To our customers,

## Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

User's Manual

# ID850QB Ver. 3.10

## Integrated Debugger

## Operation

---

**Target Device**
**V850 Series**

Document No. U17435EJ1V0UM00 (1st edition)
Date Published April 2005 CP(K)



Printed in Japan

[MEMO]

# Regional Information

Some information contained in this document may vary from country to country. Before using any NEC Electronics product in your application, please contact the NEC Electronics office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

**[GLOBAL SUPPORT]**
**http://www.necel.com/en/support/support.html**

**NEC Electronics America, Inc. (U.S.)**
Santa Clara, California
Tel: 408-588-6000
800-366-9782

**NEC Electronics (Europe) GmbH**
Duesseldorf, Germany
Tel: 0211-65030

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
- **Filiale Italiana**
  Milano, Italy
  Tel: 02-66 75 41
- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-244 58 45
- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 80 820
- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133

**NEC Electronics Hong Kong Ltd.**
Hong Kong
Tel: 2886-9318

**NEC Electronics Hong Kong Ltd.**
Seoul Branch
Seoul, Korea
Tel: 02-558-3737

**NEC Electronics Shanghai Ltd.**
Shanghai, P.R. China
Tel: 021-5888-5400

**NEC Electronics Taiwan Ltd.**
Taipei, Taiwan
Tel: 02-2719-2377

**NEC Electronics Singapore Pte. Ltd.**
Novena Square, Singapore
Tel: 6253-8311

J04.1

[MEMO]

# PREFACE

| | |
|---|---|
| **Target Readers** | This manual is intended for user engineers who design and develop application systems of the V850 Series. |
| **Purpose** | This manual is intended for users to understand the functions of the ID850QB in the organization below. |
| **Organization** | This manual consists of the following chapters: |

- OVERVIEW
- INSTALLATION
- STARTING AND TERMINATING
- ASSOCIATION WITH PM+
- DEBUG FUNCTION
- WINDOW REFERENCE
- COMMAND REFERENCE

**How to Use This Manual**

It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, microcontrollers, C language, and assemblers.

To understand the functions of the V850 Series
→ Refer to Hardware User's Manual for each product..

To understand the instruction functions of the V850 Series
→ Refer to **V850ES Architecture User's Manual (U15943E)** or **V850E1 Architecture User's Manual (U14559E)**.

**Conventions**

| | |
|---|---|
| Data significance: | Higher digits on the left and lower digits on the right |
| **Note**: | Footnote for item marked with **Note** in the text |
| **Caution**: | Information requiring particular attention |
| **Remark**: | Supplementary information |
| Numerical representation: | Binary … XXXX or XXXXB |
| | Decimal ... XXXX |
| | Hexadecimal … 0xXXXX |

Prefix indicating the power of 2 (address space, memory capacity):

K (Kilo): $2^{10} = 1024$

M (Mega): $2^{20} = 1024^2$

**Related Documents** Refer to the documents listed below when using this manual.
The related documents indicated in this publication may include preliminary versions.
However, preliminary versions are not marked as such.

**Documents related to development tools (User's Manuals)**

| Document Name | | Document No. |
|---|---|---|
| IE-V850E1-CD-NW(PCMCIA Card Type On-Chip Debug Emulator) | | U16647E |
| QB-V850ESSX2(In-circuit emulator for V850ES/SG2, V850ES/SJ2) | | U17091E |
| QB-V850IA4(In-circuit emulator for V850ES/IK1, V850E/IA3, V850E/IA4) | | U17167E |
| QB-V850ESKX1(In-circuit emulator for V850ES/Kx1, V850ES/Kx1+) | | U17214E |
| CA850 Ver. 2.70 C Compiler Package | Operation | U16932E |
| | C Language | U16930E |
| | Assembly Language | U16931E |
| | Link Directives | U16933E |
| CA850 Ver. 3.00 C Compiler Package | Operation | U17293E |
| | C Language | U17291E |
| | Assembly Language | U17292E |
| | Link Directives | U17294E |
| PM+ Ver.6.00 Project Manager | | U17178E |
| ID850QB Ver. 3.10 Integrated Debugger | Operation | This manual |
| SM+ System Simulator | Operation | U17246E |
| | User Open Interface | U17247E |
| SM850 Ver. 2.50 System Simulator | Operation | U16218E |
| SM850 Ver. 2.00 or Later System Simulator | External Part User Open Interface Specifications | U14873E |
| RX850 Ver. 3.20 Real-Time OS | Basics | U13430E |
| | Installation | U17419E |
| | Technical | U13431E |
| | Task Debugger | U17420E |
| RX850 Pro Ver. 3.20 Real-Time OS | Fundamental | U13773E |
| | Installation | U17421E |
| | Technical | U13772E |
| | Task Debugger | U17422E |
| AZ850 Ver. 3.30 System Performance Analyzer | | U17423E |
| PG-FP4 Flash Memory Programmer | | U15260E |
| TW850 Ver.2.00 Performance Analysis Tuning Tool | | U17241E |

# CONTENTS

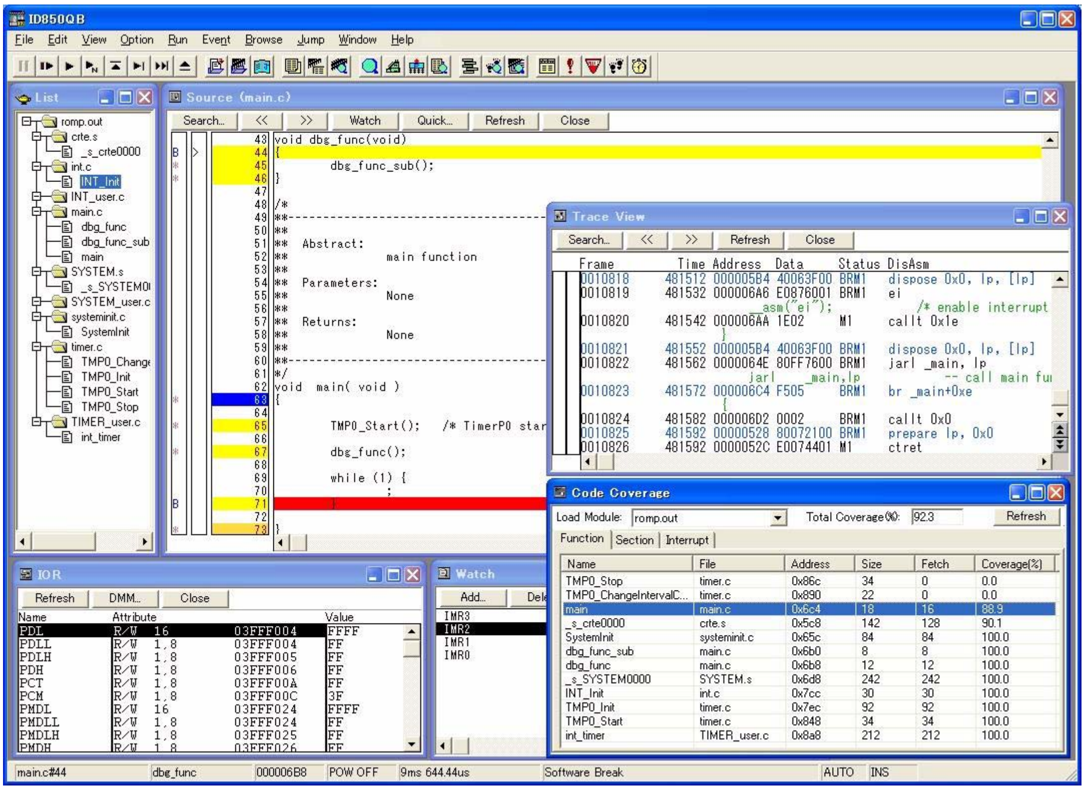
# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1  OVERVIEW

The Integrated Debugger ID850QB for the V850 Series (hereafter referred to as the ID850QB) is a software tool developed for NEC Electronics V850 Series of microcontrollers for embedded control.  This software tool is intended to enable efficient debugging of user programs.

Figure 1-1 ID850QB




This chapter explains the following items regarding the ID850QB.

- Features
- System Configuration
- Operating Environment
- Cautions During Debugging

# 1.1 Features

The ID850QB has the following features:

- New functions, enhanced functions
- Other functions

## 1.1.1 New functions, enhanced functions

**(1) Enhanced RRM function (when IECUBE is connected)**

Division setting for the sampling range of the real-time RAM monitor area is now possible (refer to "5.13 RRM Function").

Up to 8 locations can be set in 256-byte units in the 2 KB RRM area of the ID850QB.

Moreover, in the Memory window, the access status (read, write, read & write) can be displayed in different colors using this function. This can also be used as a simple coverage function.

**(2) Enhanced timer function (when IECUBE is connected)**

Using an external clock of 50 MHz, measurement of up to 7 segments + Run-Break time (time from program execution start until break) is possible (refer to "5.9 Timer Function (When IECUBE Is Connected)").

Other than during Run-Break, the maximum time, minimum time, pass count, and average time are displayed.

Measured time display during user program execution and time-over break are supported.

**(3) Fail-safe break support (when IECUBE is connected)**

In addition to the traditional guarded areas and IOR areas, fail-safe break is also supported for the guarded areas of internal ROM/internal RAM (refer to "5.4.5 Fail-safe break function").

**(4) Enhanced command function**

Script file specification is possible at ID startup (refer to "3.2 Startup Option And Argument Specification").

Testing can be done with 1 click by specifying a project file at the same time as the script file.

The Tcl/Tk core was updated to the latest version 8.4.

**(5) Settings during program execution (when IECUBE is connected)**

Timer event conditions and trace event conditions can now be set during user program execution.

**(6) Support of N-Wire CARD, MINICUBE and IECUBE**

Three emulators (IECUBE, N-Wire CARD, MINICUBE,and MINICUBE) can be connected with 1 debugger (ID850QB) (refer to "1.2 System Configuration").

In the case of IECUBE or MINICUBE, USB2.0 is supported.

**(7) Trace complementation function (when IECUBE is connected)**

This function performs complementary display of instructions between branch instructions that cannot be traced by the hardware (refer to "5.10.2 Setting trace data"). Complementation/non-complementation can be selected. In the complementation mode, assemble display in the internal ROM area is possible even during program execution (while the tracer is stopped).

**(8) Supports extension linear addresses**

In Intel-HEX-format load-module files, 1 MB or more of addresses (extension linear addresses) can be downloaded. (Refer to "5.2 Download Function, Upload Function".)

**(9) Supports multiple versions**

Multiple versions of products can be installed in the same machine.

**(10) Hardware detailed version display**

The hardware detailed version is displayed in the About Dialog Box.

The version can also be confirmed in the Configuration dialog box prior to startup, and the display information can be copied and pasted. Pasting to support mail is now easy.

**(11) Supports one-byte spaces**

Single-byte spaces can now be used for folder names.

**(12) Support of code coverage measurement (when IECUBE is connected and when memory boards are incorporated)**

Code coverage measurement (C0 coverage) can now be performed.

The code coverage can now be displayed in the Code Coverage Window, and sections for which coverage measurement is executed can now be displayed in the Source Window and Assemble Window (refer to "5.11 Coverage Measurement Function (When IECUBE Is Connected)").

**(13) Enhanced DMM function**

DMM (Dynamic Memory Modification) is now possible with memory, registers, or IORs specified.

Real-time writing to the memory can be performed with the DMM function during user program execution (refer to "5.14 DMM Function").

**(14) Support of emulation memory (when IECUBE is connected and when memory boards are incorporated)**

Up to 16 MB emulation memory can be used.

**(15) Support of flash self programming error emulation (when IECUBE is connected)**

Flash self programming can now be debugged (refer to " Flash Option Dialog Box").

## 1.1.2 Other functions

**(1) Using function of in-circuit emulator**

By using the event setting function of an in-circuit emulator, break events can be set, the user program can be traced, and time can be measured, and so on (refer to "5.12 Event Function").

**(2) Support of on-chip debugging (when N-Wire CARD, MINICUBE connected)**

A debugging function implemented by the on-chip debug unit of the Nx85ET（RCU0+TEU+TRCU）, Nx85E901（RCU0）, RCU1 is provided.

**(3) Flash memory writing function (when N-Wire CARD, MINICUBE connected)**

The internal flash memory can be written and the load module can be downloaded by the same access method as an ordinary memory operation (refer to "5.7.4 Flash memory writing function (when N-Wire CARD, MINICUBE is connected)").

**(4) Security function (when N-Wire CARD, MINICUBE is connected)**

The ID code stored in the internal ROM or internal flash memory of a product with a security unit can be authenticated (refer to " Configuration Dialog Box", "(5) ID Code").

**(5) Function expansion through Tcl**

The batch processing and hook processing, and the creation of original user custom windows are possible using the command line with Tcl/Tk (Tool Command Language) (refer to "CHAPTER 7 COMMAND REFERENCE", "APPENDIX A EXPANSION WINDOW").

**(6) Function expansion through TIP or ToolLink**

By associating with a task debugger (RD) and system performance analyzer (AZ) supporting TIP (Tool Interface Protocol) or ToolLink, the debugging efficiency of the user program using a real-time OS (RX) can be dramatically improved.

## 1.2 System Configuration

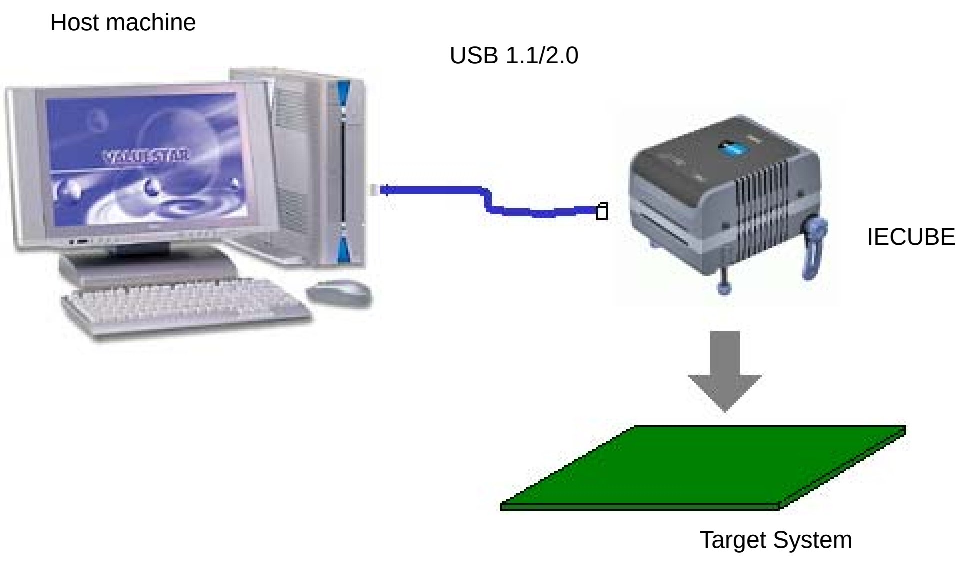
The ID850QB can be connected to the following two types of emulators.

User programs developed for the V850 Series and a pleasant debugging environment for target systems are provided.

**(1) IECUBE**

IECUBE can be manipulated from the ID850QB by connected it to the ID850QB via a USB cable.

Figure 1-2 Example of ID850QB System Configuration (IECUBE)



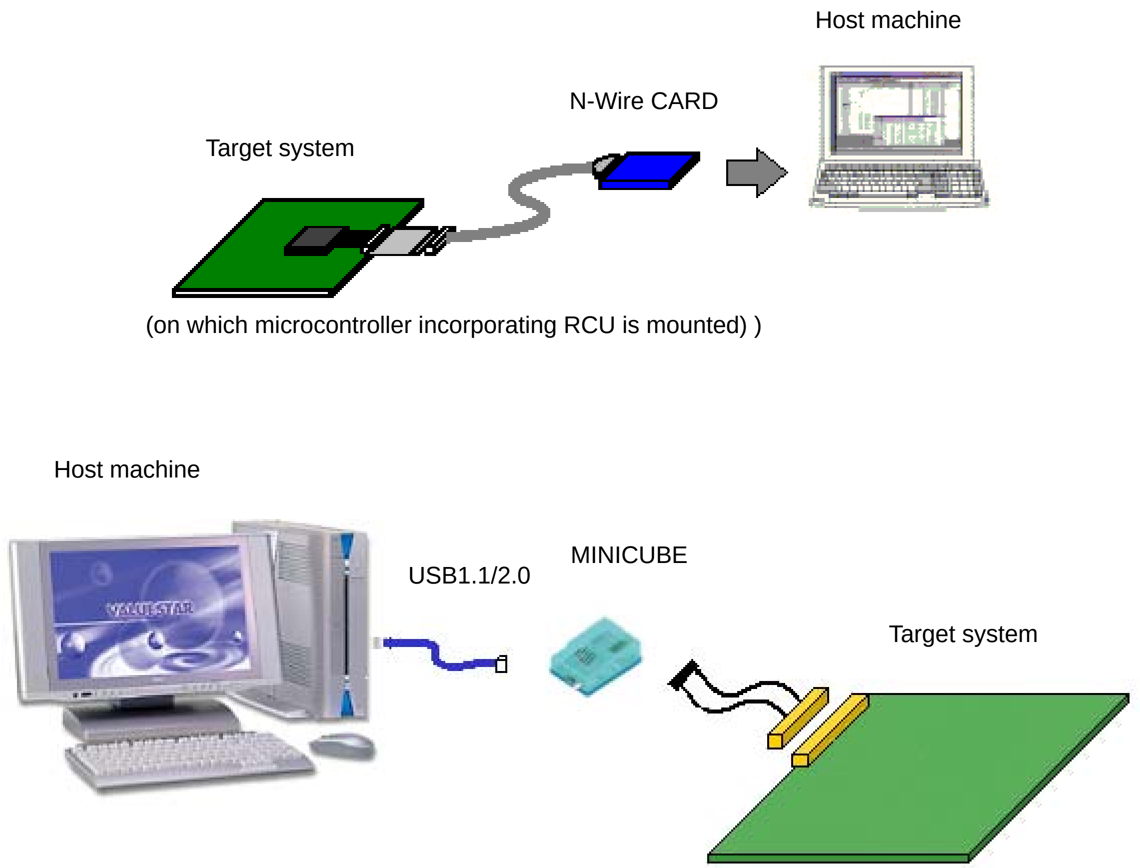
**(2) N-Wire CARD, MINICUBE**

The N-Wire CARD, MINICUBE can provide debugging functions by connecting it to a target system on which the V850ES incorporating an RCU (Run Control Unit) and the V850E1 are mounted.

The N-Wire CARD a PC card emulator, can be manipulated from the ID850QB by directly inserting it to the host machine.

The MINICUBE can be manipulated from the ID850QB by connected it to the ID850QB via a USB cable.

Figure 1-3 Example of ID850QB System Configuration (N-Wire CARD, MINICUBE)



## 1.3 Operating Environment

This section explains the following items regarding the operating environment.

- Hardware environment
- Software environment

### 1.3.1 Hardware environment

**(1) Host machine**

- The machine by which the target OS operates

**Caution:** When N-Wire CARD is connected, because it is assumed that the IECUBE is used with a notebook PC, the host machine must have a PC card slot (TYPEII).

**(2) In-circuit emulator**

- IECUBE (QB-V850Exxxx)
- MINICUBE (QB-V850MINI)
- N-Wire CARD (IE-V850E1-CD-NW)

### 1.3.2 Software environment

**(1) OS (any of the following)**

- Windows98
- Windows2000
- Windows NT4.0 *
- WindowsMe
- WindowsXP

* Can be used only when N-Wire CARD is connected.

**Caution:** Regardless of which of the OS above is used, we recommend that the latest Service Pack is installed.

**(2) Device file (Individual acquisition)**

- The device file of the target device to be used.

**Remark:** This file is available from the following Web site of NEC Electronics (ODS).
http://www.necel.com/micro/index_e.html

**(3) Supported tools (manufactured by NEC Electronics)**

- C compiler package CA850 (Version 3.00 or later)
- Project manager PM+ (Version 6.00 or later)
- System performance analyzer AZ850 (Version 3.30 or later)
- Performance tuning tool TW850 (Version 2.10 or later)

# 1.4 Cautions During Debugging

The cautions to be observed during debugging are described below.

- When performing source level debugging
- Security ID

## 1.4.1 When performing source level debugging

The object file for which source level debugging is performed must include symbol information or other information for debugging (debugging information).

Therefore, perform the following processing during source file compiling.

**(1) When using PM+**

Specify [Debug Build] when the Build mode is selected.

**(2) When using CA850 on standalone basis**

Add the -g option.

## 1.4.2 Security ID

The object file used when N-Wire CARD, MINICUBE is connected must include the security ID information.

For the security ID850QB settings, refer to "CA850 Assembler Package Manipulation".

For details about the security ID, refer to the N-Wire CARD, MINICUBE user's manual.

The security ID (ID code) from the ID850QB is set in the Configuration Dialog Box.

# CHAPTER 2 INSTALLATION

This chapter explains the following items about installation of ID850QB:

- Installing
- Uninstalling

## 2.1 Installing

The following items must be installed, when the ID850QB is used.

Table 2-1 Install

| Item | Procedure |
| --- | --- |
| ID850QB system disk | Install the contents of this disk according to the automatically executed installer. |
| Used device file | Install this file according to the DFINST.exe dedicated startup installer by selecting [Start] menu -> [Program] -> [NEC Electronics Tools] -> [Device File Installer]. |

**Caution:** When ID850NWC is already installed, ID850QB cannot be installed in the same machine. In this case, uninstall ID850NWC before installing ID850QB.

## 2.2 Uninstalling

Perform uninstallation using [Add/Remove Programs] on the Control Panel.

