, as follows:

① Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

② Register event conditions.
In this example, register two events Event01 and Event02.

③ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)
The Event Manager displays the two registered events, Event01 and Event02.



④ Open the dialog box corresponding to the type of the event to be set (Trace, Break, Timer, Event Link, Snap-Shot, or Stub dialog box).
In this example, open the Break dialog box.
(Select **Browse→BreakSet...** from the **menu bar** or click the button.)

⑤ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑥ Drag the icon into the Break dialog box.
Once the mouse pointer has been dragged into the Break dialog box, it changes from to . Dropping the icon in the Break dialog box copies the icon into the dialog box.

⑦ To register the event condition for a break event, enter a break event name, then click the Make Brk button.

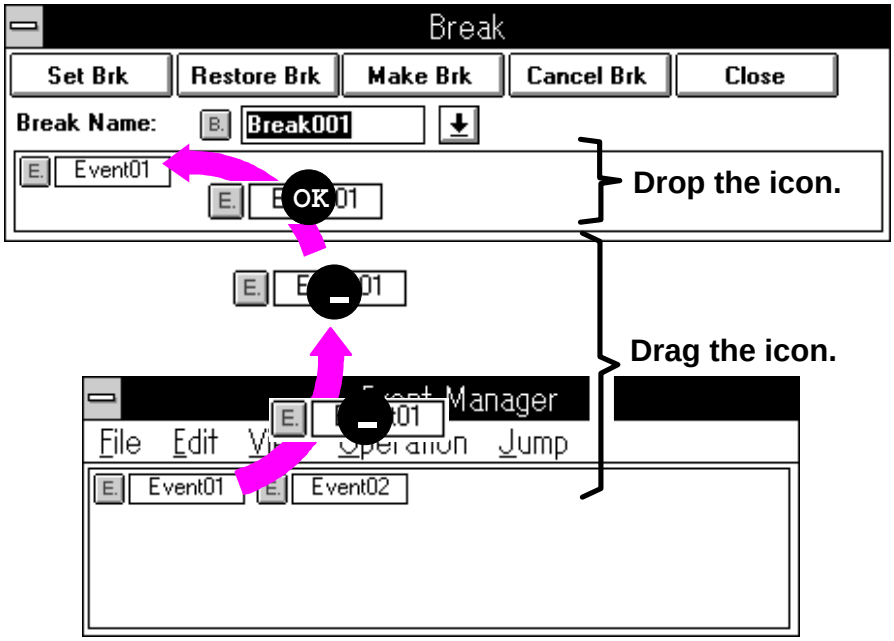


Fig. 5-27 Event Setting

《Event window relationship》

The Event Manager manages all events. The relationship between event windows is shown below.

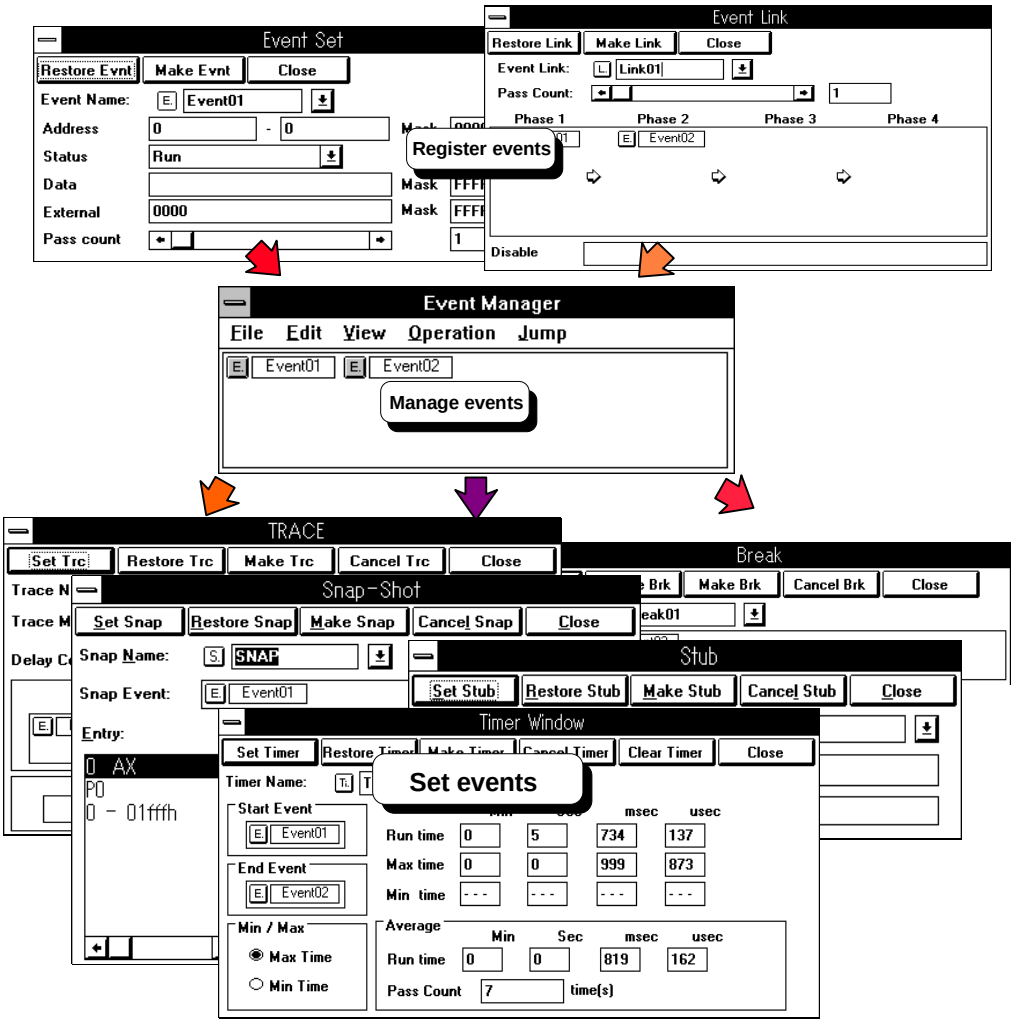


Fig. 5-28 Event Window Relationship

Event Link dialog box

Setting dialog box (Modeless)

Outline

The Event Link dialog box is used to register and display event link conditions. Once the event link conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→Event→EventLinkSet...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **E**, **and** **L** **keys**, in this order.
- In the Event Manager, select **Operation→EventLink...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **L** **keys**, in this order.

Window

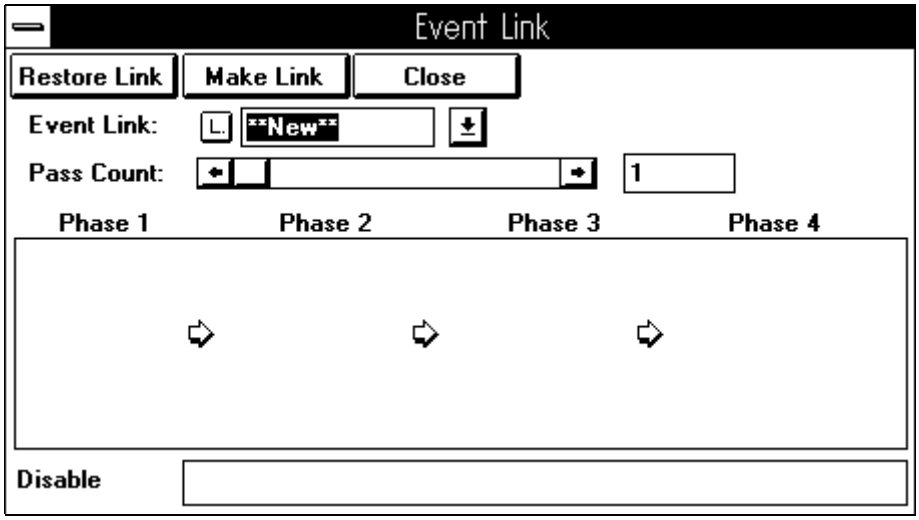


Fig. 5-29 Event Link Dialog Box

Description

The Event Link dialog box is used to register and display event link conditions.

Event link conditions are created by linking event conditions registered in the Event Manager. Up to four events can be used to define an event link condition.

The pass count for an event condition used as an event link condition must be one. Event conditions having a pass count of two or more cannot be used as an event link condition.

When an event link condition is set, the occurrence of the event is assumed only when the user program has executed the specified operations (event conditions) in the specified sequence. If a disable condition is satisfied part-way through the sequence, however, the event conditions satisfied up to that point are canceled and the first event condition is to be detected.

Up to 32,768 event link conditions can be registered. Up to two of these conditions can be simultaneously used as conditions for a break, timer, trace, snapshot, or stub event.

The Event Link dialog box consists of the following components:

- Event link name setting area
- Pass count setting area
- Link condition setting area
- Disable condition setting area
- Function buttons

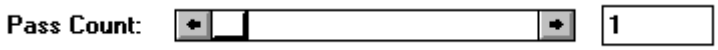
The function of each component is described below.

(1) Event link name setting area

This area is used to specify an event link name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the button displays a drop-down list, from which you can select an event link name. You can also type in, using the keyboard, a new event link name of up to eight characters .

(2) Pass count setting area



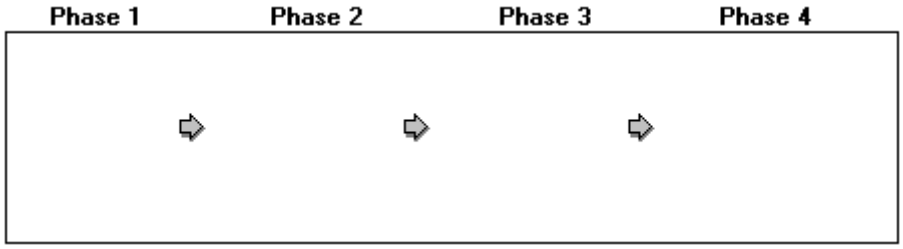
This area is used to specify the pass count condition.

Valid range: 1 ≤ Pass count ≤ 0xffff

The pass count specifies the number of times the event link conditions must be satisfied to recognize the occurrence of the event.

When the pass count is set to 1, the occurrence of the event is recognized as soon as the conditions are satisfied. When the pass count is set to two or more, no more than two event links can be enabled at the same time.

(3) Link condition setting area



This area is used to specify the event conditions to be linked and the order in which they are detected. The specified event conditions are detected in the order of **Phase 1 → Phase 2 → Phase 3 → Phase 4**. It is not necessary to specify an event for every phase. The occurrence of the event is recognized upon the detection of the entire event sequence specified in this area.

A combined total of up to ten event conditions can be specified in the link condition and disable condition setting areas.

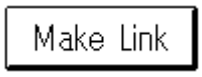
(4) Disable condition setting area

This area is used to specify disable conditions. When a disable condition is detected, the event conditions satisfied up to that point are canceled.

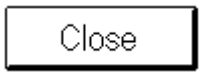
A combined total of up to ten event conditions can be specified in the link condition and disable condition setting areas.

Buttons

Ignores any selections and restores the initial settings of a specified event link condition.



Registers a specified event link condition into the Event Manager (the registered conditions are displayed with a red L. mark).



Closes the Event Link dialog box.

Notes

《How to set event link conditions》

The following is an example of setting event link conditions:

① Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

② Register event conditions in the Event Set dialog box.
In this example, register E_INIT, E_SUB0, E_SUB1, E_SUB2, E_SUB3, E_SUB4, and E_NMI.

③ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)

④ Open the Event Link dialog box.
(Select **Browse→Event→EventLinkSet...** from the **menu bar**.)

⑤ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑥ Drag the icon into the Event Link dialog box.
Once the mouse pointer has been dragged into the Event Link dialog box, it changes from to . Dropping the icon in the Event Link dialog box copies the icon into the dialog box.

⑦ By repeating steps ⑤ and ⑥, set the events from the Event Manager in the Event Link dialog box as follows:

Location	Events
Phase 1	E_INIT
Phase 2	E_SUB0
Phase 3	E_SUB1,E_SUB2
Phase 4	E_SUB4
Disable	E_SUB3,E_NMI

⑧ Enter an event link name.
In this example, enter E_LINK.

⑨ Pressing the Make Link button registers the E_LINK event link condition into the Event Manager.

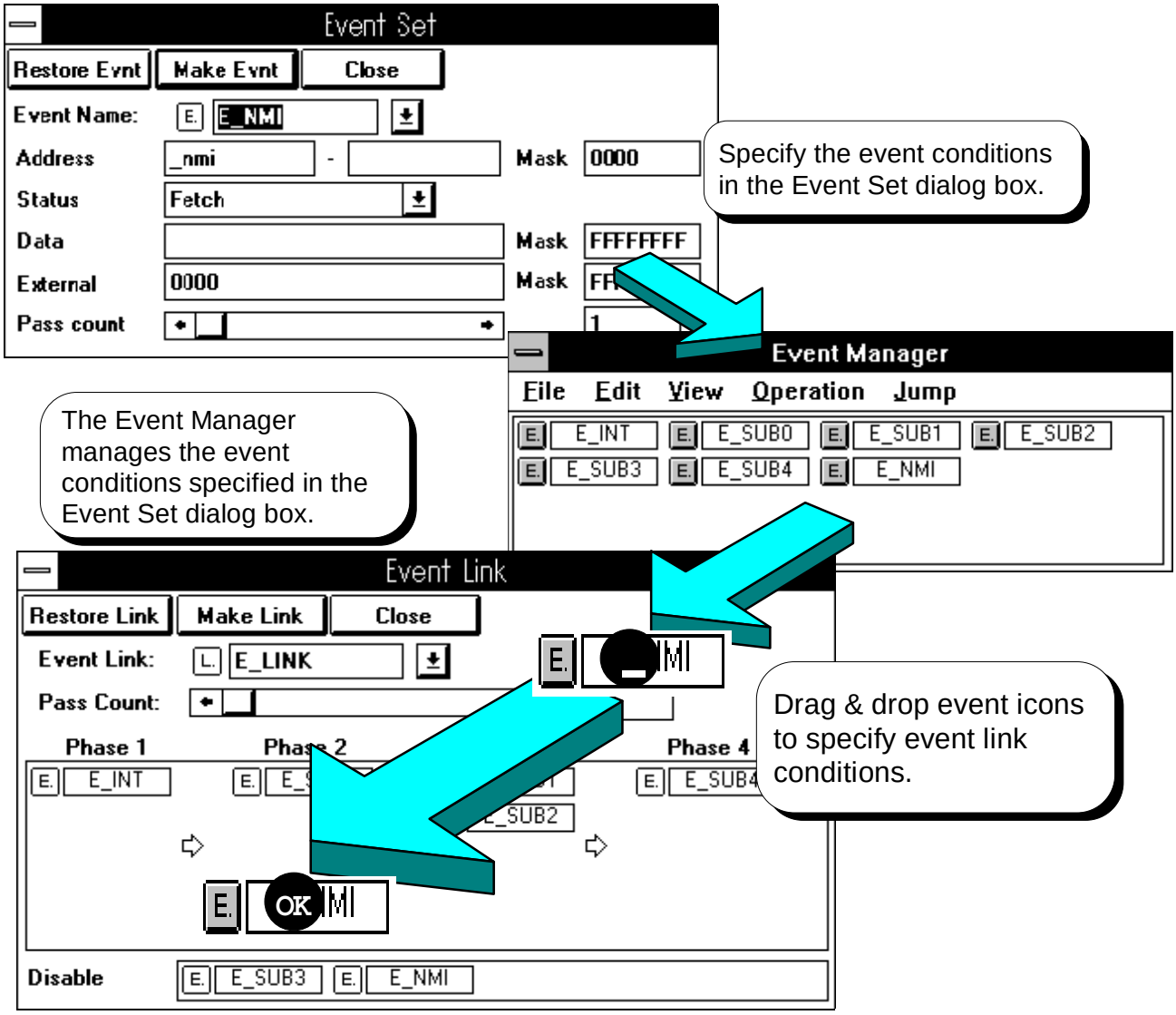


Fig. 5-30 Setting Event Link Conditions

«Application example of event link conditions»

An application example of event link conditions is described below. In this example, event link conditions are set for the program shown in Figure 5-30.

The program consists of the following processing:

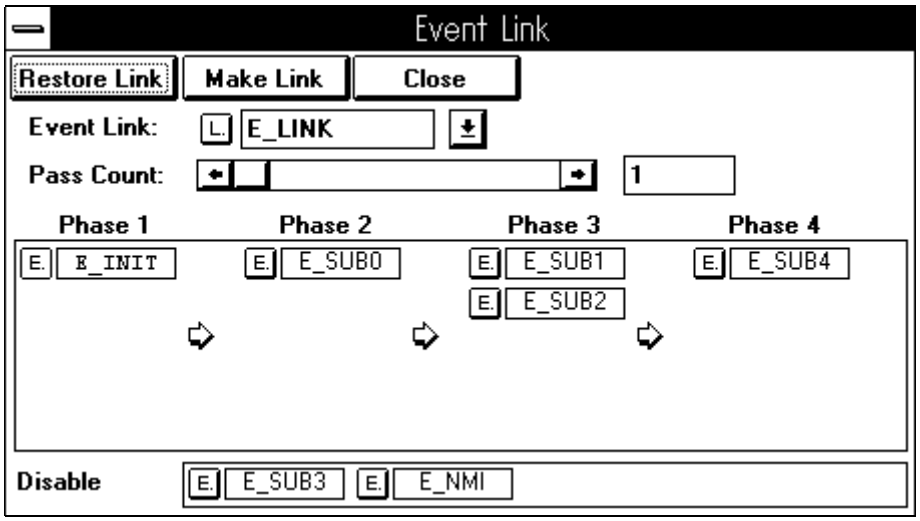
Main processing
1. Initialization (INIT)
2. Subprogram 0 (SUB0)
3. Conditional branch
   a. Subprogram 1 (SUB1)
   b. Subprogram 2 (SUB2)
   c. Subprogram 3 (SUB3)
4. Subprogram 4 (SUB4)

Interrupt handling
Interrupt handling routine (NMI)

This program may be executed according to sequence ①, ②, or ③, as shown in the figure. An interrupt (NMI) may occur during each sequence.

To generate an event only when the program is executed in sequence ① or ② with no interrupt (NMI), set the event conditions as shown in the figure and the event link conditions as follows:

Location	Events
Phase 1	E_INIT
Phase 2	E_SUB0
Phase 3	E_SUB1,E_SUB2
Phase 4	E_SUB4
Disable	E_SUB3,E_NMI



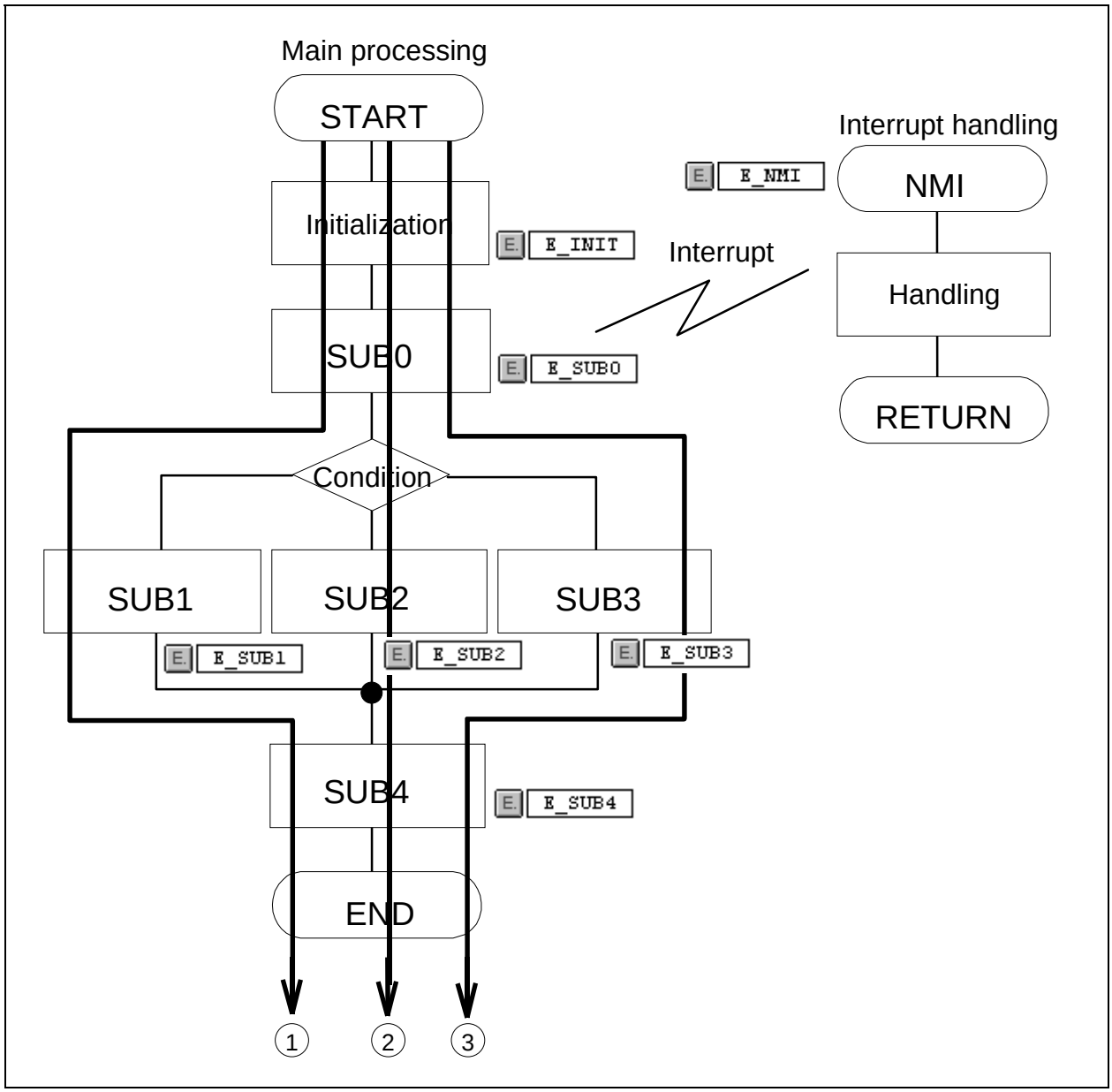


Fig. 5-31 Application Example of Event Link Conditions

Break dialog box

Setting dialog box (Modeless)

Outline

The Break dialog box is used to register, set, and display break event conditions. Once the break event conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→BreakSet...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **B** **keys**, in this order.
- In the tool bar, click the button.
- In the Event Manager, select **Operation→BreakSet...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **B** **keys**, in this order.

Window

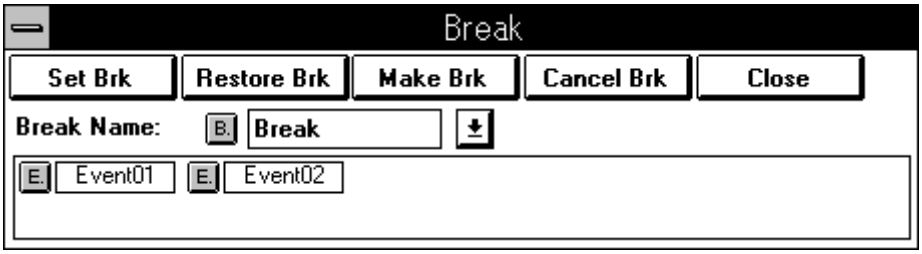


Fig. 5-32 Break Dialog Box

Description

The Break dialog box is used to register, set, and display break event conditions.

Break event conditions are registered by using event conditions and event link conditions registered in the Event Manager.

Up to 32,767 break event conditions can be registered. Up to twelve of these conditions can be simultaneously enabled.

The Break dialog box consists of the following components:

- Break event name setting area
- Break condition setting area
- Function buttons

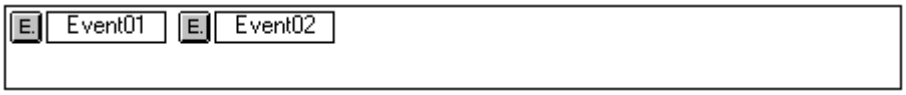
The function of each component is described below.

(1) Break event name setting area

This area is used to specify a break event name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the button displays a drop-down list, from which you can select a break event name. You can also type in, using the keyboard, a new break event name of up to eight characters.

(2) Break condition setting area



This area is used to specify break event conditions. To specify a condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

A total of up to twelve event and event link conditions can be specified in this area.

Buttons

Set Brk — Enables a specified break event condition (a break event occurs when that condition is satisfied). Clicking this button changes the color of the B. mark to red.

Restore Brk — Ignores any selections and restores the initial settings of a specified break event condition.

Make Brk — Registers a specified break event condition into the Event Manager. A break event condition is not enabled by merely registering it.

Cancel Brk — Disables a specified break event condition (a break event does not occur when that condition is satisfied). Clicking this button changes the color of the B. mark to black.

Close — Closes the Break dialog box.

Note

《How to set break event conditions》

The following is an example of setting break event conditions:

① Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

② Register event conditions in the Event Set dialog box.
In this example, register two events Event01 and Event02.

③ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)

④ Open the Break dialog box.
(Select **Browse→BreakSet...** from the **menu bar**.)

⑤ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑥ Drag the icon into the Break dialog box.
Once the mouse pointer has been dragged into the Break dialog box, it changes from to . Dropping the icon in the Break dialog box copies the icon into the dialog box.

⑦ Enter a break event name.
In this example, enter BREAK.

⑧ Pressing the Make Brk button registers the BREAK break event condition into the Event Manager.

⑨ Pressing the Set Brk button changes the mark color of the BREAK break event condition from black to red, indicating the break event is enabled.

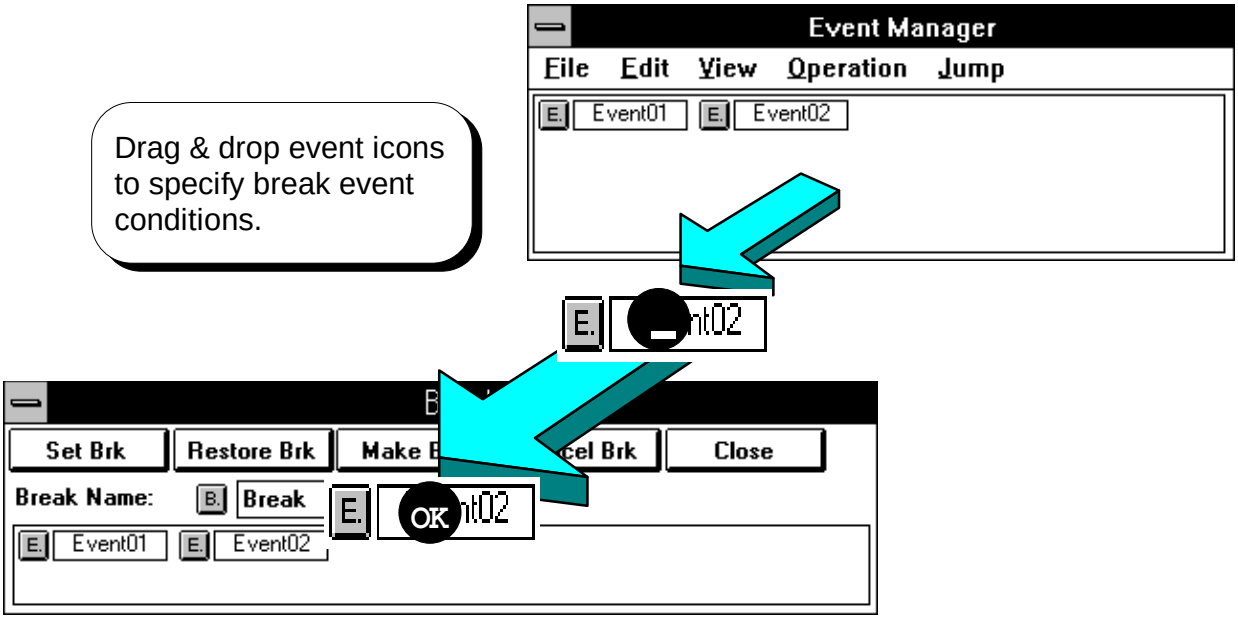


Fig. 5-33 Setting Break Event Conditions

Trace dialog box

Setting dialog box (Modeless)

Outline

The Trace dialog box is used to register, set, and display trace event conditions. Once the trace event conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→Trace→TraceSet...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **C**, **and** **T** **keys**, in this order.
- In the tool bar, click the button.
- In the Event Manager, select **Operation→TraceSet...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **T** **keys**, in this order.