# CHAPTER 3 STARTING AND TERMINATING

This chapter explains the following items related to the starting and terminating the ID850QB:

- Cautions Before Starting (When N-Wire CARD Or MINICUBE Is Connected)
- Startup Option And Argument Specification
- Starting
- Terminating
- Error Messages At Start Up

## 3.1 Cautions Before Starting (When N-Wire CARD Or MINICUBE Is Connected)

When N-Wire CARD or MINICUBE is connected, start the N-Wire Checker before the starting the ID850QB to check that the in-circuit emulator and the target system can be normally debugged.

**Caution:** For the connection between the in-circuit emulator and the target system and the power application sequence, refer to the N-Wire CARD or MINICUBE User's Manual. Incorrect connection may damage the in-circuit emulator and the target system.

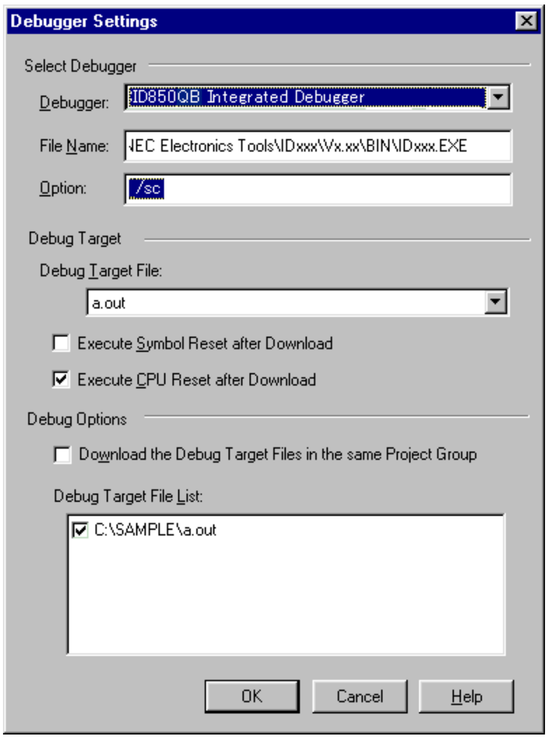
## 3.2 Startup Option And Argument Specification

The procedure for specifying the startup options and arguments for the ID850QB is described below.

By specifying the startup options and arguments, it is possible to specify the script file at startup and the project file.

**Remark:** When starting up the ID850QB from PM+, the startup option and argument settings are performed in [Debugger Settings...] in the [Tool] menu of PM+ (refer to "CHAPTER 4 ASSOCIATION WITH PM+"). The debugger startup option can be set to the option column.

Figure 3-1 [Debugger Settings] Dialog Box (PM+)



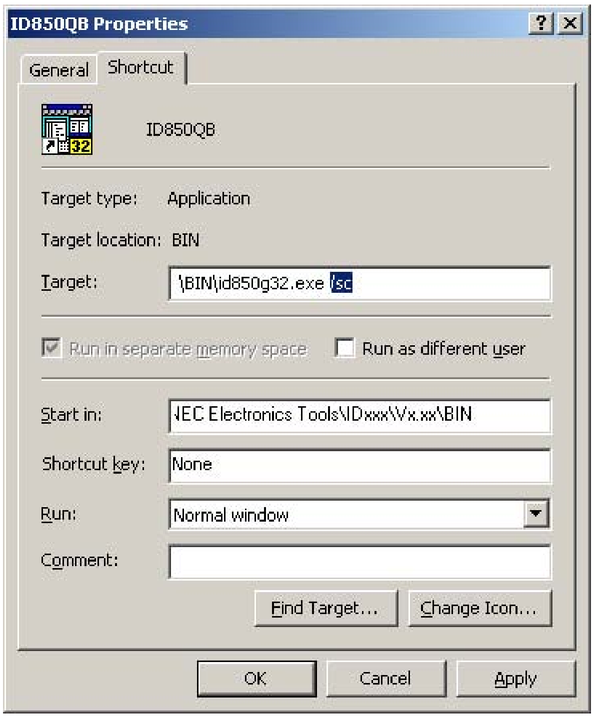
### 3.2.1 Specification method

1) Create the ID850QB shortcut on the desktop.

   The ID850QB execution file is located in the bin folder in the folder to which the installation was performed.

2) Open the properties of the created shortcut and after the execution file name displayed in [Target:], specify the option and argument (refer to "3.2.2 Specification format and options").

Figure 3-2 Startup Option (Example)



### 3.2.2 Specification format and options

**(1) Specification format**

```
id850g32.exe ?options?
id850g32.exe ?options? project
```

Each option and argument is separated by a space. The case is distinguished in the character string.

Arguments enclosed between '?' can be omitted.

When a project file is specified, that project file is read at startup.

However, during PM+ startup, the project file specification is ignored.

When there are spaces in the file names and paths, specify the project file names and script file names enclosed in double quotation marks (" "). (Refer to " Example3) Specification when there are spaces in the path".)

**(2) Specification options**

The following options can be specified.

Table 3-1 Startup Options

| Options | Meaning |
|---|---|
| /SC | Change background color of window to system color. |
| /SCRIPT:*script file name* | Specify the script file to be executed at startup. |

**(3) Specification example**

**Example1) Specification of script file only**

```
id850g32.exe /script:c:/work/script.tcl
```

**Example2) Specification of script file and project file**

```
id850g32.exe /script:c:/work/script.tcl c:/work/project.prj
```

**Example3) Specification when there are spaces in the path**

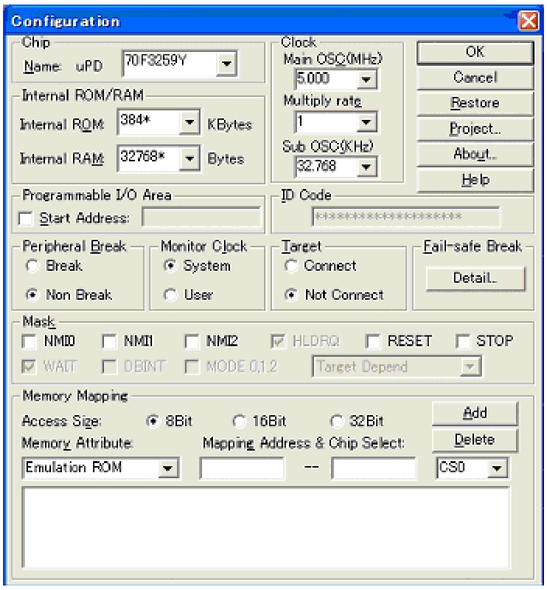
```
id850g32.exe /script:"c:/work folder/script.tcl" "c:/work folder/project.prj"
```

## 3.3 Starting

1) Start ID850QB from the [Start] menu of PM+ or by clicking the shortcut created on the desktop.
   Refer to "4.3 To Start ID850QB From PM+" when starting from PM+.
   Start the ID850QB, the Configuration Dialog Box will be opened.

**Caution:** In this case, the Configuration Dialog Box should not be displayed, but an error message should be displayed, please cope with it with reference to "3.5 Error Messages At Start Up".

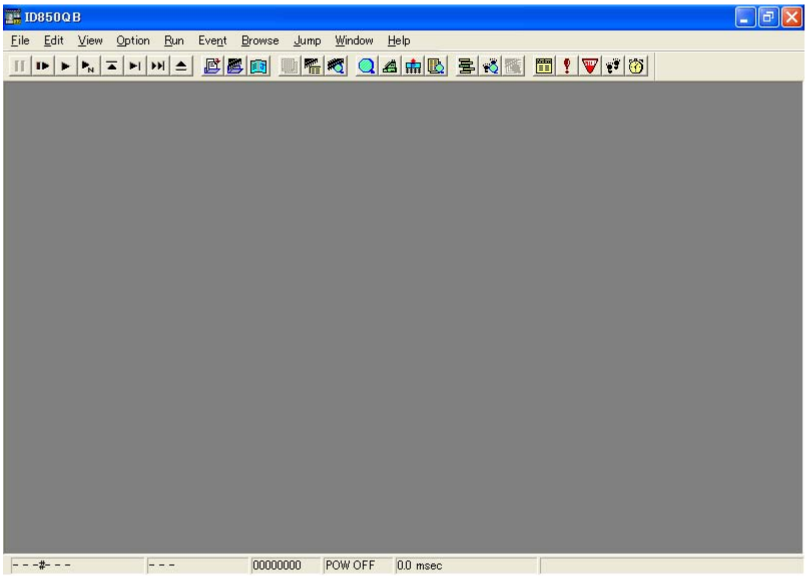
Figure 3-3 Configuration Dialog Box




2) Set the items related to the operating environment of the ID850QB in the Configuration Dialog Box. After setting each item, click the <OK> button in the dialog box.

3) The Main window will be opened and the ID850QB can be operated. Mainly use this window for debugging.

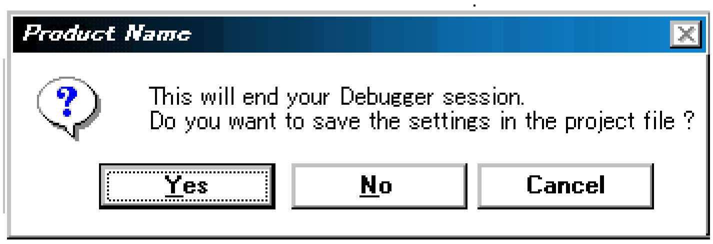
Figure 3-4 Main Window (At Startup)




## 3.4 Terminating

1) Select [File] menu -> [Exit] on the Main window. The following the Exit Debugger Dialog Box will be opened: (An execution stop confirmation message is displayed when stop operation is performed during program execution.)

Figure 3-5 Exit Debugger Dialog Box




2) To save the current debugging environment to a project file, click the <Yes> button. If the <No> button is clicked, all the windows are closed the ID850QB terminated.

# 3.5 Error Messages At Start Up

Error messages that may be output when ID850QB starts up are listed below (by order of occurrence). When these messages are output, refer to "APPENDIX D MESSAGES".

**Caution:** If both IECUBE and N-Wire CARD are connected, the USB connection (IECUBE) has priority.

## 3.5.1 When IECUBE is connected

The pattern of the output error message differs as follows depending on the connection status with the target and the settings in the Configuration Dialog Box.

Table 3-2 Error Message Output Pattern (When IECUBE Is Connected)

| Error Message | [Target] Area in the Configuration Dialog Box | | Target | | Exchange Adapter | | Target Power Supply | |
|---|---|---|---|---|---|---|---|---|
| | Connect | Not Connect | Connect | Not Connect | Used | Not Used | ON | OFF |
| Ff606: Please check connection with the target board, and power on it. | Checked | | | | | | | Checked |
| Wf607: Please check connection of the exchange adapter. | | Checked | | Checked | | Checked | | Checked |
| Ff608: Please disconnect the target board. | | Checked | Checked | | | | | Checked |
| Ff609: Please power off the target board, and disconnect it. | | Checked | | | | | Checked | |

| |
|---|
| F0100: Can not communicate with ICE. Please confirm the installation of the device driver for the PC interface board. |
| F0c43: Can not communicate with ICE. Please confirm the power of ICE, connection of the interface cable. |
| F0c70: DCU cannot be accessed. |
| F0c76: Initial state at the time of DCU access start is unusual. |
| F0c77: DCU access is unusuall. |
| Ff606: Please check connection with the target board, and power on it. |
| Wf607: Please check connection of the exchange adapter. |
| Ff608: Please disconnect the target board. |
| A0105: Failed in reading device file (d3xxx.800). |
| F0ca2: This device file does not include the on-chip debug information. |

| |
|---|
| F0ca4: This device file does not include the IECUBE information. |
| F0c71: Reset cannot be performed. |
| F0c72: Monitor memory cannot be accessed. |
| F0c73: Monitor execution cannot be performed. |
| F0c74: CPU register cannot be accessed. |
| F0c23: Bus hold under continuation. |
| A0c01: During access of register, CPU did time out. |
| A0c02: During access of memory, CPU did time out. |
| A0c03: During access of I/O register, CPU did time out. |
| A01a0: No response from the evachip. Please confirm the signal of the CLOCK or RESET WAIT, HLDRQ and so on.(IECUBE) No response from the CPU. Please confirm the signal of the CLOCK or RESET WAIT, HLDRQ and so on.(N-Wire CARD, MINICUBE) |

### 3.5.2 When N-Wire CARD or MINICUBE is connected

| |
|---|
| F0100: Can not communicate with ICE. Please confirm the installation of the device driver for the PC interface board. |
| F0c43: Can not communicate with ICE. Please confirm the power of ICE, connection of the interface cable. |
| F03a0: Target is not turned on. |
| F0c70: DCU cannot be accessed. |
| F0c76: Initial state at the time of DCU access start is unusual. |
| F0c77: DCU access is unusuall. |
| A0105: Failed in reading device file (d3xxx.800). |
| F0ca2: This device file does not include the on-chip debug information. |
| F0ca3: Unsupported information is included in the on-chip debug information in the device file. |
| F0c24: It cannot shift to debug mode. |
| F0c72: Monitor memory cannot be accessed. |
| F0c73: Monitor execution cannot be performed. |
| F0c74: CPU register cannot be accessed. |
| F0c23: Bus hold under continuation. |
| A0c01: During access of register, CPU did time out. |
| A0c02: During access of memory, CPU did time out. |
| A0c03: During access of I/O register, CPU did time out. |
| A01a0: No response from the evachip. Please confirm the signal of the CLOCK or RESET WAIT, HLDRQ and so on.(IECUBE) No response from the CPU. Please confirm the signal of the CLOCK or RESET WAIT, HLDRQ and so on.(N-Wire CARD, MINICUBE) |

# CHAPTER 4 ASSOCIATION WITH PM+

The ID850QB can automatically perform a series of operations in development processes, such as creating source files -> compiling -> debugging -> correcting source files, in association with the PM+.

This chapter explains the following items related to association with the PM+.

For details of the PM+ functions, refer to the PM+ User's Manual.

- Setting Build Mode
- Registering Debugger To PM+ Project
- To Start ID850QB From PM+
- Auto Load

**Caution:** If a load module file is created by using the Windows command prompt, the function to associate the ID850QB with the PM+ cannot be used.

## 4.1 Setting Build Mode

To debug the load module file created by the PM+ on the ID850QB at the source level, build to output symbol information for debugging must be performed to create a load module file. This setting can be performed by selecting [Debug Build] on the PM+.

## 4.2 Registering Debugger To PM+ Project

The debugger to be used or the load module files to be downloaded can be specified for each project in the PM+.

### 4.2.1 Selecting debugger

The procedure for selecting the debugger is as follows:

The ID850QB is registered as the debugger of the active project. The ID850QB icon is displayed on the tool bar of the PM+.

**(1) Creating a new workspace**

1) Select [File] menu -> [New Workspace...] on the PM+.
   -> This opens the dialog box to create a new workspace using the wizard format.
2) Creating the necessary settings for the workspace with the wizard, the [Select Debugger] dialog box will be opened. Specify ID850QB in this dialog box.
   For details of the setting, refer to the User's manual.

**(2) Using an existing workspace**

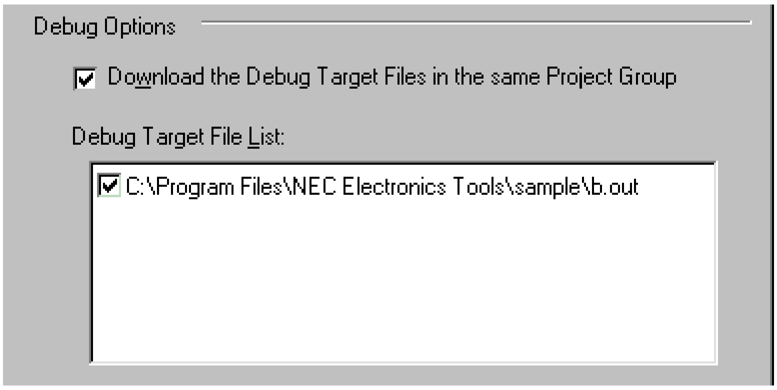
1) Select [Tool] menu -> [Debugger Settings...] on the PM+.
   -> The [Debugger Settings] dialog box will be opened.
2) Specify ID850QB and click the <OK> button in this dialog box. For details of the setting, refer to the User's manual.

### 4.2.2 Downloading multiple load module files

Load module files in the same project group can be downloaded when using ID850QB.

Download multiple load module files by specifying items in the [Debugger Settings] dialog box of PM+.

Figure 4-1 Downloading Multiple Files




For details of the project group, refer to the "User's manual".

**Remark:** Multiple load module files that are being downloaded can be checked in the Load Module List Dialog Box of the ID850QB.

**Caution:** If [Execute symbol reset after download] in the [Debugger Settings] dialog box of the product with a internal flash memory is checked, the contents of the internal flash memory are erased before downloading (when N-Wire CARD, MINICUBE is connected)

# 4.3 To Start ID850QB From PM+

The ID850QB can be started from the PM+ as follows:

- Click the ID850QB starting button on the tool bar of the PM+.
- Select the [Build] menu -> [Debug] on the PM+.
- Select the [Build] menu -> [Build and Debug] on the PM+.
- Select the [Build] menu -> [Rebuild and Debug] on the PM+.

If the debugging environment of the ID850QB is saved to a project file currently being used by the PM+, it will be started in the debugging environment saved in the project file.

If the debugging environment of the ID850QB is not saved to a project file being used by the PM+, the Configuration Dialog Box is opened. At this time, the device type (chip name) cannot be changed.

## 4.3.1 Restoring debugging environment

The previous debugging environment can be restored by the following procedure when the ID850QB is started from the PM+:

1) Create a new workspace (project file: e.g., sample.prj) on the PM+Note.
2) Start the ID850QB from the PM+. Because a new project file is created, set items other than the device type (chip name) in the Configuration Dialog Box in the same manner as when only the ID850QB is started.
3) Download the load module file to be debugged with the Download Dialog Box of the ID850QB.
4) Debug the load module file on the ID850QB.
5) Click the <Yes> button on the Exit Debugger Dialog Box when the ID850QB is terminated.
   -> The debugging environment will be saved to the project file (sample.prj) for the PM+ when the ID850QB is terminated (the debug environment can also be saved to the sample.prj file by overwriting the project file at times other than the completion of ID850QB debugging).
6) When the ID850QB is next started up after the sample.prj file is read by PM+, the debug environment at the point when the project file was saved is automatically restored.

**Note:** In the ID850QB and PM+, the environment information is saved to a project file and referenced. The extension of the project file that can be used by the ID850QB and PM+ is "prj". For the information that is saved or restored by the project file, refer to the "User's manual" of each product.

# 4.4 Auto Load

If a bug is found while the load module file is being debugged by the ID850QB, correct the source file using the following procedure. Compiling and re-downloading the file can be automatically executed. (Refer to "4.4.1 Auto load by correcting source code".)

The load module is downloaded again to the ID850QB by compiling and linking the file on the PM+ with the activated ID850QB. (Refer to "4.4.2 Auto load by starting debugger".)

**Caution:** This processing cannot be performed if it is selected that the standard editor (idea-L) is used with the PM+.

## 4.4.1 Auto load by correcting source code

Correct the source file for auto load as follows:

1) Open the source file to be corrected in the Source Window. Select [File] menu -> [Open] and specify the file to be corrected on the ID850QB (if the file is already open in the Source Window, that window is displayed in the forefront).
   -> The specified file will be opened in the Source Window.
2) Select [Edit] menu -> [Edit Source] on the ID850QB.
   -> An editor will be opened and the specified source file will be read.
3) Correct the source file on the editor.
4) Terminate the editor.

**Caution:** The CPU reset is not performed when the load module file is automatically downloaded. The debug window that was opened when the editor was called, and each event setting will be restored. If the previously used line or symbol has been deleted as a result of correcting the source file, the following happens:
- A variable that was displayed is dimmed.
- The event mark of an event condition is displayed in yellow.
- A software break point may be deleted.

5) Select [Build] menu -> [Build and Debug], or [Build] menu -> [Rebuild and Debug] on the PM+.

### 4.4.2 Auto load by starting debugger

If the following operation is performed on the PM+ with the ID850QB started, the load module will be automatically downloaded to the ID850QB.

- Selecting the [Build] menu -> [Build and Debug] on the PM+.
- Selecting the [Build] menu -> [Rebuild and Debug] on the PM+.

**Remark:** Specify whether to use a CPU reset after downloading from [Debugger Settings...] on the [Tool] menu of PM+ (a CPU reset is performed by default).

# CHAPTER 5 DEBUG FUNCTION

This chapter explains about debug function of ID850QB.

Table 5-1 Debug Function List (Flow of Debugging Operations)

| Function | Refer To |
|---|---|
| To set the debugging environment | 5.1 Setting Debugging Environment |
| To download the load module | 5.2 Download Function, Upload Function |
| To display the source file and the disassemble result | 5.3 Source Display, Disassemble Display Function |
| To set a break point | 5.4 Break Function |
| To execute the user program | 5.5 Program Execution Function |
| To check the variable value | 5.6 Watch Function |
| To check and edit the memory contents | 5.7 Memory Manipulation Function |
| To check and change the register variable | 5.8 Register Manipulation Function |
| To check the execution time | 5.9 Timer Function (When IECUBE Is Connected) |
| To check the trace data | 5.10 Trace Function (When IECUBE Is Connected) |
| To check the code coverage measurement results | 5.11 Coverage Measurement Function (When IECUBE Is Connected) |
| To manage the events | 5.12 Event Function |
| RRM function | 5.13 RRM Function |
| DMM function | 5.14 DMM Function |
| To save the debug environment and window status | 5.15 Load/Save Function |
| Jump function, linking window and cautions | 5.16 Functions Common to Each Window |

## 5.1 Setting Debugging Environment

This section explains the following items related to the setting debugging environment:

- Setting operating environment
- Setting option
- Setting mapping
- To change the value of a register required for access of an external memory

### 5.1.1 Setting operating environment

The in-circuit emulator operating environment settings are performed in the Configuration Dialog Box that is automatically displayed when ID850QB starts up.

If a project file already exists, the debugging environment can be restored by clicking the <Project...> button. (Refer to "5.15.1 Debugging environment (project file)".)

### 5.1.2 Setting option

Perform setting related to the debugger or in-circuit emulator in the following setting dialog boxes.

- Configuration Dialog Box
- Extended Option Dialog Box
- Fail-safe Break Dialog Box
- RRM Setting Dialog Box
- Flash Option Dialog Box
- Debugger Option Dialog Box

### 5.1.3 Setting mapping

The mapping settings are performed in the Configuration Dialog Box.

The following types of mapping attributes are available:

Table 5-2 Mapping Attribute

| Attribute | Meaning |
|---|---|
| Internal ROM | Internal RAM(When IECUBE is connected, with memory board)<br>A memory area specified as the internal ROM is equivalent to the internal ROM of the target device (core).If the target device attempts writing to this memory area, a write protect break occurs. |
| Internal RAM | Internal RAM(When IECUBE is connected, with memory board)<br>A memory area specified as the internal RAM is equivalent to the internal RAM of the target device.<br>The actual memory configuration depends on the target system. |
| Target | User area mapping<br>The memory area specified for user area mapping becomes the area to accesses the memory in the target system or memory incorporated in the CPU. |
| Target ROM | Target ROM (When IECUBE is connected)<br>Areas specified as target ROM are subject to write protect for fail-safe break (refer to " Fail-safe Break Dialog Box"). |
| I/O Protect | I/O protect area<br>An I/O Protect area can be set in the area specified for the "target".<br>The I/O protect area is displayed in the same manner as an area that is not mapped (display symbol: ??), on the Memory Window.  By mapping an area with this attribute, data cannot be read or written from/to this area by the Memory Window, on the area can therefore be protected from an illegal access.To read or write the value of the area mapped with this attribute, register the value in the IOR Window or Watch Window. |

### 5.1.4 To change the value of a register required for access of an external memory

When mapping has been performed for external memory, must change the values of the registers required for accessing external memory prior to downloading, using the IOR Window or the hook procedure.

For how to change register values using the hook procedure, refer to "7.8 Hook Procedure".

For the registers to be changed, refer to the hardware manual of the CPU that is used.

## 5.2 Download Function, Upload Function

ID850QB allows downloading and uploading of object files in the formats listed in the following table: Table 5-3, Table 5-4.

This section explains the following items:

- Download
- Upload

**Remark:** When N-Wire CARD, MINICUBE is connected, the internal flash memory can be written and the load module can be downloaded (refer to "5.7.4 Flash memory writing function (when N-Wire CARD, MINICUBE is connected)").

### 5.2.1 Download

Object files are downloaded in the Download Dialog Box.

The corresponding source text file (Source Window) is displayed by downloading load module files with debug information.

**Remark:** Multiple load module files can be downloaded. Loaded files can be checked in the Load Module List Dialog Box that is opened by selecting [File] menu -> [Load Module].

#### (1) Format of file that can be downloaded

Format of file that can be downloaded is as follows:

Table 5-3 Type of File That Can Be Downloaded

| Format | Extension |
| --- | --- |
| Load module (ELF(.out)) | Load Module (*.out) |
| Intel Hex format **Note1** (Standard, extension, and extension linear) | Hex Format (*.hex)**Note2** |
| Motorola Hex format S type | |
| Extended Tektronix Hex format (S0, S3, S7) | |
| Binary data | Binary Data (*.bin) |
| Coverage result | Coverage (*.cvb) |

**Note1:** Addresses of 1 MB or more can be uploaded.

**Note2:** Automatic format detection

### 5.2.2 Upload

Uploading of memory contents, etc., is performed in the Upload Dialog Box. The saving range can be set.

**(1) Format of file that can be uploaded**

Format of file that can be uploaded is as follows:

Table 5-4 Type of File That Can Be Uploaded

| Format | Extension |
|---|---|
| Intel Hex format **Note1** | Hex Format (*.hex)**Note2** |
| Motorola Hex format S type (S0, S3, S7 - 32 bit-address) | |
| Extended Tektronix Hex format | |
| Binary data | Binary Data (*.bin) |
| Coverage results (when IECUBE is connected) | Coverage (*.cvb) |

**Note1:** Standard (16-bit addresses), extension (20-bit addresses), and extension linear (32-bit addresses)
Addresses of 1 MB or more can be uploaded.

**Note2:** Specify saving format

## 5.3 Source Display, Disassemble Display Function

Source file display is performed in the Source Window. Disassemble display and online assembly are performed in the Assemble Window.

This section explains the following items:

- Source display
- Disassemble display
- Mixed display mode (Source Window)
- Convert symbol (symbol to address)

**Remark:** The locations for which coverage measurement is executed in the user program are displayed in the Source Window and Assemble Window (refer to "5.11.3 Display of locations for which coverage measurement is executed").

### 5.3.1 Source display

The corresponding text file is displayed in the Source Window by downloading a load module file having debug information.

The display start position can be changed in the Source Text Move Dialog Box displayed by selecting [View] menu -> [Move...].

Specifications related to the tab size, display font, etc., and specification of the source path are made in the Debugger Option Dialog Box. Specify a searching method in the Source Search Dialog Box opened by clicking the <Search...> button.   The search result is highlighted in the Source Window.

Table 5-5 File Type Can Be Displayed

| File Type (Extension) | Meaning |
|---|---|
| Source (*.c, *.s) | Source file<br>(The extension can be changed in the Debugger Option Dialog Box.) |
| Text (*.txt) | Text file |
| All (*.*) | All files |

### 5.3.2 Disassemble display

Disassemble display is performed in the Assemble Window.

The display start position can be changed in the Address Move Dialog Box opened by selecting [View] menu -> [Move...].

Offset display and register name display are specified in the Debugger Option Dialog Box.

Specify a searching method in the Assemble Search Dialog Box opened by clicking the <Search...> button.   The search result is highlighted in the Assemble Window.

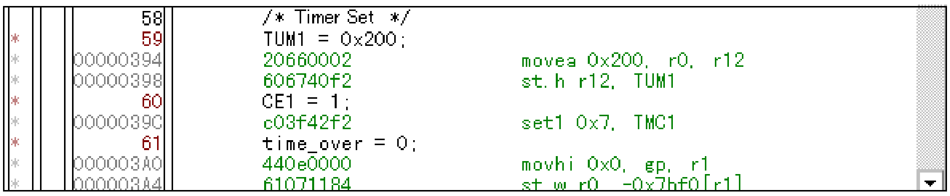
### 5.3.3 Mixed display mode (Source Window)

Programs can be disassembled and displayed combined with the source file by selecting [View] menu -> [Mix] in the Source Window. The contents displayed in the mixed display mode can be saved as a view file.

**Normal display mode**


In the normal display mode, general text files can be displayed as well as source files.

**Mixed display mode**




If a program code corresponds to the line of the displayed source file, the disassembly line is displayed next to the source line. The label of the address, code data, and disassembled mnemonic are displayed (the display start position of the mnemonic is adjusted by the set value of the tab size).

**Caution:** The mixed display mode is valid only when the load module is downloaded and the symbol information is read, and the corresponding source file is displayed.

**Remark:** When scrolling is performed using the cursor keys in the Mixed display mode, excessive scrolling may occur. Also, scrolling down to the last line may not be possible using the cursor keys.

### 5.3.4 Convert symbol (symbol to address)

In the Symbol To Address Dialog Box, can be displayed the address of the specified variable or function, or the value of the specified symbol.

Convert symbol is performed by selecting the character string to be converted in the Source Window or Assemble Window, and then selecting context menu -> [Symbol...].

The Specification symbols is indicated below.

Table 5-6 Specifying Symbols

| Conversion Target | Specification Method |
|---|---|
| Variable | var<br>file#var(to specify a static variable with file name)<br>func#var(to specify a static variable with function name)<br>file#func#var(to specify a static variable with file name and function name) |
| Function | func<br>file#func (to specify a static function with file name) |
| Label | label<br>file#label(to specify a local label with file name) |
| Line number of source file | file#no<br>prog$file#no |
| I/O port name | portname |
| I/O register name | I/O regname |
| Register name | regname |
| PSW flag name | pswname |

**Remarks1:** Separator "#"

"#" is used as a separator for file names, variables, function names, and line numbers. If a specified symbol is not found in the scope, all symbols (static variables, static functions, local labels) are searched.

**Remarks2:** Separator "$"

To specify a load module name when two or more load modules are read, use "$" as a separator to delimit the load module name from a file name, variable, function name, or symbol name.

In the default status, a symbol name takes precedence. To temporarily change the priority, prefixing "$" to a symbol gives the priority to a register name.

## 5.4 Break Function

The break function is used to stop execution of the user program by the CPU and operation of the tracer.

This section explains the following items:

## 5.4.1 Break types

The ID850QB has the following break functions.

Table 5-7 Break Types

| Item | Contents |
|---|---|
| Hardware break Note1<br>(Event detection break) | Function to stop user program execution upon detection of the set break event condition.<br>-> Refer to "5.4.2 Breakpoint setting". |
| Software break Note1 | Function to replace the instruction at the specified address software break instruction and stop the user program executed (refer to "5.4.4 Hardware break and software break").<br>-> Refer to "5.4.2 Breakpoint setting". |
| [Come Here] break Note2<br>(Simple break) | Function to stop user program execution selected by selecting [Run] menu -> [Come Here] upon detection of address specified in the Source Window or Assemble Window. |
| Break on satisfaction of condition of step execution | Function to stop execution upon satisfaction of the stop condition of each command ([Step In], [Next Over], [Return Out], [Slowmotion]). |
| Forced break | Function to forcibly stop execution by selecting [Run] menu<br>-> [Stop], or selecting the STOP button.<br>It is valid for all the execution commands. |
| Fail-safe break | Function to forcibly stop execution when the user program performs an illegal operation in relation to the memory or registers (refer to "5.4.5 Fail-safe break function").<br>-> Refer to " Fail-safe Break Dialog Box". |
| Time-out break | Function to stop user program execution when the measurement time exceeds the specified time-out time (refer to " Timer Dialog Box"). |

**Note1:** This break is valid for [Go], [Go & GO], [Come Here] and [Restart].

**Note2:** After user program execution has been stopped, the breakpoint by this function is eliminated. During execution of a user program by this function, break events set before the cursor position does not occur.

## 5.4.2 Breakpoint setting

Breakpoints can simply be set to the desired location by clicking in the Source Window or Assemble Window.

Since breakpoints are set as break event conditions and managed using the Event Function, restrictions apply to the number of breakpoints that can be set. (Refer to "5.12.4 Number of enabled events for each event condition".)

### (1) Breakpoint setting method

Breakpoints are executed by clicking lines in which " * " is displayed (lines where program code exists).

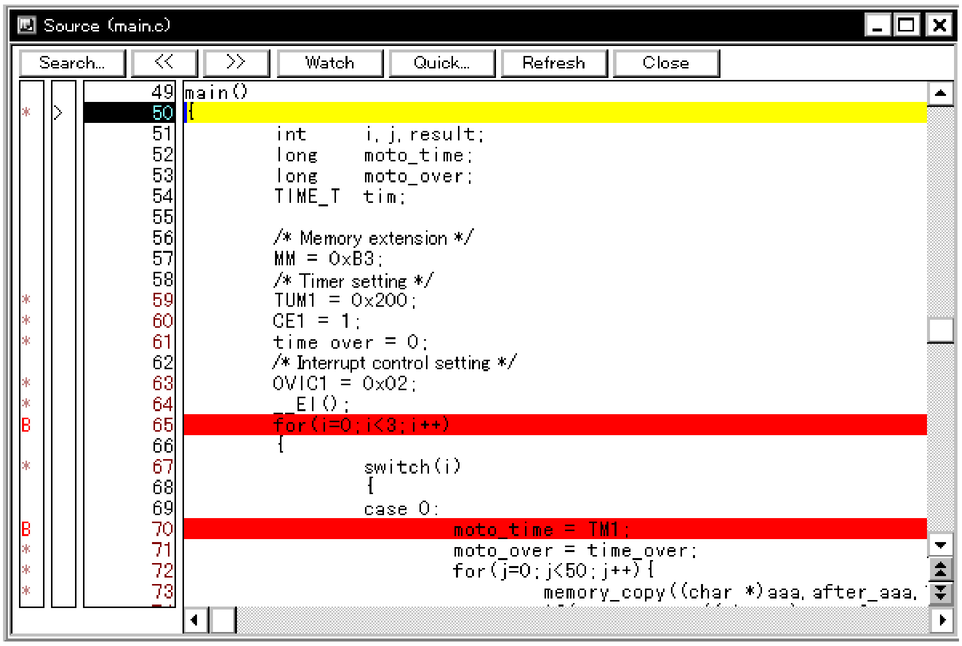
In the default setting, software breakpoint (B) is set, but if [Breakpoint] is selected in the context menu, hardware breakpoint (B, or B) is set.

If a breakpoint is set on a line on which an event breakpoint has already been set, "A" indicating that multiple events have been set is marked (refer to "Table 6-9 Event Setting Status (Event Mark)").

**Caution:** A software breakpoint cannot be set/delete in an externally mapped ROM area.

**Remark:** Breaks set by default can also be changed in the Extended Option Dialog Box.

Figure 5-1 Breakpoint Setting




### (2) Deleting a breakpoint method

Click the position at which the breakpoint to be deleted is set.

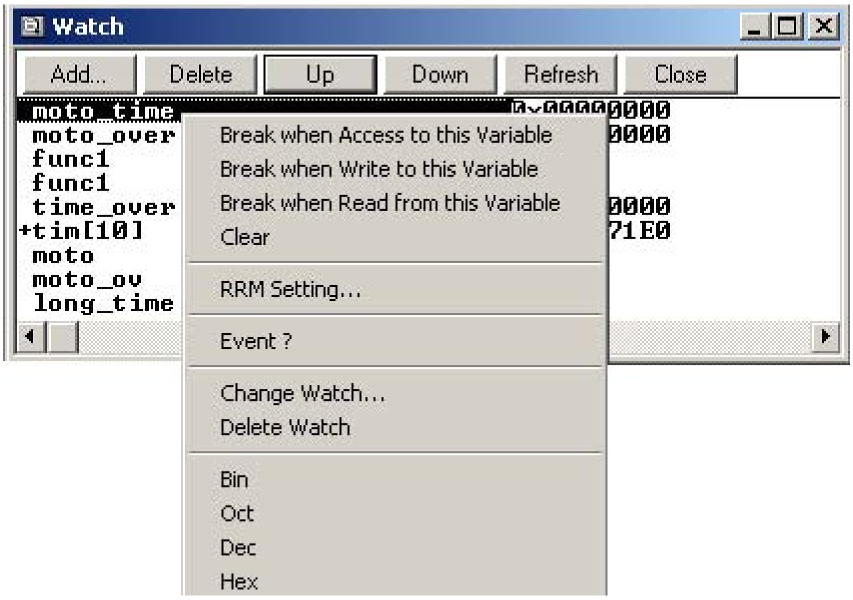
At the same time as setting is performed, in the default setting, software breakpoint (B) is deleted, but if [Breakpoint] is selected in the context menu, hardware breakpoint (B, or B) is deleted.

As a result of deletion, If another event remains, however, the mark of that event is displayed.

### 5.4.3 Setting breaks to variables

Access breaks can easily be set to variables from the context menu in the Source Window or Watch Window.

Figure 5-2 Setting Break to Variable



### 5.4.4 Hardware break and software break

**(1) Hardware break**

Hardware breaks are breaks that are set using one hardware resource per event condition.

Therefore, in the ID850QB, they are managed using "5.12 Event Function" as break event conditions.

The number of valid break points varies depending on the device (refer to "5.12.4 Number of enabled events for each event condition").

**(2) Software break**

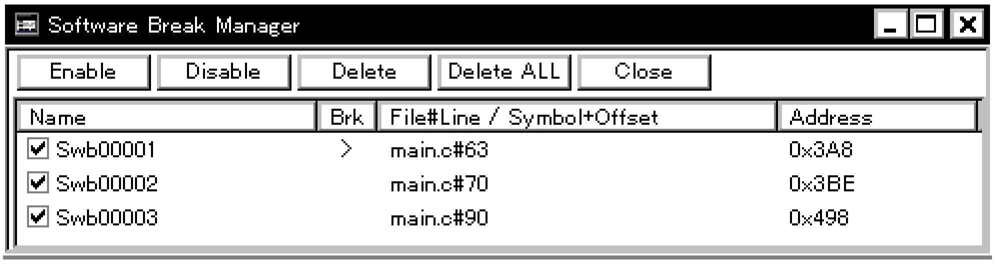
Software breaks are breaks that are set by rewriting instructions of specified addresses to software break instructions. Therefore, the number of software breaks that can be set is not limited, but settings to external ROM, stopping at variable access timing, etc., cannot be specified.

Table 5-8 The Number of Valid Software Break

| Connected IE | Valid Number |
|---|---|
| IECUBE | 2000 |
| N-Wire CARD,<br>MINICUBE | 2000<br>**Note:**<br>Software breaks in relation to internal ROM and internal flash memory are automatically set by the ROM collection function.<br>The maximum number of software breaks that can be set with the ROM collection function depends on the product (0, 4, or 8). Moreover, the software breaks set with the ROM collection function are temporarily disabled by target reset or internal reset, but they are enabled when a break occurs. |

Software break is managed by the Software Break Manager.

Figure 5-3 Management of Software Breaks



### 5.4.5 Fail-safe break function

The fail-safe break settings are performed in the Fail-safe Break Dialog Box.

Individual settings are possible by selecting check boxes.

**Remark:** However, the protect setting for the internal RAM area is performed through verify processing by software, and therefore a warning is displayed during breaks.
During IOR Illegal and internal RAM verify checks, the address is displayed on the status bar.

Figure 5-4 Fail-safe Break Setting

## 5.5 Program Execution Function

The program execution function is used to start/stop execution of the user program by the CPU and operation of the tracer.

Through user program execution, the program counter (PC) advances until the set breakpoint or forced break. (Refer to "5.4 Break Function".)

**Remark:** While the user program is being executed, trace event condition and timer event condition can be set. (Refer to " Trace Dialog Box", " Timer Dialog Box".)

The following types of ID850QB execution functions are provided. They are operated using the execution buttons on the tool bar , or from the [Run] menu.

Figure 5-5 Execution Button


Figure 5-6 [Run] Menu

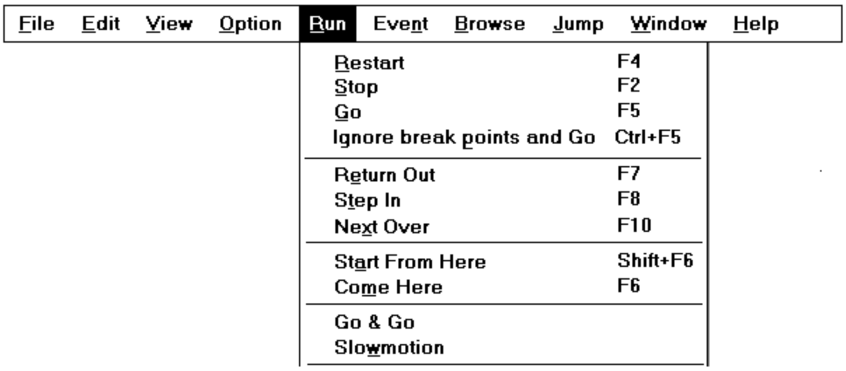


Table 5-9 Type of Execution

| Items | Contents |
|---|---|
| [Restart] | The CPU is reset and the user program is executed starting from address 0. This is the same operation as "resetting the CPU before execution of the user program and executing [Go]". |
| [Go] | The user program is executed starting from the address indicated by the current PC register and is stopped if a set break event condition is satisfied. |
| [Ignore break points and Go] | The user program is executed starting from the address indicated by the current PC register Execution of the user program continues, ignoring set breakpoints. |
| [Return Out] | The user program is executed until execution returns to the calling function described in C language. |
| [Step In] | In the source mode, step execution of one line of the source text is performed starting from the current PC register value and the contents of each window are updated.<br>In the instruction mode, one instruction is executed from the current PC register value and the contents of each window are updated. |
| [Next Over] | **jarl instruction:**<br>Next step execution is performed, assuming the function or subroutine called by the jarl instruction as one step (step execution continues until the nesting level becomes the same as when the jarl instruction was executed).<br>**Instruction other than jarl:**<br>The same processing as [Step In] is performed. |
| [Start From Here] | This command executes the user program starting from the specified address. Execution of the user program is stopped when a set break event condition is satisfied. |
| [Come Here] | The user program is executed from the address indicated by the current PC register to the address selected in the line/address display area of the Source Window or Assemble Window, and then a break occurs.<br>While the user program is being executed, the break event currently set does not occur. |
| [Go & Go] | The user program is executed starting from the address indicated by the current PC register and stopped if a set break event condition is satisfied. The contents of each window are updated, and execution of the user program is resumed from the address at which the program was stopped.<br>This operation is repeated until the user executes [Stop]. |
| [Slowmotion] | Step execution of one line is performed from the address indicated by the current PC register value in the source mode.   In the instruction mode, step execution of one instruction is performed.   The contents of each window are updated each time step execution is performed.<br>This operation is repeated until the user executes [Stop]. |
| [Stop] | Forcibly stops program execution. |

## 5.6 Watch Function

This section explains the following items related to the watch function:

- Displaying and changing data values
- Displaying and changing local variable values
- Registering and deleting watch data
- Changing watch data
- Temporarily displaying and changing data values
- Callout watch function
- Stack trace display function

## 5.6.1 Displaying and changing data values

Data values are displayed and changed in the Watch Window. Shifts in data values can be checked by registering watch data.

The display format is specified in the Debugger Option Dialog Box.