Window

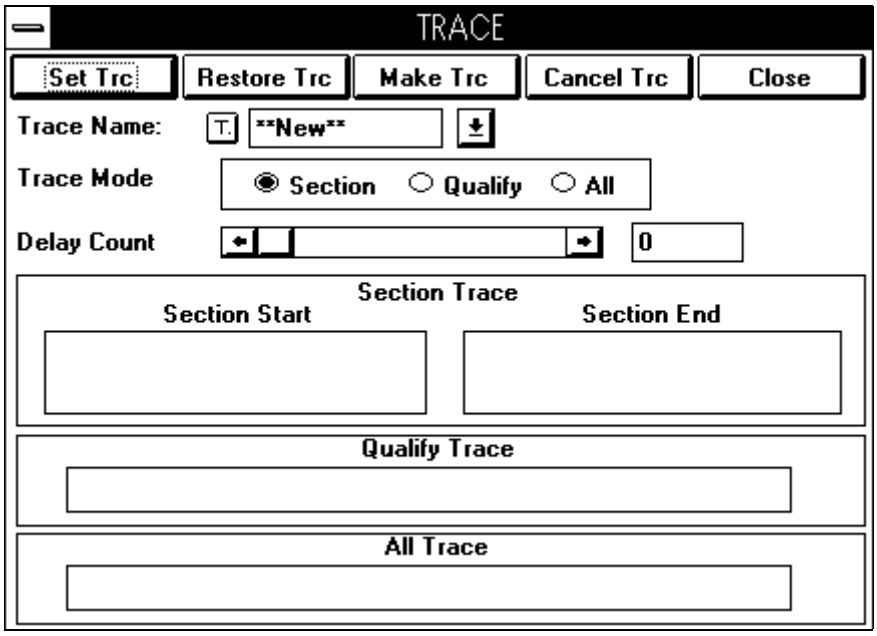


Fig. 5-34 Trace Dialog Box

Description

The Trace dialog box is used to register, set, and display trace event conditions.

Trace event conditions are registered by using event conditions and event link conditions registered in the Event Manager.

Up to 32,767 trace event conditions can be registered. Only one of these conditions can be simultaneously enabled.

To operate the tracer according to the specified trace event condition, select **Execute→ Cond. Trace ON** from the **menu bar** of the main window.

The Trace dialog box consists of the following components:

- Trace event name setting area
- Trace mode setting area
- Delay count setting area
- Sectional trace condition setting area
- Qualified trace condition setting area
- Function buttons

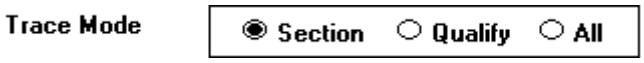
The function of each component is described below.

(1) Trace event name setting area

This area is used to specify a trace event name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the  button displays a drop-down list, from which you can select a trace event name. You can also type in, using the keyboard, a new trace event name of up to eight characters.

(2) Trace mode setting area



This area is used to specify the trace mode.

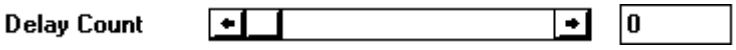
There are three trace modes, all trace, sectional trace, and qualified trace.

Trace mode	Description
All trace	Traces all sources.
Sectional trace	Traces only in a range specified with event conditions.
Qualified trace	Traces only when a specified event condition is satisfied.

To specify each trace mode, the relevant item (under **Execute**) on the menu bar of the main window must also be set. The table below lists the combination of settings to specify each trace mode.

Trace mode	Setting for Execute on the menu bar of the main window	Trace mode setting in Trace dialog box	Delay condition
All trace	**Uncond. Trace ON**	----	No
	Cond. Trace ON	**ALL**	Yes
Sectional trace		**Section**	Yes
Qualified trace		**Qualify**	Yes
No trace	**Trace OFF**	----	----

(3) Delay count setting area



This area is used to specify the delay count.

Valid range: $0 \leq$ Delay count ≤ 65535

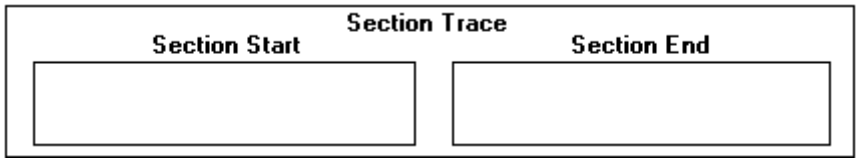
When a delay count is set, the detection of a delay event causes the tracer to trace source lines or instructions as many as the delay count, then stop.

When the delay count is set to 0, no delay event is detected.

The following events are used as a delay event depending on the trace mode:

Trace mode	Delay event
All trace	Event condition specified in the all trace condition setting area
Sectional trace	Event conditions specified for **Section End** in the sectional trace condition setting area
Qualified trace	Event condition specified in the qualified trace condition setting area

(4) Sectional trace condition setting area



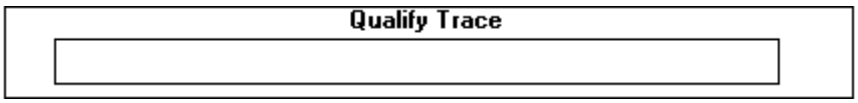
This area is used to specify the event conditions for sectional trace.

Section Start: Specifies the trace start event condition.
Section End: Specifies the trace end event condition.

To specify an event condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

A total of up to twelve event and event link conditions can be specified in this area.

(5) Qualified trace condition setting area

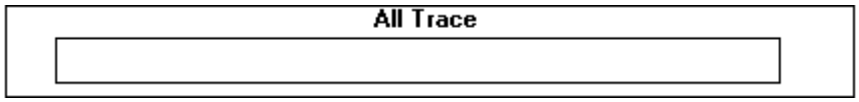


This area is used to specify the event condition for qualified trace.

To specify an event condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

A total of up to twelve event and event link conditions can be specified in this area.

(6) All trace condition setting area



This area is used to specify the delay event condition for all trace.

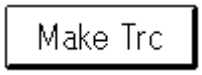
To specify an event condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

A total up to twelve event and event link conditions can be specified in this area.

Buttons

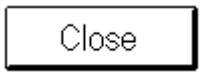
Enables a specified trace event condition (a trace event occurs when that condition is satisfied). Clicking this button changes the color of the T. mark to red.

Ignores any selections and restores the initial settings of a specified trace event condition.



Registers a specified trace event condition into the Event Manager. A trace event condition is not enabled by merely registering it.

Disables a specified trace event condition (a trace event does not occur when that condition is satisfied). Clicking this button changes the color of the T. mark to black.



Closes the Trace dialog box.

Note

《How to set trace event conditions》

The following is an example of setting trace event conditions:

① Select **Execute→Cond. Trace ON** from the **menu bar** of the main window.

② Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

③ Register event conditions in the Event Set dialog box.
In this example, register two events Event01 and Event02.

④ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)

⑤ Open the Trace dialog box.
(Select **Browse→Trace→TraceSet...** from the **menu bar**.)

⑥ Specify the trace mode and delay count.

⑦ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑧ Drag the icon into the Trace dialog box.
Once the mouse pointer has been dragged into the Trace dialog box, it changes from to OK. Dropping the icon in the Trace dialog box copies the icon into the dialog box.

⑨ Enter a trace event name.
In this example, enter TRACE.

⑩ Pressing the Make Trc button registers the TRACE trace event condition into the Event Manager.

⑪ Pressing the Set Trc button changes the mark color of the TRACE trace event condition from black to red, indicating the trace event is enabled.

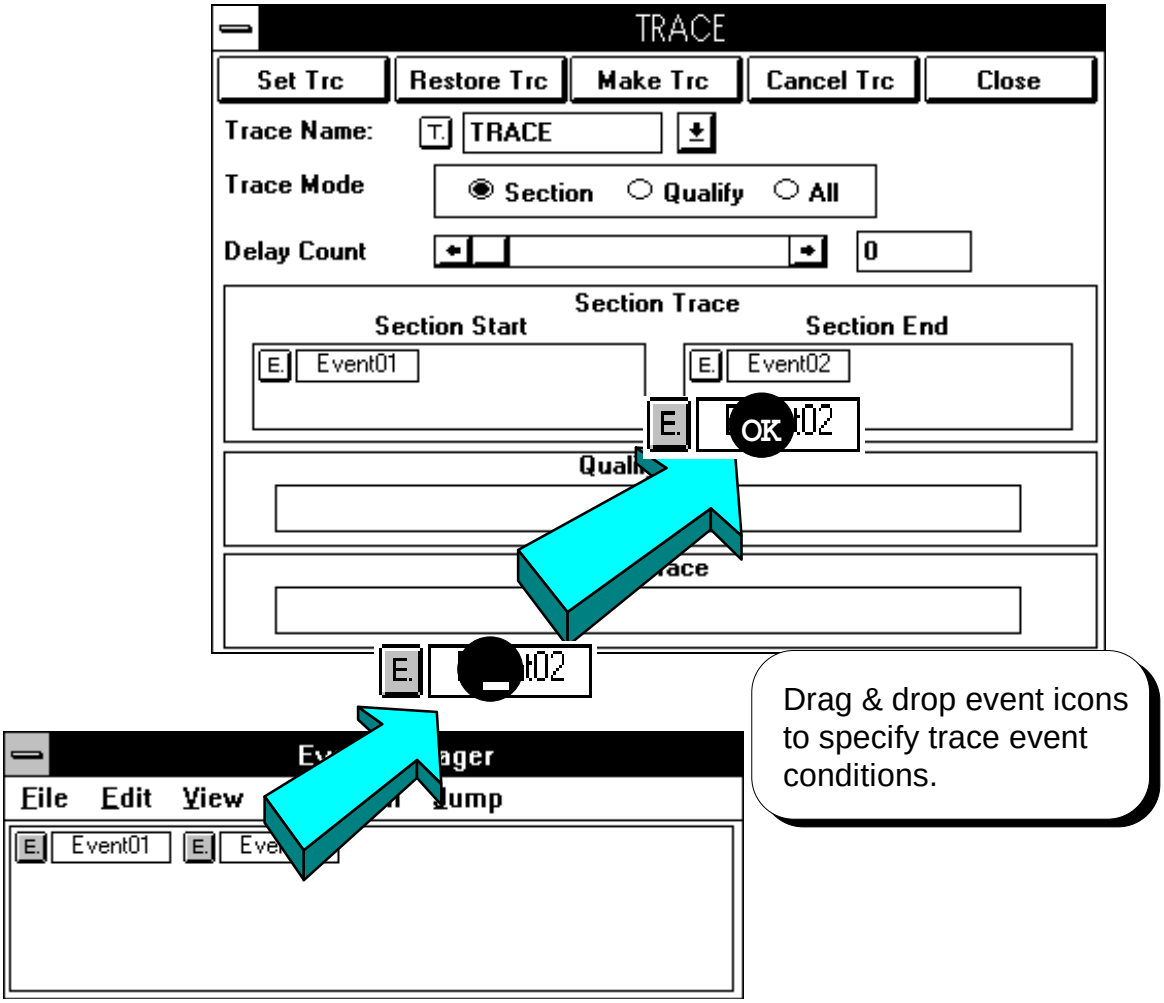


Fig. 5-35 Setting Trace Event Conditions

Snap-Shot dialog box

Setting dialog box (Modeless)

Outline

The Snap-Shot dialog box is used to register, set, and display snapshot event conditions. Once the snapshot event conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→Trace→SnapShotTraceSet...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **C**, **and** **N** **keys**, in this order.
- In the Event Manager, select **Operation→SnapShotTraceSet...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **N** **keys**, in this order.

Window

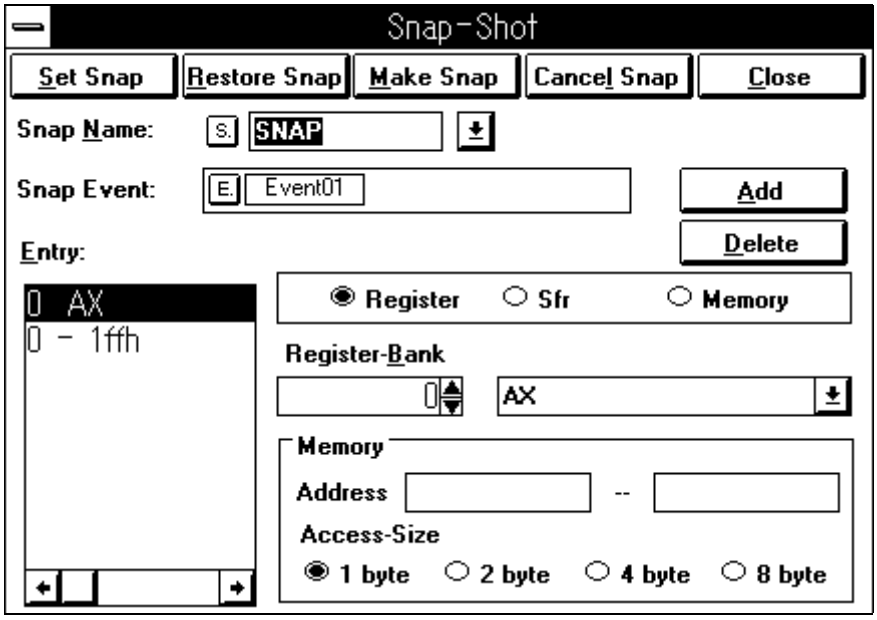


Fig. 5-36 Snap-Shot Dialog Box

Description

The Snap-Shot dialog box is used to register, set, and display snapshot event conditions.

Snapshot event conditions registered by using event conditions and event link conditions registered in the Event Manager.

Up to 32,767 snapshot event conditions can be registered. Only one of these conditions can be simultaneously enabled.

The Snap-Shot dialog box consists of the following components:

- Snapshot event name setting area
- Snapshot event condition setting area
- Snapshot data selection area
- Register bank setting area
- Register & SFR data setting area
- Memory data setting area
- Snapshot data view area
- Function buttons

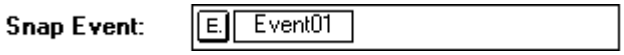
The function of each component is described below.

(1) Snapshot event name setting area

This area is used to specify a snapshot event name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the button displays a drop-down list, from which you can select a snapshot event name. You can also type in, using the keyboard, a new snapshot event name of up to eight characters.

(2) Snapshot event condition setting area

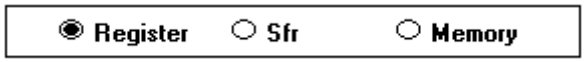


This area is used to specify a snapshot event condition.

To specify an event condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

Only one event or event link condition can be specified in this area.

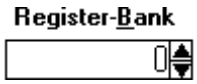
(3) Snapshot data selection area



This area is used to select the type of data to be subject to snapshot, from the following three types:

Type	Description
Register	Enables the registration of register data (general-purpose and control registers). A register can be selected using the register bank setting area and register & SFR data setting area.
Sfr	Enables the registration of SFR data. An SFR can be selected using the register & SFR data setting area.
Memory	Enables the registration of memory data. An address range and access size can be selected using the memory data setting area.

(4) Register bank setting area



This area is used to specify the bank number of the register to be subject to snapshot. When **Current** is specified, the current bank register upon a break is subject to snapshot. This area is enabled only when **Register** is selected in the snapshot data selection area.

(5) Register & SFR data setting area

The contents of the drop-down list displayed in this area vary with whether **Register** or **Sfr** is selected in the snapshot data selection area.

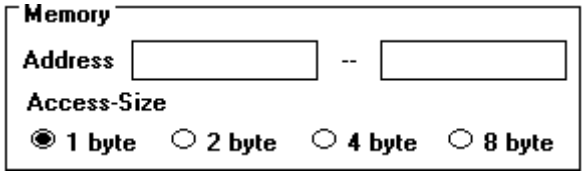
When **Register** is selected

This area is used to select a register. Clicking the  button displays a drop-down list, from which you can select a register.

When **Sfr** is selected

This area is used to select an SFR. Clicking the  button displays a drop-down list, from which you can select an SFR.

(6) Memory data setting area



This area is used to specify the address range and access size for memory data to be subject to snapshot.

This area is enabled only when **Memory** is selected in the snapshot data selection area.

Specify an address range and access size as follows:

Item	Description
Address	Specify an address range as start address -- end address.
Address-Size	Select an access size from the following: 1 byte 2 bytes 4 bytes 8 bytes

Symbols can also be used to specify an address, as follows:

Function or variable	_fnc file**#**_fnc (for static function or variable)
Line number in source text	file**:**no

fnc: Function or variable name
file: File name no: Line number

When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

(7) Snapshot data view area

This area lists the registered snapshot data.

A total of up to 16 data items can be registered for snapshot, including register, SFR, and memory data.

The registered snapshot data items are written into the tracer upon the detection of a snapshot event.

To register a snapshot data item, make necessary settings then click the Add button. To delete a snapshot data item, select a data item to be deleted then click the Delete button.

Buttons

Set Snap	Enables a specified snapshot event condition. The color of the S. mark changes to red.
Restore Snap	Ignores any selections and restores the initial settings.
Make Snap	Registers a specified snapshot event into the Event Manager.
Cancel Snap	Disables a specified snapshot event condition. The color of the S. mark changes to black.
Close	Closes the Snap-Shot dialog box.
Add	Registers snapshot data.
Delete	Deletes a specified snapshot data item from the Entry area.

Note

《How to set snapshot event conditions》

The following is an example of setting snapshot event conditions:

① Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

② Register event conditions in the Event Set dialog box.
In this example, register Event01.

③ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)

④ Open the Snap-Shot dialog box.
(Select **Browse→Trace→SnapShotTraceSet...** from the **menu bar**.)

⑤ Register snapshot data.

⑥ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑦ Drag the icon into the Snap-Shot dialog box.
Once the mouse pointer has been dragged into the Snap-Shot dialog box, it changes from to . Dropping the icon in the Snap-Shot dialog box copies the icon into the dialog box.

⑧ Enter a snapshot event name.
In this example, enter SNAP.

⑨ Pressing the Make Snap button registers the SNAP snapshot event condition into the Event Manager.

⑩ Pressing the Set Snap! button changes the mark color of the SNAP snapshot event condition from black to red, indicating the snapshot event is enabled.

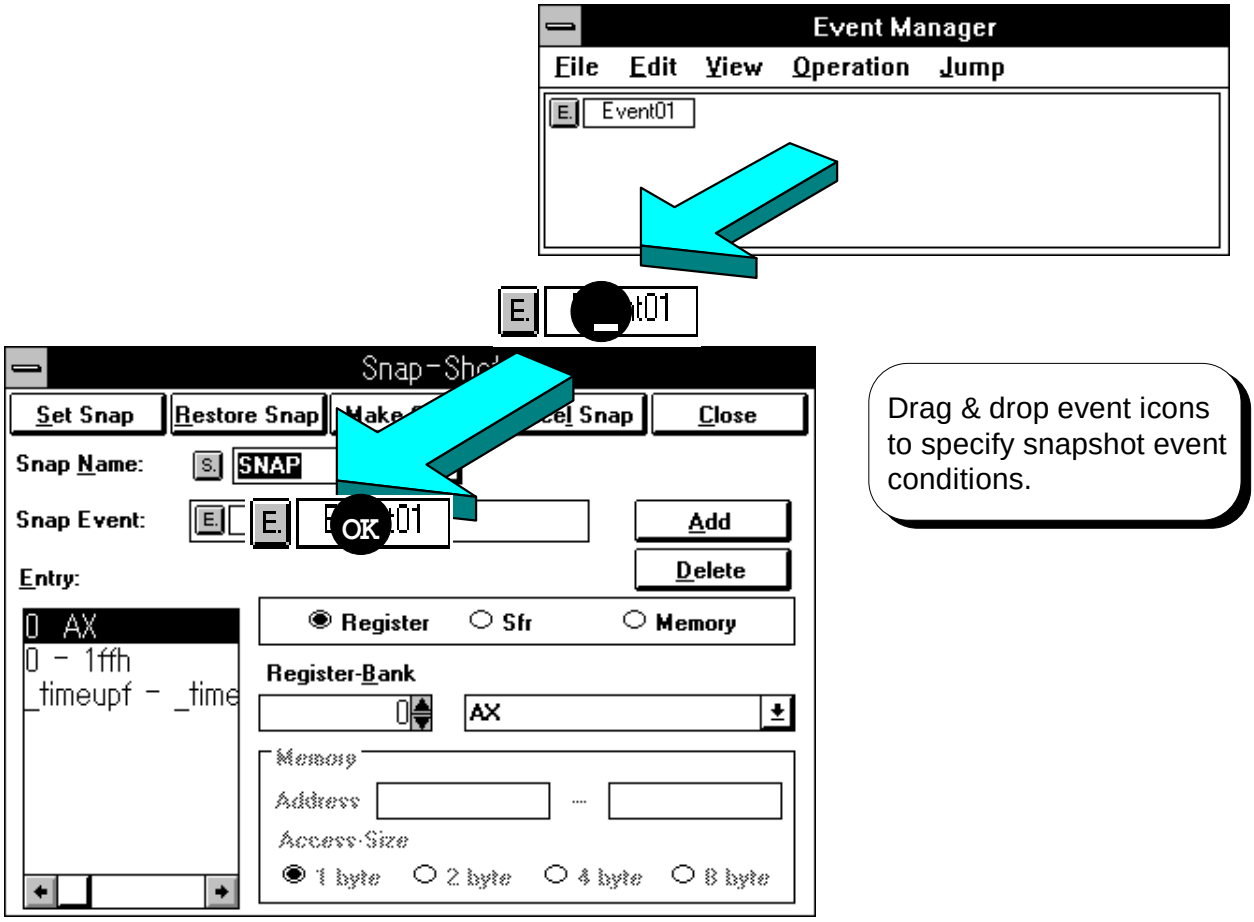


Fig. 5-37 Setting Snapshot Event Conditions

Stub dialog box

Setting dialog box (Modeless)

Outline

The Stub dialog box is used to register, set, and display stub event conditions. Once the stub event conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→Stub Set...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **U** **keys**, in this order.
- In the Event Manager, select **Operation→Stub Set...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **U** **keys**, in this order.

Window

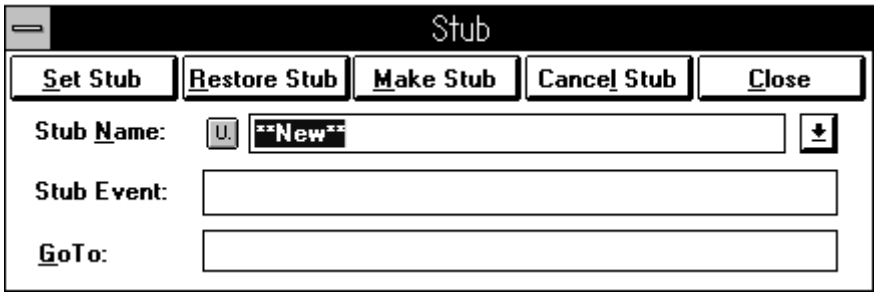


Fig. 5-38 Stub Dialog Box

Description

The Stub dialog box is used to register, set, and display stub event conditions.

Stub event conditions are registered by using event conditions and event link conditions registered in the Event Manager.

Up to 32,767 stub event conditions can be registered. Only one of these conditions can be simultaneously enabled.

The Stub dialog box consists of the following components:

- Stub event name setting area
- Stub event condition setting area
- Jump destination address setting area
- Function buttons

The function of each component is described below.

(1) Stub event name setting area

This area is used to specify a stub event name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the  button displays a drop-down list, from which you can select a stub event name. You can also type in, using the keyboard, a new stub event name of up to eight characters.

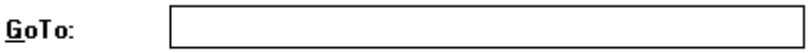
(2) Stub event condition setting area

This area is used to specify a stub event condition.

To specify an event condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

Only one event or event link condition can be specified in this area.

(3) Jump destination address setting area



This area is used to specify the first address of the function to be executed upon the detection of a stub event. Specify a function which ends with a RET instruction.

Symbols can also be used to specify the first address of a function, as follows:

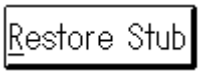
Function or variable	_fnc file#_fnc (for static function or variable)
Line number in source text	file:no

fnc: Function or variable name
file: File name no: Line number

When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

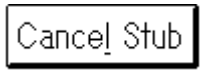
Buttons

Enables a specified stub event condition. The color of the U. mark changes to red. When that condition is satisfied, the function starting from the address specified in the Goto: area is executed.

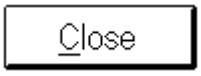


Ignores any selections and restores the initial settings of a specified stub event condition.

Registers a specified stub event condition into the Event Manager.



Disables a specified stub event condition. The color of the U. mark changes to black. A stub event does not occur when that condition is satisfied.



Closes the Stub dialog box.

Note

《How to set stub event conditions**》**

The following is an example of setting stub event conditions:

① Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

② Register event conditions in the Event Set dialog box.
In this example, register Event01.

③ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)

④ Open the Stub dialog box.
(Select **Browse→Stub Set...** from the **menu bar**.)

⑤ Specify the jump destination address for a function.

⑥ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑦ Drag the icon into the Stub dialog box.
Once the mouse pointer has been dragged into the Stub dialog box, it changes from to . Dropping the icon in the Stub dialog box copies the icon into the dialog box.

⑧ Enter a stub event name.
In this example, enter STUB.

⑨ Pressing the Make Stub button registers the STUB stub event condition into the Event Manager.

⑩ Pressing the Set Stub button changes the mark color of the STUB stub event condition from black to red, indicating the stub event is enabled.