Figure 5-7 Watch Window

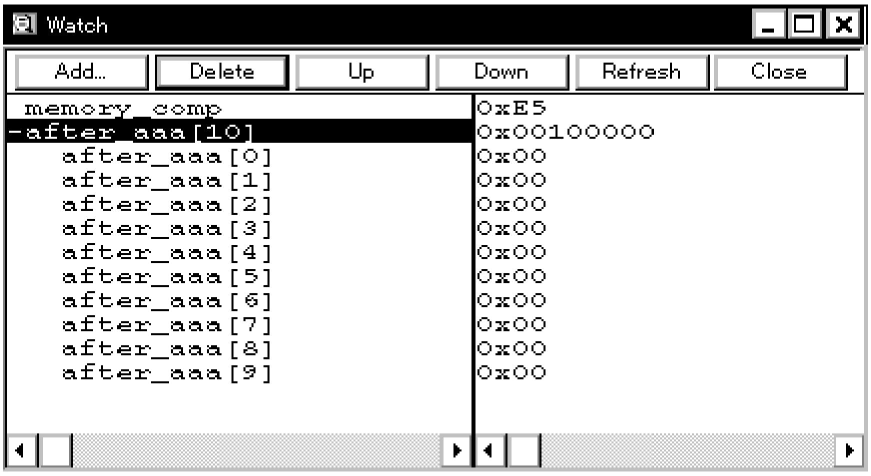
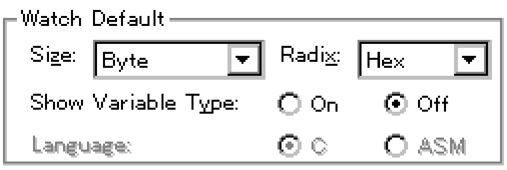



Figure 5-8 Specification of the Display Format (Debugger Option Dialog Box)

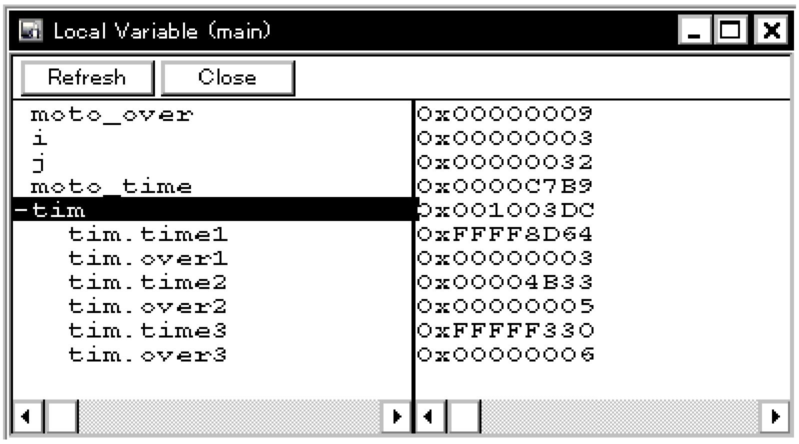


### 5.6.2 Displaying and changing local variable values

Local variables are displayed and changed in the Local Variable Window.

Local variables within the current function are automatically displayed in this window. (Variable addition/deletion is not possible.)

Figure 5-9 Local Variable Window




### 5.6.3 Registering and deleting watch data

Data can be registered to the Watch Window from the Source Window or Assemble Window. This is simply done by selecting the variable or symbol name in the respective window, and then clicking the <Watch> button. Registration is also possible with the following method.

- Drag and drop the selected variable or symbol name directly on the Watch Window. (Refer to "5.16.4 Drag & drop function".)
- Click the <Add> button in the Quick Watch Dialog Box or Add Watch Dialog Box.

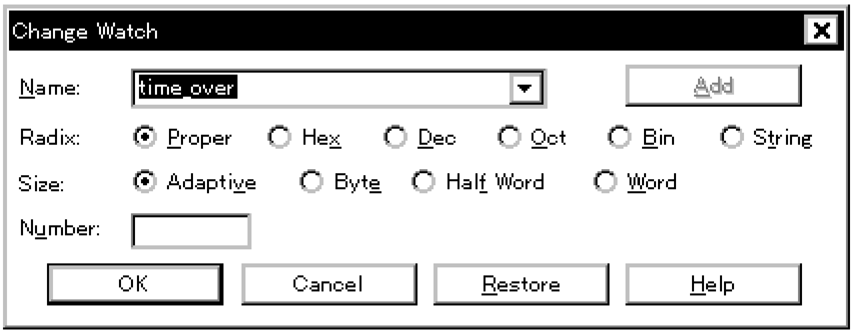
To delete watch data, click the variable name or symbol name (multiple selections can also be made using the Shift key or Ctrl key), and then click the <Delete> button. However, lines with an expanded hierarchy, such as elements of an array, and members of structures and unions, cannot be deleted.

### 5.6.4 Changing watch data

Watch data is changed in the Change Watch Dialog Box.

Note that the symbol name can be changed even if it results in duplication of a name already in use with existing data.

Figure 5-10 Change Watch Dialog Box



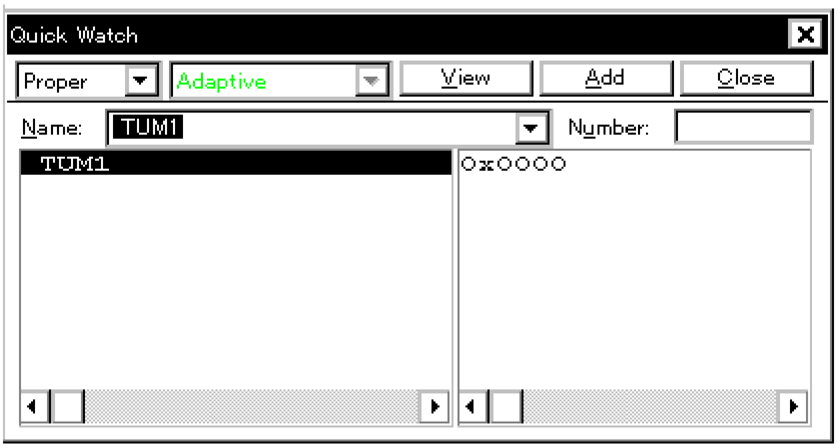

### 5.6.5 Temporarily displaying and changing data values

Data values are temporarily displayed and changed in the Quick Watch Dialog Box.

Select the desired variable or symbol name in the Source Window or Assemble Window and click the <Quick...> button to perform watch data registration.

The display radix, display size, and display number can be changed in this window.

Figure 5-11 Quick Watch Dialog Box



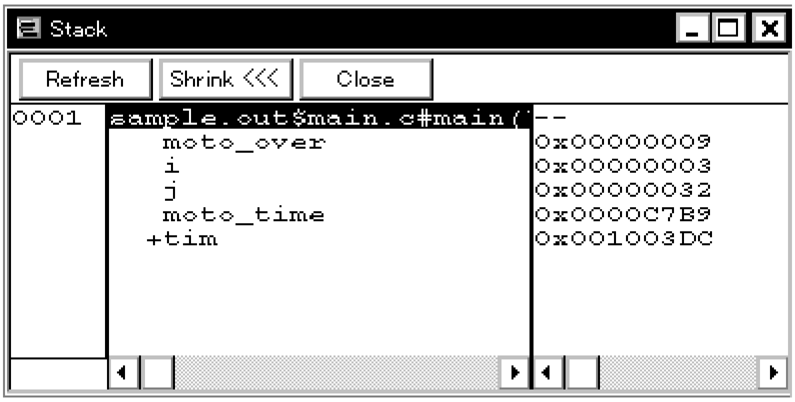
### 5.6.6 Callout watch function

The corresponding variable value pops up when the mouse cursor is placed over a selected variable in the Source Window or Assemble Window.

### 5.6.7 Stack trace display function

This function displays the stack contents of the current user program in the Stack Window.

Figure 5-12 Stack Window



## 5.7 Memory Manipulation Function

This section explains the following items related to the memory manipulation:

Verify check, etc., is specified in the Extended Option Dialog Box.

- Displaying and changing memory contents
- Filling, copying, and comparing memory contents
- Access monitor function (when IECUBE is connected)
- Flash memory writing function (when N-Wire CARD, MINICUBE is connected)

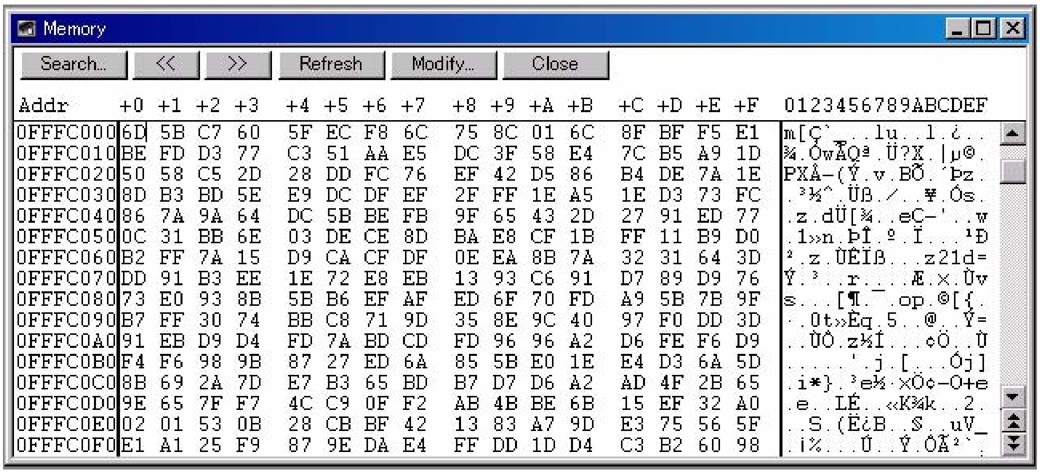
### 5.7.1 Displaying and changing memory contents

In the Memory Window, the memory contents can be displayed or changed by using mnemonic codes, hexadecimal codes, and ASCII codes. Searching is done in the Memory Search Dialog Box displayed by clicking the <Search...> button. The results of search is highlighted in the Memory Window.

The display start position can be changed in the Address Move Dialog Box displayed by selecting [View] menu -> [Move...].

The variables and data allocated to the sampling range can be displayed in real time even during program execution. (Refer to "5.13 RRM Function".)

Figure 5-13 Displaying and Changing Memory Contents




### 5.7.2 Filling, copying, and comparing memory contents

Memory contents are Filled, copied, and compared in the Memory Fill Dialog Box, Memory Copy Dialog Box, and Memory Compare Dialog Box displayed by selecting [Edit] menu -> [Memory] -> [Fill.../Copy.../Compare...].

The comparison results are displayed in the Memory Compare Result Dialog Box.

### 5.7.3 Access monitor function (when IECUBE is connected)

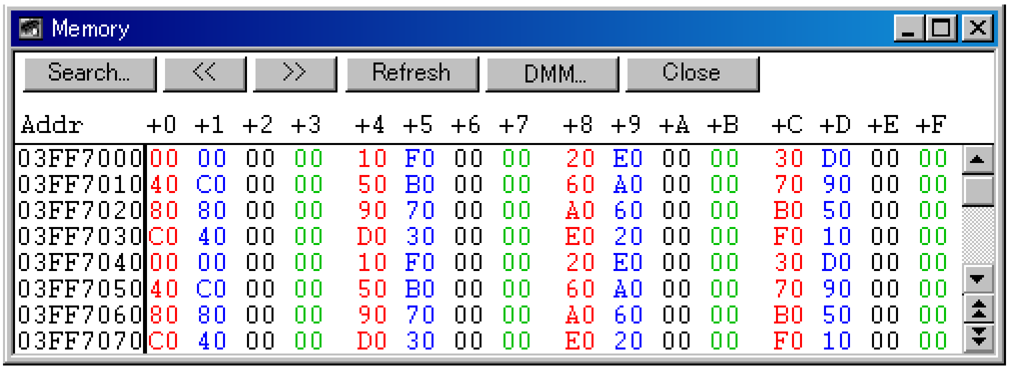
The access monitor function displays the access status (read, write, read & write) for the sampling range of the The RRM Function using different colors in the Memory Window.

The access monitor color-marking function is executed only during byte display. Colors are not displayed in the ASCII display area. Cumulative display setting and access status display can be cleared by selecting [View] menu -> [Access Monitoring].

**Cautions1:** The value of memory rewritten via DMA during program execution, and the value of memory rewritten from the debugger cannot be displayed on the access monitor.

**Cautions2:** This function is enabled only when [Option] menu -> [RRM Function] is selected.

Figure 5-14 Access Monitor Function (Memory Window)




### 5.7.4 Flash memory writing function (when N-Wire CARD, MINICUBE is connected)

With the ID850QB, the internal flash memory can be written and the load module can be downloaded by the same access method as an ordinary memory operation.

The data on the internal flash memory can be changed from the Memory Window, Assemble Window, Watch Window, Memory Fill Dialog Box and Memory Copy Dialog Box, without having to be aware that the data is that of the internal flash memory. The load module can also be downloaded to the internal flash memory by using the flash self-programming function (refer to " Flash Option Dialog Box").

**Caution:** No data can be written to the internal flash memory during user program execution.

**Remark:** With the ID850QB, the remaining area contents after the load module was downloaded to the internal flash memory are erased.

## 5.8 Register Manipulation Function

This section explains the following items related to the register manipulation function.

- Displaying and changing register contents
- Displaying and changing peripheral I/O registers contents
- Displaying and changing I/O port contents

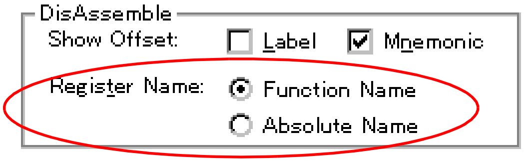
### 5.8.1 Displaying and changing register contents

Register contents can be displayed and changed in the Register Window.

Register name display switching (absolute name/function name) can be done in the Debugger Option Dialog Box.

**Remark:** The display register is selected in the Register Select Dialog Box.

Figure 5-15 Absolute Name/Function Name Switching



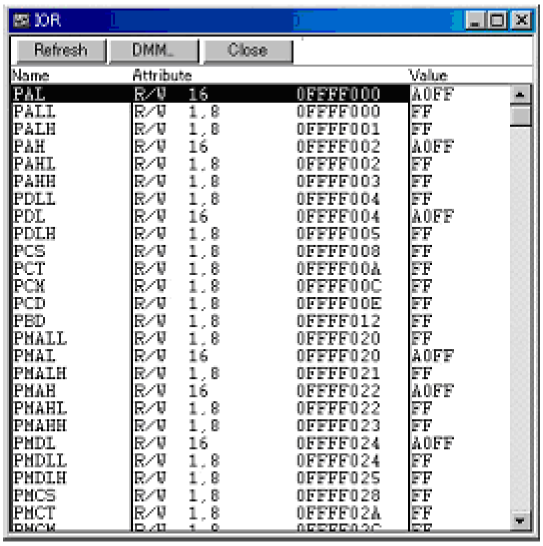

### 5.8.2 Displaying and changing peripheral I/O registers contents

The peripheral I/O registers contents can be displayed and changed in the IOR Window.

The display start position can be changed in the Address Move Dialog Box displayed by selecting [View] menu -> [Move...].

The display register is selected in the IOR Select Dialog Box.

Figure 5-16 Displaying IOR Contents

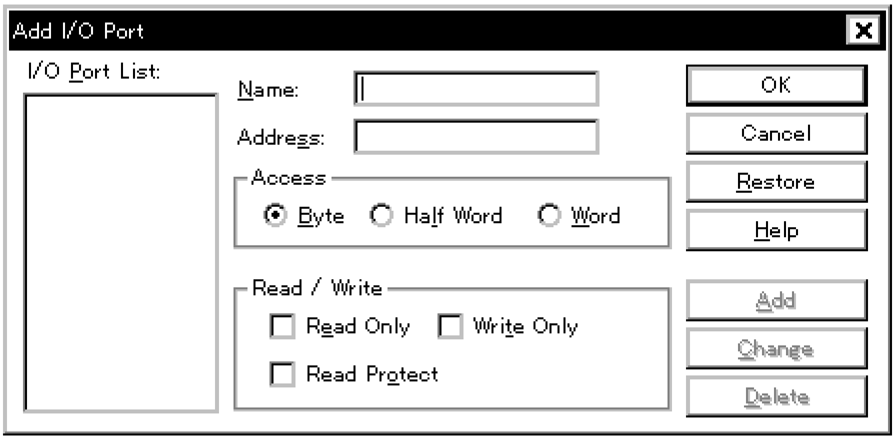


### 5.8.3 Displaying and changing I/O port contents

User-defined I/O ports can be displayed and changed in the IOR Window once they have been registered in the Add I/O Port Dialog Box.

In the case of products that support programmable I/O registers, programmable I/O register contents can be displayed and changed by setting programmable I/O area use in the Configuration Dialog Box.

Figure 5-17 Register I/O Port



## 5.9 Timer Function (When IECUBE Is Connected)

The timer function measures the execution time (run-break time) from the start of user program execution until a break, or the execution time in a specific user program interval using timer events.

The ID850QB timer function performs measurements using an external clock. Therefore, the measurable time differs based on the setting in "Table 6-8 The Relationship Between the Timer Count Division Ratio and Maximum Measurement Time (Timer counter (Timer))"」(when IECUBE is connected).

The Run-Break time is also displayed in the status bar in the Main window.

This section explains the following items:

- Timer event conditions
- Run-Break event

**Remarks1:** Timer event condition setting/enable/disable/delete operations are possible even during user program execution.

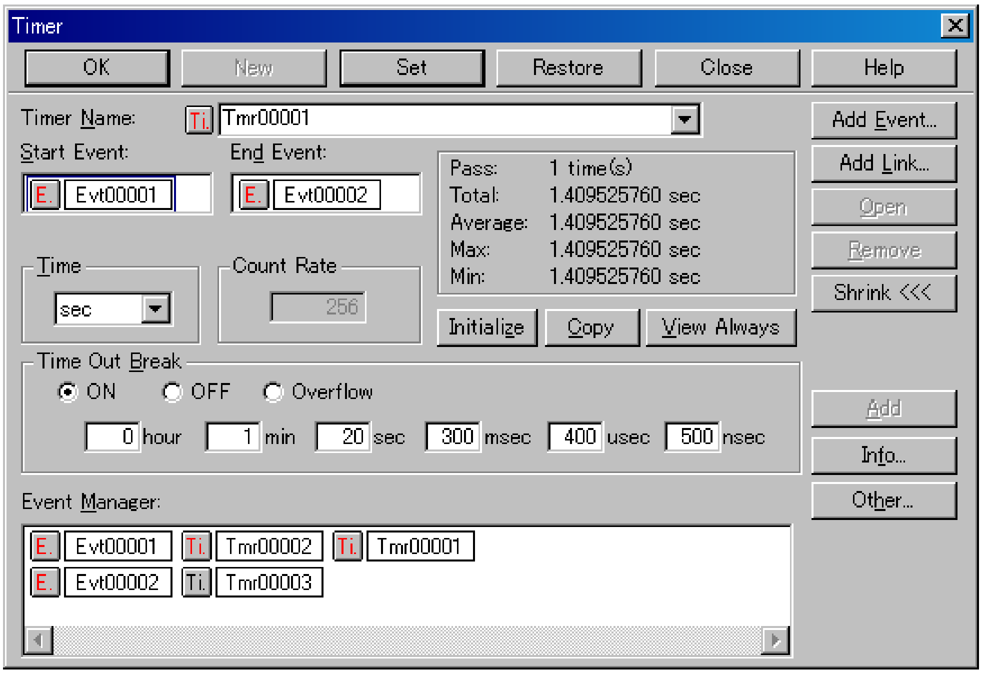
**Remarks2:** When N-Wire CARD, MINICUBE is connected, there is no timer event setting function and only Run-Break time measurement is possible. The Run-Break measurement result is displayed in the status bar of the Main window.

## 5.9.1 Timer event conditions

A timer event condition specifies the trigger by which time measurement is started or stopped. Timer event conditions are set in the Timer Dialog Box. (Refer to "5.12 Event Function".)

In the ID850QB, timeout break settings can be performed in the Time Out Break area.

Figure 5-18 Sets and Displays Timer Event (Timer Dialog Box)




Continuous display in the Timer Result Dialog Box can be selected by clicking the <View Always> button.

Timer manipulations during program execution are performed by selecting [Run] -> [Timer Start/Timer Stop].

## 5.9.2 Run-Break event

Run-Break event is a timer event name given to a timer event condition that measures the execution time from execution to break. Run-break events are registered in advance and the run-break time can be displayed through specification in the Timer Dialog Box.

The Run-Break time is also displayed in the status bar in the Main window.

Since Run-Break events are included in the number of timer events that can be simultaneously enabled (refer to "5.12.4 Number of enabled events for each event condition"), they can be used added to the number of valid timer event conditions.

## 5.10 Trace Function (When IECUBE Is Connected)

The trace function is used to save the history of the data indicating the execution process of the user program to the trace memory.

The DMA (Direct Memory Access) start point and end point are traced regardless of the trace condition. (Refer to "5.10.7 DMA point trace function".)

This section explains the following items:

- Trace memory
- Setting trace data
- Checking Trace Data
- Mixed display mode (Trace View Window)
- Tracer operation
- Setting conditional trace
- DMA point trace function

**Caution:** RRM Function, Trace Function (When IECUBE Is Connected) and Coverage Measurement Function (When IECUBE Is Connected) are functions that are used on a mutually exclusive basis (refer to "5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis").
To switch between the three functions, go to the [Option] menu.

**Remark:** Trace event condition setting/enable/disable/delete operations are possible even during user program execution. In this case, the tracer operation is momentarily stopped during manipulation.

### 5.10.1 Trace memory

ID850QB has trace memory with a ring structure. Size specification is done in the Extended Option Dialog Box.

The maximum trace memory capacity is as follows.