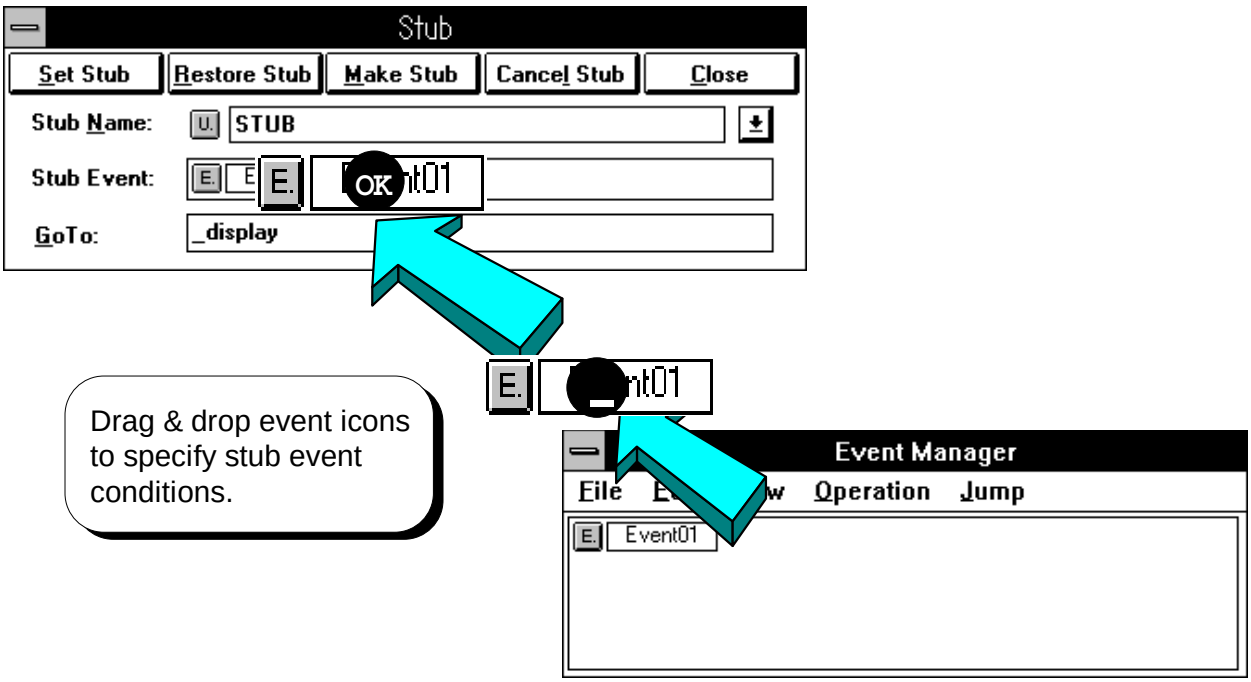


Fig. 5-39 Setting Stub Event Conditions

Timer dialog box

View/setting dialog box (Modeless)

Outline

The Timer dialog box is used to display the results of execution time measurement, and register and set timer event conditions. Once the timer event conditions have been set using this dialog box, they are automatically registered into the Event Manager.

This dialog box can be opened in any of the following ways:

- In the main window, select **Browse→Timer...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **I** **keys**, in this order.
- In the tool bar, click the button.
- In the Event Manager, select **Operation→Timer...** from the **menu bar**.
- In the Event Manager, press the **GRPH**, **O**, **and** **I** **keys**, in this order.

Window

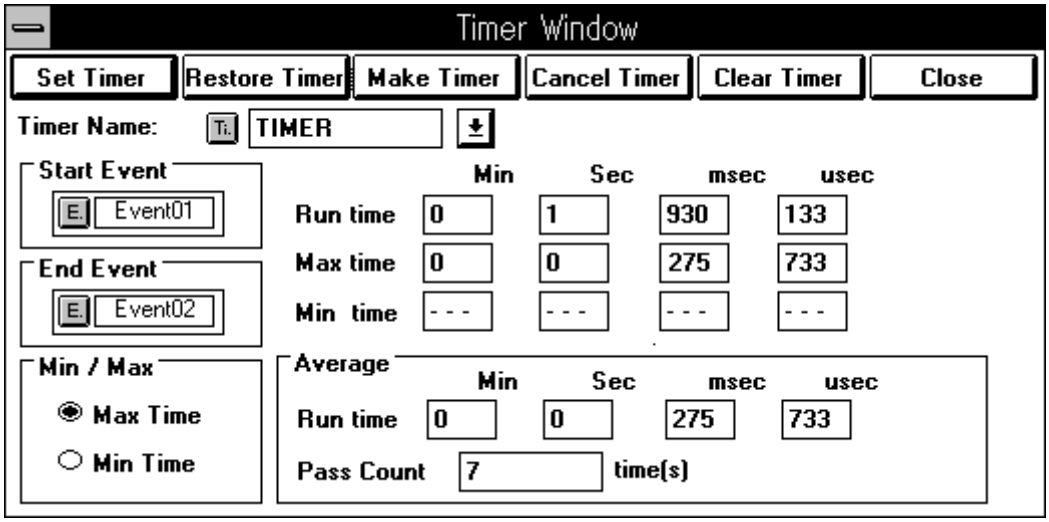


Fig. 5-40 Timer Dialog Box

Description

The Timer dialog box is used to display the results of execution time measurement, and register and set timer event conditions.

Selecting a set timer event displays the results of the execution time measurement triggered by that event.

Timer event conditions are registered by using event conditions and event link conditions registered in the Event Manager.

Up to 32,767 timer event conditions can be registered. Up to three of these conditions can be simultaneously enabled.

When less than three timer event conditions are enabled, the time between the start of program execution and a break can also be displayed, by selecting Run-Break from the timer event name setting area.

The Timer dialog box consists of the following components:

- Timer event name setting area
- Timer event condition setting area
- Measurement mode selection area
- Execution time view area
- Average execution time view area
- Function buttons

The function of each component is described below.

(1) Timer event name setting area

This area is used to specify a timer event name. When the dialog box is opened, ****NEW**** is displayed.

Clicking the button displays a drop-down list, from which you can select a timer event name. Timer event name "Run-Break", which displays the time between the start of program execution and a break, is already registered in the list (but not displayed within the Event Manager). You can also type in, using the keyboard, a new timer event name of up to eight characters.

(2) Timer event condition setting area

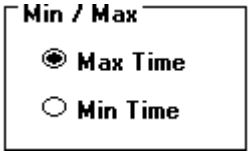
This area is used to specify timer event conditions.

Start Event: Specifies the event condition for starting timer measurement.
End Event: Specifies the event condition for ending timer measurement.

To specify an event condition, drag & drop the icon for a desired event or event link condition from the Event Manager into this area.

Only one event or event link condition can be specified for each of the start and end events.

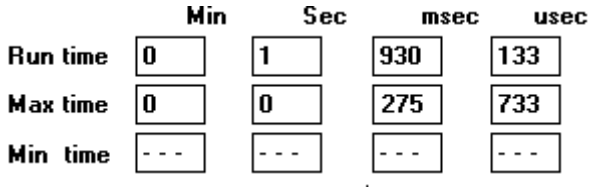
(3) Measurement mode selection area



This area is used to select the time measurement mode from the following:

Max Time: Maximum execution time mode.
In this mode, the longest time required for unit processing between the start and end event conditions is measured.
Min Time: Minimum execution time mode.
In this mode, the shortest time required for unit processing between the start and end event conditions is measured.

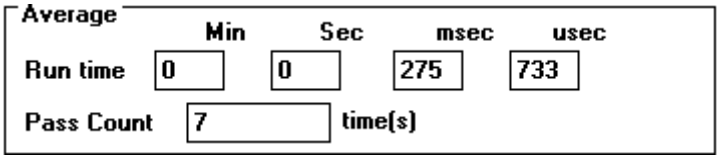
(4) Execution time view area



This area displays the results of program execution time measurement. Up to approximately 14 minutes and 18 seconds can be measured. In addition to the total execution time between the specified start and end event conditions, the maximum or minimum execution time between the conditions is measured at the same time.

Execution time	Description
Run time	Displays the total execution time.
Max time	Displays the maximum execution time (only when **Max time** is selected in the measurement mode selection area).
Min time	Displays the minimum execution time (only when **Min time** is selected in the measurement mode selection area).

(5) Average execution time view area

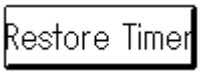


This area displays the average execution time required for unit processing and the number of times unit processing has been executed.

Item	Description
Run time	Displays the average execution time.
Pass Count	Displays the number of times unit processing has been executed. Valid range: 0 ≤ number ≤ 65535

Buttons

Enables a specified timer event condition (a timer event occurs when that condition is satisfied). Clicking this button changes the color of the  mark to red.

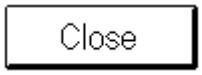


Ignores any selections and restores the initial settings of a specified timer event condition.

Registers a specified timer event condition into the Event Manager. A timer event condition is not enabled by merely registering it.

Disables a specified timer event condition (a timer event does not occur when that condition is satisfied). Clicking this button changes the color of the  mark to black.

Initializes the timer.



Closes the Timer dialog box.

Caution

When measured execution time has exceeded the maximum measurable time (14 minutes and 18 seconds), selecting the relevant timer event name displays message "Result of Timer measurement is over." to indicate that measurement has failed.

Note

《How to set timer event conditions》

The following is an example of setting timer event conditions:

① Open the Event Set dialog box.
(Select **Browse→Event→EventSet...** from the **menu bar**.)

② Register event conditions in the Event Set dialog box.
In this example, register two events Event01 and Event02.

③ Open the Event Manager.
(Select **Browse→Event→EventManager...** from the **menu bar**.)

④ Open the Timer dialog box.
(Select **Browse→Timer...** from the **menu bar**.)

⑤ Select the icon for the event condition to be used, in the Event Manager.
Position the mouse pointer to the icon, press the mouse button then, keeping the mouse button held down, move the mouse. The mouse pointer changes from to . As the mouse is moved, so too does the icon.

⑥ Drag the icon into the Timer dialog box.
Once the mouse pointer has been dragged into the Timer dialog box, it changes from to . Dropping the icon in the Timer dialog box copies the icon into the dialog box.

⑦ Enter a timer event name.
In this example, enter TIMER.

⑧ Pressing the Make Timer button registers the TIMER timer event condition into the Event Manager.

⑨ Pressing the Set Timer button changes the mark color of the TIMER timer event condition from black to red, indicating the timer event is enabled.

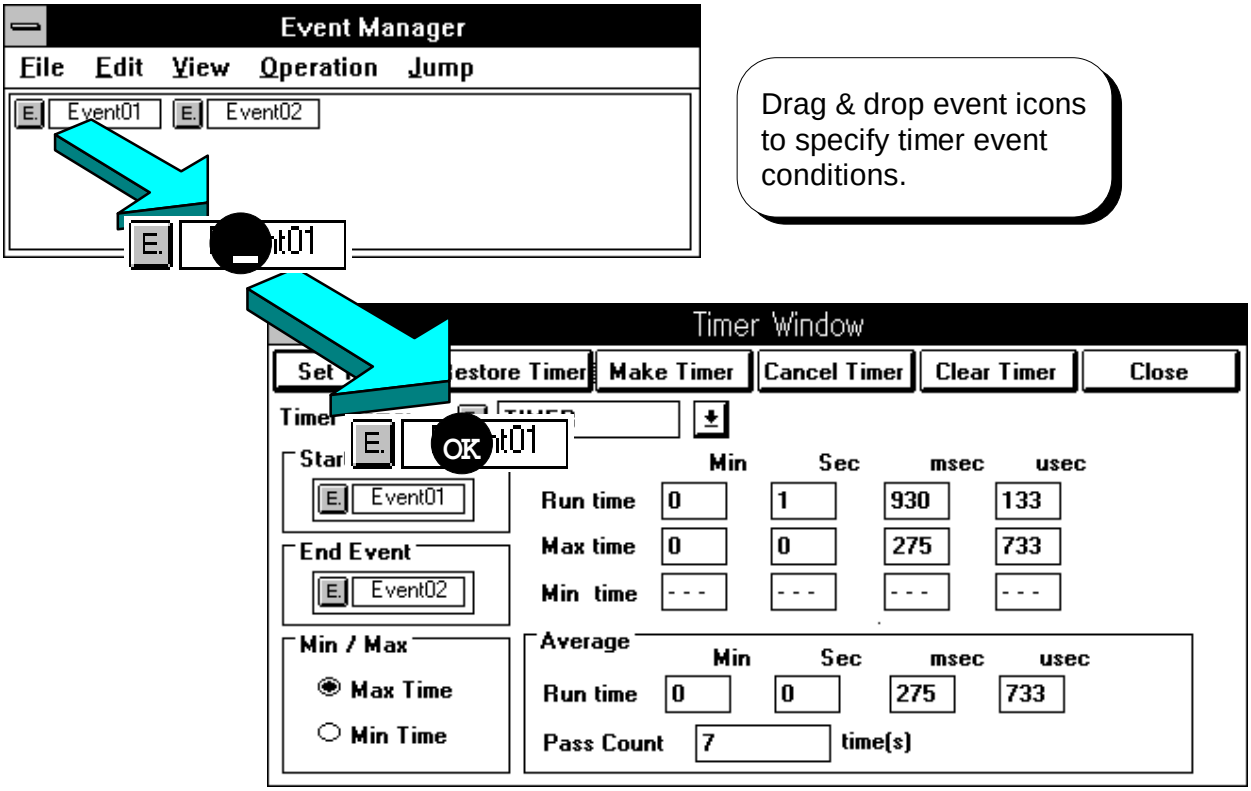


Fig. 5-41 Setting Timer Event Conditions

Trace View window

View window

Outline

The Trace View window is used to display trace results.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Trace→TraceView...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **C**, **and** **V** **keys**, in this order.
- In the tool bar, click the  button.

Window

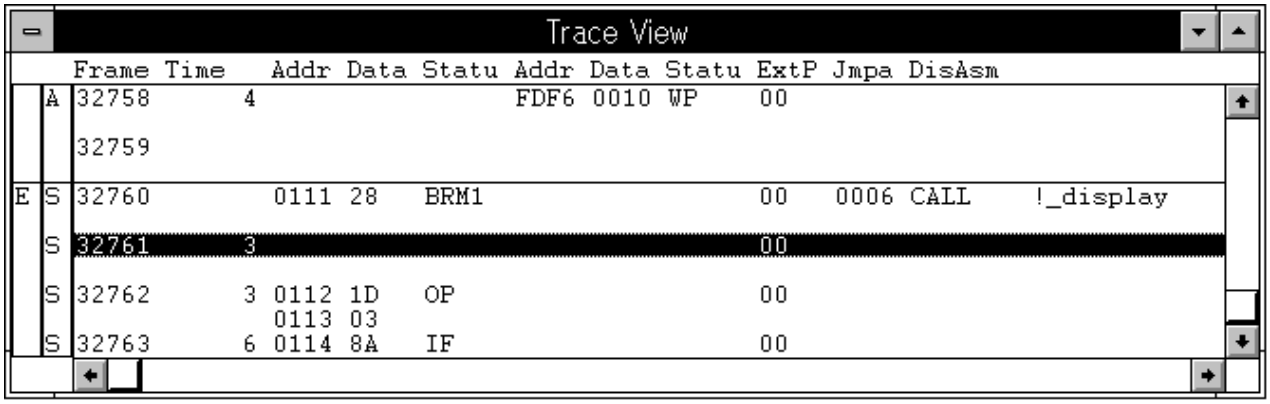


Fig. 5-42 Trace View Window

Description

The Trace View window displays trace results.

The tracer has a capacity of 32,768 frames and is of a ring structure. If the amount of data to be written is more than 32,768 frames, the oldest data is overwritten. In the window, frame numbers are assigned sequentially, with frame 0 being assigned to the oldest data item.

Block information is written to the tracer at breaks of user program execution. Block information is represented by a horizontal line displayed in each area. Block information is written when the preceding and current execution modes are as follows:

Mode of most recent execution	Current execution mode
Real-time execution	Real-time execution Step execution
Step execution	Real-time execution Step execution by changing the execution address

The Trace View window consists of the following components:

- Point mark area
- Trace mode view area
- Trace result view area

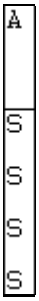
The function of each component is described below.

(1) Point mark area

This area displays the setting condition of each type of events. When an execution event or access fetch event is set for a trace address, the mark corresponding to the event type is displayed on the left of the trace address.

Mark	Description
E	An event condition is set.
L	The final phase of an event link is set.
B	A break event is set.
T	A trace event is set.
Ti	A timer event is set.
S	A snapshot event is set.
U	A stub event is set.
A	Two or more events are set.

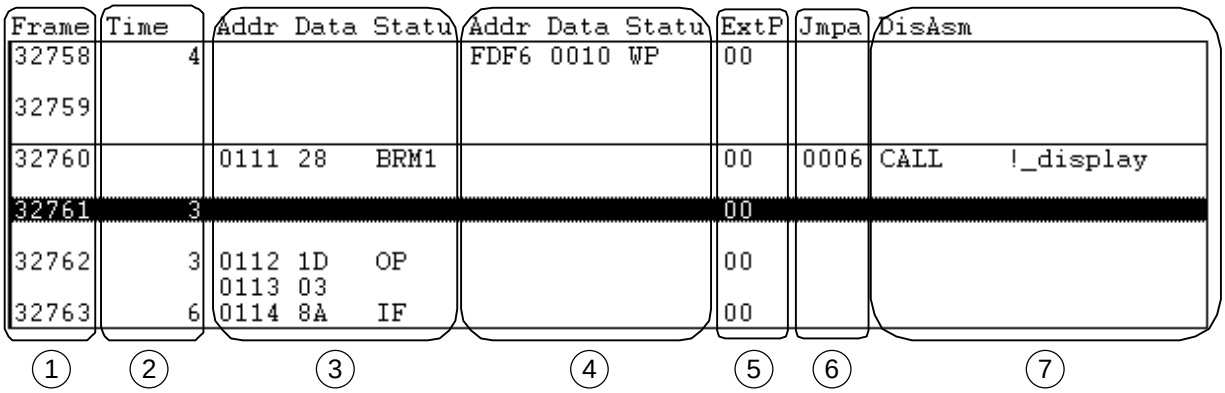
(2) Trace mode view area



The trace mode view area displays trace modes.

A: Overall trace or sectional trace
Q: Qualified trace
S: Step execution trace

(3) Trace result view area



The trace result view area displays trace results. Selecting this area enables the jump and window linkage functions to be used.

The window linkage function is used as follows:

1. Select the window to be linked among the items displayed by selecting **Operation→ Window Connect** from the **menu bar** of the Main window.

Item under Window Connect	Window to be linked
SourceText	Source window
Assemble	Assemble window
Memory	Memory window

2. Open the window selected in step 1 and the Trace View window.

3. Select a trace address in the trace result view area of the Trace View window, using the mouse.

4. The contents of the location identified by the address selected in step 3 are displayed in reverse video in the view area of the window selected in step 1.

The window linkage function differs from the jump function in that scrolling the view area in the Trace View window also scrolls the view area in the linked window accordingly.

The trace result view area displays the following information:

①Trace frame number
②Time tag
③Fetch access result
④Data access result
⑤External sense data
⑥Branch destination address
⑦Mnemonics

a. Trace frame number (Frame)

The trace frame number field displays trace frame numbers.

Valid range: 0 ≤ Trace frame number ≤ 32,767

To display trace frame numbers, choose from the items displayed by selecting **View→ Trace View→Frame** from the menu bar of the Main window.

b. Time tag (Time)

The time tag field displays the number of clock pulses taken by the target chip between the start of the execution of the immediately preceding trace address and the start of the execution of the current trace address.

Measurement range: 1 ≤ CPU clock rate ≤ 0xffffff

To display time tags, choose from the items displayed by selecting **View→Trace View→ Timetag** from the menu bar of the Main window.

c. Fetch access result (Address Data Statu)

The fetch access result field displays program fetch results and snapshot data. This field displays the following information depending on the fetch status displayed in the Statu field:

Statu	Information displayed	
BRM1	Program fetch results	Fetch of the first byte of the first instruction encountered after a branch
M1		Fetch of the first byte of an instruction
OP		Operation code fetch
IF		Invalid fetch
SNAP	Snapshot data	
Others	None	

For program fetch results, the following information is displayed:

Address	Fetch address
Data	Fetch data

To display fetch access results, choose from the following items in the menu bar of the Main window.

Item	Selection method
For fetch addresses	Choose from the items displayed by selecting **View→Trace View →Instruction Fetch Address.**
For fetch data	Choose from the items displayed by selecting **View→Trace View → Instruction Fetch Data.**
For fetch statuses	Choose from the items displayed by selecting **View→Trace View → Instruction Fetch Status.**

For snapshot data, the following information is displayed:

Item	Snapshot type	Information displayed
Address	Register	Register name
	SFR	SFR name
	Memory	Memory address
Data	Register	Register value
	SFR	SFR value
	Memory	Memory contents

d. Data access result (Address Data Statu)

The data access result field displays data access results.

Statu	Information displayed
VECT	Vector read
RWP	Data read or write by a user program
RP	Data read by a user program
WP	Data write by a user program
RWM	Data read or write by a macro service
RM	Data read by a macro service
WM	Data write by a macro service

Address	Address
Data	Data

To display data access results, choose from the following items in the menu bar of the Main window.

Item	Selection method
For addresses	Choose from the items displayed by selecting **View→Trace View →Memory access Address**.
For data	Choose from the items displayed by selecting **View→Trace View →Memory access Data.**
For statuses	Choose from the items displayed by selecting **View→Trace View →Memory access Status.**

e. External sense data (Ext P)

The external sense data field displays the input level of the external sense clips when a trace was performed. Each bit is assigned as follows:

External sense data	External sense clip number
Bit 7	No. 8
Bit 6	No. 7
Bit 5	No. 6
Bit 4	No. 5
Bit 3	No. 4
Bit 2	No. 3
Bit 1	No. 2
Bit 0	No. 1

To display external sense data, choose from the items displayed by selecting **View→ Trace View→External Probe** from the menu bar of the Main window.

f. Branch destination address (Jmpadd)

The branch destination address field displays the last address of a branch destination. This information is displayed only when the fetch status is **BRM1**.

To display a branch destination address, choose from the items displayed by selecting **View →Trace View→Jump Address** from the menu bar of the Main window.

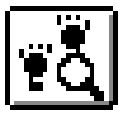
g. Mnemonics (DisAsm)

The mnemonics field displays disassembly results.
This information is displayed only when the fetch status is **BRM1** or **M1**.

To display mnemonics, choose from the items displayed by selecting **View→Trace View→ DisAssemble** from the menu bar of the Main window.

Icon

The Trace View window can be reduced to the following icon by clicking the button on the title bar:



Trace View

Trace pick-up dialog box

Specification dialog box (Modal)

Outline

The trace pick-up dialog box is used to specify conditions for displaying trace results in the Trace View window.

This dialog box can be opened in either of the following ways when the current window is the Trace View window:

- In the main window, select **View→Trace View→Pick Up...** from the **menu bar**.
- In the main window, press the **GRPH**, **V**, **T**, **and** **P** **keys**, in this order.

Window

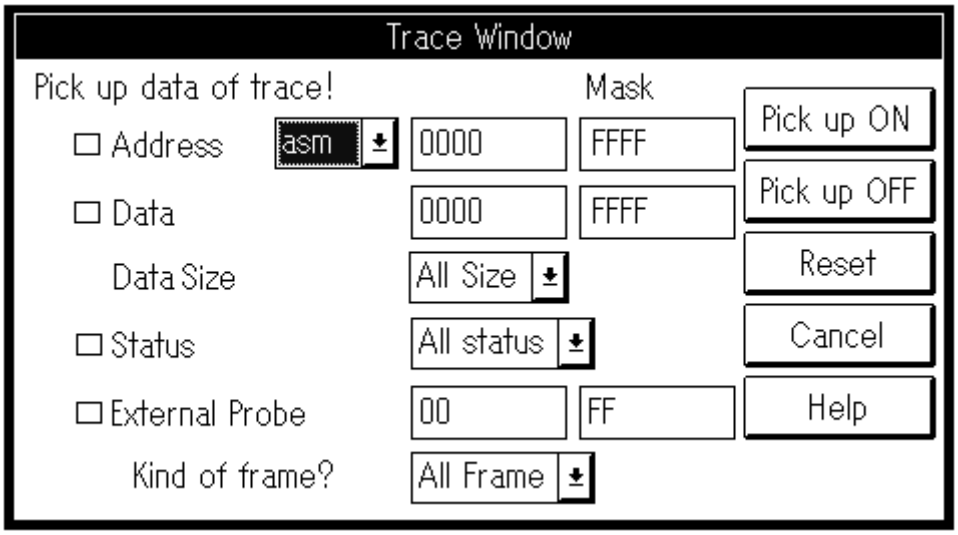


Fig. 5-43 Trace Pick-Up Dialog Box

Description

The trace pick-up dialog box is used to specify conditions for displaying trace results in the Trace View window.

The trace pick-up dialog box consists of the following components:

- Address condition specification area
- Data condition specification area
- Status condition specification area
- External sense data condition specification area
- Displayed-frame condition selection area
- Function buttons

The function of each component is described below.

(1) Address condition specification area

Use the address condition specification area to specify an address value as a pick-up condition. Specify an address specification mode, address value, and address mask value.

The address value and address mask value to be specified must be in the range of 0 to 0xffff.

Accepted address values vary with the specified mode as follows:

Mode	Input data
Source mode	Variable name, function name, or line number
Instruction mode	Immediate address or symbolic name

(2) Data condition specification area

Use the data condition specification area to specify a data value as a pick-up condition. Specify a data value, data mask value, and data size.

The data value and data mask value to be specified must be in the range of 0 to 0xffff. Accepted data sizes are listed below.

Data Size	Meaning
All Size	Perform retrieval for any data size.
Byte	Retrieve only frames accessed in bytes.
Word	Retrieve only frames accessed in words.

(3) Status condition specification area

Use the status condition specification area to specify the status as a pick-up condition. Accepted status conditions are listed below.

Status condition	Meaning
All status	Pick up all frames.
BRM1	Pick up only the first frame for which an M1 fetch operation was performed after program control branched.
M1	Pick up only the frames for which an M1 fetch operation was performed.
OP	Pick up only the frames for which a fetch operation was performed.
R	Pick up only the frames for which a read operation was performed.
RM	Pick up only the frames for which a read operation was performed in macro service processing.
RP	Pick up only the frames for which a read operation was performed in a user program.
RW	Pick up only the frames for which a read or write operation was performed.
RWM	Pick up only the frames for which a read or write operation was performed in macro service processing.
RWP	Pick up only the frames for which a read or write operation was performed in a user program.
VECT	Pick up only the frames for which a vector read operation was performed.
W	Pick up only the frames for which a write operation was performed.
WM	Pick up only the frames for which a write operation was performed in macro service processing.
WP	Pick up only the frames for which a write operation was performed in a user program.

(4) External sense data condition specification area

Use the external sense data condition specification area to specify an external sense data value as a pick-up condition. Specify a data value and data mask value.

The data value and data mask value to be specified must be in the range of 0 to 0xff.

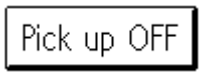
(5) Displayed-frame condition selection area

Select a displayed-frame type. Accepted frame types are listed below.