Table 5-10 Trace Memory Size

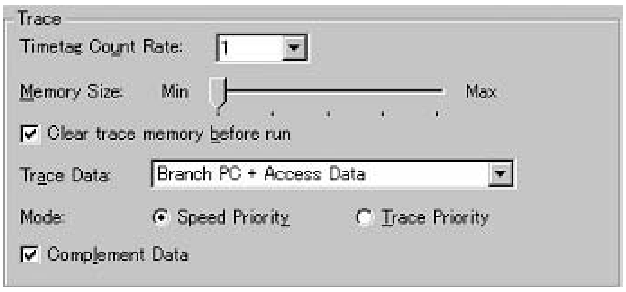
| Connected IE | Maximum Value |
|---|---|
| IECUBE | 256 KB |

### 5.10.2 Setting trace data

The detailed settings for the collected traced data are done in the Extended Option Dialog Box.

Complement display of instructions between branch instructions that cannot be traced by hardware is possible in the complement mode (enabled by selecting the Complement Data area checkbox). When the complement mode is enabled, assemble display of the internal ROM area is possible during user program execution (while the tracer is stopped).

Figure 5-19 Setting trace data



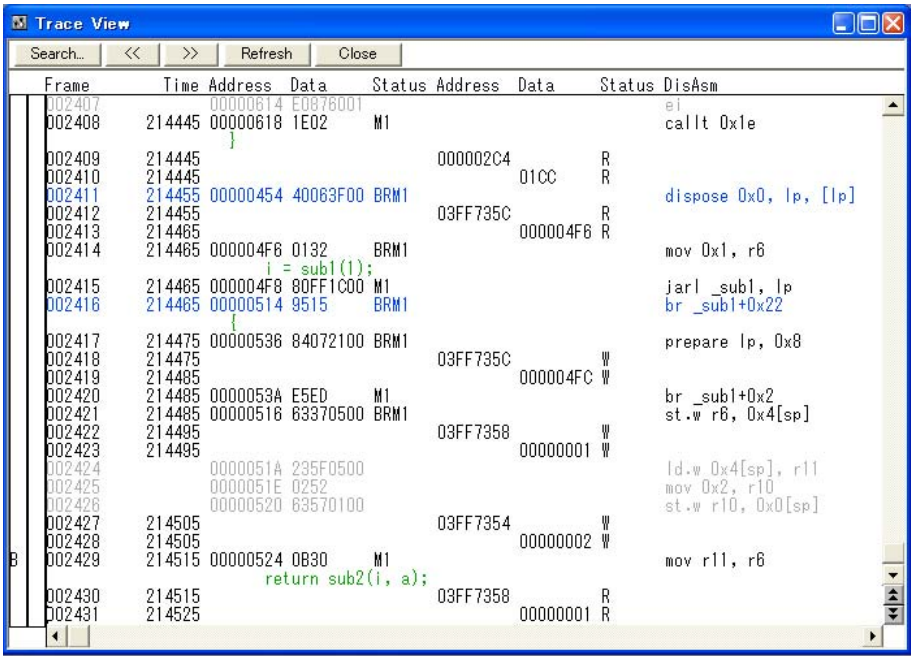
### 5.10.3 Checking Trace Data

The trace data saved to the trace memory can be checked in the Trace View Window. Trace data can be searched in the Trace Search Dialog Box displayed by clicking the < Search...> button.

The display start position can be changed in the Trace Move Dialog Box displayed by selecting [View] -> [Move].

The display items in the Trace View Window can be selected in the Trace Data Select Dialog Box.

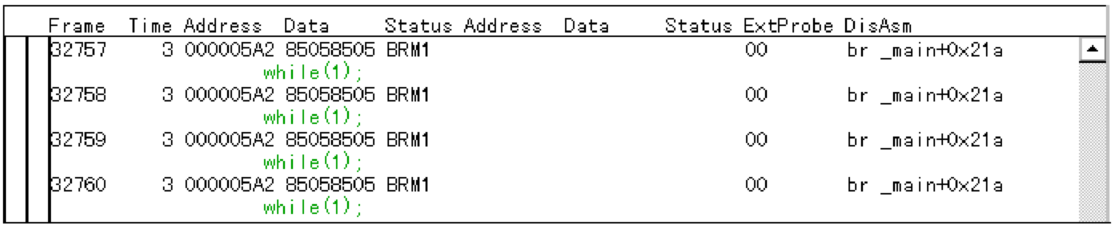
Figure 5-20 Checking Trace Data




### 5.10.4 Mixed display mode (Trace View Window)

Source file display combined with trace results can be done by selecting [View] -> [Mix] in the Trace View Window (mixed display mode).

If a program code corresponds on the program fetch address to be displayed, a source file line is displayed before the line indicating the result of tracing that program fetch address.




The source file line is displayed, emphasized in green.

**Caution:** The mixed display mode is valid only when the load module has been downloaded and symbol information is read, and when a fetch address, fetch data, fetch status, or result of disassembly is displayed.

### 5.10.5 Tracer operation

The trace operation differs as follows according to the user program execution format and the tracer control mode.

Tracer manipulations during program execution are performed by selecting [Run] -> [Tracer Start/Tracer Stop].

**(1) Operation during execution**

The tracer operation differs as follows according to [Run] -> [Cond. Trace ON/Cond. Trace ON] selection.

Table 5-11 Types of Trace Mode

| Item | Contents |
|---|---|
| Unconditional trace | Trace is started when execution of user program, and ends when a break occurs.<br>At this time, the set trace event conditions are ignored. |
| Conditional trace | Trace is started or stopped by the condition set in the Trace Dialog Box (refer to "5.10.6 Setting conditional trace").<br>If a break occurs while a trace is being executed, however, trace is stopped immediately. |

**(2) Operation during Step In execution**

The tracer operates every step execution, and trace data of one step is successively added to the trace memory.

**(3) Operation during Next Over execution**

The operation of the tracer differs depending on the instruction to which Next Over is to be executed.

(a) jarl disp22, [lp] instruction

The jarl instruction and the subroutine that was called are traced.

(b) Other instructions

The same operation as that during Step In execution is performed.

**(4) Tracer control mode**

There are the following types of trace control mode. These trace mode settings are performed from the [Run] menu.

Table 5-12 Types of Tracer Control Mode

| Mode | Contents |
|---|---|
| Non Stop | Goes around the trace memory and overwrites data from the oldest frame (default). |
| Full Stop | Goes around the trace memory and then stops the tracer. |
| Full Break | Goes around the trace memory and then stops the tracer and program execution |
| Delay Trigger Stop | Traces data by the number of delay count frames and stops the tracer when a delay trigger event has occurred. |

### 5.10.6 Setting conditional trace

A trace event condition triggers starting/stopping trace execution when a conditional trace is set.

By setting a trace event condition in the Trace Dialog Box, the conditional trace can be set (refer to "5.12 Event Function").

There are the following types of conditional trace.

Table 5-13 Types of Conditional Trace

| Item | Contents, Setting Method |
|---|---|
| Section trace | Executes a trace between two specified conditions (in a specific zone).<br>A section trace can be executed by setting a trace start event and trace end event in the Trace Dialog Box and selecting [Run] -> [Cond. Trace ON]. |
| Qualify trace | Executes a trace only when a condition is satisfied.<br>If two or more events are set as qualify trace events, a qualify trace can be executed by executing a conditional trace.<br>A qualify trace can be executed by setting a qualify trace event in the Trace Dialog Box and selecting [Run] -> [Cond. Trace ON]. |
| Delay trigger trace | Executes a trace by the number of delay counts after a condition has been satisfied.<br>A delay trigger trace can be executed by setting a delay trigger event in the Trace Dialog Box, setting a delay count in the Delay Count Dialog Box and selecting [Run] -> [Cond. Trace ON]. |

### 5.10.7 DMA point trace function

The DMA point trace (Direct Memory Access Trace) function is performed prior to normal trace functions.

For frames to be accessed using the DMA point trace function, the "M" mark is displayed in the Trace View Window.

Below are the types of DMA point trace function operations.

#### (1) During unconditional trace

Not only all the points that are traced using the normal all trace function but also the DMA start and end points are always traced using the DMA point trace function.

#### (2) During section trace

Not only the points that are traced using the normal section trace function but also the DMA start and end points, which may be located outside of the section-trace area, are always traced using the DMA point trace function.

#### (3) During qualify trace

Not only the points that are traced using the normal qualify trace function but also the DMA start and end points are always traced even when conditions for the qualify trace are not satisfied using the DMA point trace function.

## 5.11 Coverage Measurement Function (When IECUBE Is Connected)

Although there are several types of coverage measurement, the ID850QB performs measurement for C0 coverage.

C0 coverage (instruction coverage): A percentage that all statements in a code are executed at least once

Download or upload the coverage measurement result (coverage data) via the Download Dialog Box or Upload Dialog Box, respectively.

This section explains the following items:

- Coverage mesurement result display
- Coverage measurement range
- Display of locations for which coverage measurement is executed
- RRM function, trace function, and coverage function used on a mutually exclusive basis

**Cautions1:** This function is disabled when no coverage boards are incorporated.

**Cautions2:** RRM Function, Trace Function (When IECUBE Is Connected) and Coverage Measurement Function (When IECUBE Is Connected) are functions that are used on a mutually exclusive basis (refer to "5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis").
To switch between the three functions, go to the [Option] menu.

**Remark:** Coverage data is cleared when the debugger is started.

### 5.11.1 Coverage mesurement result display

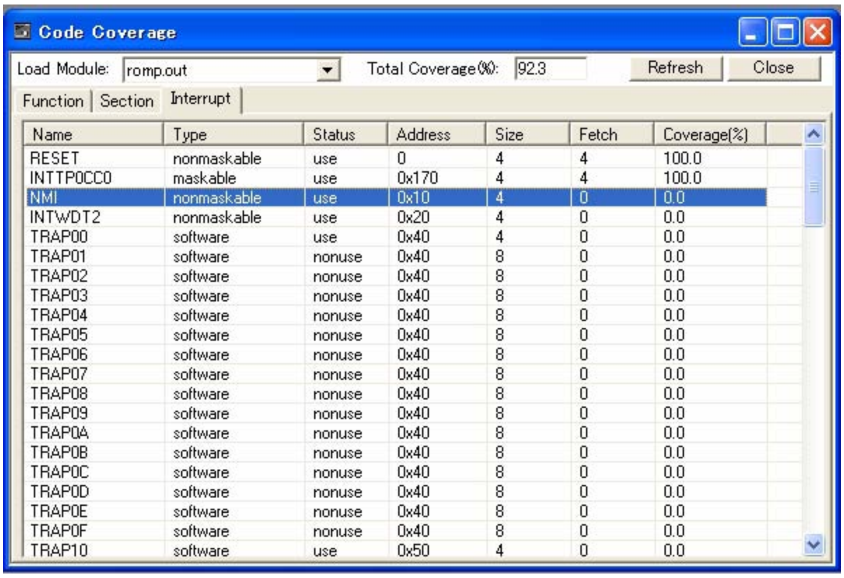
The coverage measurement result can be checked in the Code Coverage Window.

In the Code Coverage Window, the measurement result is displayed individually for functions, sections, and interrupt handlers (vectors). The coverage measurement result is updated at a break (it is not updated automatically during user program execution).

Clear the coverage data by selecting [Option] menu ->[Coverage] ->[Clear].

Coverage data can be saved in the CSV format (refer to "5.15.2 Window display information (view file)").

Figure 5-21 Coverage Mesurement Result Display




### 5.11.2 Coverage measurement range

The coverage measurement range is as follows.

Table 5-14 Code Coverage Measurement Range

| Connected IE | Code Coverage Measurement Range |
| --- | --- |
| IECUBE | Internal ROM space + any 1 MB space (selectable by Coverage-Address Dialog Box) |

### 5.11.3 Display of locations for which coverage measurement is executed

The locations for which coverage measurement is executed in the user program are displayed in the Source Window and Assemble Window based on the coverage measurement information.

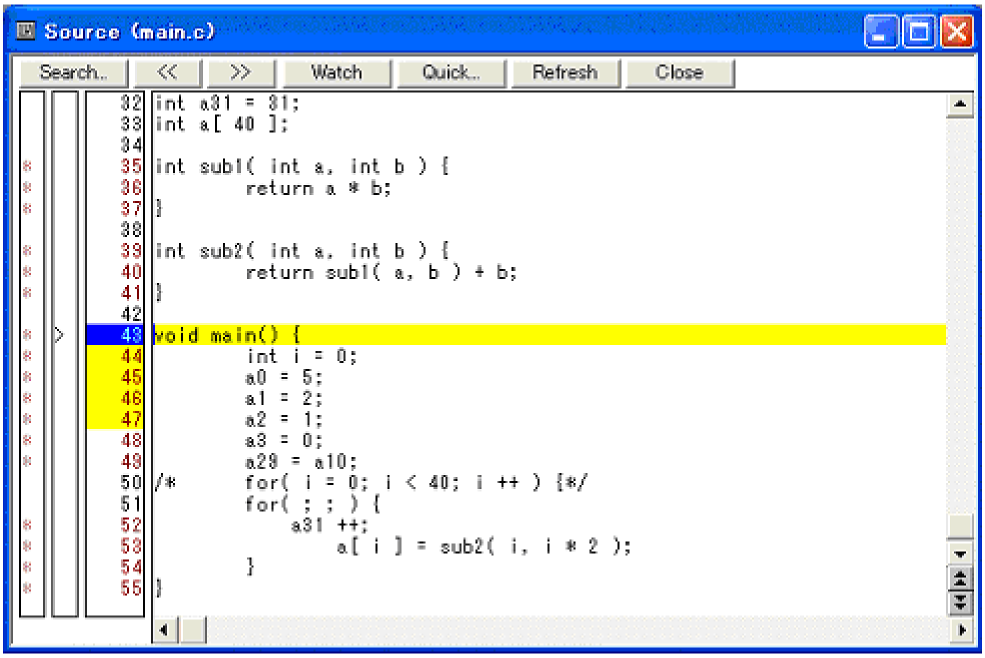
The display result can be saved as view files for the Source Window and Assemble Window (refer to "5.15.2 Window display information (view file)").

The numbers of line or addresses for which coverage measurement is executed are highlighted as shown in the table below. In the view file, the marks in the table below are appended to the relevant line number or address instead of using the background color.

Table 5-15 Format of View of Locations for Which Coverage Measurement Executed

| Coverage | Source Window | | Assemble Window | |
|---|---|---|---|---|
| | Background Color | Mark | Background Color | Mark |
| Code on this line has been executed by 100% | Yellow | '@' | Yellow | '@' |
| Code on this line has been executed by 1 to 99% | Orange | '+' | | |
| Code on this line has been executed by 0% (not yet executed) | No highlight color | - | No highlight color | - |

Figure 5-22 View of Locations for Which Coverage Measurement Executed



### 5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis

RRM Function, Trace Function (When IECUBE Is Connected) and Coverage Measurement Function (When IECUBE Is Connected) are functions that are used on a mutually exclusive basis.

Accordingly, the trace and coverage functions cannot be used when the RRM function is selected; the RRM and coverage functions cannot be used when the trace function is selected; and the RRM and trace functions cannot be used when the coverage function is selected.

To switch between the three functions, go to the [Opiton] menu, and select one of the [RRM Function/Trace Function/Coverage Function].

Basically, the three functions are used on a mutually exclusive basis; however the operations below are possible.

- The switching of the functions does not clear coverage data.
- Even when a function other than the coverage function is selected, coverage data can be downloaded, uploaded, and cleared.
- When the coverage function is selected, it is possible to perform traces, with the trace mode fixed at "All PC." (Conditional traces cannot be performed.)
- When the RRM function is selected, the trace data displayed in the Trace View Window indicates only branch information and access information in the sampling range of Real-time RAM monitor function. (Traces are performed, with the trace mode fixed at "Branch PC.")

## 5.12 Event Function

Events specify specific states of the target system during debugging ,such as "fetched address 0x1000" or "Wrote data to address 0x2000".

In ID850QB, such events are used as action triggers for each debugging function, such as break and trace.

This section explains the following items:


- Using event function
- Creating events
- Setting event conditions
- Number of enabled events for each event condition
- Managing events

### 5.12.1 Using event function

Events (event conditions and event rink conditions) consist of the event conditions listed in the following table, by assigning various debugging functions. As a result, event conditions can be utilized according to the debugging purpose.

Table 5-16 Various Event Conditions

| Event Condition | Mark | Contents<br>->Setting Dialog Box |
|---|---|---|
| Break event | B | Condition in which the execution of the user program or operation of a tracer is stopped. (refer to "5.4 Break Function").<br>->Break Dialog Box |
| Trace event | T | Condition in which the process of user program execution is saved to the trace memory (refer to "5.10 Trace Function (When IECUBE Is Connected)").<br>->Trace Dialog Box |
| Timer event | Ti | Condition for specifying the time measurement start timing and stop timing (refer to "5.9 Timer Function (When IECUBE Is Connected)").<br>->Timer Dialog Box |

### 5.12.2 Creating events

Events can be used as action triggers of various event conditions described before through registration of event conditions and event link conditions, individually naming states called events.

**(1) Creating and registering events**

The creation of event conditions is done in the Event Dialog Box.

Set an address condition, status condition, and data condition in this dialog box. Specify a combination of these as one event condition and name and register this event condition.

A simple method consists in using event conditions generated by setting breakpoint in the Source Window and Assemble Window. (Refer to "5.4.2 Breakpoint setting".)

**(2) Creating and registering event links**

Event link conditions are conditions for single events that provide ordered restrictions for event conditions, and are generated when user programs are executed according to the specified sequence.

To create an event link condition, use the Event Link Dialog Box.

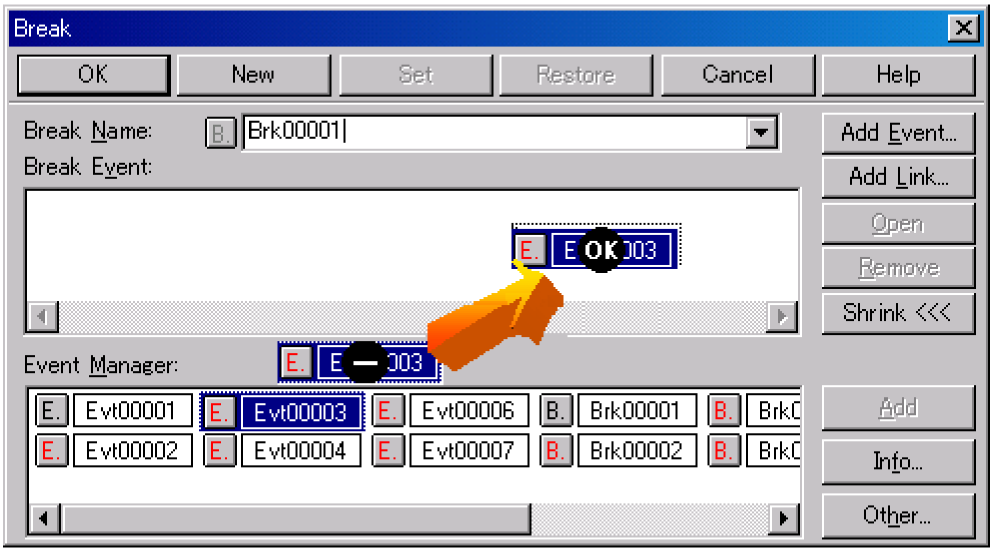
## 5.12.3 Setting event conditions

Various event conditions listed in Table 5-16 are individually created in the corresponding dialog box.

### (1) Setting of Various Event Conditions

The setting of the various event conditions is done by selecting the event icon of the desired event conditionor event link condition displayed in the event manager area (or Event Manager) in the respective setting dialog box, and dragging and dropping this icon in the condition area to be set.

Figure 5-23 Setting of Various Event Conditions




The shape of the mouse cursor changes to "OK" when it is dragged over a settable condition area.

Regarding the created event conditions, the event icon mark becomes red and the setting is enabled by clicking the <Set> button or <OK> button in the Setting dialog box. After the event has been set, a debugging action occurs as various event conditions.

### (2) Settings using selection mode (settings after checking contents)

The Event Dialog Box or Event Link Dialog Box are open in the selection mode by placing the focus on the condition area to be set and then clicking the <Add Event...> button or <Add Link...> button. Because when a condition set in the dialog box is selected, the corresponding detailed condition is displayed, conditions can be set after checking the contents.

### (3) Copying and moving event icons

In the event condition setting area, event conditions can be copied and moved through drag & drop operation using the following methods.

- If event condition was dropped using only the mouse, move event condition.
- If the event condition was dropped while pressing the Ctrl key, copy the event condition.

**(4) Manipulation in event manager area**

Event conditions can be set by clicking the <Add> button after placing the focus on the condition area to be set and selecting an event icon.

**Event setting content display**

Select an event and click the <Open> button or double-click the event. The setting dialog box corresponding to the selected event will be opened and the set contents of the event will be displayed.

**Deletion**

An event can be deleted by selecting the event and then clicking the <Remove / Delete> button or pressing the Delete key.

**Changing display mode and sorting**

The display mode of and sorting in the event manager area can be selected by clicking the <Info...> button.

**Area non-display**

An area can be hidden by clicking the <Shrink<<< > button.

### 5.12.4 Number of enabled events for each event condition

Up to 256 conditions can be registered as event conditions or various event conditions.

One event condition or link event condition can be set for multiple types of events such as break and trace.

However, the number of event conditions that can be simultaneously set (enabled) is limited as follows.

Therefore, if the valid number is exceeded or if the used event conditionsor event link conditions exceed the maximum number that can be used simultaneously, it is necessary to disable the set various event conditions once and then register them again. (Refer to "5.12.5 Managing events".)