Frame type	Meaning
All Frame	Pick up all frames.
Step	Pick up only step execution frames.
Next	Pick up only Next step execution frames.
Real Time	Pick up only real-time execution frames.
Snap	Pick up only snapshot frames.

Buttons

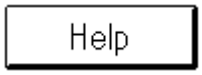
Displays only frames that satisfy specified conditions.



Displays all frames.

Ignores any selections and resets the initial state.

Closes the trace pick-up dialog box.



Opens the help window.

Register window

View/setting window

Outline

The Register window displays registers (general-purpose registers and control registers), and is used to modify their contents.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Register...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **R** **keys**, in this order.
- In the tool bar, click the  button.

Window

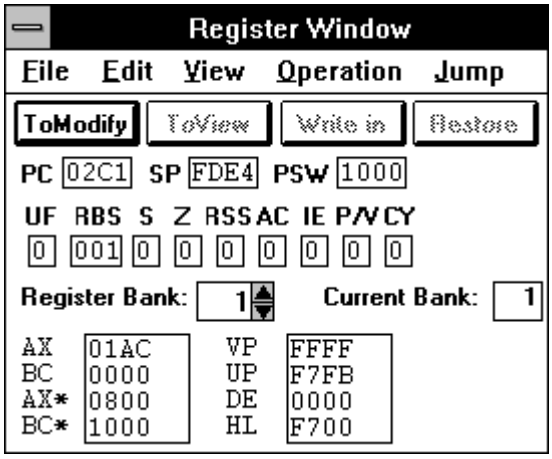


Fig. 5-44 Register Window

Description

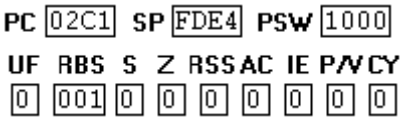
The Register window displays registers (general-purpose registers and control registers), and is used to modify their contents. This window can be set to view mode or modify mode.

The Register window consists of the following components:

- Control register view area
- Register bank selection area
- General-purpose register view area
- Menu bar
- Function buttons

The function of each component is described below.

(1) Control register view area



The control register view area displays control registers, and is used to modify their contents. To modify the contents of a control register, click the ToModify button. To write new data to the target device, click the Write in button after modification.

In addition to being used to display and modify the contents of control registers, this area serves as the jump pointer for the jump function.

The jump function causes a jump to the Source, Assemble, or Memory window, with the value of a selected control register being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

This function is used as follows (when jumping to the Source window):

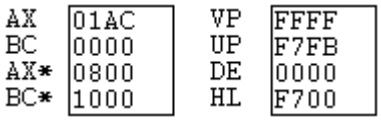
1. Select a control register.
2. In the Register window, select **Jump→SourceText...** from the **menu bar** or press the **GRPH**, **J**, and **S** keys, in this order. Or, press the **CTRL**+**U** shortcut keys.

(2) Register bank selection area

The register bank selection area displays the number of a general-purpose register bank, and is used to select a bank.

Item	Description
Register Bank:	Indicates the register bank to be displayed in the general-purpose register view area, and is used to select such a bank. To change the bank number, use the button.
Current Bank:	Displays the number of the register bank which is currently set in the target device (current bank).

(3) General-purpose register view area

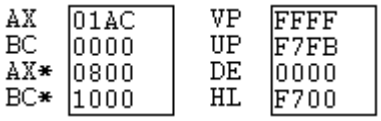


The general-purpose register view area displays the registers that belong to the bank indicated in the **Register Bank:** area, and is used to modify their contents.

To modify the contents of a general-purpose register, click the ToModify button. To write new data to the target device, click the Write in button after modification.

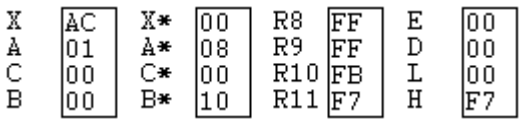
The mode of general-purpose register view can be switched between absolute-name view and functional-name view and also between register view and register pair view, by choosing from the items displayed by selecting **View** from the menu bar of the Register window.

Functional name and register pair view



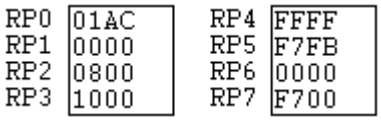
(Select **Functional Name** and **Register Pair**.)

Functional name and register view



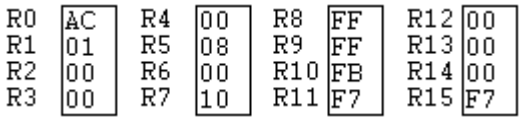
(Select **Functional Name** and **Register**.)

Absolute name and register pair view



(Select **Absolute Name** and **Register Pair**.)

Absolute name and register view



(Select **Absolute Name** and **Register**.)

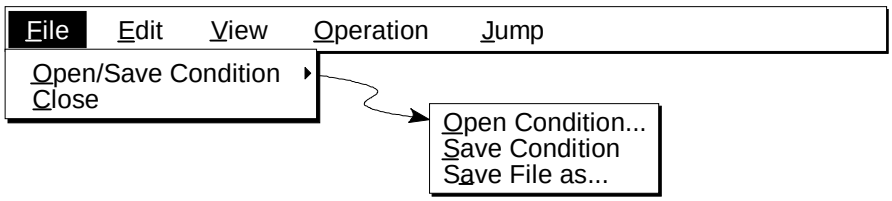
If **View→Functional Name** is selected from the **menu bar** of the Register window, the **A**, **X**, **B**, **C**, **AX**, and **BC** registers are displayed as follows depending on the value of the **RSS bit**:

Displayed name	When RSS = 0	When RSS = 1
X	X	R0
A	A	R1
C	C	R2
B	B	R3
X*	R4	X
A*	R5	A
C*	R6	C
B*	R7	B
AX	AX	RP0
BC	BC	RP1
AX*	RP2	AX
BC*	RP3	BC

Menu bar

Double-clicking the mouse button on an item of the menu bar opens the corresponding pull-down menu.

(a) File



Open/Save Condition ▸ Loads or saves the contents of general-purpose registers.

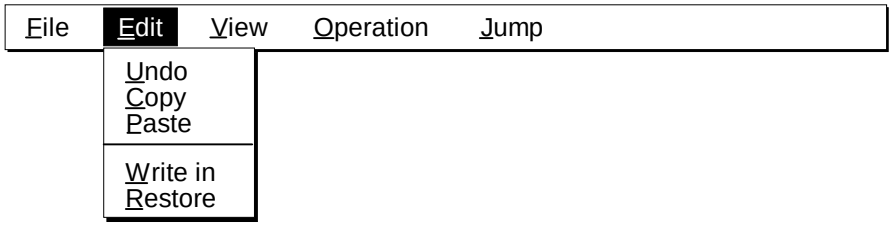
Open Condition... Opens the selected file for reference. The view file load dialog box is opened.

Save Condition Saves the contents of the window into a view file.

Save File as... Saves the contents of the window into a view file. The view file save dialog box is opened.

Close Closes the Register window.

(b) Edit



Undo Cancels the most recent editing.

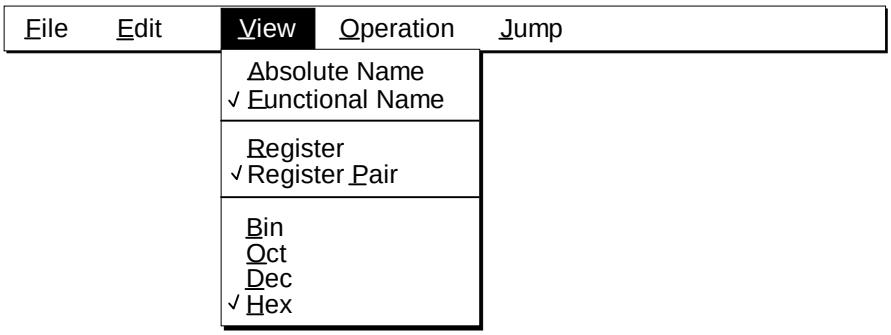
Copy Copies a selected character string into the clipboard buffer.

Paste Pastes the contents of the clipboard buffer at the point to which the text cursor is positioned.

Write in Writes the modified contents into the target device.

Restore Cancels the modified contents.

(c) View



Absolute Name Displays absolute register names.

Functional Name Displays functional register names.

Register Displays registers individually.

Register Pair Displays register pairs.

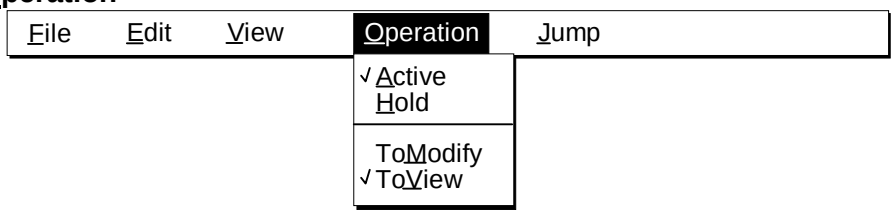
Bin Displays data in binary format.

Oct Displays data in octal format.

Dec Displays data in decimal format.

Hex Displays data in hexadecimal format.

(d) Operation



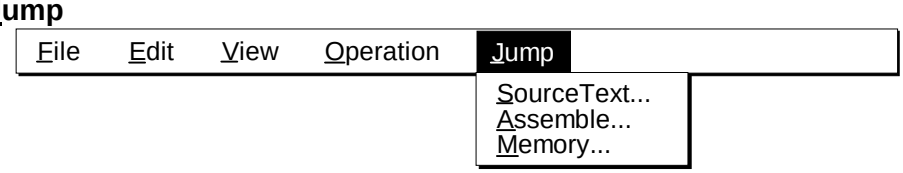
Active Sets the Register window to the active state.

Hold Sets the Register window to the hold state.

ToModify Sets the Register window to modify mode.

ToView Sets the Register window to view mode.

(e) Jump



SourceText... Sets the contents of the selected register as the jump address, and displays the source text and source line starting from that address: The Source window is opened.

Assemble... Sets the contents of the selected register as the jump address, and displays the disassembled text starting from that address: The Assemble window is opened.

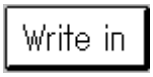
Memory... Sets the contents of the selected register as the jump address, and displays the memory contents starting from that address: The Memory window is opened.

Buttons

Switches the window to modify mode. This button can be clicked only in view mode. Clicking this button enables the contents of a register to be modified. When the window is placed in modify mode, the window is highlighted and this button is disabled.

To modify the contents of a register, click the current value of the register to display a text cursor, then enter a new value using a keyboard. Clicking the Write in button writes the new value into the target device.

Switches the window to view mode. This button can be clicked only in modify mode. When the window is placed in view mode, the window is not highlighted any more and this button is disabled.



Writes the new contents of a register into the target device.

Cancels modification. Clicking this button restores the initial contents of registers which have been modified in modify mode. If the Write in button has been clicked, modification before clicking the button is not canceled.

SFR window

View/setting window

Outline

The SFR window displays SFRs, and is used to modify their contents.

This window can be opened in any of the following ways:

- In the main window, select **Browse→Sfr...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **and** **F** **keys**, in this order.
- In the tool bar, click the button.

Window

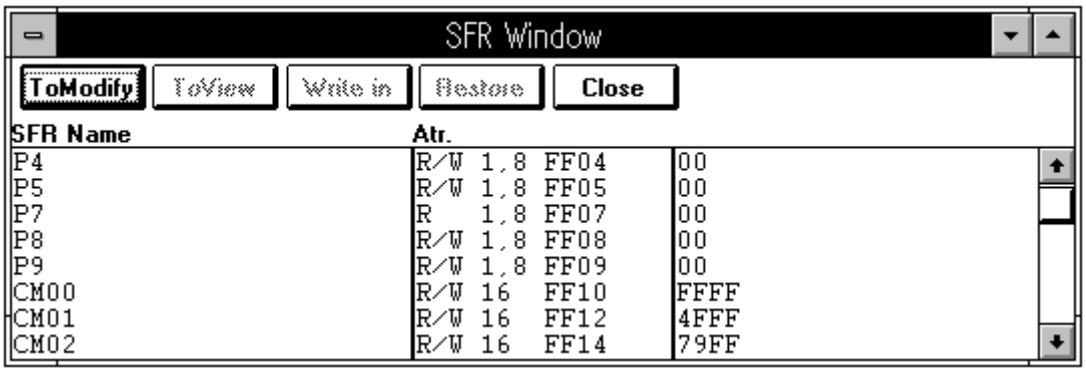


Fig. 5-45 SFR Window

Description

The SFR window displays SFRs, and is used to modify SFR contents.

SFRs that can only be read from are displayed in gray. Display in this color indicates that the contents of the registers cannot be modified.

Choosing from the items displayed by selecting **View→Sfr** from the **menu bar** of the Main window can determine how to display and read from SFRs.

Item under Sfr	Description
Address Sort	Specifies the display order. Without check mark: Alphabetical order With check mark (✓): In order of addresses
Pick Up	Displays only modified SFRs.
Attribute	Displays or hides the SFR attribute.
Compulsion Read	Forcibly reads read-protected SFRs.
Synchronize	Writes changed contents to the target device.

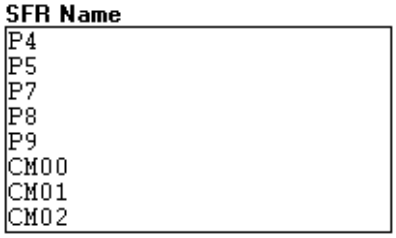
This window can be set to view mode or modify mode.

The SFR window consists of the following components:

- SFR name view area
- Attribute view area
- SFR contents view area
- Function buttons

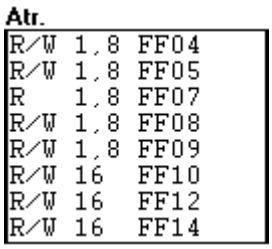
The function of each component is described below.

(1) SFR name view area



The SFR name view area displays SFR names.

(2) Attribute view area



The attribute view area displays the read/write attributes, access types, and addresses of SFRs.

Choosing from the items displayed by selecting **View→Sfr→ Attribute** from the menu bar of the Main window can determine whether the attribute view area is to be displayed.

Read/write attributes are classified as follows:

Attribute	Description
R	The SFR can only be read from. It is displayed in gray.
W	The SFR can only be written to.
R/W	The SFR can be both read from and written to.

Access types are classified as follows:

Access type	Description
1	The SFR can be accessed bit by bit.
8	The SFR can be accessed byte by byte.
16	The SFR can be accessed word by word.

(3) SFR contents view area

The SFR contents view area displays SFR contents, and is used to modify them.

Contents are displayed in different ways depending on the attribute of the SFR as follows:

Read-only SFR: The contents are displayed in gray.
Write-only SFR: -- is displayed.
Read/write SFR: The contents are displayed in black.
SFR modified by reading: ** is displayed.

To modify the contents of an SFR, click the ToModify button.
To write new data to the target device, click the Write in button after modification.

Buttons

ToModify

Switches the window to modify mode. This button can be clicked only in view mode. Clicking this button enables the contents of an SFR to be modified. When the window is placed in modify mode, the window is highlighted and this button is disabled.

To modify the contents of an SFR, click the current value of the SFR to display a text cursor, then enter a new value using a keyboard. Clicking the Write in button writes the new value into the target device.

ToView

Switches the window to view mode. This button can be clicked only in modify mode. When the window is placed in view mode, the window is not highlighted any more and this button is disabled.

Write in

Writes the new contents of an SFR into the target device.

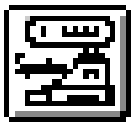
Cancels modification. Clicking this button restores the initial contents of SFRs which have been modified in modify mode. If the 

button has been clicked, modification before clicking the button is not canceled.

Closes the SFR window.

Icon

The SFR window can be reduced to the following icon by clicking the  button on the title bar:



SFR Window

Coverage window

View window

Outline

The Coverage window displays coverage results.

This window can be opened in either of the following ways:

- In the main window, select **Browse→Coverage→View...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **O**, **and** **V** **keys**, in this order.

Window

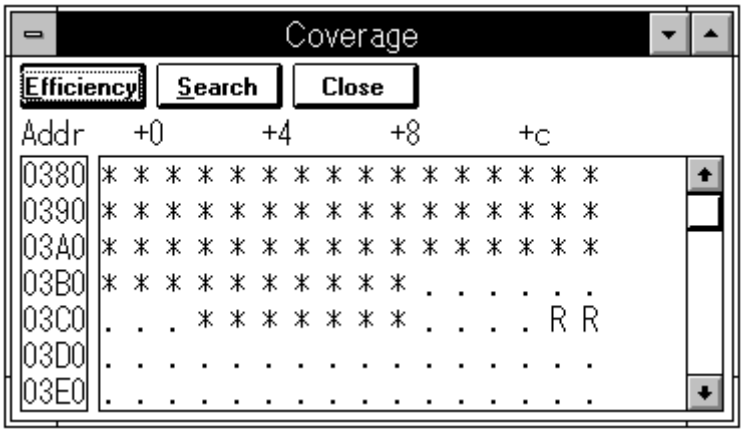


Fig. 5-46 Coverage Window

Description

The Coverage window displays coverage results.

To measure coverage, be sure to select **Execute→Coverage ON** from the **menu bar** of the Main window.

The Coverage window consists of the following components:

- Address view area
- Coverage view area
- Function buttons

The function of each component is described below.

(1) Address view area

The address view area displays coverage addresses.

In addition to displaying coverage addresses, this area can be used to perform the following function:

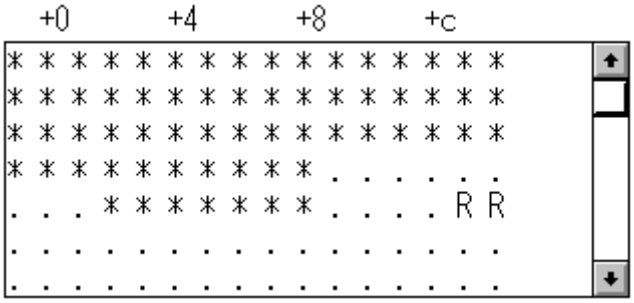
a. Jump function

The jump function causes a jump to the Source, Assemble, or Memory window, with a selected address being the jump pointer. The jump destination window is displayed from the location indicated by the jump pointer.

This function is used as follows (when jumping to the Source window):

1. Select an address.
2. In the Main window, select **Jump→SourceText...** from the **menu bar** or press the **GRPH**, **J**, and **S** keys, in this order. Or, press the **CTRL**+**U** shortcut keys.

(2) Coverage view area



The coverage view area displays coverage results.

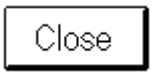
The symbols displayed in this area have the following meaning:

Symbol	Meaning
.	No execution, reading, or writing was performed.
R	Only reading was performed.
W	Only writing was performed.
*****	Only execution was performed.
$	Execution and reading were performed.
#	Execution and writing were performed.
A	Reading and writing were performed.
%	Execution, reading, and writing all performed.

Buttons

Displays coverage efficiency. The Coverage Efficiency View dialog box is opened.

Searches for coverage information to be displayed. The Find dialog box is opened.

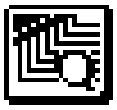


Closes the Coverage window.

When the Coverage window is closed, the Coverage Efficiency View and Coverage Condition Setting dialog boxes, if open, are also closed.

Icon

The Coverage window can be reduced to the following icon by clicking the  button on the title bar:



Coverage

Coverage Efficiency View dialog box

View dialog box (Modeless)

Outline

The Coverage Efficiency View dialog box displays coverage results in efficiency.

This dialog box can be opened in any of the following ways when the Coverage window is open:

- In the main window, select **Browse→Coverage→Efficiency...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **O**, **and** **E** **keys**, in this order.
- In the Coverage window, click the Efficiency button.
- In the Coverage Condition Setting dialog box, click the View button.

Closing the Coverage window also closes the Coverage Efficiency View dialog box.

Window

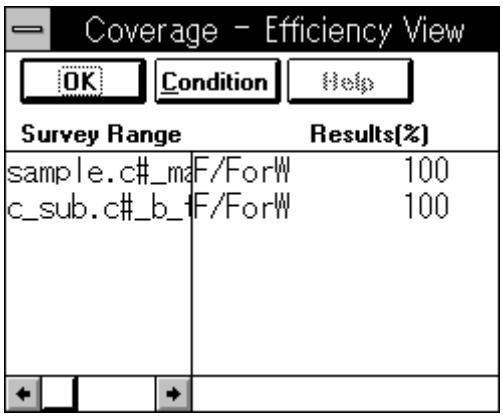


Fig. 5-47 Coverage Efficiency View Dialog Box

Description

The Coverage Efficiency View dialog box displays the coverage efficiency in the range specified in the Coverage Condition Setting dialog box.

The Coverage Efficiency View dialog box consists of the following components:

- Coverage range view area
- Coverage efficiency view area
- Function buttons

The function of each component is described below.

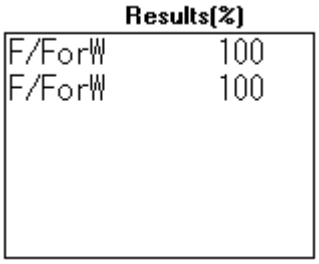
(1) Coverage range view area

The Coverage range view area displays the coverage efficiency measurement range.

This area displays the areas specified in the Coverage Condition Setting dialog box.

If the range has been specified with a function name, it is displayed in the format "file-name#_function-name."

(2) Coverage efficiency view area



The Coverage efficiency view area displays coverage efficiency.

Coverage efficiency is the percentage to which specified states (i.e., execution, reading, and writing) have occurred in the measurement range.

This area displays the coverage efficiency for each specified computation condition. The meaning of a computation condition is described below, after which the supported computation conditions are listed.

The computation condition is a combination of states such as F, R, W, and ALL.

F	Point at which a program was executed.
R	Point at which memory was read-accessed.
W	Point at which memory was write-accessed.
ALL	An entire function or an entire range of specified addresses.

A computation condition is expressed by combining states with a character string 'or' and a slash (/). The slash represents a relationship between a denominator and a numerator, while 'or' represents a set.

Example

$$F / ForW = \frac{F}{F \cup W} \qquad ForW / ForWorR = \frac{F \cup W}{F \cup W \cup R}$$

Computation conditions

Computation condition	Description
F/ForW	Ratio of program execution at a point where a program was executed or memory was write-accessed.
W/ForW	Ratio of memory writing at a point where a program was executed or memory was write-accessed.
F/ForR	Ratio of program execution at a point where a program was executed or memory was read-accessed.
R/ForR	Ratio of memory reading at a point where a program was executed or memory was read-accessed.
W/WorR	Ratio of memory writing at a point where memory was read- or write-accessed.
R/WorR	Ratio of memory reading at a point where memory was read- or write-accessed.
F/ForWorR	Ratio of program execution at a point where a program was executed or memory was read- or write-accessed.
W/ForWorR	Ratio of memory writing at a point where a program was executed or memory was read- or write-accessed.
R/ForWorR	Ratio of memory reading at a point where a program was executed or memory was read- or write-accessed.
ForW/ForWorR	Ratio of program execution or memory writing at a point where a program was executed or memory was read- or write-accessed.
ForR/ForWorR	Ratio of program execution or memory reading at a point where a program was executed or memory was read- or write-accessed.
WorR/ForWorR	Ratio of memory reading or writing at a point where a program was executed or memory was read- or write-accessed.
F/ALL	Ratio of program execution in the entire selected range.
W/ALL	Ratio of memory writing in the entire selected range.
R/ALL	Ratio of memory reading in the entire selected range.
ForW/ALL	Ratio of program execution or memory writing in the entire selected range.
ForR/ALL	Ratio of program execution or memory reading in the entire selected range.
WorR/ALL	Ratio of memory reading or writing in the entire selected range.
ForWorR/ALL	Ratio of program execution, memory reading, or writing in the entire selected range.

Buttons

Closes the Coverage Efficiency View dialog box.

Sets coverage efficiency view conditions. The Coverage Condition Setting dialog box is opened.

Opens the help window.

Coverage Condition Setting dialog box

Setting dialog box (Modeless)

Outline

The Coverage Condition Setting dialog box is used to set the range for coverage efficiency measurement.

This dialog box can be opened in any of the following ways when the Coverage window is open:

- In the main window, select **Browse→Coverage→Condition...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **O**, **and** **O** **keys**, in this order.
- In the Coverage Efficiency View dialog box, click the Condition button.

Closing the Coverage window also closes the Coverage Condition Setting dialog box.

Window

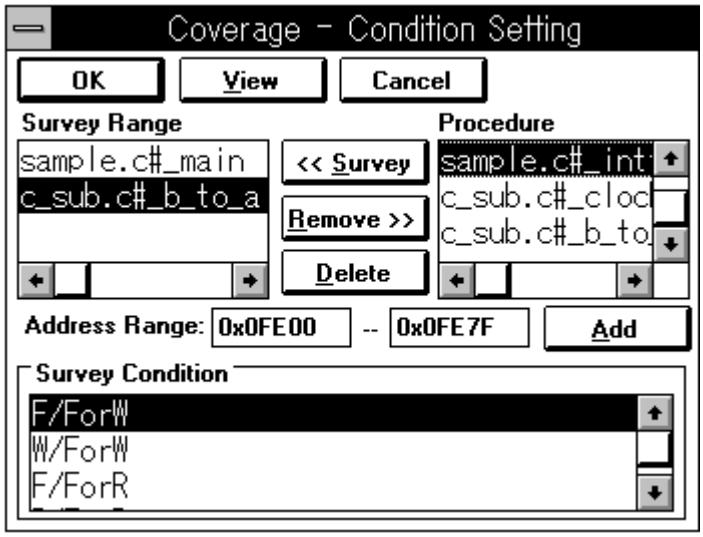


Fig. 5-48 Coverage Condition Setting Dialog Box

Description

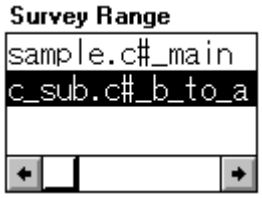
The Coverage Condition Setting dialog box is used to specify the information to be displayed in the Coverage Efficiency View dialog box.

The Coverage Condition Setting dialog box consists of the following components:

- Selection list view area
- Function list view area
- Address specification area
- Coverage condition specification area
- Function buttons

The function of each component is described below.

(1) Selection list view area



The selection list view area lists the currently selected items.

An item can be added to the selection list in either of the following ways:

a. Adding a function in the function list

1. Specify the coverage condition.
2. Select the function to be added in the function list view area, and click the [<< Survey] button.

To delete a function from the selection list, use the [Remove >>] button.

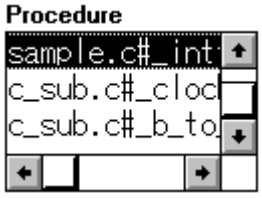
b. Adding an address range

1. Specify the coverage condition.
2. Enter an address range in the address specification area, and click the [Add] button.

To delete an address range from the selection list, use the [Delete] button.

To display the coverage efficiency for the items registered in the list, click the [OK] button. The contents of the list are displayed in the Coverage Efficiency View dialog box.

(2) Function list view area



The function list view area is used to specify a function as a coverage efficiency address condition.

This area displays the names of the functions registered in the load module file.

Select a function name and coverage condition (in the coverage condition specification area), and click the [<< Survey] button. The selected function is registered in the selection list view area. A function name that has been added to the list is displayed in gray.

(3) Address specification area

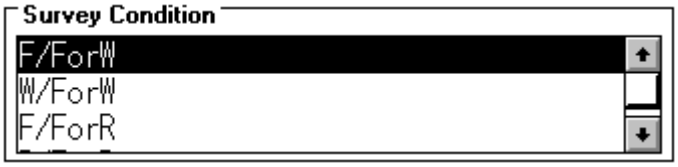
The address specification area is used to specify the address range for measuring coverage efficiency. Symbols can also be used to specify an address, as follows:

Function or variable	_fnc file#_fnc (for static function or variable)
SFR	sfrname
Line number in source text	file:no

fnc: Function or variable name sfrname: SFR name
file: File name no: Line number

When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

(4) Coverage condition specification area



The coverage condition specification area is used to select the computation condition for coverage efficiency. The coverage efficiency is computed according to a computation condition for each function or for a specified address range.
The computation condition is a combination of states such as F, R, W, and ALL.

F	Point at which a program was executed.
R	Point at which memory was read-accessed.
W	Point at which memory was write-accessed.
ALL	An entire function or an entire range of specified addresses.

A computation condition is expressed by combining states with a character string 'or' and a slash (/). The slash represents a relationship between a denominator and a numerator, while 'or' represents a set. Examples of compensation conditions follow.

Example

$$F / ForW = \frac{F}{F \cup W} \qquad ForW / ForWorR = \frac{F \cup W}{F \cup W \cup R}$$

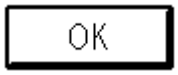
Computation conditions (1/2)

Computation condition	Description
F/ForW	Ratio of program execution at a point where a program was executed or memory was write-accessed.
W/ForW	Ratio of memory writing at a point where a program was executed or memory was write-accessed.
F/ForR	Ratio of program execution at a point where a program was executed or memory was read-accessed.
R/ForR	Ratio of memory reading at a point where a program was executed or memory was read-accessed.
W/WorR	Ratio of memory writing at a point where memory was read- or write-accessed.
R/WorR	Ratio of memory reading at a point where memory was read- or write-accessed.
F/ForWorR	Ratio of program execution at a point where a program was executed or memory was read- or write-accessed.
W/ForWorR	Ratio of memory writing at a point where a program was executed or memory was read- or write-accessed.
R/ForWorR	Ratio of memory reading at a point where a program was executed or memory was read- or write-accessed.

Computation conditions (2/2)

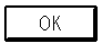
Computation condition	Description
ForW/ForWorR	Ratio of program execution or memory writing at a point where a program was executed or memory was read- or write-accessed.
ForR/ForWorR	Ratio of program execution or memory reading at a point where a program was executed or memory was read- or write-accessed.
WorR/ForWorR	Ratio of memory reading or writing at a point where a program was executed or memory was read- or write-accessed.
F/ALL	Ratio of program execution in the entire selected range.
W/ALL	Ratio of memory writing in the entire selected range.
R/ALL	Ratio of memory reading in the entire selected range.
ForW/ALL	Ratio of program execution or memory writing in the entire selected range.
ForR/ALL	Ratio of program execution or memory reading in the entire selected range.
WorR/ALL	Ratio of memory reading or writing in the entire selected range.
ForWorR/ALL	Ratio of program execution, memory reading, or writing in the entire selected range.

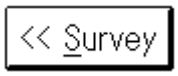
Buttons



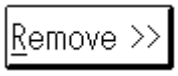
Displays the contents of the selection list view area in the Coverage Efficiency View dialog box.

Displays the coverage efficiency. The Coverage Efficiency View dialog box is opened.

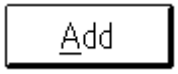
Cancels modification. Clicking this button restores the initial settings. If the  button has been clicked, modification before clicking the button is not canceled.



Registers a function name specified in the function list view area into the selection list view area.



Deletes a specified function name from the selection list view area.



Registers an address range specified in the address specification area into the selection list view area.

Deletes a specified address range from the selection list view area.

Coverage Memory Clear dialog box

Setting dialog box (Modal)

Outline

The Coverage Memory Clear dialog box is used to clear coverage results.

This dialog box can be opened in either of the following ways when the Coverage window is open:

- In the main window, select **Browse→Coverage→Clear...** from the **menu bar**.
- In the main window, press the **GRPH**, **B**, **O**, **and L** **keys**, in this order.

Window

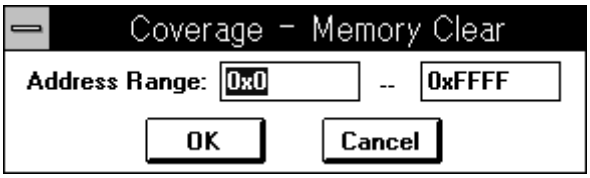


Fig. 5-49 Coverage Memory Clear Dialog Box

Description

The Coverage Memory Clear dialog box is used to clear coverage results.

The Coverage Memory Clear dialog box consists of the following components:

- Address specification area
- Function buttons

The function of each component is described below.

(1) Address specification area

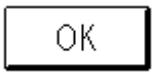
The address specification area is used to specify the coverage memory address range to be cleared. Symbols can also be used to specify an address, as follows:

Function or variable	_fnc file#_fnc (for static function or variable)
SFR	sfrname
Line number in source text	file:no

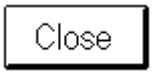
fnc: Function or variable name sfrname: SFR name
file: File name no: Line number

When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

Buttons



Clears the specified address range in coverage memory.



Closes the Coverage Memory Clear dialog box.

View file load dialog box

Selection dialog box (Modal)

Outline

The view file load dialog box is used to load a view file corresponding to the current window, and open a window for referencing the view file.

This dialog box can be opened as follows:

To load a view file for the Local Variable, Assemble, Memory, Stack, SFR, Trace View, or Coverage window:

- In the main window
  1. Select the relevant window as the current window.
  2. Select **File→Open...** from the **menu bar**.

- In the main window
  1. Select the relevant window as the current window.
  2. Press the **GRPH**, **F**, **and** **O** **keys**, in this order.

- Using short cut keys
  1. Select the relevant window as the current window.
  2. Press the **CTRL** + **O** **keys**.

To load a view file for the Variable window:

- In the Variable window
  Select **File→Open/save Condition→Open Condition...** from the **menu bar**.

- In the Variable window
  Press the **GRPH**, **F**, **O**, **and** **O** **keys**, in this order.

To load a view file for the Register window:

- In the Register window
  Select **File→Open/save Condition→Open Condition...** from the **menu bar**.

- In the Register window
  Press the **GRPH**, **F**, **O**, **and** **O** **keys**, in this order.

Window

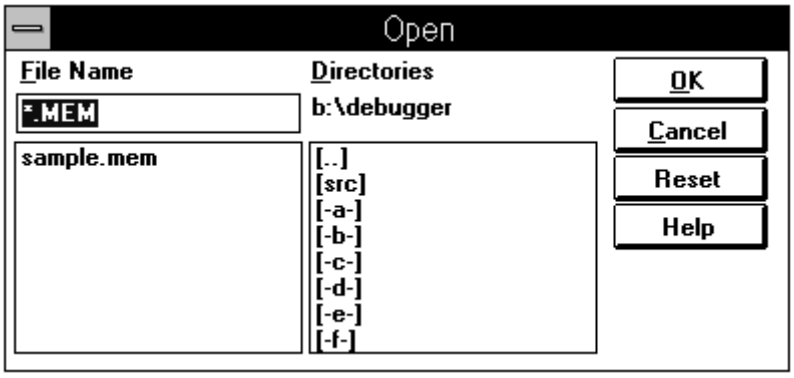
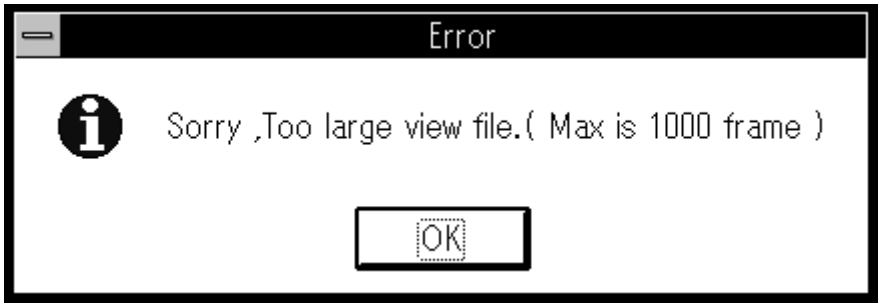


Fig. 5-50 View File Load Dialog Box

Description

The view file load dialog box loads a view file and opens a window for referencing the view file.

When the loaded view file contains more than 1000 lines, no reference window is opened. Instead, the following error message dialog box appears.



In that case, use other Windows applications to reference the view file.

The view file load dialog box consists of the following components:

- File selection area
- Path setting area
- Function buttons

The function of each component is described below.

(1) File selection area

Specify the name of the view file to be loaded.

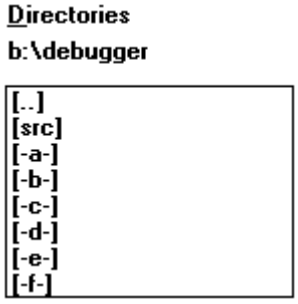
You can select a view file from the list by clicking it. The selected file name is highlighted and displayed in the area above the list.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the OK button.

The default extension for a view file name is as follows:

Window	Default extension
Variable window	VAR
Local Variable window	LOC
Assemble window	DIS
Memory window	MEM
Register window	REG
Stack window	STK
SFR window	SFR
Coverage window	COV
Trace View window	TVW
Event Manager	EVN

(2) Path setting area



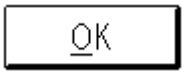
Specify the path of the view file to be loaded.

Double-clicking a path name in the list displays the view files under the path, in the view file list.

Directories and drives are distinguished in the list as follows:

[xxx]: Directory name
[-x-]: Drive name

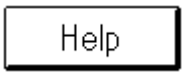
Buttons



Loads the specified view file.

Closes the view file load dialog box.

Ignores any selections and resets the initial state.



Opens the help window.

View file save dialog box

Selection dialog box (Modal)

Outline

The view file save dialog box is used to save the contents of the current window into a view file.

This dialog box can be opened as follows:

To save the contents of the Local Variable, Assemble, Memory, Stack, SFR, Trace View, or Coverage window:

- In the main window
  1. Select the relevant window as the current window.
  2. Select **File→Save As...** from the **menu bar**.

- In the main window
  1. Select the relevant window as the current window.
  2. Press the **GRPH**, **F**, **and** **A** **keys**, in this order.

To save the contents of the Variable window:

- In the Variable window
  Select **File→Open/save Condition→Save File as...** from the **menu bar**.

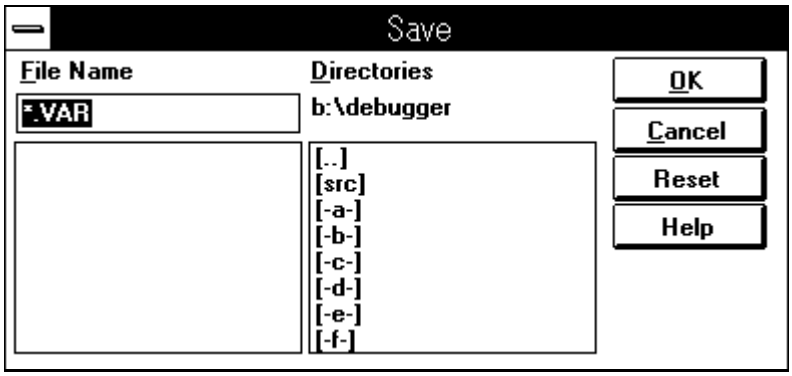
- In the Variable window
  Press the **GRPH**, **F**, **O**, **and** **A** **keys**, in this order.

To save the contents of the Register window:

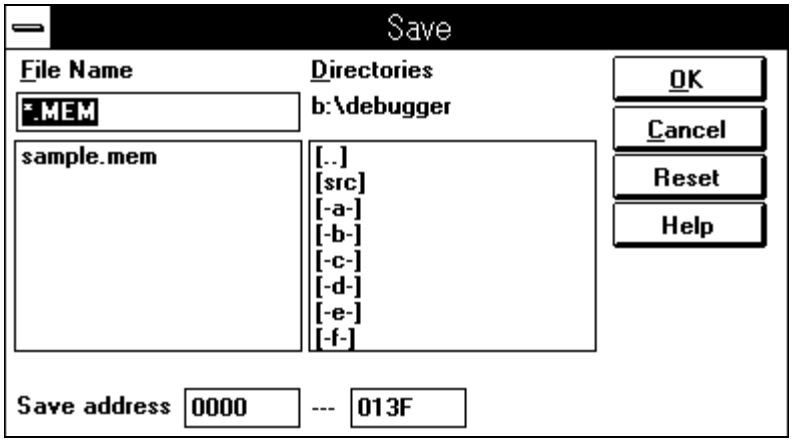
- In the Register window
  Select **File→Open/save Condition→Save File as...** from the **menu bar**.

- In the Register window
  Press the **GRPH**, **F**, **O**, **and** **A** **keys**, in this order.

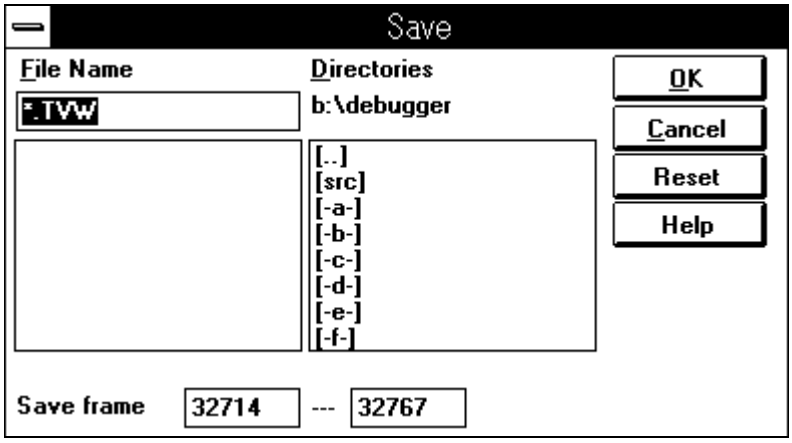
Window



When the current window is the Local Variable, Assemble, Variable, Stack, SFR, Coverage, or Register window



When the current window is the Memory window



When the current window is the Trace View window

Fig. 5-51 View File Save Dialog Box

Description

The view file save dialog box saves the contents of the current window into a view file.

The view file save dialog box consists of the following components:

- File selection area
- Path setting area
- Save range setting area
- Function buttons

The function of each component is described below.

(1) File selection area

Specify the name of the view file into which the contents of the current window will be saved.

Enter the file name from the keyboard. You can also select a view file from the list by clicking it, when that file is to be overwritten.

Double-clicking a file name in the list has the same effect as selecting the file name and clicking the OK button.

The default extension for a view file name is as follows:

Window	Default extension
Variable window	VAR
Local Variable window	LOC
Assemble window	DIS
Memory window	MEM
Register window	REG
Stack window	STK
SFR window	SFR
Coverage window	COV
Trace View window	TVW
Event Manager	EVN

(2) Path setting area

Specify the path under which the view file will be stored.

Double-clicking a path name in the list displays the view files under that path, in the view file list.

Directories and drives are distinguished in the list as follows:
[xxx]: Directory name
[-x-]: Drive name

(3) Save range setting area

This area is displayed only when the current window to be saved is the Memory or Trace View window.

a. When the current window is the Memory window

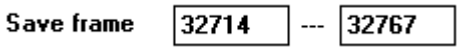
Specify the address range to be saved. Symbols can also be used to specify an address, as follows:

Function or variable	_fnc file#_fnc (for static function or variable)
Line number in source text	file:no

fnc: Function or variable name
file: File name no: Line number

When specifying a function or variable name, precede it with an underscore (_). A file name must be separated from a function or variable name with a sharp (#). A file name must be separated from a line number with a colon (:).

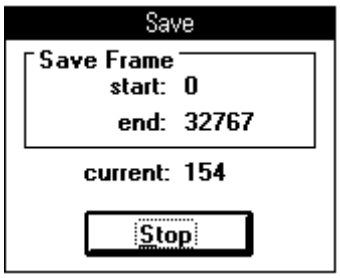
b. When the current window is the Trace View window



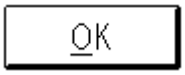
Specify the trace frame range to be saved.

Valid range: $0 \leq$ Frame number $\leq 32{,}767$

If a range of 100 or more frames is specified, the following message dialog box can be displayed to get hold of save conditions. A save operation can be aborted by pressing the Stop button in the Message dialog box.



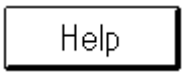
Buttons



Saves the contents of the current window into the specified view file.

Closes the view file save dialog box.

Ignores any selections and resets the initial state.



Opens the help window.

Error/Warning dialog box	Confirmation dialog box (Modal)

Outline

The Error/Warning dialog box appears when an error or warning occurs. The user shall confirm the message displayed in this dialog box.

Window

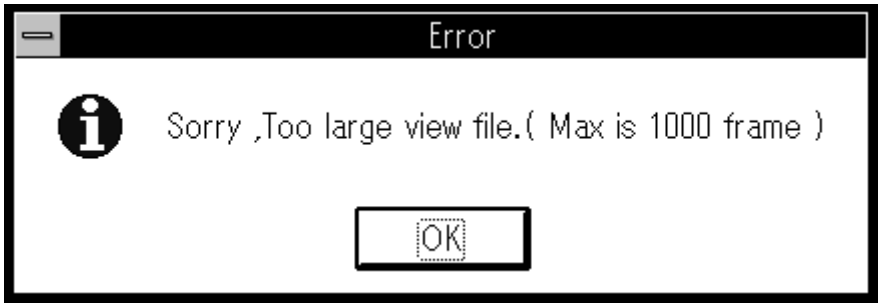


Fig. 5-52 Error/Warning Dialog Box

Description

The Error/Warning dialog box displays the reason for the error or warning.

Buttons

OK	Closes the Error/Warning dialog box.

Reset Debugger dialog box

Confirmation dialog box (Modal)

Outline

The Reset Debugger dialog box is used to initialize the debugger or emulation CPU.

This dialog box can be opened in either of the following ways:

- In the main window, select **Execute→CPU Reset...** from the **menu bar**.
- In the main window, press the **GRPH**, **X**, **and U** **keys**, in this order.

Window

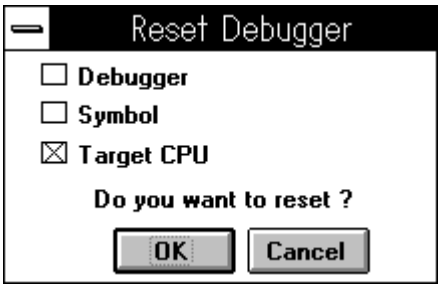


Fig. 5-53 Reset Debugger Dialog Box

Description

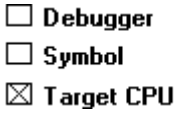
Specify what is to be initialized, using the check boxes. The default setting is resetting only the emulation CPU.

The Reset Debugger dialog box consists of the following components:

- Address selection area
- Function buttons

The function of each component is described below.

(1) Reset system selection area

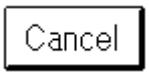


Select the system to be initialized.

Item	Description
Debugger	Restarts the entire debugger system.
Symbol	Initializes the entire symbolic information which has been loaded and registered.
Target CPU	Resets only the emulation CPU.

Buttons

Initializes the debugger or emulation CPU according to the selection.



Closes the Reset Debugger dialog box.

About dialog box

View dialog box (Modal)

Outline

The About dialog box is used to display the version of the debugger.

This dialog box can be opened in either of the following ways:

- In the main window, select **Help→About...** from the **menu bar**.
- In the main window, press the **GRPH**, **H**, **and** **A** **keys**, in this order.

Window

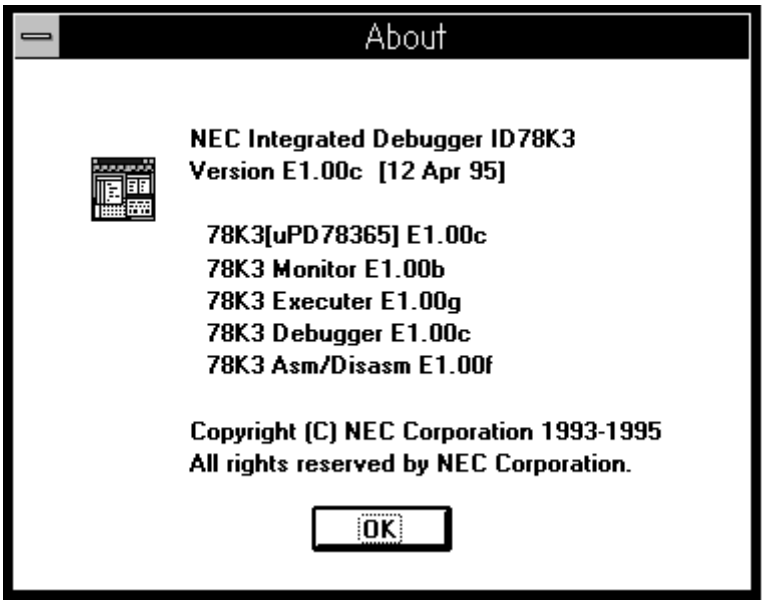


Fig. 5-54 About Dialog Box

Description

The About dialog box displays version information including the versions of the debugger and device file.

Buttons

Closes the About dialog box.

Exit Debugger dialog box

Confirmation dialog box (Modal)

Outline

The Exit Debugger dialog box is used to exit the debugger.

You can save the current debugging environment into a project file when exiting the debugger.

This dialog box can be opened in either of the following ways:

- In the main window, select **File→Exit** from the **menu bar**.
- In the main window, press the **GRPH**, **F**, **and** **X** **keys**, in this order.

Window

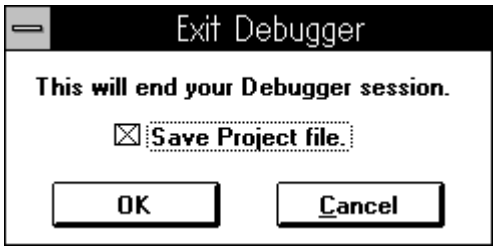


Fig. 5-55 Exit Debugger Dialog Box

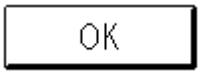
Description

Select whether to save the current debugging environment into a project file when exiting the debugger, using the check box. By default, the current environment is not saved.

When the box is checked, clicking the OK button opens the project file save dialog box, saves the current debugging environment into a specified project file, closes all windows, then terminates the debugger.

When the box is not checked, clicking the OK button closes all windows, then terminates the debugger.

Buttons



When the box is checked, opens the project file save dialog box, saves the current debugging environment into a specified project file, closes all windows, then terminates the debugger.

When the box is not checked, closes all windows, then terminates the debugger.

Closes the Exit Debugger dialog box without any processing.

Chapter 6 Explanation of Debugger Functions

This chapter explains each of the functions of the integrated debugger.

6.1 System Operating Modes

The system operating mode indicates the current operating state of the system, i.e., whether the "user program execution (emulation) functions" and the "analyzer functions" are activated.

6.1.1 Operating Mode Types

There are three system operating modes. Commands are restricted according to the current system operating mode.

Break mode

The execution of both "user program execution (emulation) functions" and "analyzer functions" is stopped.

Emulation mode

The "user program execution (emulation) functions" are activated but execution of the "trace functions" is stopped. This mode is used when the user does not want the execution of a user program to be stopped. In this mode analyzer functions other than the tracer, such as coverage and timer measurement, are activated.

Trace mode

Both the "user program execution (emulation) functions" and "analyzer functions" are activated.

6.1.2 System Operating Mode

The current system operating mode is displayed in the status bar of the main window.

System operating mode	CPU	Tracer
Break mode	Stopped	Stopped
Emulation mode	Activated	Stopped
Trace mode	Activated	Activated

6.1.3 System Operating State

The relationship between an emulation CPU and the analyzer functions is shown in the following figure. The relationship shown here is just one example.

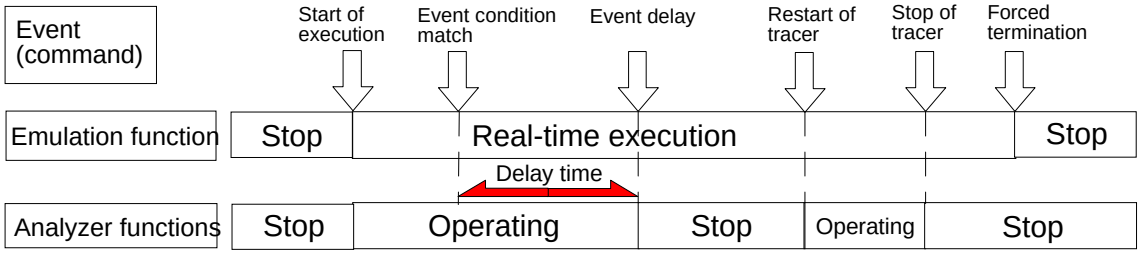


Fig. 6-1 Example of System Operating State

6.2 Using the Basic Functions

The following sections explain the basic functions of the debug functions supported by the debugger.

6.2.1 Clock Selection Function

The clock selection function is used to specify the clock source to be supplied to an emulation CPU (target device). One of three clock sources can be selected:

Fixed clock (25 MHz) within the in-circuit emulator (Internal)
User-established clock (External)
Clock obtained by multiplying the user-established clock pulses (Multiple Ext)

The desired clock source can be selected when the debugger is first started or from within the Configuration dialog box.

There are two ways of specifying a user-established clock to supply any clock:

a. Create a clock on the target system and supply the clock to an emulation CPU via the emulation probe.
b. Install a transmitter in the clock socket on the break board in the main section of the in-circuit emulator and supply the clock to an emulation CPU.

When the clock source is changed, the emulation CPU is reset.

6.2.2 Mapping Functions

There are five mapping functions, which allow the following settings for address areas other than SFRs. These settings can be made when the debugger is first started or from within the Configuration dialog box.

Internal ROM

The memory area specified as internal ROM represents the internal ROM of the target device. With this type of mapping, the target device can access the memory of the in-circuit emulator. When the target device writes data into this memory area, a write protect break is generated.

Internal RAM

The memory area specified as internal RAM represents the internal RAM of the target device. With this type of mapping, the target device can access the memory of the in-circuit emulator.

User area mapping (Target)

The memory area specified for user area mapping allows access to the memory of the target system. With this type of mapping, the target device can access the memory of the target system.

IE alternate ROM (Emulation ROM)

The memory area specified for IE alternate ROM represents the additional ROM of the target device. With this type of mapping, the target device can access the memory of the in-circuit emulator. When the target device writes data into this memory area, a write protect break is generated.

IE alternate RAM (Emulation RAM)

The memory area specified for IE alternate RAM represents the additional RAM of the target device. With this type of mapping, the target device can access the memory of the in-circuit emulator.

6.2.3 Reset Functions

The reset functions are used to reset the entire in-circuit emulator system or the emulation device only.