Table 5-17 Number of Enabled Events For Each Event Condition

| Connected IE | | Event | | Event Link | Break | Trace | Timer |
|---|---|---|---|---|---|---|---|
| | | Execute | Access | | | | |
| IECUBE | | 10[*a] | 6[*b] | 1 | 10+6 | 1 | 7[*g] |
| N-Wire CARD, MINICUBE | Nx85ET（RCU0+TEU+TRCU） | 10[*a] | 4[*c] | 1[*d] | 10+4 | - | - |
| | Nx85E901（RCU0）, RCU1 | 2[*e] | | 1[*f] | 2[*e] | - | - |

***a** 2 before executions (usable only for breaks, address range not specifiable), 8 post-execution events (4 when address range is specified, because 2 events are used for range specification)

***b** 3 when address range is specified, because 2 events are used for range specification

***c** 2 when address range is specified, because 2 events are used for range specification

***d** Use from Phase 1 to Phase 4.

***e** Address range specification is not possible.

***f** Can be set only for Phase 1 and Phase 2.

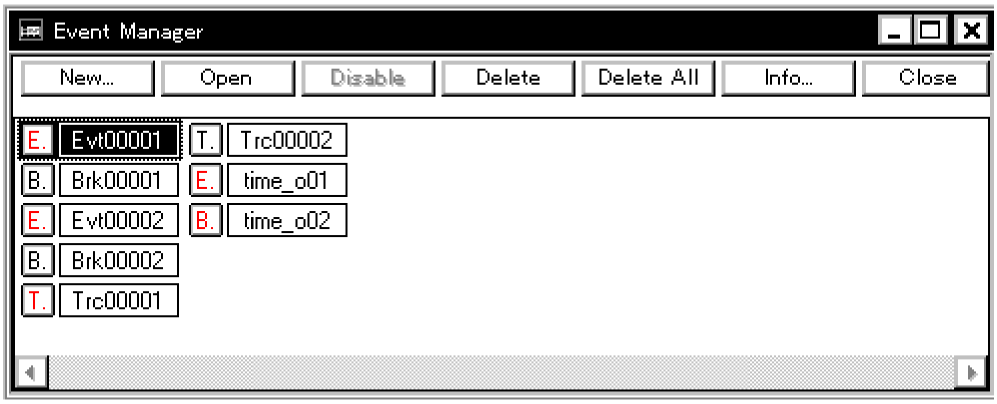
***g** Can not be set the event condition that range address is specified for Start Event or End Event area. Can be set the event link condition that range address is specified.

### 5.12.5 Managing events

Managing events is done with the Event Manager.

The Event Manager allows display, enabling/disabling, and deletion of the Various Event Conditions.

Figure 5-24 Managing Events (the Event Manager)




**(1) Event icon**

Event icons consist of a mark and an event name indicating the type of event. The color of each event icon indicates the setting status of that event.

Enable/disable is switched by clicking the mark part.

Table 5-18 Event Icon

| Character Color | Mark | Meaning |
|---|---|---|
| Red | E.L. | Indicates that the event condition or event link condition which is used for various event conditions is enabled. |
| | B.T.Ti. | Indicates that the Various Event Conditions is valid.<br>The various events occur when its condition is satisfied. |
| Black | E.L. | Indicates that the event condition or event link condition which is used for various event conditions is disabled. |
| | B.T.Ti. | Indicates that the Various Event Conditions is invalid.<br>The various events do not occur even when its condition is satisfied. |
| Yellow | E.L. | Indicates that the symbol specified for an event is held pending because it cannot be recognized by the program currently loaded. |
| | B.T.Ti. | Indicates that the Various Event Conditions is held pending.<br>The various events do not occur even when its condition is satisfied. |

## 5.13 RRM Function

This section explains the following items related to the RRM function.

- Real-time RAM monitor function
- Pseudo real-time RAM monitor function (Break When Readout)

**Caution:** RRM Function, Trace Function (When IECUBE Is Connected) and Coverage Measurement Function (When IECUBE Is Connected) are functions that are used on a mutually exclusive basis (refer to "5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis").
To switch between the three functions, go to the [Option] menu.

### 5.13.1 Real-time RAM monitor function

Table 5-19 shows the range of data that can be loaded using the real-time RAM monitor function.

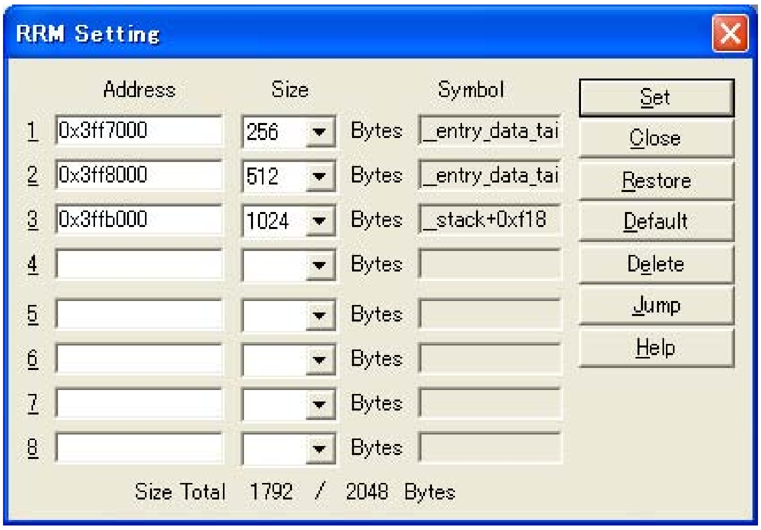
The variables and data allocated to this area can be displayed in real time in the Watch Window and Memory Window.

The sampling interval can be specified in the Extended Option Dialog Box.

Table 5-19 Real-time RAM Monitor Function

| Connected IE | Sampling Range |
|---|---|
| IECUBE | Areas specified in the RRM Setting Dialog Box |
| N-Wire CARD, MINICUBE | None |

Figure 5-25 RRM Setting Dialog Box



### 5.13.2 Pseudo real-time RAM monitor function (Break When Readout)

This is a function that performs pseudo real-time RAM monitoring via software emulation. When this function is used for reading data, a break occurs instantaneously upon a read during the user program execution.

The variables and data allocated to this area can be displayed in close to real time in the Watch Window and Memory Window**Note**.

Specify turning on/off of the pseudo real-time RAM monitor function and the sampling range in the Extended Option Dialog Box.

**Note:** When IECUBE is connected, this function is available only for the data and variables displayed in the Watch Window.

Figure 5-26 Specification of Pseudo Real-Time RAM Monitor Function

## 5.14 DMM Function

DMM (Dynamic Memory Modification) is a function that rewrites the contents of the memory (RAM) in real time during user program execution.

The DMM Dialog Box is opened by clicking the <DMM... > button on the Memory Window, Register Window, or IOR Window. Specify the DMM target address and data.

**Note:** When IECUBE is connected, a pseudo DMM function is performed. With pseudo DMM, a break occurs instantaneously upon a write during the user program execution.

Figure 5-27 DMM Dialog Box

## 5.15 Load/Save Function

ID850QB allows saving and loading the following types of information as files. As a result, recovery of these various types of information is possible.

- Debugging environment (project file)
- Window display information (view file)
- Window setting information (setting file)

**Remark:** The simple window status can be maintained by selecting [Window] menu -> [Static]. (Refer to "5.16.1 Active status and static status".)

### 5.15.1 Debugging environment (project file)

A project file (*.prj) is a file that records the debugging environment.

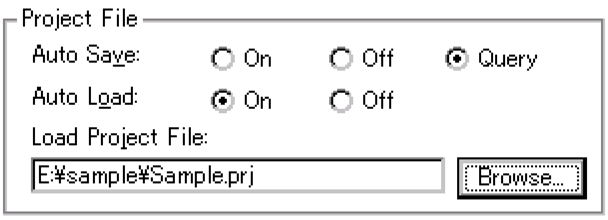
A project file is created when the debugging environment at a particular point in time is saved, and that debugging environment can be restored by loading this file at a subsequent time.

Project files are loaded and saved in the Project File Save Dialog Box and Project File Load Dialog Box, respectively. To load a project file at startup, press the <Project...> button in the Configuration Dialog Box.

#### (1) Automatic save/automatic load of project file

The project file to be loaded/saved when starting up and exiting in the Debugger Option Dialog Box can also be set in advance.

Figure 5-28 Automatic Save/Automatic Load Setting for Project File



**(2) Contents saved to project File**

The following contents are saved to the project file:

Table 5-20 Contents Saved to Project File

| Window Name | Saved Contents |
|---|---|
| Configuration Dialog Box | All items (target device, clock setting, pin mask setting, mapping information) |
| Main window | Display position, tool bar/status bar/button display information, execution mode information, trace on/off information |
| Download Dialog Box | File information to be downloaded |
| Extended Option Dialog Box<br>Debugger Option Dialog Box<br>Fail-safe Break Dialog Box<br>RRM Setting Dialog Box | Set information |
| Source Window | Display information of window |
| Assemble Window<br>Memory Window | Display information of window, display start address |
| Stack Window<br>IOR Window<br>Local Variable Window<br>Trace View Window<br>Code Coverage Window<br>Event Manager<br>Console Window<br>Expansion Window | Display information of window |
| Event Dialog Box | Display information of window, event information |
| Event Link Dialog Box | Display information of window, link event information |
| Break Dialog Box | Display information of window, break event information |
| Trace Dialog Box | Display information of window, trace event information |
| Timer Dialog Box | Display information of window, timer event information |
| Register Window | Display information of window |
| Watch Window | Display information of window[Note], watch registration information |
| Add I/O Port Dialog Box | Added I/O port information |
| DMM Dialog Box | DMM information |
| Delay Count Dialog Box | Delay count value |
| Software Break Manager | Display information of window, software break information |

**Note:** The display status of members of a structure pointer, array pointer and so on, and radix for displaying individual member are not saved.

### 5.15.2 Window display information (view file)

A view file is a file that records window display information.

View files can be loaded and saved for each window.

When a view file is loaded, a reference window (Source Window in the static status) is displayed and the display information at the time of saving is displayed.

View files are loaded and saved in the View File Load Dialog Box and View File Save Dialog Box, respectively.

Table 5-21 Type of the View Files

| File Type | Current Window Name, File Name |
|---|---|
| Source Text (*.svw) | Source Window **Note1** |
| Assemble (*.dis) | Assemble Window **Note1** |
| Memory (*.mem) | Memory Window |
| Watch (*.wch) | Watch Window |
| Register (*.rgw) | Register Window |
| I/O Register (*.ior) | IOR Window |
| Local Variable (*.loc) | Local Variable Window |
| Stack Trace (*.stk) | Stack Window |
| Trace (*.tvw)* | Trace View Window |
| Code Coverage (*.cov) | Code Coverage Window |
| Console (*.log) | Console Window |
| All (*.*) | All files |
| Source (*.c, *.s) | Source file**Note2** |
| Text (*.txt) | Text file |

**Note1:** When IECUBE is connected, the mark for indicating the code coverage measurement result (executed/not yet executed) is added to the contents of the displayed file (refer to "Table 5-15 Format of View of Locations for Which Coverage Measurement Executed").

**Note2:** The extension of the source file can be changed in the Extended Option Dialog Box.

### 5.15.3 Window setting information (setting file)

A setting file is a file that records the window setting information (watch data settings, peripheral I/O registers settings, and event settings).

Setting files can be loaded and saved for each window.

When a setting file is loaded, the target window is displayed and the setting information that was saved is restored.

Setting files are loaded and saved in the Environment Setting File Load Dialog Box and Environment Setting File Save Dialog Box, respectively.

Table 5-22 Type of the Setting Files

| File Type | Current Window Name |
|---|---|
| Watch (*.wch) | Watch Window**Note** |
| I/O Register (*.ior) | IOR Window**Note** |
| Event (*.evn) | Event Manager |

**Note:** A variable value can not be loaded.

## 5.16 Functions Common to Each Window

The windows have the following common functions.

- Active status and static status
- Jump function
- Trace Result with Linking Window (when IECUBE is connected)
- Drag & drop function
- Cautions

### 5.16.1 Active status and static status

Each of the Windows below has two statuses: The Active status and Static status.

- Source Window (that is displaying the source file to which symbol information is read)
- Assemble Window
- Memory Window

Only one window can be opened in the active status. However, because two or more windows in the static status can be opened, the current status of the windows can be temporarily held.

Select this status by the [Window] menu.

**(1) Active status**

The display position and contents of the window in the active status are automatically updated in association with the current PC value.

This window is also the jump destination of the Jump function. If this window is linked with the Trace View Window, the contents displayed in the active window are updated in association with the Trace View Window.

Only one window can be opened in the active status.

**(2) Static status**

The display position of the window in the static status does not move in association with the current PC value, but the displayed contents are updated.

The static window is not used as the jump destination of the Jump function. In addition, it is not linked with the Trace View Window.

If an active window is already open, the next window is opened in the static status.

Two or more static windows can be opened at the same time.

### 5.16.2 Jump function

This is a function that jumps to any of the Windows below from a line or address (a jump pointer) on which the cursor is put. The Window to which the jump is made is displayed on the jump pointer.

- Source Window
- Assemble Window
- Memory Window

You can jump among the above windows, or from the Trace View Window, Stack Window, Event Manager and Register Window to the above windows.

#### (1) Jump method

The jump method is as follows:

1) Move the cursor to the line or address that is to be used as the jump pointer, on the window from which jumping is possible (select an event icon on the Event Manager).
2) Select the following menu item to which execution is to jump from the [Jump] menu.

**Caution:** If a program code does not exist on the line at the cursor position, the first address of the line with a program code above or below that line is used as the jump pointer.

**(2) Details of jump source address**

The details of jump source address is as follows:

Table 5-23 Details of Jump Source Address

| Target Window | Details of Jump Pointer | |
|---|---|---|
| From the Register Window | Registers selected | |
| From the Memory Window | Address at the cursor position | |
| From the Event Manager | If the selected event icon is that of an event condition, an address condition is used as the jump pointer. | |
| | If the address condition is set in point | Jump to specified address |
| | If the address condition is set in range | Jump to lower address (point address before the mask if a mask is specified) |
| | If the address condition is set in bit | Jump to address at the bit position |
| From the Stack Window | A function at the cursor position that stack flame number indicates is used as the jump pointer. | |
| | **With current function** | |
| | If the jump destination is the Source Window | Jumps to the current PC line |
| | Other than above | Jumps to the current PC address |
| | **With function other than current function** | |
| | If the jump destination is the Source Window | Jump to the line that calls a nested function. |
| | Other than above | Jump to the address next to the instruction that calls a nested function. |
| From the Trace View Window | Jump to the Memory Window | |
| | If the cursor position is at an access address, access data, or access status | Access address |
| | Other than above | Fetch address |
| Jump to the Source Window or Assemble Window | Fetch address | |

### 5.16.3 Trace Result with Linking Window (when IECUBE is connected)

By linking the Trace View Window with each window (Source Window, Assemble Window or Memory Window), the corresponding part can be displayed on the linked window, by using the address at the cursor position on the Trace View Window as a pointer.

If the cursor is moved on the Trace View Window, the corresponding part on the linked window is highlighted or indicated by the cursor position.

**(1) Linking method**

The linking method is as follows:

1) Set the Trace View Window as the current window.
2) Select [View] menu -> [Window Synchronize] to select a window to be linked.
3) Move the cursor to the line to be linked in the trace result display area of the Trace View Window.
4) Using the address of the line selected in 3) as a pointer, the corresponding part is highlighted (or indicated by the cursor position) in the display area of the window selected in 2).

**Remark:** The linking source address differs as follows depending on the cursor position in the trace result display area if the Memory Window is linked.
Access address, access data, access status -> Access address
Others -> Fetch address
When the Source Window or Assemble Window is linked, the fetch address is always used as the pointer.

### 5.16.4 Drag & drop function

Selected and highlighted line numbers, addresses, and text can be dragged and dropped in another window using the following method.

1) Drag the selected line number, address, or text.
   -> The shape of the mouse cursor changes from an arrow to "-".
2) Drop the selection in a window or area where it can be dropped.
   -> The shape of the cursor changes from "-" to "OK" when the cursor is placed over a window or area where the selection can be dropped.

In the window in which the line number of the address has been dropped, an operation is performed on the dropped address or the address that is obtained from the dropped line number. For example, a variable can be simply registered by dragging and dropping in the Watch Window such a variable located in the Source Window.

#### (1) Drag & drop details

The operation to be performed after dropping the line number or address differs, depending on the window or area in which the line number or address has been dropped.

Table 5-24 Details of Drag & Drop Function (Line/Address)

| Window/Area to Drop to | Operation After Drop |
|---|---|
| The Event Manager or the event manager area in each various event setting dialog box | Automatically creates an execution event condition by using the dropped line number or address as an address condition.<br>Event condition names are automatically created as Evt00001, Evt00002, and so on. A path count is not specified.<br>The address condition is set for the closest symbol in the format of symbol name + offset value. |
| Condition setting area in each various event setting dialog box (other than address and data setting areas) | Automatically creates an execution event condition by using the dropped line number or address as an address condition.<br>The automatically created event condition is set in each condition setting area in which the line number or address has been dropped.<br>Event condition names are automatically created as Evt00001, Evt00002, and so on. A path count is not specified.<br>The address condition is set for the closest symbol in the format of symbol name + offset value. |
| Condition setting area in each various event setting dialog box (address and data setting areas) | The text of the dropped line number or address is set in the area in which the line number or address has been dropped.<br>The address condition is set for the closest symbol in the format of symbol name + offset value. |

Table 5-25 Details of Drag & Drop Function (Character String)

<table>
<tr><th>Window/Area to Drop to</th><th colspan="2">Operation After Drop</th></tr>
<tr><td rowspan="7">The Event Manager or the event manager area in each various event setting dialog box</td><td colspan="2">If the dropped text can be converted as a symbol into an address value, an event condition in the Access status (all access statuses) or Execute status is automatically created, using the converted address value as an address condition.<br>Event condition names are automatically created as Evt00001, Evt00002, and so on. A data condition and path count are not specified.<br>The address condition is set by the dropped text.<br>The relationship between the event condition to be created and the symbol is as follows:</td></tr>
<tr><td>Symbols</td><td>Status</td></tr>
<tr><td>Variable</td><td>Access (R/W)</td></tr>
<tr><td>Function</td><td>Execute</td></tr>
<tr><td>Symbol in data section</td><td>Access (R/W)</td></tr>
<tr><td>Symbol in code section</td><td>Execute</td></tr>
<tr><td>Others</td><td>Access (R/W)</td></tr>
<tr><td rowspan="7">Condition setting area in each various event setting dialog box (other than address and data setting areas)</td><td colspan="2">If the dropped text can be converted as a symbol into an address value, an event condition in the Access status (all access statuses) or Execute status is automatically created, using the converted address value as an address condition.<br>The automatically created event condition is set in each condition setting area in which the line number or address has been dropped.<br>Event condition names are automatically created as Evt00001, Evt00002, and so on. A data condition and path count are not specified.<br>The address condition is set by the dropped text.<br>The relationship between the event condition to be created and the symbol is as follows:</td></tr>
<tr><td>Symbols</td><td>Status</td></tr>
<tr><td>Variable</td><td>Access (R/W)</td></tr>
<tr><td>Function</td><td>Execute</td></tr>
<tr><td>Symbol in data section</td><td>Access (R/W)</td></tr>
<tr><td>Symbol in code section</td><td>Execute</td></tr>
<tr><td>Others</td><td>Access (R/W)</td></tr>
<tr><td>Condition setting area in each various event setting dialog box (address and data setting areas)</td><td colspan="2">The dropped text is set in the area.</td></tr>
<tr><td>Watch Window</td><td colspan="2">If the dropped text is recognizable as a symbol, the contents of the symbol are displayed.</td></tr>
</table>

**Remark:** Each various event setting dialog box are as follows.

- Event Dialog Box
- Event Link Dialog Box
- Break Dialog Box
- Trace Dialog Box
- Timer Dialog Box

### 5.16.5 Cautions

(1) The number of characters that can be displayed on 1 line in each area of a window is 319.

(2) If the width of the display area is narrow, the display may become corrupted. In this case, increase the width of the window.

(3) Redrawing may not successfully be performed in a window with a <Refresh> button when the cursor position is changed while the window is active. Click the <Refresh> button to perform redrawing.

(4) The help that is opened using the F1 key is the help corresponding to the window on which the cursor is placed. Consequently, because the cursor cannot be placed on the Trace View Window in which no trace results are displayed, such as immediately after startup, the help may not open even if the F1 key is pressed. In this case, open the help by selecting [Current Window Help] from the [Help] menu.

(5) Do not select [Slowmotion] from the [Run] menu during Go & Go execution. [Slowmotion] on the [Run] menu is usually dimmed during Go & Go execution, but there is a moment when it can be selected, so if [Slowmotion] is selected at this time, the program will not be able to be stopped even if [Stop] is selected from the [Run] menu (or the STOP button is clicked).

(6) If for some reason or other the application switches while event icons are in the process of being dragged, the icons will no longer be able to be dropped.
Use the ESC key to escape from drag, then reattempt the drag.

# CHAPTER 6  WINDOW REFERENCE

This chapter explains in detail the functions of the windows and dialog boxes of ID850QB.

- Window List
- Explanation of Windows

## 6.1 Window List

The list is the windows of the ID850QB.