Reset of entire system (Debugger)
Reset of emulation device only (Target CPU)

These functions can be specified from within the Reset Debugger dialog box.

6.2.4 Load Function

The load function is used to load specified files such as the debugging environment, object, load module, and symbol files individually.

There are two types of files to be loaded: View files for screen reference and information files for updating information in the debugger.

A view file stores screen information that was current when the file was saved. When a view file is loaded, a reference window opens.

The following table lists the types of view files.

File	Window	Explanation
Variable view file (file name: XXXXXXXX.VAR)	Variable window	Stores variable information.
Assemble view file (file name: XXXXXXXX.DIS)	Assemble window	Stores assemble information.
Memory view file (file name: XXXXXXXX.MEM)	Memory window	Stores memory information.
Register view file (file name: XXXXXXXX.REG)	Register window	Stores register information.
Stack trace view file (file name: XXXXXXXX.STK)	Stack window	Stores stack trace information.
SFR view file (file name: XXXXXXXX.SFR)	SFR window	Stores SFR information.
Local variable view file (file name: XXXXXXXX.LOC)	Local Variable window	Stores local variable information.
Trace view file (file name: XXXXXXXX.TVW)	Trace View window	Stores trace information.

The following table lists the types of information files.

File	Window	Explanation
Object file (file name: XXXXXXXX.HEX)	Load Module dialog box	Stores the object code (in Intel standard hexadecimal format) of a user program.
Symbol table file (file name: XXXXXXXX.SYM)	Load Module dialog box	Stores the symbols defined by the user on a source for a user program.
Load module file (file name: XXXXXXXX.LNK)	Load Module dialog box	Stores the object code, symbols, and source information of a user program.
Project file (file name: XXXXXXXX.PRJ)	Project file load dialog box	Stores a debugging environment. This file contains the setting information for the following windows: • Configuration dialog box • Extended Option dialog box • Load Module dialog box • Source window • Source Path dialog box • Assemble window • Memory window • Stack window • SFR window • Local Variable window • Trace View window • Event Manager • Event Link dialog box • Break dialog box • Trace dialog box • Snap-Shot dialog box • Stub dialog box • Timer dialog box • Register window • Variable window • Coverage window • Coverage Efficiency View dialog box
Event setting file (file name: XXXXXXXX.EVN)	Event Manager	Stores event setting information.

6.2.5 Emulation Execution Functions

The emulation execution functions are used to start "user program execution (emulation)" by an emulation CPU and the analyzers. The functions are classified according to the emulation execution mode, as follows:

Real-time execution functions:

Go (button)	Performs real-time execution. The program breaks upon the occurrence of a break event.
Return (button)	Performs real-time execution until control returns to the calling function.
Go & Go	Performs real-time execution. The program breaks upon the occurrence of a break event and, after window updating, real-time execution is performed.
Go & Come	Performs real-time execution until a specified address or source line is reached. No break event occurs while a program is being executed.
CPU Reset & Go	Resets the emulation CPU then performs real-time execution.

Non-real-time execution functions:

Step (button)	If Source Mode is selected Performs step execution at the source level. If Instruction Mode is selected Performs real-time execution at the instruction level.
Next (button)	If Source Mode is selected Performs Next step execution at the source level. If Instruction Mode is selected Performs Next step execution at the instruction level.
Slowmotion	Performs step execution continuously.

Real-time execution functions

There are four real-time execution functions: "Go" which executes a user program until the occurrence of a break event; "Go & Go" which updates each window and reexecutes a user program upon the occurrence of a break event; "Go & Come" which executes a user program up until a specified point, when it breaks; and “Return” which executes a user program until control returns to the calling function.

Go command (button)

With real-time execution by the Go command, the user program is executed starting from a specified address and is stopped upon the occurrence of a break event. Each analyzer becomes operable when the program is executed, and is executed or stopped depending on the events.

The relationship between the CPU, tracer, timer, and coverage during real-time execution by the Go command is shown in the following figure.

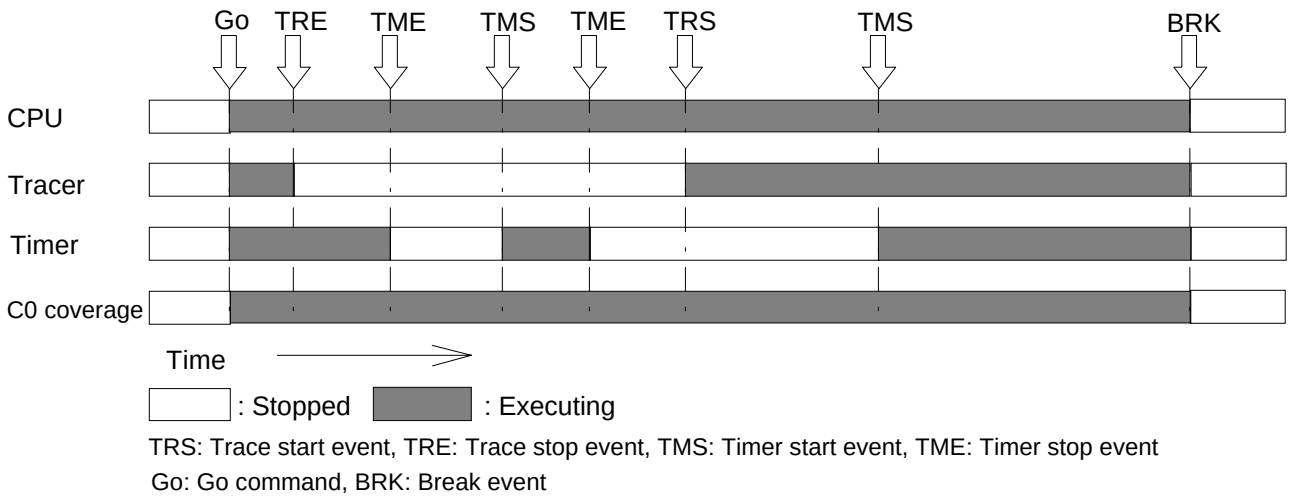


Fig. 6-2 Example of System Operating State (Go)

Return command

With real-time execution by the Return command, real-time execution is performed until control returns to the calling function. If no calling function exists, no operation is performed.
The concept of real-time execution by the Return command is shown in the following figure.

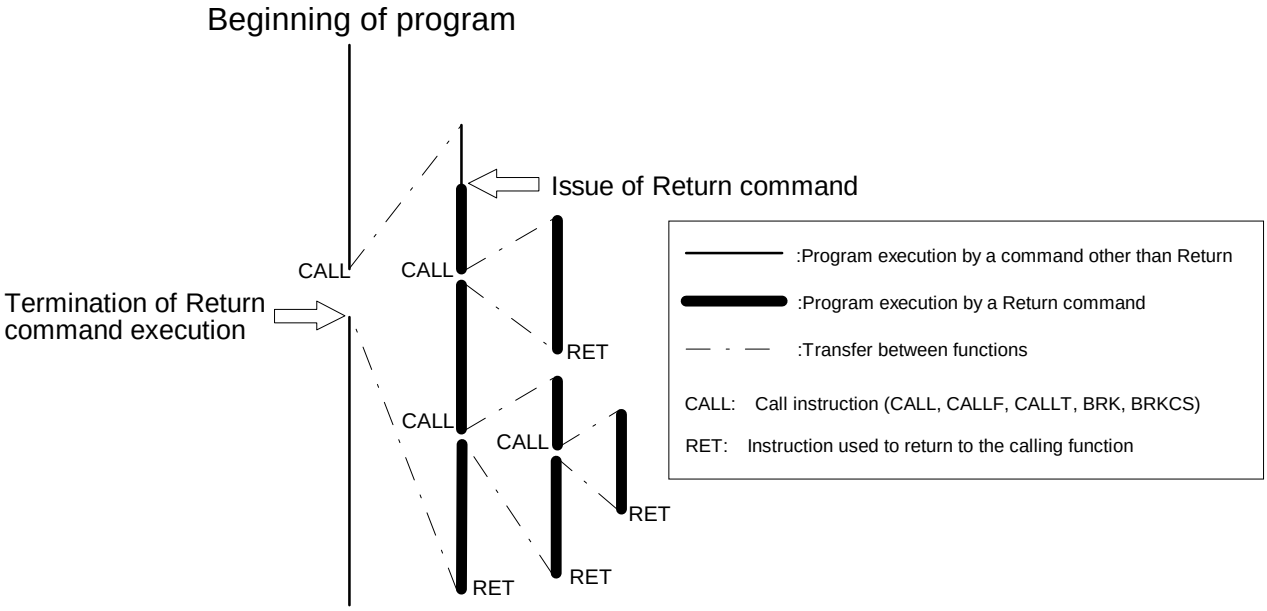


Fig. 6-3 Concept of Return Command Execution

The Return command realizes real-time execution by specifying an execution break at the return address of a function. The following diagram shows the relationships among the CPU, tracer, timer, and coverage used in real-time execution based on the Return command.

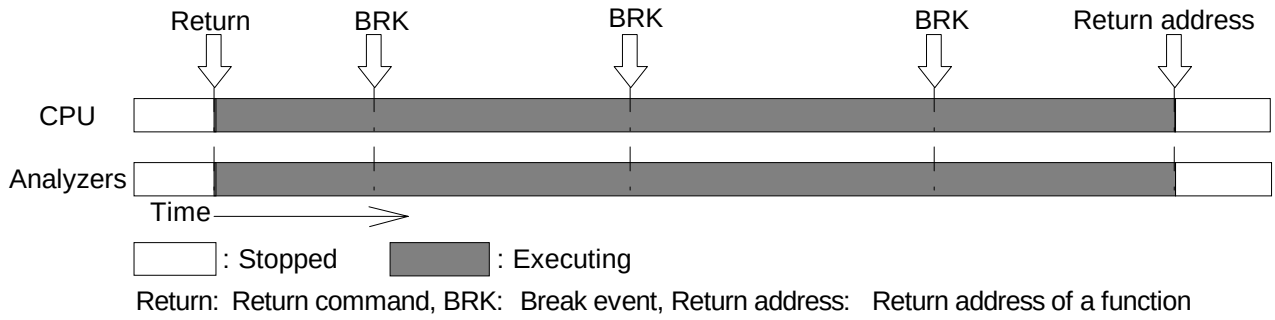


Fig. 6-4 Example of System Operating State (Return)

Go & Go command

With real-time execution by the Go & Go command,

(1) The user program is executed starting from a specified address.
(2) The program is stopped when a break event occurs.
(3) Each window is updated.
(4) The program is reexecuted starting from the address where it was stopped.
(5) (2), (3), and (4) are repeated until the Stop command is issued.

Each analyzer becomes operable when the program is executed, and executed or stopped depending on the events.

The relationship between the CPU and the analyzers during real-time execution by the Go & Go command is shown in the following figure.

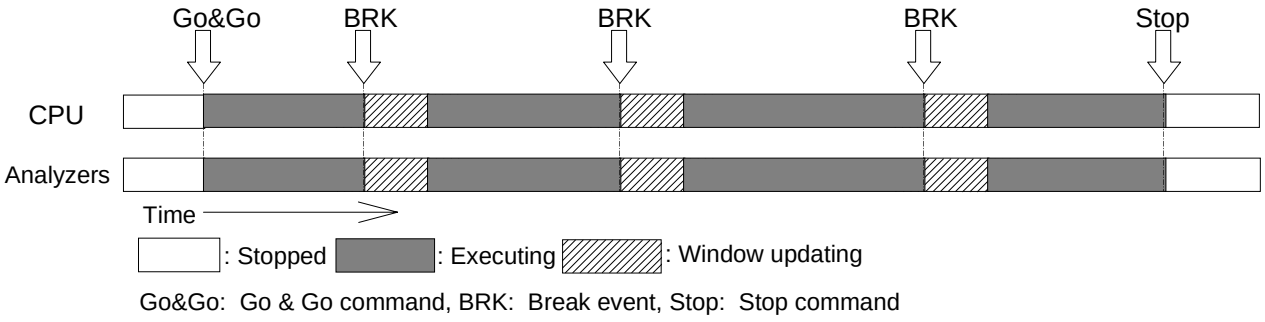


Fig. 6-5 Example of System Operating State (Go & Go)

Go & Come command

With real-time execution by the Go & Come command,

(1) In either the Source or Assemble window, position the cursor to the point where you want the program to be stopped.
(2) Issue a Go & Come command to execute the user program starting from the address stored in the program counter.
(3) The program is executed up to the point to which the cursor is positioned, after which it breaks.

The program does not break upon the occurrence of a break event during the course of program execution.

The relationship between the CPU and the analyzers during real-time execution by the Go & Come command is shown in the following figure.

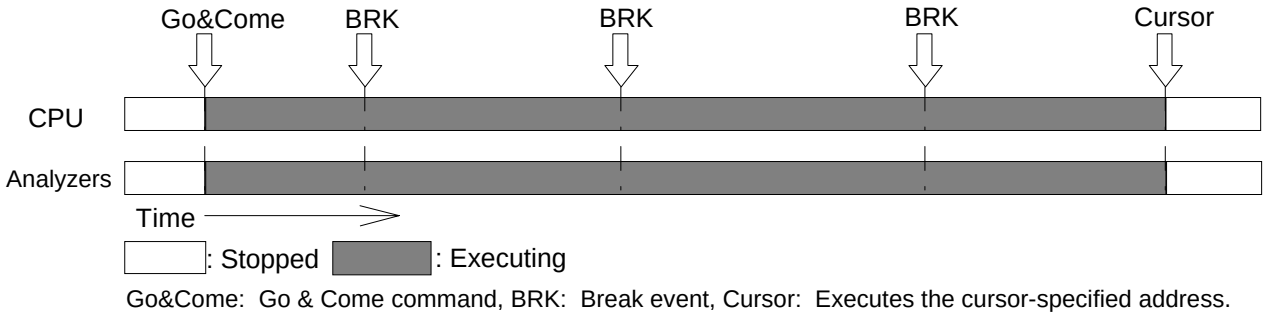


Fig. 6-6 Example of System Operating State (Go & Come)

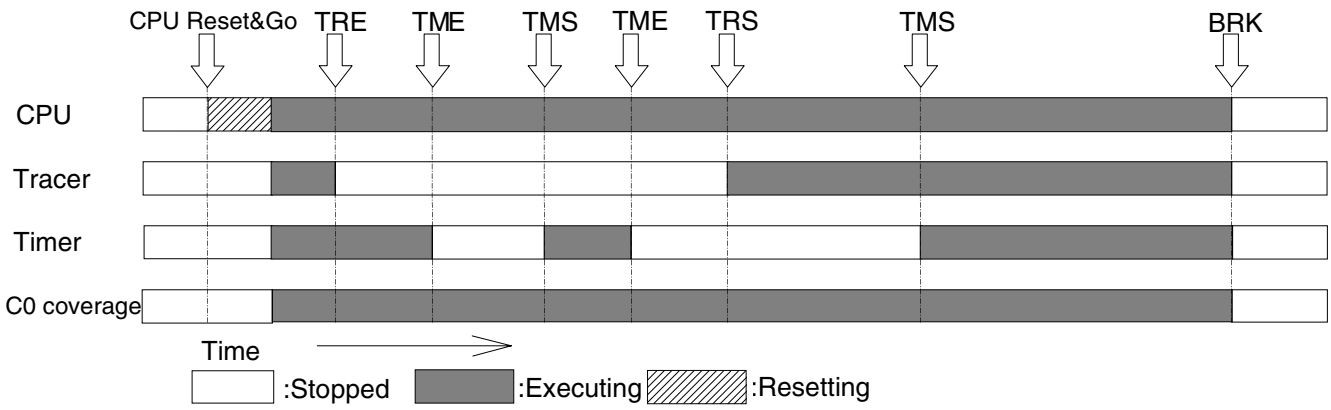
CPU Reset & Go command

With CPU Reset & Go execution,

(1) The emulation CPU is reset.
(2) The program is executed with a reset vector.

The emulation CPU is reset before the program is executed. Subsequently, the operation is the same as that of the Go command.

The relationship between the CPU and the analyzers during CPU Reset & Go execution is shown in the following figure.



TRS: Trace start event, TRE: Trace stop event, TMS: Timer start event, TME: Timer stop event
CPU Reset&Go: CPU Reset & Go command, BRK: Break event

Fig. 6-7 Example of System Operating State (CPU Reset & Go)

Non-real-time execution functions

The non-real-time execution functions are roughly divided into "Step" which performs step execution; "Next" which performs Next step execution; and "Slowmotion" which performs continuous step execution.

Step command

With step execution by the Step command,

- In Source Mode,
  Step execution is performed for one line starting from a specified source line.
- In Instruction Mode,
  One instruction is executed starting from a specified address.

After execution, each window is updated.

The relationship between the CPU and the analyzers during step execution by the Step command is shown in the following figure.

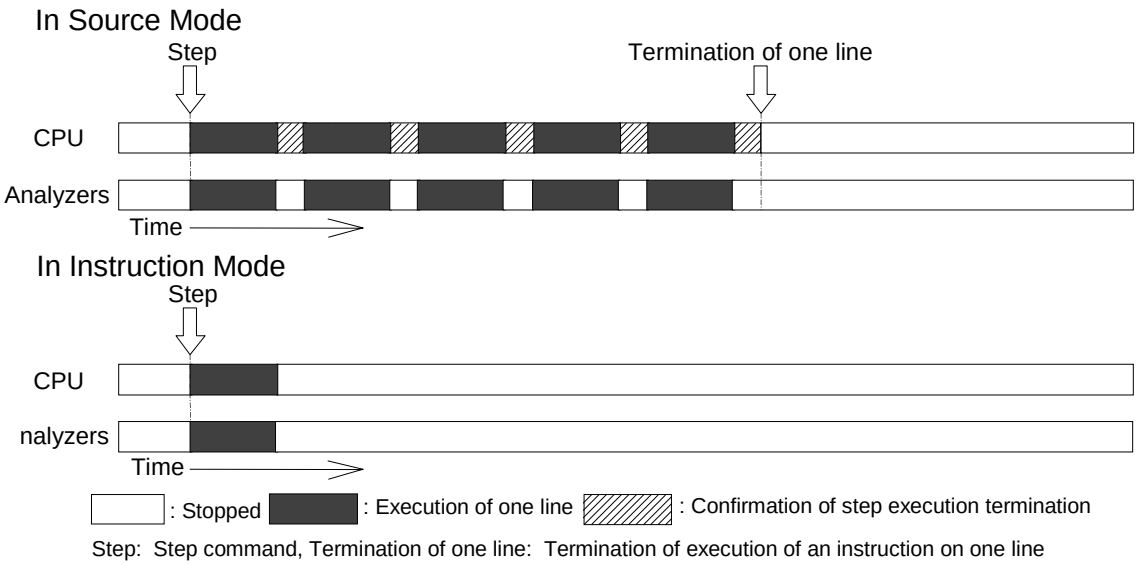


Fig. 6-8 Example of System Operating State (Step)

Next command

Next step execution by the Next command differs depending on whether a call statement or a statement other than a call statement is executed. Depending on the debug mode, the following can be used as a call instruction:

- In source mode,
  Line calling a function
- In instruction mode,
  CALL, CALLF, CALLT, BRK, and BRKCS instructions

The operation of the Next command is explained below:

- If a call statement is executed,
  an execution break is set for the line or instruction immediately after the call statement. Then, real-time execution is performed.
- If a statement other than a call statement is executed,
  the same processing as that resulting from execution of the Step command is performed.

The concept of Next step execution by the Next command is as shown in the figure below.

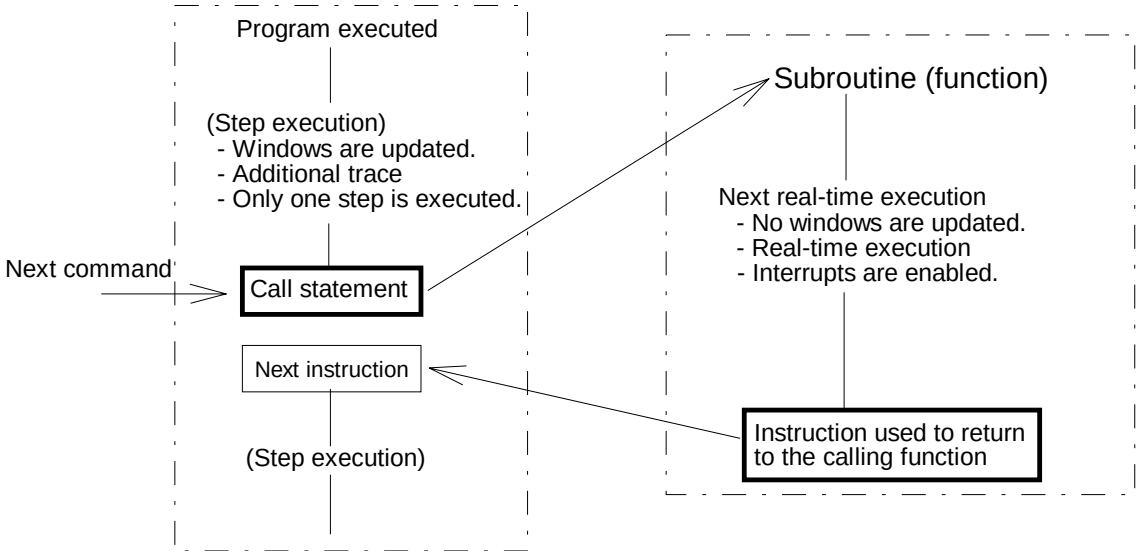


Fig. 6-9 Concept of Next Step Execution

Slowmotion command

With step execution by the Slowmotion command,

(1) Starting from a specified address, step execution is performed line-by-line if the debug mode is source mode, or instruction-by-instruction if it is instruction mode.
(2) Each window is updated.
(3) (1) and (2) are repeated until the Stop command is issued.

6.2.6 Break Functions

The break functions are used to stop "user program execution (emulation)" by an emulation CPU and the "analyzers (tracer, timer, and coverage)".

There are basically five break functions:

- Event detection break
- Software break
- Temporary break
- Break caused by a condition being satisfied during step execution
- Forcible break
- Fail-safe break

The table below lists the relationship between these "break functions" and the "emulation execution functions".

	Event detection break	Temporary break	Software break	Break due to a step condition being satisfied	Forcible break	Fail-safe break
Real-time execution by Go command	○	×	○	×	○	○
Real-time execution by Go & Go command	○	×	○	×	○	○
Real-time execution by Go & Come command	×	○	×	×	○	○
Real-time execution by CPU Reset & Go command	○	×	○	×	○	○
Non-real-time execution by Step command	×	×	×	○	○	○
Non-real-time execution by Return command	×	×	×	○	○	○
Non-real-time execution by Next command	×	×	×	○	○	○
Non-real-time execution by Slowmotion command	×	×	×	×	○	○

○: Applied ×: Not applied

Event detection break

Event detection break enables the stopping of the execution of a user program upon the detection of a specified event condition. This break is effective for the Go, Go & Go, and CPU Reset & Go commands. For the Go & Go command, each window is redrawn and the program is reexecuted after an event detection break. A break event must be set as an event detection condition by using the Event Set dialog box, Event Manager, or Break dialog box.

Temporary break

Temporary break enables the stopping of the execution of a user program upon the detection of a specified address. Once execution of the user program stops, the temporary break point is cleared.

Software break

Software break enables the stopping of the execution of a user program upon the detection of a specified address. Compared with event detection break or temporary break, which uses one hardware resource for each event condition, software break allows breakpoints to be set at multiple addresses for one event condition.

Since software break allows breakpoints to be set in multiple addresses with a single event detector, it performs the following processing before and after execution of a user program:

Immediately before execution:

(1) The instruction at the location where a software break is set is changed to a CALLT instruction immediately before execution of a user program.
(2) In the event detector, a break point is set in the vector address of the above CALLT instruction.

After a break:

(1) Correction of the analyzers (tracer and coverage) used when the CALLT instruction was executed
(2) The CALLT instruction is restored to the original instruction.

Since a software break uses a CALLT instruction as stated above, a vector table must be left open for the CALLT instruction. These settings are made from within the Extended Option dialog box.

The analyzers are corrected after the break. Therefore, the result of executing the CALLT instruction may not be displayed correctly in the tracer.

Break caused by a condition being satisfied during step execution

Break caused by a condition being satisfied during step execution enables the execution of a program to be stopped when the termination condition for a command (Step, Next, Return, and Slowmotion) is satisfied. The processing time is longer than that for real-time execution because execution, stop, and condition confirmation is performed for each instruction.

Forcible break

Forcible break enables the execution of a user program to be stopped forcibly. This break is effective for all commands used to execute a program.

There are two types of forcible break:

1. Stop command
   Forcibly stops the execution of a user program.
2. Reset command
   Forcibly stops the execution of a user program, then resets the devices.

The Stop command is useful for temporarily stopping a program. The Reset command is useful for executing a program from the beginning.

Fail-safe break

Fail-safe break enables the execution of a user program to be stopped forcibly if the program attempts to perform execution that is prohibited, using the memory and registers.

There are three types of fail-safe break:

1. Non-map break
   Generated when an attempt is made to access a non-mapping area.
2. Write-protect break
   Generated when an attempt is made to write to write-protected memory such as ROM.
3. SFR illegal access break
   Generated when an attempt is made to gain illegal access to the SFR area.

A fail-safe break can be generated by either of two causes: an error in the user program or an environment setting error in the debugger.

Caution: If a program is written up to the vicinity of the boundary between the mapping and non-mapping areas or of the internal RAM area for which fetching is not allowed, a non-map break may be generated.

More specifically, a non-map break may be generated if:

Maximum address in a mapping area - 5 ≤ Program address ≤ Maximum address in a mapping area

or

Minimum address in unfetchable internal RAM - 5 ≤ Program address

≤ Minimum address in unfetchable internal RAM

Example: For a mapping area of 0x00000 to 0x03FFF and a non-mapping area of 0x04000 and above, a non-map break may be generated if a program is written to addresses 0x3FFA to 0x3FFF.

If a non-map break is generated, it is related to prior fetch and a queue buffer.

6.2.7 Trace Functions

The trace functions allow a user to write data such as external sense clip values and data obtained by memory access during the execution of a user program to "trace memory" in real-time mode. By opening the Trace View window, the data written to trace memory can be referenced to check the progress of the target program.

The main functions of trace execution and view are as follows. Trace conditions can be set in the **Trace dialog box**. Trace data view can be selected from the **trace pick-up dialog box** and with **View→Trace View** in the **menu bar** of the main window.

Trace operations:
- Operation during real-time execution
- Operation during step execution
- Operation during Next step execution

Trace condition setting function (Trace dialog box)
- Trace mode specification
- Qualified trace setting
- Section trace setting

Trace data view, format, and retrieval condition setting
- Trace data view specification
- Trace data retrieval condition setting

Relationship between trace execution and trace memory

Trace is divided into trace blocks according to the corresponding period:

①Block from real-time execution to a break caused by the occurrence of an event
②Block from emulation execution until the generation of a fail-safe break
③Block from emulation execution until application of a forcible break
④Block of step execution groups which are consecutive as viewed from the program

Trace memory is a ring buffer having 32K frames. If trace exceeds 32K frames, the oldest frame is overwritten with the newest trace data.