Table 6-1 Window List

| Window Name | Contents |
|---|---|
| Main window | This window is displayed first, when the ID850QB is started. It controls execution of the user program. Various windows are opened from this window. |
| Configuration Dialog Box | Displays and sets the ID850QB operation environment. |
| Extended Option Dialog Box | Displays and sets the extended options of the ID850QB. |
| Fail-safe Break Dialog Box | Sets the fail-safe breaks (IECUBE) |
| RRM Setting Dialog Box | Sets the RRM sampling range (IECUBE) |
| Flash Option Dialog Box | Makes the flash self programming emulation settings. (IECUBE) |
| Debugger Option Dialog Box | Displays and sets other options. |
| Project File Save Dialog Box | Saves the current debug environment to project file |
| Project File Load Dialog Box | Loads the debug environment. |
| Download Dialog Box | Loads an object file and binary file. |
| Upload Dialog Box | Saves the memory contents to a file. |
| Load Module List Dialog Box | Lists the names of the downloaded load module files. |
| Source Window | Displays a source file and text file |
| Source Search Dialog Box | Searches in the Source Window |
| Source Text Move Dialog Box | Specifies a file to be displayed in the Source Window and the position from which displaying the file is to be started. |
| Assemble Window | Disassembles the program and executes online assembly. |
| Assemble Search Dialog Box | Searches in the Assemble Window |
| Address Move Dialog Box | Specifies the start address to display the contents of the Memory Window or Assemble Window. |
| Symbol To Address Dialog Box | Displays the address of the specified variable or function, or the value of the specified symbol |
| Watch Window | Displays and changes specified watch data |
| Quick Watch Dialog Box | Displays temporarily specified watch data |
| Add Watch Dialog Box | Registers watch data todisplay in the Watch Window |
| Change Watch Dialog Box | Changes watch data todisplay in the Watch Window |
| Local Variable Window | Displays and changes the local variable in the current function. |
| Stack Window | Displays the current stack contents |
| Memory Window | Display the contents of memory. |
| Memory Search Dialog Box | Searches in the Memory Window |

| Window Name | Contents |
|---|---|
| Memory Fill Dialog Box | Fills the memory contents with specified data. |
| Memory Copy Dialog Box | Copies the memory. |
| Memory Compare Dialog Box | Compares the memory. |
| Memory Compare Result Dialog Box | Displays the results of comparing the memory. |
| DMM Dialog Box | Sets addresses and data subject to DMM. |
| Register Window | Displays the contents of registers. |
| Register Select Dialog Box | Selects registers to be displayed in the Register Window. |
| IOR Window | Displays the contents of peripheral I/O registers |
| IOR Select Dialog Box | Selects peripheral I/O registers and I/O ports to be displayed in the IOR Window |
| Add I/O Port Dialog Box | Registers an I/O port to be displayed in the IOR Window. |
| Timer Dialog Box | Registers and sets timer event conditions, and displays execution time measurement result. (IECUBE) |
| Timer Result Dialog Box | Displays execution time measurement results. (IECUBE) |
| Trace View Window | Displays trace results. (IECUBE) |
| Trace Search Dialog Box | Searches trace data. (IECUBE) |
| Trace Data Select Dialog Box | Selects items to be displayed in the Trace View Window. (IECUBE) |
| Trace Move Dialog Box | Specifies the start address to display the contents of the Trace View Window. (IECUBE) |
| Trace Dialog Box | Registers and sets trace event conditions. (IECUBE) |
| Delay Count Dialog Box | Sets the delay count of a delay trigger trace event.(IECUBE) |
| Code Coverage Window | Display of code coverage results (IECUBE) |
| Coverage-Address Dialog Box | Selects the code coverage measurement range.(IECUBE) |
| Event Manager | Displays, enables/disables, and deletes each event condition. |
| Software Break Manager | Display, enable or disable, and delete software breaks. |
| Event Dialog Box | Registers event conditions. |
| Event Link Dialog Box | Registers event link conditions. |
| Break Dialog Box | Registers and sets break event conditions. |
| View File Save Dialog Box | Saves the display information of the current window to a view file. |
| View File Load Dialog Box | Loads the view file of each window. |
| Environment Setting File Save Dialog Box | Saves the setting information of the current window to a setting file. |
| Environment Setting File Load Dialog Box | Loads the setting file of each window. |
| Reset Debugger Dialog Box | Initializes the ID850QB,CPU, and symbol information. |
| Exit Debugger Dialog Box | Terminate the ID850QB. |

| Window Name | Contents |
|---|---|
| About Dialog Box | Displays the version of the ID850QB. |
| Console Window | Inputs commands. |
| Font Dialog Box | Displays the types of fonts displayed. |
| Browse Dialog Box | Selects the file to be set |

## 6.2 Explanation of Windows

This section explains each window or dialog box as follows:

# Window Name / Dialog box Name

Briefly explains the function of the window or dialog box and points to be noted.

If an invalid window/dialog box exists due to a connected IE, the name of the valid connected IE is indicated at the lower right of the window/dialog box name.

In addition, the display image of the window or dialog box is also illustrated.

Items of related operation are also explained.

### Opening

Explains how to open the window or dialog box.

### Explanation of each area

Explains items to be set to or displayed in each area of the window or dialog box.

### Context menu

Explains the context menu that is displayed in the window when the right mouse button is clicked. From the context menu, convenient functions often used in this window can be selected with a single action (window only).

### Function buttons

Explains the operation of each button in the window or dialog box.

### Related operations

Explains the operation of a window or dialog box related to this window or dialog box.

# Main window

This window is automatically opened when the ID850QB is started up and initialized.

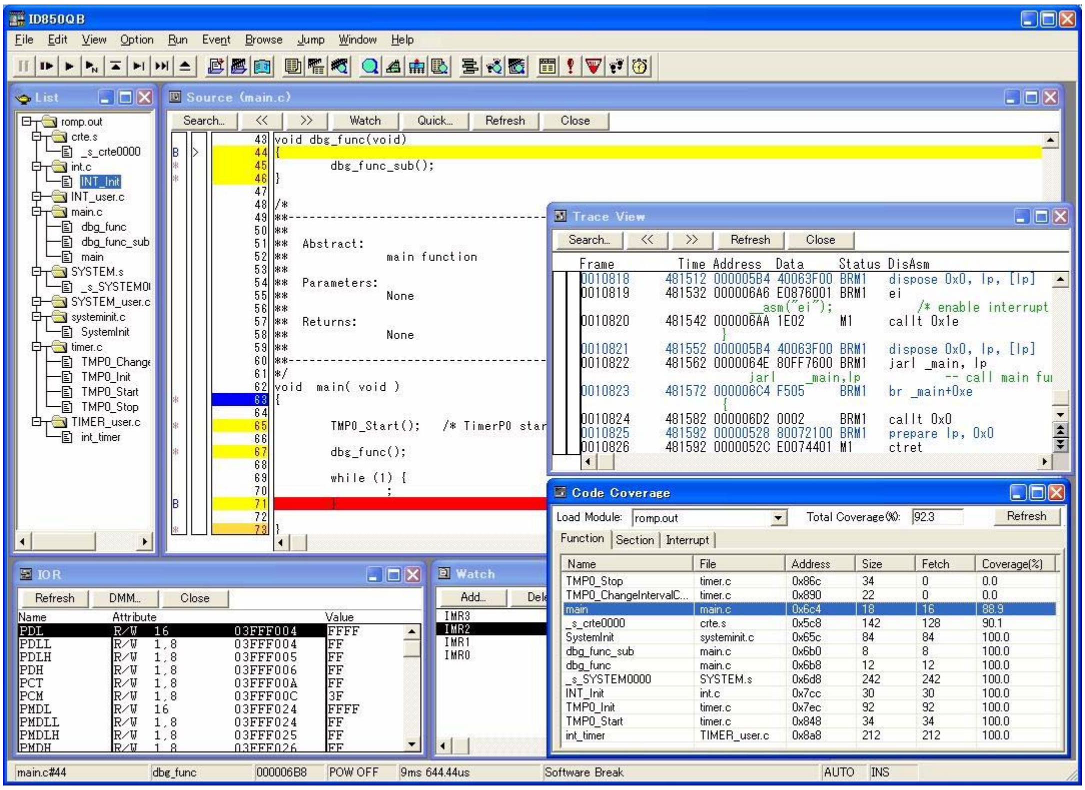
In ID850QB, other windows are manipulated from this window (refer to "Table 6-1 Window List").

Execution of the user program is controlled in this window.

Execution of the user program is controlled in the following three modes:

- Source mode (Debugs the user program at the source level.)
- Instruction mode (Debugs the user program at the instruction level.)
- Auto mode (Automatically selects the source mode or instruction mode.) (default)

Figure 6-1 Main Window



- Menu bar
- Toolbar
- Window display area
- Status bar

## Menu bar



### (1) [File] menu

| | | |
|---|---|---|
| Open... | | Loads a view file, source file, or text file.<br>Opens the View File Load Dialog Box.<br>The operation differs depending on the extension of the file selected in the dialog box. |
| Save As... | | Saves the contents displayed on the current window to the file whose name is specified.<br>Opens the View File Save Dialog Box. |
| Close | | Closes the current window. |
| Download... | | Downloads a file.<br>Opens the Download Dialog Box. |
| Load Module... | | Lists the names of the files that have been downloaded.<br>Opens the Load Module List Dialog Box. |
| Upload... | | Uploads a program.<br>Opens the Upload Dialog Box. |
| Project | | Manipulates a project file. |
| | Open... | Opens a project file.<br>Opens the Project File Load Dialog Box. |
| | Save | Overwrites the current status to the project file currently being read to the ID850QB. |
| | Save As... | Saves the current status to a specified project file.<br>Opens the Project File Save Dialog Box. |
| Environment | | Manipulates a setting file. |
| | Open... | Opens a setting file.<br>Opens the Environment Setting File Load Dialog Box. |
| | Save As... | Saves the setting in the current window to the setting file.<br>Opens the Environment Setting File Save Dialog Box. |
| Debugger Reset... | | Initializes the CPU, symbols, and ID850QB.<br>Opens the Reset Debugger Dialog Box. |
| Exit | | Terminate the ID850QB.<br>Opens the Exit Debugger Dialog Box. |

| | |
|---|---|
| (Open file) | Lists the names of the files opened. |

**(2) [Edit] menu**

| | | |
|---|---|---|
| Cut | | Cuts a selected character string and saves it to the clipboard buffer. |
| Copy | | Copies a selected character string and saves it to the clipboard buffer. |
| Paste | | Pastes the contents of the clipboard buffer to the text cursor position. |
| Write in | | Writes the modified contents to the target. |
| Restore | | Cancels the modification. |
| Memory | | Manipulates the memory contents. |
| | Fill... | Fills the memory contents with specified codes.<br>Opens the Memory Fill Dialog Box. |
| | Copy... | Copies the memory.<br>Opens the Memory Copy Dialog Box. |
| | Compare... | Compares the memory.<br>Opens the Memory Compare Dialog Box. |
| DMM... | | Rewrites the memory contents to real time during user program execution.<br>Opens the DMM Dialog Box. |
| Edit Source | | Opens the source file displayed in the active Source Window with the editor specified by the PM+ when the PM+ runs. |

**(3) [View] menu**

The [View] menu contains common parts as well as dedicated parts added according to the active window. For details about the dedicated parts, refer to the description of each window.

(a) Common items

| | |
|---|---|
| Search... | Performs a search.<br>Opens the search dialog box corresponding to the current window.<br>Same operation as the <Search> button. |
| Move... | Moves the display position.<br>Opens the specification dialog box corresponding to the current window. |
| Quick Watch... | Temporarily displays the contents of the specified data.<br>Opens the Quick Watch Dialog Box. |
| Add Watch... | Registers the specified data to the Watch window.<br>Opens the Add Watch Dialog Box. |
| View Watch | Adds the selected data to the Watch window.<br>If the data is a symbol, it is added in accordance with the setting of Debugger Option Dialog Box. |
| Change Watch... | Changes the data on the line selected by the Watch window.<br>Opens the Change Watch Dialog Box.<br>This menu is valid only when a variable is selected in the Watch Window. |
| Delete Watch | Deletes the selected watch point from the Watch Window.This menu is valid only when a variable is selected in the Watch Window. |

| | |
|---|---|
| Symbol... | Displays the address of the specified variable or function, or the value of the specified symbol.<br>Opens the Symbol To Address Dialog Box. |

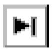
**(4) [Option] menu**

| | | |
|---|---|---|
| Tool Bar | | Selects whether the tool bar is displayed (default) or not. |
| Status Bar | | Selects whether the tool bar is displayed (default) or not. |
| Button | | Selects whether the buttons on each window are displayed (default) or not. |
| Source Mode | | Executes step execution at the source level (in line units). |
| Instruction Mode | | Executes step execution at the instruction level (in instruction units). |
| Auto Mode | | Automatically selects step execution at the source level or step execution at the instruction level (default).<br>Step execution is performed at the source level (in a mode other than mixed display mode) if the Source Window is active.  It is performed at the instruction level if the Assemble Window is active.  If neither window is active, step execution is performed at the source level. |
| Configuration... | | Sets the environment.<br>Opens the Configuration Dialog Box. |
| Extended Option... | | Sets extended options.<br>Opens the Extended Option Dialog Box. |
| RRM Setting... | | Sets the sampling range for RRM function.<br>Opens the RRM Setting Dialog Box. (when IECUBE is connected) |
| Flash Option Setting... | | This dialog box is used to make the flash self programming emulation settings. (When IECUBEis connected.)<br>Opens the Flash Option Dialog Box. |
| Debugger Option... | | Sets ID850QB options.<br>Opens the Debugger Option Dialog Box. |
| Add I/O Port... | | Adds user-defined I/O ports.<br>Opens the Add I/O Port Dialog Box. |
| Trace Clear | | Clears the trace data.<br>This item is displayed only when Trace View Window is active (when IECUBE connected). |
| Coverage | | Manipulates the coverage measurement. (when IECUBE is connected) |
| | Clear | Clears the coverage measurement results. (when IECUBE is connected) |
| | Select... | Selects the coverage measurement range as a space of 1 MB or more. (when IECUBE is connected)<br>Opens the Coverage-Address Dialog Box. |
| RRM Function | | Select the RRM function. (when IECUBE is connected) (refer to "5.13 RRM Function")<br>This function is used on a mutually exclusive basis with the RRM function and coverage function. (-> Refer to "5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis".<br>When this is selected, the following menu cannot be selected.<br>[Trace start/Trace end], [Uncond. Trace ON/Cond.Trace ON], [Tracer Control Mode] |

| | |
|---|---|
| Trace Function | Select the trace function. (when IECUBE is connected) (refer to "5.10 Trace Function (When IECUBE Is Connected)")<br>This function is used on a mutually exclusive basis with the RRM function and coverage function. ( Refer to "5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis".)<br>When this is selected, the following menu cannot be selected.<br>[RRM Setting...] |
| Coverage function | Select the coverage function. (when IECUBE is connected) (refer to "5.11 Coverage Measurement Function (When IECUBE Is Connected)")<br>This function is used on a mutually exclusive basis with the RRM function and trace function. ( Refer to "5.11.4 RRM function, trace function, and coverage function used on a mutually exclusive basis".)<br>When this is selected, the following menu cannot be selected.<br>[Trace start/Trace end], [Uncond. Trace ON/Cond.Trace ON], [Tracer Control Mode], [RRM Setting...] |

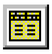
### (5) [Run] menu

| | |
|---|---|
| Restart | Resets the CPU and executes the program.<br>Same operation as this button. |
| Stop | Forcibly stops program execution.<br>Same operation as this button. |
| Go | Executes the program from the current PC.<br>Same operation as this button. |
| Ignore break points and Go | Ignores break points being set, and executes the program.<br>Same operation as this button. |
| Return Out | The user program is executed until execution returns<br>Same operation as this button.<br>**Note:** This command is used for a function described in C language. |
| Step In | Executes the instructions in the program one by one (step execution).<br>If a function or subroutine is called, its instructions are executed one by one.<br>Same operation as this button. |
| Next Over | Executes the instructions in the program one by one (Next step execution).<br>If a function or subroutine is called, its instructions are not executed on a step-by-step basis.<br>Same operation as this button. |
| Start From Here | Executes the program from the cursor position on the Source Window or Assemble Window. |
| Come Here | Executes the program from the current PC to the cursor position in the Source Window or Assemble Window. |
| Go & Go | Continues executing the program.<br>If a break occurs because a break condition is satisfied, the window is updated and the program is executed again.<br>Same operation as clicking this button each time a break has occurred. |

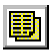
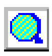
| | |
|---|---|
| Slowmotion | Continues step execution.<br>Each time step execution has been performed, the window is updated and then step execution is performed again.<br>Same operation as clicking this button each time a break has occurred. |
| CPU Reset | Resets the CPU.<br>Same operation as this button. |
| Change PC | Sets the address at the cursor position in the Source Window or Assemble Window to the PC. |
| Break Point | Sets or deletes a breakpoint at the cursor position in the Source Window or Assemble Window. |
| Software Break Point | Sets or deletes a software breakpoint at the cursor position in the Source Window or Assemble Window. |
| Delete All Breakpoints | Deletes all the set break events. |
| Uncond. Trace ON | Validates unconditional trace so that trace can always be executed during program execution (default). (When IECUBE is connected.)<br>At this time, the set trace event conditions are ignored.<br>The trace mode cannot be changed while the tracer is activated. |
| Cond. Trace ON | Validates conditional trace and traces in accordance with the trace event condition during program execution (when IECUBE is connected ).<br>The trace mode cannot be changed while the tracer is activated. |
| Tracer control mode | Sets trace control mode (when IECUBE is connected). |
| Non Stop | Goes around the trace memory and overwrites data from the oldest frame (default). |
| Full Stop | Goes around the trace memory and then stops the tracer. |
| Full Break | Goes around the trace memory and then stops the tracer and program execution |
| Delay Trigger Stop | Traces data by the number of delay count frames and stops the tracer when a delay trigger event has occurred. |

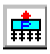
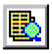
| | |
|---|---|
| Timer Start/Timer Stop | Starts timer measurement when it is stopped, or stops it when it is in progress (when IECUBE is connected).<br>This item is invalid if the program is not being executed and if a timer event is not used. Immediately after program execution has been started, timer measurement is in progress. |
| Tracer Start/Tracer Stop | Starts the tracer when it is stopped, or stops it when it is in progress (when IECUBE is connected).<br>This item is invalid if the program is not being executed.<br>Immediately after program execution has been started , the tracer is executed. |

### (6) [Event] menu

| | |
|---|---|
| Event Manager | Manages various event conditions.<br>Opens the Event Manager.<br>Same operation as this button. |
| Software Break Manager | Manages software break event conditions.<br>Opens the Software Break Manager. |
| Event... | Registers an event condition.<br>Opens the Event Dialog Box.<br>Same operation as this button. |
| Event Link... | Registers an event link condition<br>Opens the Event Link Dialog Box. |
| Break... | Registers and sets a break event condition.<br>Opens the Break Dialog Box.<br>Same operation as this button. |
| Trace... | Registers and sets a trace event condition (when IECUBE is connected).<br>Opens the Trace Dialog Box.<br>Same operation as this button. |
| Timer... | Registers and sets a timer event condition (when IECUBE is connected).<br>Opens the Timer Dialog Box.<br>Same operation as this button. |
| Delay Count... | Sets the delay count (when IECUBE is connected).<br>Opens the Delay Count Dialog Box. |

### (7) [Browse] menu

| | |
|---|---|
| Source Text | Displays a source text.<br>Opens the Source Window.<br>If there is this window already open in the active status, it is opened in the static status.<br>Same operation as this button. |
| Assemble | Displays the disassemble results.<br>Opens the Assemble Window.<br>If there is this window already open in the active status, it is opened in the static status.<br>Same operation as this button. |
| Memory | Displays the contents of the memory.<br>Opens the Memory Window.<br>If there is this window already open in the active status, it is opened in the static status.<br>Same operation as this button. |
| Watch | Displays the watch contents.<br>Opens the Watch Window.<br>Same operation as this button. |

| | |
|---|---|
| Register | Displays the register contents.<br>Opens the Register Window.<br>Same operation as this button. |
| I/O Register | Displays the contents of the Peripheral I/O registers.<br>Opens the IOR Window.<br>Same operation as this button. |
| Local Variable | Displays the local variable.<br>Opens the Local Variable Window.<br>Same operation as this button. |
| Stack Trace | Displays the stack trace results.<br>Opens the Stack Window.<br>Same operation as this button. |
| Trace | Displays the trace results (when IECUBE is connected) .<br>Opens the Trace View Window.<br>Same operation as this button. |
| Code Coverage | Displays the code coverage measurement results (when is IECUBE is connected).<br>Opens the Code Coverage Window.<br>Same operation as this button. |
| Console | Opens the Console Window. |
| Others | Displays other windows. (refer to "A.2 Sample Window" )<br>Displays a user-defined window list. |

**(8) [Jump] menu**

| | |
|---|---|
| Source Text | Displays the corresponding source text and source line, using the data value selected in the current window as the jump destination address.<br>If no line information exists at the jump destination address, however, you cannot jump.<br>Opens the Source Window.<br>If the Source Window in active is open, that window is displayed in the forefront (so that it can be manipulated). |
| Assemble | Disassembles and displays the results from the jump destination address specified by the data value selected in the current window.<br>Opens the Assemble Window.<br>If the Assemble Window in active is open, that window is displayed in the forefront (so that it can be manipulated). |
| Memory | Displays the memory contents from the jump destination address specified by the data value selected in the current window.<br>Opens the Memory Window.<br>If the Memory Window in active is open, that window is displayed in the forefront (so that it can be manipulated). |

### (9) [Window] menu

| | |
|---|---|
| New Window | Opens a new window displaying the same contents as those of the current window.This menu is valid only when the current window is the Source Window, Assemble Window or Memory Window. |
| Cascade | Cascade display of the windows in the Main window. |
| Tile | Cascade display of the windows in the Main window. |
| Arrange Icons | Rearranges the icons in the Main window. |
| Close All | Closes all windows, except the Main window. |
| Refresh | Updates the contents of the window with the latest data. |
| Active | Sets the window in the active status. |
| Static | Sets the window in the static status. |
| (Open Window) | Lists the windows that are open.<br>The window with the check mark shown on the side of the figure is the current window. By selecting a window name, the selected window is used as the current window. |

### (10) [Help] menu

| | |
|---|---|
| ID850QB Help | Displays the help. |
| Command Reference | Opens the Help window of COMMAND REFERENCE. |
| Main Window | Displays the help of the Main window. |
| Current Window | Displays the help of the current window. |
| About... | Displays the version of the ID850QB.<br>Opens the About Dialog Box. |

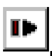
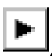
## Toolbar

(1) Meaning of each button

(2) Operation of Toolbar

### (1) Meaning of each button

The meaning of each button on the toolbar is as follows.When the mouse cursor is placed on a button of the toolbar, a tool hint pops up several seconds later.