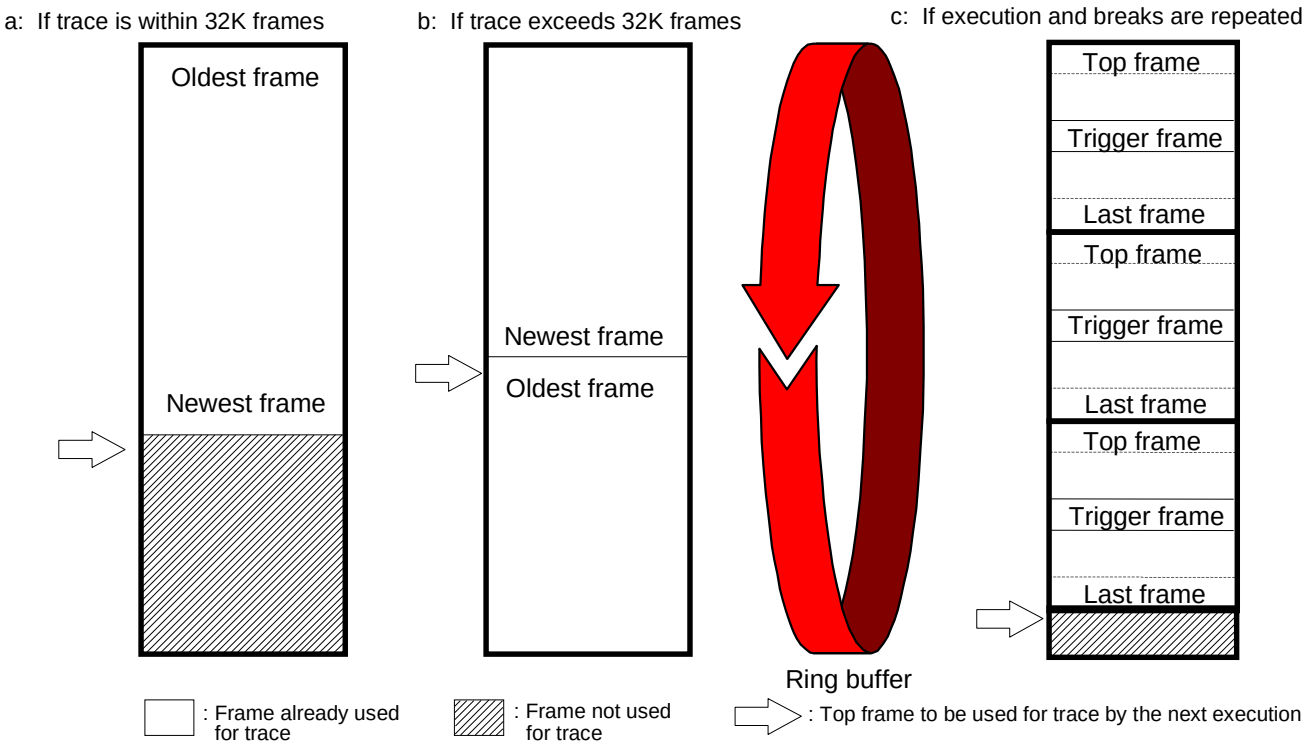


Fig. 6-10 Concept of Trace Memory

Trace operation

The operation of the tracer varies with the execution mode:

Operation during real-time execution

The tracer starts tracing with real-time execution specified and stops tracing when the event condition (including a delay condition), specified as a break condition from within the Trace dialog box, is satisfied.

Operation during step execution

The tracer operates upon the execution of each step, trace information for each step being added to the tracer every time a step is executed.

Operation during Next step execution

If an instruction to be executed is other than a call instruction (CALL, CALLF, CALLT, BRK, and BRKCS)

The tracer performs the same processing as that performed during step execution.

If the instruction to be executed is a call instruction (CALL, CALLF, CALLT, BRK, and BRKCS)

The tracer performs the same processing as that performed during real-time execution.
Real-time execution is canceled when control returns to the calling function.

Trace condition setting function (Trace dialog box)

The following specifications enable the user to specify the conditions for tracing. If these specifications are omitted, "all trace" is assumed, i.e., trace information is recorded for each instruction of a user program.

Trace mode specification

Specifies whether to execute all trace or conditional trace. There are two types of conditional trace: Qualified trace and sectional trace. The trace mode specified with this setting becomes valid.

Qualified trace setting

Qualified trace involves tracing only when a previously specified address is executed or accessed. The condition to be specified can be created in the Event Set dialog box.

Sectional trace setting

In sectional trace, tracing begins when a previously specified trace start condition is satisfied and stops when a stop condition is satisfied. In other words, it is a range-specified trace. The condition to be specified can be created in the Event Set dialog box.

Trace data view, format, and retrieval condition setting

The display/hiding of data in the Trace View window, as well as the view conditions, can be set.

Trace data view specification

Specifying trace data view enables effective use of the screen. When specifying trace data view, display/hiding can be specified for the following data items with **View→Trace View** in the **menu bar** of the main window.

Table 6-1 Explanation of Trace Data View

Menu bar item	Item in Trace View window	Explanation
Frame	Frame	Time sequence in which data is written to trace memory using trace memory frame numbers. (Range: 00000-32767)
TimeTag	Time	Execution time for each frame
Instruction Fetch Address	Addr	Fetch address
Instruction Fetch Data	Data	Fetch data
Instruction Fetch Status	Statu	Fetch status • M1: Fetch of the first byte of an instruction • BRM1: Fetch of the first byte of the first instruction after a branch • OP: Fetch of operation code • IF: Invalid fetch
Memory access Address	Addr	Access address
Memory access Data	Data	Access data
Memory access Status	Statu	Access status • VECT: Interrupt handling • RWP: Data read/write by a user program • RP: Data read by a user program • WP: Data write by a user program • RWM: Data read/write by a macro service • RM: Data read by a macro service • WM: Data write by a macro service
External Probe	ExtP	External sense clip data
Jump Address	Jumpa	Jump address field
DisAssemble	DisAsm	Disassemble result

Setting of trace data retrieval conditions

Trace data retrieval conditions can be specified. Retrieval conditions can be specified by selecting any or all of the items listed in the table below, in the trace pick-up dialog box. The specified data becomes valid by clicking the Pick up ON button.

Table 6-2 Trace Retrieval Items

Specifiable item	Explanation	Specifiable range	Default
Address	Address to be retrieved	0-0FFFFFH	0XXXXXH
Data	Data to be retrieved	0-0FFFFFFFH	0XXXXXXXH
Status	Status to be retrieved • All status: All status • M1: Fetch of the first byte of an instruction • BRM1: Fetch of the first byte of the first instruction after a branch • OP: Fetch of operation code • IF: Invalid fetch • VECT: Interrupt processing • RWP: Data read/write by a user program • RP: Data read by a user program • WP: Data write by a user program • RWM: Data read/write by a macro service • RM: Data read by a macro service • WM: Data write by a macro service	Same as left	All status
External Probe	External sense clip data to be retrieved	0-0FFH	0XXH
Kind of frame?	Type of data to be retrieved • All Frame: All frames • Step: Step execution frames • Next: Frames other than step execution frames	Same as left	All Frame

6.2.8 Snapshot Function

The snapshot function is used to interrupt real-time execution and output specified information to trace memory as snap data upon the detection of a specified snap event during real-time execution. Once the output of information to trace memory has been completed, the function resumes real-time execution.

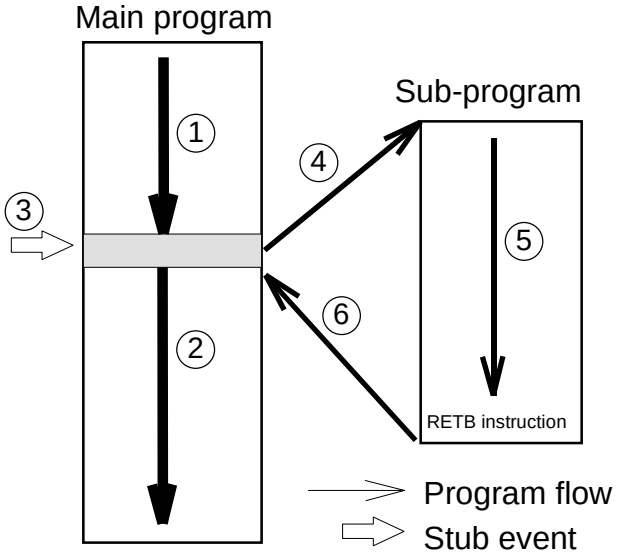
The following can be specified as snap data:

- Data in a general-purpose register
- Data in an SFR
- Data in memory

Snap data output can be specified for a maximum of sixteen points.

6.2.9 Stub Function

The stub function is used to insert a simple program into the user program currently being checked. When a stub event is generated, the user program stops and another user-supplied program is executed. A RETB instruction must be specified at the end of the sub-program to be executed when a stub event is generated. Failing to specify this instruction can cause a malfunction.



The stub function enables a user to easily insert a program.

When the stub function is not used
① is executed.
② is executed.

When the stub function is used
① is executed.
A stub event is generated at ③ and the main program breaks.
In ④, IE pushes the current address into the stack and overwrites the PC with the entry address of the sub-program.
⑤ is executed.
In ⑥, control is returned to the main program by execution of the RETB instruction which appears at the end of the sub-program.
② is executed.

Fig. 6-11 Concept of Stub Function

6.2.10 Event Setting and Detection Functions

The event setting and detection functions are used to set the conditions for stopping "user program execution" by an emulation CPU and for starting and stopping "trace operation" and "timer measurement" by the analyzers.

The following event condition setting and detection functions are provided:

Event detection condition setting functions
- Bus event condition setting function
- Execution event condition setting function
- Event condition link setting function

Event detection condition integration functions
- Break event setting
- Trace event setting
- Timer event setting

Event condition setting function

The event condition setting function is used to set, in the event condition register, the conditions for stopping "user program execution" by an emulation CPU and for starting and stopping "trace operation" and "timer measurement" by the analyzers. The event detection condition specified by the event detection condition setting function (Event Set dialog box and Event Link dialog box) does not become effective until set in the event mode register by the event detection condition integration function (Event Manager, Break dialog box, Trace dialog box, Snap-Shot dialog box, Stub dialog box, and Timer dialog box).

Three functions are provided for setting the event detection conditions:

Bus event condition setting function

When the user program accesses specified memory or data is input to an external sense clip, it is possible to set this as an event detection condition in the bus event condition register.

(a) Bus event condition register

Using the Event Set dialog box, up to seven conditions can be set in the bus event condition register (BRA).

(b) Event conditions

The following items can be set as event detection conditions:

Item	Status	Explanation
Address	Address	Address (address range)
	Mask	Address mask
Status	Fetch	Program fetch
	Program Read	Read by a program
	Program Write	Write by a program
	Program R/W	Read/write by a program
	Macro Read	Read by a macro service
	Macro Write	Write by a macro service
	Macro Read/Write	Read/write by a macro service
	Program/Macro Read	All read
	Program/Macro Write	All write
	Program/Macro R/W	All read/write
	VECT	Vector read by interrupts
	ALL(No Condition)	All accesses
Data	Data	Data value
	Mask	Data mask value
External	External	External sense data value
	Mask	External sense data mask value
Pass count	Pass count	Pass count value

Caution: When Fetch is selected as a Status condition, an event is generated with the execution of a Fetch operation. That is, an event is generated prior to the fetched program being executed. To generate an event with the actual execution of the fetched program, use the execution event condition setting function.

Execution event condition setting function

When the user program attempts to execute the instruction at a specified address and data is simultaneously input to an external sense clip, it is possible to set this as an event detection condition in the execution event condition register.

(a) Execution event condition register

Using the Event Set dialog box, up to three conditions can be set in the execution event detection register (BRS).

(b) Event conditions

The following items can be set as event detection conditions:

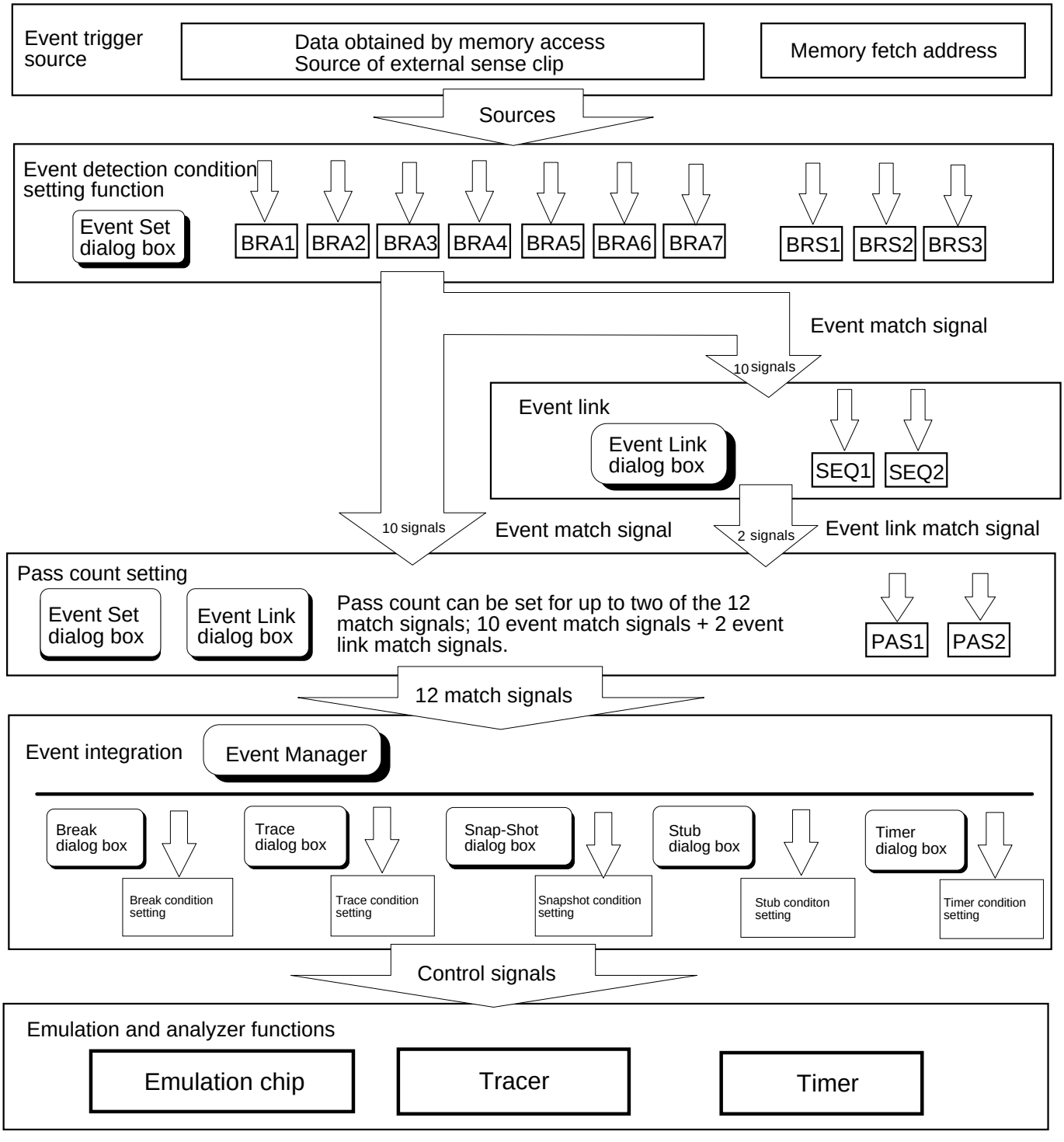
Item	Status	Explanation
Address	Address	Address (address range)
	Mask	Address mask
Status	Run	Program execution
Data	Data	Data value
	Mask	Data mask value
External	External	External sense data value
	Mask	External sense data mask value
Pass count	Pass count	Pass count value

Event condition link setting function

In the Event Link dialog box, an event link condition can be registered for the event condition registered in the Event Set dialog box.

Concept of event detection

The figure below illustrates the concept of event detection starting from event condition setting.



:Corresponding window, BRAx: Bus event detector, BRSx: Execution event detector

SEQx: Event link detector, PASx: Pass counter

Fig. 6-12 Concept of Event Detection

6.2.11 Register Operation Functions

The register operation functions are used to view and modify the general-purpose registers and SFRs. The main functions are explained below.

(1) General-purpose register operation function (Register window)
This function is used to view and modify the control registers and general-purpose registers.

- Control registers: PC, SP, PSW
- General-purpose registers: RP0, RP1, RP2, RP3, AX, BC, DE, HL, VP, UP

PSW can also be viewed and modified using the PSW flag names given below:

- PSW flag names: UF, RSB, S, Z, RSS, AC, IE, P/V, CY

(2) Special function register operation function (SFR window)
This function is used to view and modify the special function registers (SFRs).
Bit operation is possible for SFRs.

6.2.12 Memory Operation Functions

The memory operation functions are used to view and modify memory using mnemonic codes, hexadecimal codes, and ASCII characters. These functions can be used in the Assemble and Memory windows.

6.2.13 Save Function

The save function is used to store, to a file, the object code and debugging environment on the in-circuit emulator, via the disk device connected to the host machine.

6.2.14 Time Measurement Function

This function is used to measure the total execution time from the start of execution to a break and the time from one event to another.

Time can be measured at up to three locations. The following table lists the specifications of the timers.

Item	Explanation
Number of timers	Three independent timers
Maximum measurable time	Approximately 14 minutes and 18 seconds
Minimum resolution	200 ns

6.2.15 Source Debugging

ID78K3 can be used to debug not only object programs but also source programs. The debugging of a source program is referred to as source debugging.

The debugging of a source program offers the following advantages over the debugging of an object program:

- Debugging can be carried out while viewing the source, created by the editor in C or structured assembler.
- Breakpoints can be set and step execution can be performed for the source.

To set a breakpoint, for example, the actual address of the breakpoint must normally be set. In source debugging, on the other hand, a breakpoint can be set by specifying the position in the source program where the breakpoint is desired, using the mouse. Also, in source debugging, the line of the source program that is currently being executed is indicated by a '>' during step execution. This allows the user to comprehend the operation of the program more accurately.

Source debugging is particularly useful when debugging a program written in C or structured assembler.

Note the following when performing source debugging:

(1) Before assembling or compiling a source program, an appropriate option must be specified so that the object program includes source debugging information.

Type of source program to be debugged	Necessary action
C program	Specify the -G option before compiling.
Structured assembler program	Specify the -GS option before structured assembling.
Assembler program	Specify the -GA option before assembling.
Link	Specify the -G option before linking.

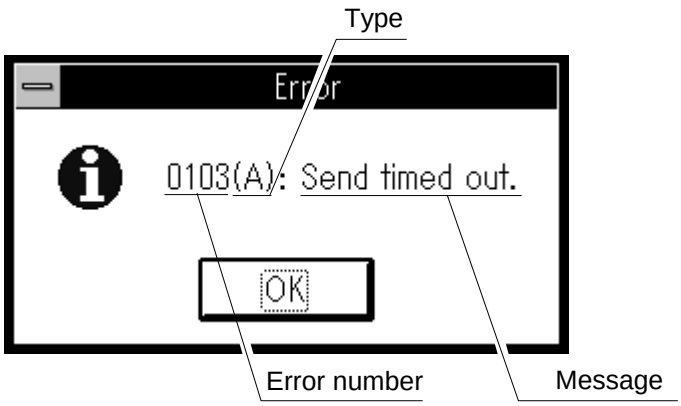
(2) Information on the path of the source program must be specified in the Source Path dialog box.

(3) To perform source debugging, the load module file created by the linker must always be loaded. Source debugging is not possible when the object file created by the object converter is loaded.

Appendix A Error Messages

This appendix lists the error and warning messages output by ID78K3.

An error message consists of error number + type + message.



A type is represented by an alphabetic character. There are three types:

Type	Explanation
A	Abort error. Processing is interrupted and the debugger ends. If this error occurs, debugging cannot be continued.
F	Format (syntax) error. Processing is interrupted. The currently open windows and dialog boxes are closed.
W	Warning. Processing is interrupted. The currently open windows and dialog boxes remain as is.

A message contains the names of the file, variable, and device related to the error, as follows:

Representation in message	Explanation
xxx	Low-order three digits of device name
yyy	File name
zzz	Function name

Errors messages (1/11)

Error No.	Type	Message	Explanation
---	--	Can't open this file. please make sure, now Active Window.	The project file format is incorrect, or the file content has collapsed. Loading the project file was discontinued.
---	--	Cannot find "character string".	The search character was not found. The search was discontinued. Alternatively, opening the specified file was discontinued because no data was in the file.
---	--	Event Name is not set.	There is no event name. Specify the name of the event when adding it.
---	--	Event number already exist.	It is impossible to add an event having the same number as an existing event. Change the number of the event to be added or of the existing event.
---	--	Not enough memory.	Because of insufficient memory, a window cannot be displayed, its content cannot be changed, or changes to it cannot be retained. Assign sufficient memory, and retry.
---	--	Other view mode window exist.	Two or more active windows of the same type cannot be opened simultaneously. An active window that was already open was closed.
---	--	Sorry, Too large view file. (Max is 1000 frame)	The specified view file (.MEM, .TVW, or .DIS) contains more than 1000 lines. Its display was discontinued.
---	--	"event name" is already exist.	It is impossible to add an event having the same name as an existing event. Change the name of the event to be added or of the existing event.
0103	A	Send timed out	Data was not transmitted to the in-circuit emulator (IE). The interface board may not be correctly set or the IE may not be turned on. Check the above and restart the debugger.
0104	A	Receive timed out	No response is returned from the IE. There may be an error in the IE. Check the IE and restart the debugger.

Errors messages (2/11)

Error No.	Type	Message	Explanation
0105	A	Invalid D3xxx.78K	The device file (D3xxx.78K) was not read correctly. The device file may not exist in the specified directory or it may have been destroyed. Install the device file again and restart the debugger.
01a0	A	Monitor timed out	Data was not transferred to and from the monitor program. Clock pulses may not be being supplied to the target CPU or power may not be supplied. Check the above and restart the debugger.
01a1	A	Invalid EX78Kx.OM0	The executor file (EX78K3.OM0) was not read correctly. The executor file may not exist or may have been destroyed. Install the executor file again and restart the debugger.
01a2	A	Unconnected Break-board	The break board (IE-784000-R-BK) is not correctly connected. Connect the IE-784000-R-BK to the IE-784000-R correctly.
01a3	A	Unconnected Emulation-board	The emulation board (IE-784000-R-EM) is not correctly connected. Connect the IE-784000-R-EM to the IE-784000-R correctly.
01a4	A	Contradictory Board-set	The board configuration in the IE is contradictory. Reconfigure the boards and restart the debugger.
01a5	A	Unconnected I/O emulation-board	Emulation board 1 (IE-783xx-R-EM1) is not correctly connected. Connect the IE-783xx-R-EM1 to the IE-784000-R correctly.
01a8	A	Invalid EXPC.INI	The initialize file (EXPC.INI) was not read correctly. The initialize file may not exist or may have been destroyed. Install the initialize file again and restart the debugger.
02a0	F	Bus hold error	The bus is on hold, in which case the user program cannot be executed.
0300	F	User program is running	A user program is running. This command cannot be executed.
0301	F	User program is stopped	A user program is at a break. This command cannot be executed.
0302	F	User program is tracing	The tracer is running. This command cannot be executed.

Errors messages (3/11)

Error No.	Type	Message	Explanation
0303	F	No tracing	No trace measurement has been performed.
0304	F	Now, trace memory is off	The tracer is off.
0305	F	Cannot over trace block	It is impossible to move beyond a trace block.
0306	F	There is no trace block	There is no trace block.
0307	F	There is no event-No	There is no event condition.
0308	F	Not doing Timer measurement	Timer measurement has not been performed.
0309	F	There is no trigger frame	There is no trigger frame.
030a	F	Traces off	An attempt was made to stop the tracer when it was already at a stop.
030b	F	No entry snap number	An attempt was made to reference or delete a snapshot event that had not been registered.
030c	F	No entry stub number	An attempt was made to reference or delete a stub event that had not been registered.
030d	F	Timer is running	The timer is running. No timer event can be changed.
030e	F	Illegal memory range	A memory copy range overlaps with another.
030f	F	Already specified mode	Tracing is already on.
0310	F	Illegal event number	No event condition has been set up.
0311	F	Full timer number	An attempt was made to register more than three timer conditions.
0312	F	Not specified timer number	This timer event has not been set up.
0313	F	Mapping range over.	The specified mapping range is incorrect. An attempt was made to specify mapping that cannot be specified.
03a0	W	Target power off	The target power supply is off.
03a1	F	Now stepping	Step execution is in progress. This command cannot be used.
03a2	F	Timer and Tracer are running	The timer and tracer are running. This command cannot be used.
0400	F	Illegal parameter	The parameter is illegal.
0401	F	Result of Timer measurement is over	The timer has overflowed.
0402	F	Pass count conditions overflow	More than two event conditions cannot be used simultaneously to specify a pass.
0403	F	Specified address range is over	An attempt was made to specify more event conditions than the maximum allowable quantity for an address range. Up to three execution event conditions and one bus event condition can be specified.
0404	F	Event conditions overflow	An attempt was made to specify more event conditions than usable simultaneously. Up to seven bus event conditions and three execution event conditions can be specified.
0405	F	Snapshot number conditions overflow	An attempt was made to register more than 32,767 snapshot events.

Errors messages (4/11)

Error No.	Type	Message	Explanation
0406	F	Stub number conditions overflow	An attempt was made to register more than 32,767 stub events.
0407	F	Initialized data overflow	The amount of initialization data is too large to fit the initialization area.
0408	F	Search data number over	The search data is a string consisting of more than 16 bytes. The maximum allowable size of search data is 16 bytes.
0409	F	Search range over	The search data is larger than the search range.
04a0	F	Number of Trigger condition overflow	An attempt was made to specify more than 100 software breaks.
04a1	F	Emulation memory is not enough	An attempt was made to map IE alternate memory in an area of 1 Mbyte or more.
04a2	F	Bus size conditions overflow	The bus size is larger than 8. An event may not be set up correctly.
04a3	F	BRS event conditions overflow	An attempt was made to set up more than three execution event conditions. (Up to three execution event conditions can be specified.)
04a4	F	BRA event conditions overflow	An attempt was made to set up more than seven bus event conditions. (Up to seven bus event conditions can be specified.)
05a0	A	Evade runaway hardware	The IE is unstable. The IE was reset to bring the user program to a break forcibly.
0600	A	Communication buffer error	The area for the buffer used for exchanging data with the IE was not reserved. End other MS-Windows applications, change the setting of the MS-Windows swap file, or install additional main memory in the host machine.
1000	A	failure in initialization	An attempt to initialize the IE failed. Check whether the IE is abnormal.
1003	F	Illegal relocation address	It is impossible to relocate to a specified address.
1004	F	Illegal parameter	The parameter is illegal.
1006	F	Illegal address	The address is illegal.
1007	A	Not enough substitute memory	An attempt was made to map IE alternate memory in an area of 1 Mbyte or more.
100b	F	Program Is Running	A user program is running. This command cannot be executed.
100c	F	Different Bussize	An attempt was made to make duplicate specification in areas having different bus sizes.
100d	F	Total Maximum Over	An attempt was made to specify a bus larger than the maximum size (8).
100e	F	Enable Maximum Over	The bus size is larger than 8.
100f	W	Wrong Target Status(Power Off)	The target state is unstable.
10ff	A	Communication Error	It is impossible to communicate with the IE. Check whether the IE is abnormal.

Errors messages (5/11)

Error No.	Type	Message	Explanation
2000	F	Illegal sfr name	The SFR name is illegal.
2002	F	User program is running	A user program is running. This command cannot be executed.
2003	F	Illegal SFR number	An attempt was made to access a nonexistent SFR.
2004	F	Illegal bit number	There is no bit SFR at the specified bit position.
2005	W	Redraw sfr name	The SFR has been disabled from redrawing.
2006	F	This SFR is hidden SFR	This SFR is not open to general use. It is impossible to display or change data for the SFR.
2007	F	Can't Read/Write	An attempt was made to write to a write-protected SFR or read from a read-protected SFR.
2008	F	Too big number	The specified SFR was not found.
200a	F	Illegal Bit Pattern	An attempt was made to specify an illegal value for an SFR.
20ff	A	Communication Error	Communication with the IE is impossible. Check whether the IE is abnormal.
3000	F	Illegal address	The address is illegal.
3001	F	Different data	There is a memory content mismatch.
3002	F	Illegal source address	The specified source address range does not fall within the mapping range (during a memory search, comparison, or copy).
3003	F	Illegal destination address	The specified destination address range does not fall within the mapping range (during a memory search, comparison, or copy).
3004	F	Illegal address (source & destination)	The specified address range does not fall within the mapping range (during a memory search, comparison, or copy).
3005	F	Illegal parameter	The parameter is illegal.
3006	F	User program is running	A user program is running. This command cannot be executed.
3008	F	No Parameter	There is no parameter.
3009	F	Parameter Size Alignment Error	The parameter size is illegal. Change the parameter according to the memory access size.
300a	F	Memory Alignment Error	The address value is illegal. Change the address value according to the memory access size.
300b	F	Source Start Address Alignment Error	The source address is illegal. Change the source address according to the memory access size.
300c	F	Error, Destination Start Address Alignment Error	In the destination address range, a memory range with a conflicting access memory size was specified.
300d	F	End Address Alignment Error	The end address is illegal. Change the end address according to the memory access size.
300e	F	Different Access Size in This Area	In the address range, a memory range with a conflicting access memory size was specified.
300f	F	Different Access Size in Source Area	In the source address range, a memory range with a conflicting access memory size was specified.

Errors messages (6/11)

Error No.	Type	Message	Explanation
3010	F	Different Access Size in Destination Area	In the destination address range, a memory range with a conflicting access memory size was specified.
3011	F	Different Access Size, Source & Destination	The access size conflicts between the source and destination address ranges.
30ff	A	Communication Error	Communication with the IE is impossible. Check whether the IE is abnormal.
4000	F	Number is referenced now	The specified event condition cannot be deleted.
4001	F	Illegal table number	The specified table number is illegal.
4002	F	Illegal start address	The start address is illegal.
4003	F	Illegal end address	The end address is illegal.
4004	F	Illegal status	The status is illegal.
4005	F	Illegal data	The data is illegal.
4006	F	Can't action number	An attempt was made to use an event number that was already in use.
4007	F	Can't empty number	An attempt was made to register more than 32,767 events of the same type.
4008	F	Table not found	The specified event has not been registered.
4009	F	Illegal data size	The data size is illegal.
400a	F	Illegal type mode	The mode is illegal.
400b	F	Illegal parameter	The parameter is illegal.
400c	F	Illegal type number	The type is illegal.
400d	F	Table overflow	An attempt was made to register more than 32,767 events of the same type.
400e	F	No entry event number	The specified event does not exist.
400f	F	Illegal Elink data	An event condition specified with a range condition or pass condition was used as an event link condition. Alternatively, only one event condition was specified.
4010	F	Function not found	The specified function was not found.
4011	A	No free memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
4012	F	Timer not enabled	The timer is disabled. Enable it if timer measurement must be made.
4013	W	Data access size mismatch at the bus size	The access size in an event condition does not match the bus size for mapping.
4014	F	Can't use software break	At present, no software break can be used. Specify that a software break be usable, using the Extended Option dialog box.
4015	F	Not point-address	It is impossible to use, as an address condition, an event condition specifying a range.
4016	F	Not renew event condition.	This event condition is being used for another event. It is impossible to change the address range condition or pass count condition.
4017	F	Specified odd-address by word-access.	The data value was not detected in word data beginning at an odd address. Do not include that data value in the setting.

Errors messages (7/11)

Error No.	Type	Message	Explanation
5000	A	Illegal type number	The type is illegal.
5002	A	Illegal file name	The device file cannot be opened.
5003	A	Cannot file seek	An attempt to seek the file failed.
5004	A	Cannot file close	An attempt to close the file failed.
5005	A	Illegal device format	The format of the device file is illegal.
5006	A	Cannot device initialize	An attempt to initialize the IE failed.
5007	A	Illegal device information	There is no device information.
5008	F	Cannot open device file	The specified device file cannot be opened.
5009	F	Cannot open EX78KX.OM0 file	The EX78K3.OM0 cannot be opened.
500a	F	No match device file of version	The version of the device file is illegal.
500b	W	Device has no relocatable iram.	The currently selected device does not support relocation in internal RAM.
6001	F	Illegal entry symbol name	The symbol name is illegal.
6002	F	Illegal parameter	The parameter is illegal.
6003	F	Illegal entry function name	The function name is illegal.
6004	F	Out of Buffer flow	Function display in the Stack window is incomplete. The maximum allowable line size is 512 characters.
6005	F	Illegal expression	The expression is illegal.
7001	F	User program is running	A user program is running. This command cannot be executed.
7002	F	User program is stopped	A user program is at a break. This command cannot be executed.
7003	F	Trace function is active	The tracer is running. This command cannot be executed.
7004	F	Trace memory is OFF	The tracer is off.
7005	F	No Return Address, Can't Execute	The return address of the current function was not found. Step execution based on the Return command is not carried out.
7010	W	Warning, No Source Line Information	Instruction-level step execution was carried out because there was no source information.
7012	A	Not enough memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
70fe	A	Bus Hold Error	The bus is on hold. The user program cannot be executed.
70ff	A	Communication Error	Communication with the IE is impossible. Check whether the IE is abnormal.
7801	F	Step wait canceled	Step execution was discontinued. So, communication with the IE may become impossible.
7802	F	Step aborted	An illegal access break occurred during step execution. Check the user program.

Errors messages (8/11)

Error No.	Type	Message	Explanation
7f00	F	Interrupted step	Step execution was forced to end.
7f02	F	Suspended step	Step execution was suspended.
7f03	A	Run/Step cancel failed. CPU resetted	An attempt to break the user program failed. The IE is unstable because the evaluation chip was reset. Make sure that the IE is normal, then restart it.
7f04	F	Illegal address	An attempt was made to execute in an non-mapped area.
8000	F	File not found	The file was not found.
8001	F	Illegal line number	The line number is illegal.
8002	F	Current data is not set	The current information has not been set.
8003	F	Ilegal address	The address is illegal.
9002	F	Illegal set value	The specified value cannot be set in a register. Specify a value that can be set.
a001	F	Illegal expression	The expression is illegal.
a002	F	Start address bigger than end address	The start address is greater than the end address (start address > end address). Check the addresses.
a003	F	Source path not found	The specified source path information is illegal. Specify the correct source path information.
a004	F	Expression is too big	The size of the expression is greater than 127 characters.
a005	A	Not enough memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
a006	F	Illegal argument	The argument is illegal.
a008	F	Source path not set	The source path has not be specified.
a009	F	File not found	The file was not found.
a00a	F	File not open	The file cannot be opened.
a00b	A	File not close	An attempt to close the file failed.
a00c	A	File not read	An attempt to read the file failed. It is likely that the file has collapsed.
a00d	F	Not source file of LM	The specified source file has not been registered for the load module file. A file not registered for the load module file cannot be displayed in the Source window.
a00e	F	Illegal line number	The line number is illegal.
a00f	F	Illegal variable	The variable does not exist.
a010	A	Communication failed	Communication with the IE is impossible. Check whether the IE is abnormal.
a011	F	Can't access register	The register cannot be accessed. Check the IE.

Errors messages (9/11)

Error No.	Type	Message	Explanation
a012	F	Can't access memory	The specified memory (variable) cannot be accessed. Check the IE or map setting.
b000	F	Command line error	The parameter is illegal.
b001	F	Task type not found	The load module file does not contain program information.
b002	F	File not found	The file was not found.
b003	F	Function not found	The specified function was not found.
b004	F	Illegal magic number	The magic number for the load module file is illegal.
b005	F	Symbol not found	The symbol was not found.
b008	F	Illegal value	The expression is illegal.
b009	A	Not enough memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
b00a	F	Illegal symbol entry	There is an illegal symbol in the load module file. It is likely that there is a bug related to the programming language.
b00b	F	Current type noting	There is no debug information. Load the load module file.
b00c	F	Current file noting	There is no current source file. Alternatively the source file cannot be opened because the load module file has not be loaded.
b012	F	Line number too large	The line number is illegal.
b015	A	Read error	An attempt to read the file failed. It is likely that the file has collapsed.
b016	A	Open error	The file cannot be opened.
b017	A	Write error	An attempt to write to the file failed.
b019	A	Seek error	An attempt to seek the file failed.
b01a	A	Close error	An attempt to close the file failed.
b01d	F	Address not found	There is no source line that corresponds to the current PC value.
b01e	F	No line information(not compile with -g)	There is no source line information in the load module file. Attach the debug option, and carry out recompilation, assembly, and linkage.
b01f	F	Cannot find member	No member was found in the specified structure.
b020	F	Cannot find value	The specified enumeration constant is illegal.
b021	F	Striped LM	There is no symbol information in the load module file.
b022	F	Null statement line	The line number is illegal.
b026	F	Max dimension array over	A four-dimensional or greater-scale array cannot be displayed.
b027	F	End of file	The file is not complete.
b029	F	Illegal address	The address is illegal.

Errors messages (10/11)

Error No.	Type	Message	Explanation
b02a	A	Communication failed	Communication with the IE is impossible. Check whether the IE is abnormal.
b02b	F	No stack frame point	Stack tracing is impossible with the current PC value.
b02c	F	Max block overflow	The maximum number of blocks in one function is exceeded. The function cannot be displayed. (The maximum number of blocks per function is 256.)
b02d	F	Illegal argument	The argument is illegal.
c001	F	Cannot open file	The file cannot be opened.
c002	A	Cannot close file	An attempt to close the file failed.
c003	A	Cannot read file	An attempt to read the file failed. It is likely that the file has collapsed.
c004	A	Cannot seek file	An attempt to seek the file failed.
c005	F	Illegal file type	The format of the file is illegal. This file cannot be handled.
c006	F	Illegal magic number	The magic number for the load module file is illegal.
c007	F	This file is not load-module file	The specified file is not a load module file.
c008	F	Old coff version	The version of the load module file is illegal.
c009	A	Not enough memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
c00a	F	Illegal address	The address is illegal.
c00b	F	LM not load	The load module file has not been loaded.
c00c	F	Illegal argument	This is an internal error.
c00d	F	User program is emulating	A user program is running. This command cannot be executed.
c00e	F	User program is tracing	The tracer is running. This command cannot be executed.
c010	A	Communication failed	Communication with the IE is impossible. Check whether the IE is abnormal.
c011	F	Illegal file format	The format of the load module file (LNK) is illegal.
c012	F	Check sum error	A checksum error occurred in reading the load module file. Check the load module file.
c013	F	Too big size	The address range for uploading has exceeded 1 Mbytes.
c014	F	Cannot write file	An attempt to write to the file failed.
c100	F	Not support	The Tektronix format is not supported.

Errors messages (11/11)

Error No.	Type	Message	Explanation
d001	F	Not enough memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
e000	F	Illegal argument	This is an internal error.
e001	F	Illegal start address	The start address is illegal.
e002	F	Illegal end address	The end address is illegal.
e003	F	Size too long	The address value is illegal.
e004	F	Can't open file	The specified file cannot be opened.
e005	F	Can't read file	An attempt to read the file failed. It is likely that the file has collapsed.
e006	F	Can't seek file	An attempt to seek the file failed.
e007	F	Can't write file	An attempt to write to the file failed.
e008	F	Not enough memory	There is no sufficient memory. End unnecessary applications, or close the Debugger window.
e009	F	Illegal file format	The format of the file is illegal.

Appendix B Key Functions

Debugging can be carried out more effectively when ID78K3 is operated using the special function keys. In the following explanation of the special function keys, general key representations (generic key representations) are used. For the IBM-PC/AT Series, the key representations may differ slightly depending on the keyboard type.

B.1 Functions of Special Function Keys

Key		Function
PC-9801 and 9821 Series	**IBM-PC/AT Series**	
BS	BackSpace	Deletes the character immediately before the cursor and moves the cursor to the position of the deleted character. The character string following the cursor is moved back.
COPY	PrintScreen	Captures the entire screen into the clipboard as a bit image. (Windows function)
ESC	Esc	①Closes the pulldown menu. ②Closes the modal dialog box.
GRPH	Alt	Moves the cursor to the menu bar.
HELP	End	Displays the last line. Also, the cursor is positioned to the last line.
HOME CLR	Home	Displays the first line. Also, the cursor is positioned to the first line.
ROLL UP	PageUp	Scrolls the display up by one screen. Also, the cursor is positioned to the top of the screen.
ROLL DOWN	PageDown	Scrolls the display down by one screen. Also, the cursor is positioned to the top of the screen.
SPACE	Space	Inserts one blank.
TAB	Tab	Positions the cursor to the next item.
↑	↑	Moves the cursor up. Scrolls the screen down by one line when the cursor is positioned to the top of the screen.
↓	↓	Moves the cursor down. Scrolls the screen up by one line when the cursor is at the bottom of the screen.
←	←	Moves the cursor to the left. Scrolls the screen to the right by one item when the cursor is in the leftmost column.
→	→	Moves the cursor to the right. Scrolls the screen to the left by one item when the cursor is in the rightmost column.
↵	↵	Confirms input data.

B.2 Functions of Special Function Keys (CTRL + Key)

Key (Common to the PC-9801, 9821, and IBM-PC/AT Series)	Function
A	Using the data value in the current window as an address to jump to, disassembles and displays the program starting from that address. Opens the Assemble window.
B	Sets a breakpoint in a selected line.
C	Copies a selected character string to the clipboard buffer.
F	Switches a window to modify mode. This has the same effect as clicking the ToModify button.
G	Executes a program. This has the same effect as clicking the ▶ button.
H	Switches a window to the Hold state.
I	Switches a window to the Active state.
M	Using the data value in the current window as an address to jump to, displays the contents of memory starting from that address. Opens the Memory window.
O	If the Source window is current: Allows the user to select a source view file. Opens the source file select dialog box. Otherwise: Displays an appropriate view file in the current window. Opens the view file save dialog box.
P	Stops the execution of a program. This has the same effect as clicking the ❚❚ button.
R	Performs step execution until control returns to the calling function. This has the same effect as clicking the button.
S	Saves the contents of the current window to a view file.
T	Performs step execution. This has the same effect as clicking the ▶❙ button.
U	Using the data value in the current window as an address to jump to, displays an appropriate source text and source line. Opens the Source window.
V	Pastes the contents of the clipboard buffer to the text cursor position.
W	Switches a window to view mode. This has the same effect as clicking the ToView button.
X	Performs Next step execution. This has the same effect as clicking the ⏭ button.
Z	Cancels the previous editing operation.

Appendix C Menus

This Appendix lists the menus supported by ID78K3.

Symbols used in the menu lists

Symbol	Meaning
[Item]	Item on a menu bar
No symbol	Item in a pull-down menu
→ (arrow)	Item in a cascaded menu The number of arrows corresponds to the nesting level.

Table C-1 Main Window (1/5)

Menu	Mnemonic	Explanation
[File]		
Open...	CTRL+O	Opens a file.
Save	CTRL+S	Saves the contents of the current window into the view file.
Save As...		Save the contents of the current window into a view file having a different name.
Close		Closes the current window.
Print		Print the contents of the current window.
Down load...		Downloads a program.
Up load...		Uploads a program.
Open/Save Project ▶		
→Open Project...		Open a project file.
→Save		Overwrites the project file with the current debugging environment.
→Save As...		Saves the current debugging environment into a project file.
Open/Save Log		Records the history of execution.
Exit		Exits from the debugger.
[Edit]		
Undo	CTRL+Z	Cancel the most recent editing.
Copy	CTRL+C	Copies a selected character string into the clipboard buffer.
Paste	CTRL+V	Pastes the contents of the clipboard buffer at the point to which the text cursor is positioned.
Write in		Writes the modified contents into the target device.
Restore		Cancels the modified contents.
Memory ▶		
→Memory Fill...		Initializes memory.
→Memory Copy...		Copies the contents of memory
→Memory Compare...		Compares the contents of memory.
→File Compare...		Compares the view file with the contents of memory.

Table C-1 Main Window (2/5)

Menu	Mnemonic	Explanation
[View]		
Search...		Searches for a character string or numerical value.
Address...		Displays the contents of memory at a specified address.
View Variable...		Displays the value of a specified variable temporarily.
Watch Variable...		Displays the value of a specified variable continuously.
Add Variable...		Adds a variable to the Variable window.
Sym To Adr...		Converts symbols.
Delete		Deletes a specified value.
Bin		Selects binary display format.
Oct		Selects octal display format.
Dec		Selects decimal display format.
Hex		Selects hexadecimal display format.
Proper		Selects a default display format for each variable.
Event ?		Displays event information.
Memory ▶		
→Nibble		Displays data in nibble format.
→Byte		Displays data in byte format.
→Word		Displays data in word format.
→Long		Displays data in long format.
→Ascii		Switches on or off ASCII view mode.
Sfr ▶		
→Address Sort		Selects alphabetic display order or display in order of addresses.
→Pick Up		Displays only modified SFRs.
→Attribute ▶		
→→Show		Displays the attribute view area.
→→Hide		Hides the attribute view area.
→Compulsion Read		Performs forced reading of a read-protected SFR.
→Synchronize		Writes the modified SFRs to the target device.
Trace View		
→Frame ▶		
→→Show		Displays the frame number field.
→→Hide		Hides the frame number field.
→Timetag		
→→Show		Displays the time tag field.
→→Hide		Hides the time tag field.
→Instruction Fetch Address ▶		
→→Show		Displays the instruction fetch address field.
→→Hide		Hides the instruction fetch address field.

Table C-1 Main Window (3/5)

Menu	Mnemonic	Explanation
→Instruction Fetch Data ▶		
→→Bin		Displays instruction fetch field data in binary format.
→→Oct		Displays instruction fetch field data in octal format.
→→Dec		Displays instruction fetch field data in decimal format.
→→Hex		Displays instruction fetch field data in hexadecimal format.
→→Hide		Hides instruction fetch field data.
→Instruction Fetch Status ▶		
→→Show		Displays the instruction fetch status.
→→Hide		Hides the instruction fetch status.
→Memory access Address ▶		
→→Show		Displays the memory access address field.
→→Hide		Hides the memory access address field.
→Memory access Data ▶		
→→Bin		Displays memory access field data in binary format.
→→Oct		Displays memory access field data in octal format.
→→Dec		Displays memory access field data in decimal format.
→→Hex		Displays memory access field data in hexadecimal format.
→→Hide		Hides memory access field data.
→Memory access Status ▶		
→→Show		Displays the memory access status.
→→Hide		Hides the memory access status.
→External Probe ▶		
→→Bin		Displays external sense field data in binary format.
→→Oct		Displays external sense field data in octal format.
→→Dec		Displays external sense field data in decimal format.
→→Hex		Displays external sense field data in hexadecimal format.
→→Hide		Hides external sense field data.
→Jump Address ▶		
→→Show		Displays the jump address field.
→→Hide		Hides the jump address field.
→DisAssemble ▶		
→→Show		Displays the disassembly view field.
→→Hide		Hides the disassembly view field.
→Open Frame...		Specifies a view frame number.
→Pick Up...		Selects a view frame.
Coverage ▶		
→ 1 Byte		Displays data in 1-byte units.
→64 Byte		Displays data in 64-byte units.
→1024 Byte		Displays data in 1024-byte units.

Table C-1 Main Window (4/5)

Menu	Mnemonic	Explanation
[Option]		
Tool Bar		Displays or hides the tool bar.
Status Bar		Displays or hides the status bar.
Button		Displays or hides the buttons in the window.
Source Mode		Selects the source mode.
Instruction Mode		Selects the instruction mode.
Configuration...		Sets the environment.
Source Path...		Sets source path information.
Extended Option...		Sets extended options.
[Execute]		
Stop	CTRL+P	Stops the execution of a program.
Go	CTRL+G	Executes a program.
Return	CTRL+R	Executes a program, step by step, until control is returned to the calling function.
Step	CTRL+T	Executes a program step by step.
Next	CTRL+X	Performs Next step execution of a program.
Go & Go		Repeatedly executes a program.
Go & Come		Executes a program up to a specified address.
Slowmotion		Continues step-by-step execution.
CPU Reset & Go		Resets the CPU before starting execution.
CPU Reset...		Resets the CPU.
Set BP	CTRL+B	Sets a breakpoint.
Set PC		Sets the address in the program counter.
Uncond. Trace ON		Sets unconditional tracing.
Cond. Trace ON		Sets conditional tracing.
Trace OFF		Disables the tracer.
Coverage ON		Enables coverage measurement.
Coverage OFF		Disables coverage measurement.
[Operation]		
Active	CTRL+I	Put the window in the active state.
Hold	CTRL+H	Put the window in the hold state.
ToModify	CTRL+F	Puts the window in modify mode.
ToView	CTRL+W	Puts the window in view mode.
Window Connect ▶		
→SourceText		Links to the Source window.
→Assemble		Links to the Assemble window.
→Memory		Links to the Memory window.

Table C-1 Main Window (5/5)

Menu	Mnemonic	Explanation
[Browse]		
SourceText...		Opens the Source window.
Variable...		Opens the Variable window.
Assemble...		Opens the Assemble window.
Memory...		Opens the Memory window.
Register...		Opens the Register window.
Stack Trace...		Opens the Stack window.
Sfr...		Opens the SFR window.
Local Variable...		Opens the Local Variable window.
BreakSet...		Opens the Break dialog box.
Timer...		Opens the Timer dialog box.
Stub Set...		Opens the Stub dialog box.
Trace ▶		
→TraceSet...		Opens the Trace dialog box.
→TraceView...		Opens the Trace View dialog box.
→SnapShotTraceSet...		Opens the Snap-Shot dialog box.
Event ▶		
→EventSet...		Opens the Event Set dialog box.
→EventManager...		Opens the Event Manager.
→EventLinkSet...		Opens the Event Link dialog box.
Coverage ▶		
→View...		Opens the Coverage window.
→Clear...		Opens the Coverage Memory Clear dialog box.
→Condition...		Opens the Coverage Condition Setting dialog box.
→Efficiency...		Opens the Coverage Efficiency View dialog box.
[Jump]		
SourceText...	CTRL+U	Jumps to the Source window.
Assemble...	CTRL+A	Jumps to the Assemble window.
Memory...	CTRL+M	Jumps to the Memory window.
[Window]		
Cascade		Displays the window in cascade style.
Tile		Displays the window in tile style.
Arrange Icons		Re-arranges the icons.
Close All		Closes all windows except the main window.
[Help]		
About...		Displays the information about the version.

Table C-2 Event Manager

Menu	Mnemonic	Explanation
[File]		
Open...		Opens an event setting file.
Save		Saves the current event settings into the event setting file, overwriting the previously saved setting.
Save As...		Saves the current event settings into a specified event setting file.
Close		Closes the Event Manager.
Print		Prints the event registration/setting information.
[Edit]		
Undo		Cancel the most recent editing.
Copy		Copies a specified icon using a different name.
All Select		Selects all icons.
Delete		Deletes a specified icon.
[View]		
Name		Sorts the icons into event name order.
Kind		Sorts the icons into event type order.
Detail		Switches between normal view and detail view.
[Operation]		
BreakSet...		Opens the Break dialog box.
Timer...		Opens the Timer dialog box.
StubSet...		Opens the Stub dialog box.
TraceSet...		Opens the Trace dialog box.
SnapShotTraceSet...		Opens the Snap-Shot dialog box.
EventSet...		Opens the Event Set dialog box.
EventLinkSet...		Opens the Event Link dialog box.
[Jump]		
SourceText...		Jumps to the Source window.
Assemble...		Jumps to the Assemble window.
Memory...		Jumps to the Memory window.

Table C-3 Register Window

Menu	Mnemonic	Explanation
[File]		
Open/save Condition ▸		
→Open Condition...		Opens the selected file for reference.
→Save Condition		Saves the contents of the window into a view file.
→Save File as...		Saves the current event settings into a specified view file.
Close		Closes the Register window.
[Edit]		
Undo		Cancel the most recent editing.
Copy		Copies a selected character string into the clipboard buffer.
Paste		Pastes the contents of the clipboard buffer at the point to which the text cursor is positioned.
Write in		Writes the modified contents into the target device.
Restore		Cancels the modified contents.
[View]		
Absolute Name		Displays absolute register names.
Functional Name		Displays functional register names.
Register		Displays registers individually.
Register Pair		Displays register pairs.
Bin		Displays data in binary format.
Oct		Displays data in octal format.
Dec		Displays data in decimal format.
Hex		Displays data in hexadecimal format.
[Operation]		
Active		Puts the Register window in the active state.
Hold		Puts the Register window in the hold state.
ToModify		Puts the Register window in modify mode.
ToView		Puts the Register window in view mode.
[Jump]		
SourceText...		Jumps to the Source window.
Assemble...		Jumps to the Assemble window.
Memory...		Jumps to the Memory window.

Table C-4 Variable Window

Menu	Mnemonic	Explanation
[File]		
Open/save Condition ▸		
→Open Condition...		Opens the selected file for reference.
→Save Condition		Saves the contents of the window into a view file.
→Save File as...		Saves the currents of the window into a specified view file.
Close		Closes the Variable window.
[Edit]		
Undo		Cancel the most recent editing.
Copy		Copies a selected character string into the clipboard buffer.
Paste		Pastes the contents of the clipboard buffer at the point to which the text cursor is positioned.
Write in		Writes the modified contents into the target device.
Restore		Cancels the modified contents.
[View]		
Bin		Displays variable values in binary format.
Oct		Displays variable values in octal format.
Dec		Displays variable values in decimal format.
Hex		Displays variable values in hexadecimal format.
Proper		Displays variable values in default format for each variable.
[Operation]		
Active		Puts the Variable window in the active state.
Hold		Puts the Variable window in the hold state.
ToModify		Puts the Variable window in modify mode.
ToView		Puts the Variable window in view mode.
Delete		Removes a specified variable from the Variable window.

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

V

W