| | |
|---|---|
|  **Stop** | Stops execution of the user program.<br>Same function as [Run] menu -> [Stop]. |
|  **ReGo** | Resets the CPU and executes the user program.<br>Same function as [Run] menu - > [Restart]. |
|  **Go** | Executes the user program from the current PC without resetting the CPU.<br>Same function as [Run] menu -> [Go]. |

| | |
|---|---|
| **Go** | Ignores break points being set, and executes the user program.<br>Same function as [Run] menu -> [Ignore break points and Go]. |
| **Ret** | The user program is executed until execution returns<br>Same function as [Run] menu -> [Return Out].<br>**Note:** This command is used for a function described in C language. |
| **Step** | Step execution (executes instructions in the program one by one.)<br>If a function or subroutine is called, its instructions are executed one by one.<br>Same function as [Run] menu -> [Step In]. |
| **Over** | Next step execution (executes the program, assuming a function/call statement as one step.)<br>If a function or subroutine is called, its instructions are not executed on a step-by-step basis.<br>Same function as [Run] menu -> [Next Over]. |
| **Res** | Resets the CPU.<br>Same function as [Run] menu -> [CPU Reset]. |
| **Open** | Opens the View File Load Dialog Box.<br>Same function as [File] menu -> [Open...]. |
| **Load** | Opens the Download Dialog Box.<br>Same function as [File] menu -> [Download...]. |
| **Proj** | Opens the Project File Load Dialog Box.<br>Same function as [File] menu -> [Project] -> [Open...]. |
| **Src** | Displays the source text.<br>Opens the Source Window.<br>Same function as [Browse] menu -> [Source Text]. |
| **Asm** | Displays the disassemble results.<br>Opens the Assemble Window.<br>Same function as [Browse] menu -> [Assemble]. |
| **Mem** | Displays the contents of the memory.<br>Opens the Memory Window.<br>Same function as [Browse] menu -> [Memory]. |
| **Wch** | Displays the watch contents.<br>Opens the Watch Window.<br>Same function as [Browse] menu -> [Warch]. |
| **Reg** | Displays the register contents.<br>Opens the Register Window.<br>Same function as [Browse] menu -> [Register]. |
| **IOR** | Displays the contents of the Peripheral I/O registers.<br>Opens the IOR Window.<br>Same function as [Browse] menu -> [I/O Register] . |
| **Loc** | Displays the local variable contents.<br>Opens the Local Variable Window.<br>Same function as [Browse] menu -> [Local Variable]. |
| **Stk** | Displays the stack trace results.<br>Opens the Stack Window.<br>Same function as [Browse] menu -> [Stack Trace]. |
| **TrW** | Displays the trace results. (when IECUBE is connected)<br>Opens the Trace View Window.<br>Same function as [Browse] menu -> [Trace]. |
| **Cov** | Displays the code coverage measurement results. (when IECUBE is connected)<br>Opens the Code Coverage Window.<br>Same function as [Browse] menu -> [Code Coverage]. |

| | |
|---|---|
| Mgr | Opens the Event Manager.<br>Same function as [Event] menu -> [Event Manager...]. |
| Evn | Registers and sets events.<br>Opens the Event Dialog Box.<br>Same function as [Event] menu -> [Event...]. |
| Brk | Registers and sets break events.<br>Opens the Break Dialog Box.<br>Same function as [Event] menu -> [Break...]. |
| Trc | Registers and sets a trace event (when IECUBE is connected).<br>Opens the Trace Dialog Box.<br>Same function as [Event] menu -> [Trace...]. |
| Tim | Registers and sets a timer event (when IECUBE is connected).<br>Opens the Timer Dialog Box.<br>Same function as [Event] menu -> [Timer...]. |

**(2) Operation of Toolbar**

Whether the toolbar is displayed or not can be specified by selecting [Option] -> [Tool Bar] from the menu bar.

This toolbar can be displayed in the following two modes. The modes are selected in the Debugger Option Dialog Box.

Figure 6-2 Toolbar (Picture Only)

Figure 6-3 Toolbar (Picture and Text)


## Window display area

This area displays various debug windows.

The displayed window can be changed in size or an icon can be created in this area.

## Status bar

The status bar displays the status of the ID850QB and in-circuit emulator.

While the user program is being executed, the status bar is displayed in red.

Whether the status bar is displayed or not can be specified by selecting [Option] -> [Status Bar] from the menu bar.

**Remark:** If the screen resolution is low (800 - 600, etc.), all the statuses may not be displayed on the status bar.

Figure 6-4 Status Bar


| | |
|---|---|
| (1) Program name | Displays the program file name indicated by the PC value. |
| Source name | Displays the source file name indicated by the PC value. |
| Line number | Displays the line number indicated by the PC value. |
| (2) Function name | Displays the function name indicated by the PC value. |
| (3) PC value | Displays the current PC value. |
| (4) CPU status | Refer to "Table 6-2 CPU Status". |
| (5) IE status | Refer to "Table 6-3 IE Status".<br>(If there are two or more the statuses, they delimited with '\|' and displayed.) |
| (6) Break Cause | Refer to "Table 6-4 Break Cause". |
| (7) STEP mode | Displays the step execution mode.<br>Displays that the following modes are selected from the [Option] menu:<br>SRC............. Source mode<br>INST ............ Instruction mode<br>AUTO ........... Automatic mode |
| (8) Key input mode | Displays the key input mode.<br>INS ............... Insertion mode<br>OVR............. Overwrite mode<br>The Memory Window is fixed to OVR mode. |

Table 6-2 CPU Status

| Display | Meaning |
|---|---|
| HALT | Halt mode |
| STOP/IDLE | Software stop mode, hardware stop mode, Idle mode |
| HOLD | Bus hold mode |
| WAIT | Wait mode |
| RESET | Reset mode |
| POW OFF | Power is not supplied to the target |

Table 6-3 IE Status

| Display | Meaning |
|---|---|
| RUN | User program execution in progress (the color of the status bar changes). |
| STEP | Step execution in progress. |
| TRC | Trace operating*1 |
| TIM | Timer operating*1 |
| COV | Coverage operating*1 |
| BREAK | Break occurring. |
| *Time* | Displays the result of measuring the time from the start of user program execution to the occurrence of break (Run-Break time) (refer to "Table 6-14 Measurable Values") . |
| TIMER OVERFLOW | Measurement result overflowed. |

Table 6-4 Break Cause

| Display | Meaning |
|---|---|
| Manual Break | Forced break |
| Temporary Break | Temporary break |
| Event Break | Break by event |
| Software Break | Software break |
| Trace Full Break | Break due to trace full*1 |
| Non Map Break | Non-mapped area is accessed. *1 |
| Write Protect | An attempt has been made to write to a write-protected area. *1 |
| IOR Illegal | Illegal access to peripheral I/O register was performed *1 |
| Timer Over Break | Execution time-over detected *1 |
| Flash Macro Service | Flash macro service in progress*2 |
| IRAM Write Protect (xxx xxx) | During break, performed verify check of IRAM guarded area and rewrote value. *1<br>xxx xxx indicates the relevant address and data (in case of multiple items, only first item is displayed). |
| Illegal Opcode Trap | Break due to illegal opcode trap*1 |

***1** When IECUBEis connected.

***2** When N-Wire CARD, MINICUBEis connected.

# Configuration Dialog Box

This dialog box is used to display and set the ID850QB operation environment. (Refer to "5.1 Setting Debugging Environment".)

This dialog box is automatically displayed after the ID850QB is started up.

However, no setting is required to read a project as the results of reading the project file are reflected in this dialog box. (Refer to "5.15.1 Debugging environment (project file)".)

Figure 6-5 Configuration Dialog Box (When IECUBE Is Connected)

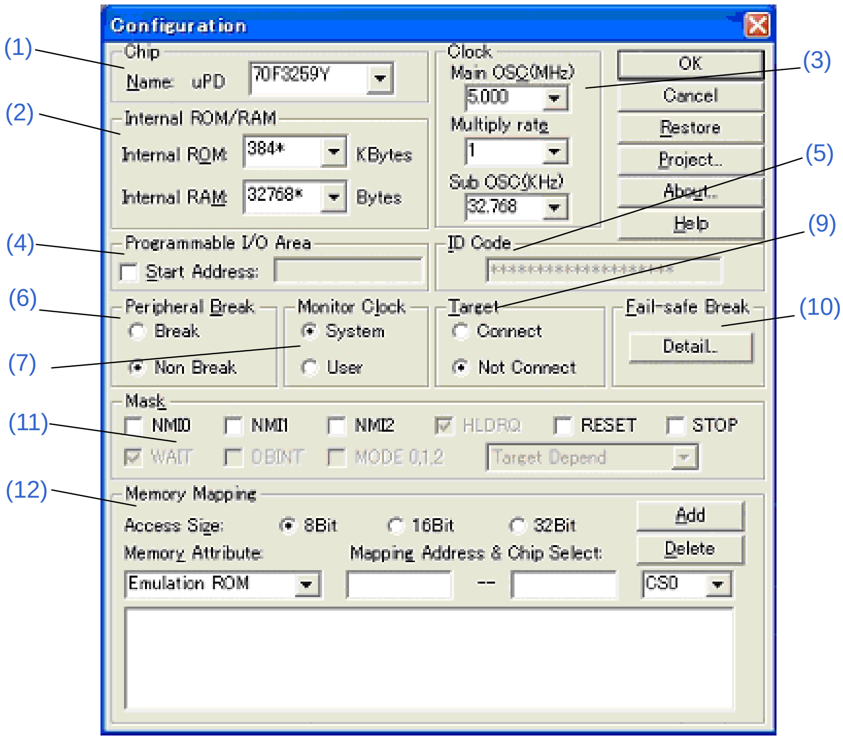
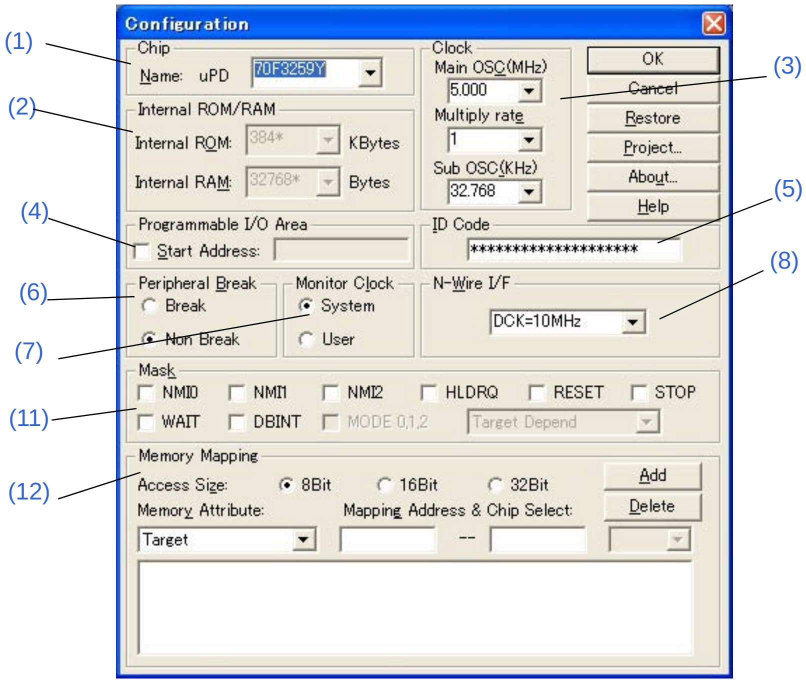


Figure 6-6 Configuration Dialog Box (When N-Wire CARD, MINICUBE is connected)




- Opening
- Explanation of each area
- Function buttons

## Opening

(Automatically when the ID850QB is started up)

Select [Option] -> [Configuration...] from the menu bar.

## Explanation of each area

**(1) Chip**

This area is used to select the chip name.

A chip name is selected from the drop-down list.

On the drop-down list, only the chip names registered to the registry from the device file installer are displayed. This area can be specified only when the debugger is started up.

**Remark:** By default, the type selected at the previous startup is displayed, but if that type is not registered, the first type registered is displayed.

**(2) Internal ROM/RAM**

This area is used to set the size of the internal RAM and internal ROM of the CPU. (Refer to "Table 5-2 Mapping Attribute".)

When N-Wire CARD, MINICUBE is connected, the sizes of the ROM and RAM in the device file are fixed.

A mapping size is selected from the drop-down list.

The default size is obtained from the device file through selection in (1) Chip , and displayed (value with '*').

Table 6-5 Range and Unit of Internal ROM/RAM Setting (When IECUBE Is Connected)

| Area | Meaning | Settable Range |
|---|---|---|
| Internal ROM | Sets the Internal ROM size | 0,8,32,64,128,256,512,1024 (KB) |
| Internal RAM | Sets the Internal RAM size | 4096,12288,28672,61440 (bytes) |

**(3) Clock**

Settings related to the main clock and subclock are performed here.

| | |
|---|---|
| Main OSC (MHz) | Specifies a frequency before the main clock is multiplied.<br>A frequency can be selected from the drop-down list (5.000, 8.000, 13.500, or 18.000) or directory input. |
| Multiply rate | Specifies the main clock multiplication rate.<br>This value can be selected from the drop-down list (1 to 10), or directly input. |
| Sub OSC (KHz) | Specifies the subclock frequency (setting this parameter is disabled for types that do not have a subclock).<br>A frequency can be selected from the drop-down list or directory input. |

**(4) Programmable I/O Area**

This area is used to specify use of the programmable I/O area and the start address.

The start address of the programmable I/O area can be specified, only if the device selected by (1) Chip supports the programmable I/O area. The start address of the programmable I/O area can be input by checking the check box when the programmable I/O area is used. The address is aligned to 16 KB.

**Remark:** In the case of a device with an extended I/O area with fixed addresses, the device selected by (1) Chip, Setting of this area is performed automatically.

**(5) ID Code**

This area is used to input the ID code to be used when the code on the internal ROM or internal flash memory is read by ID850QB (ID code authentication) (When N-Wire CARD, MINICUBE is connected)

This area does not have to be set with a ROMless product or a product without a RCU (security unit).

Input a hexadecimal number of 20 digits (10 bytes) as the ID code (all 'F' by default).

The ID code is saved to the registry. If inputting the ID code fails three times, the ID850QB is forcibly terminated.

For the details of ID code authentication, refer to N-Wire CARD, MINICUBE User's Manual.

### (6) Peripheral Break

This area is used to specify whether the peripheral emulation function of in-circuit emulator is stopped during a break.

| | |
|---|---|
| Break | Stopped |
| Non Break | Not stopped (default) |

### (7) Monitor Clock

This area is used to specify whether the operation clock of the monitor program is switched from the sub clock to the main clock during a break. (when IECUBE is connected)

This area does not have to be set with a product without a sub clock.

| | |
|---|---|
| System | The operation clock is switched to the main clock and the monitor program is executed (default).<br>**Note:**<br>In theID850QB, the clock is changed by manipulating PCC, but not while the main clock is stopped. If the operation clock is switched to the main clock during a break, the clock is returned to the previous setting when execution returns to the user program. |
| User | The monitor program is executed with the clock selected by the user program. |

### (8) N-Wire I/F

This area is used to select a clock supplied form N-Wire CARD or MINICUBE to the on-chip debug unit (DCU) (when N-Wire CARD, MINICUBE is connected). In default, a 10 MHz clock is supplied.

**Caution:** Usually, 10 MHz clocks must be selected. When a 20 MHz clock is selected, the ID850QB may not start operating.

| | |
|---|---|
| DCK=10MHz | The DCK clock is 10 MHz (in default).<br>**Remark:**<br>During selection, the maximum value of the measurement time of the execution time measurement function is doubled, and that of the resolution of the execution time measurement function is reduced by half. |
| DCK=20MHz | In this case, the DCK clock is 20 MHz. |

### (9) Target

This area is used to select whether the target board is to be connected to the in-circuit emulator or not (when IECUBE is connected).

**Remark:** It is used to detect an illegal power supply status.
The default is determination by detecting the power (TVDD) of the target. (Refer to "Table 3-2 Error Message Output Pattern (When IECUBE Is Connected)".)

| | |
|---|---|
| Connect | Be connected |
| Not Connect | Not be connected |

**(10) Fail-safe Break**

Clicking the <Detail...> button opens the Fail-safe Break Dialog Box, so that the fail-safe break function can be individually set. (When IECUBE is connected.)

**(11) Mask**

This area is used to mask the signal sent from the target.

The signal of a masked pin is not input to the in-circuit emulator.

Mask a pin only when the operation of the target system is not stable at the debugging stage.

**Remarks1:** By checking RESET, the external reset or internal reset generated by the watchdog timer can be masked. At this time, whether internal reset can be masked or not depends on the device.

**Remarks2:** If a device file supporting TM tag is used, however, select Mode00 to Mode1F (the modes to be displayed are determined by the definition of the device file). When the IECUBE is connected and when the target is connected, Target Depend can be selected.

**(12) Memory Mapping**

This area is used to set mapping by specifying the access size, memory attribute, and address.

(a) Access Size

Selects memory access size.

This setting is used to specify the access size on the ID850QB software; the operation of the external bus hardware is set in accordance with the settings of the MODE pin and I/O register.

| | |
|---|---|
| 8Bit | Accesses memory with ld.b instruction/st.b instruction. |
| 16Bit | Accesses memory with ld.h instruction/st.h instruction. |
| 32Bit | Accesses memory with ld.w instruction/st.w instruction. |

(b) Memory Attribute

The following mapping attributes can be selected. Select a mapping attribute according to the usage. (Refer to "Table 5-2 Mapping Attribute".)

| | |
|---|---|
| Emulation ROM | Selects the in-circuit emulator alternate ROM.<br>(when IECUBE is connected, or when memory boards are incorporated)<br>(Mapping unit:1MB) |
| Emulation RAM | Selects the in-circuit emulator alternate RAM.<br>(when IECUBE is connected, or when memory boards are incorporated)<br>(Mapping unit:1MB) |
| Target | Selects the target memory.<br>(Mapping unit:1MB) |
| Target ROM | Selects the target ROM (when IECUBE is connected) |
| I/O Protect | Selects the I/O protect area.<br>(This area can be set only in the area set as Target.)<br>(Mapping unit:1 byte) |

**Caution:** The area set as "I/O Protect" is not read unless it is registered to the IOR Window or Watch Window as an I/O port. To read this area, forcibly read it on these windows.

(c) Mapping Address & Chip Select

Specify the address to be mapped.

Input the higher and lower addresses from the keyboard.

Since the areas that are specified in the emulation memory are composed of 16 MB (16 banks of 1 MB memory), those areas can be allocated to any location in CS0 through CS7 with the chip select function. (Select one from the drop-down list.)

Allocation addresses can be allocated to any 1 MB boundary.

Multiple banks can be assigned to one single CS.

**Remark:** In the case of the V850ES Series, the allocation of chip selection is fixed, or no chip-selection functions are provided, no selection can be made. No selections can be made if the option board is not installed when N-Wire CARD, MINICUBE is connected or when IECUBE is connected.

(d) <Add>button, <Delete> button

These buttons are used to set and delete mapping.

When the <Add> button is clicked, the mapping specified with (b) Memory Attribute and (c) Mapping Address & Chip Select is set and displayed.

**Remark:** Moreover, mapping unit adjustment is performed except I/O Protect, and if the mapping units is not matched, the minimum settable range including the specified address becomes the mapping target.

**Caution:** If the external memory is mapped, change the value of a register required for access of an external memory.(refer to "5.1.4 To change the value of a register required for access of an external memory").

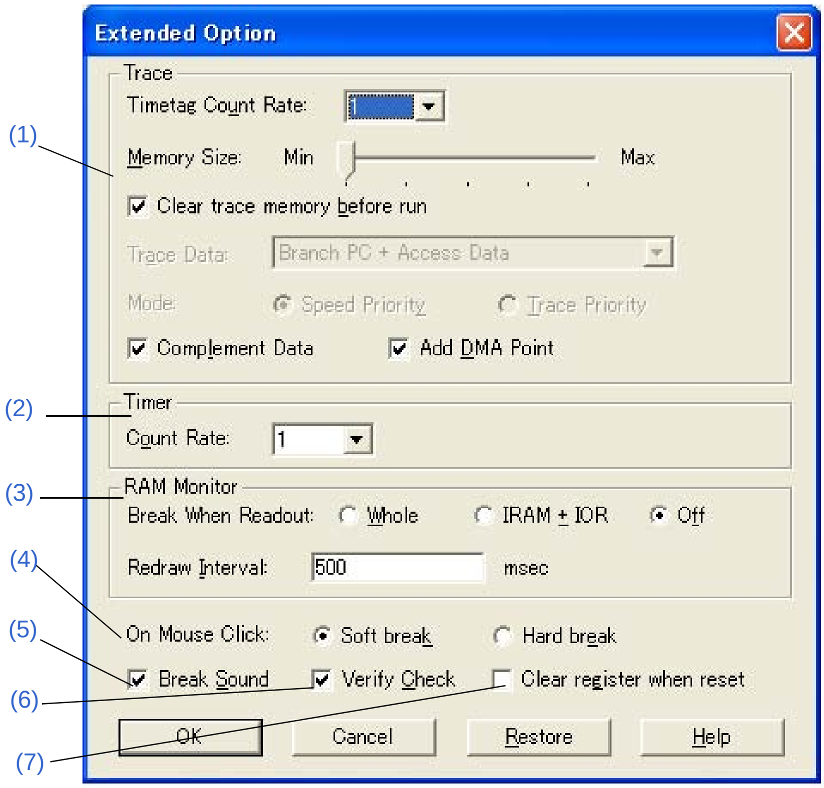
## Function buttons

| | |
|---|---|
| OK | Validates the current environment.<br>Sets the environment and closes this dialog box. |
| Cancel | Cancels the changes and closes this dialog box. |
| Restore | Restores the previous settings before this dialog box was opened. |
| Project... | Opens the Project File Load Dialog Box. If an error occurs while a project file is being opened or read, the ID850QB can no longer continue and is terminated. |
| About... | Opens the About Dialog Box. |
| Help | Displays the help window of this window. |

# Extended Option Dialog Box

This dialog box is used to display and set the extended options of the ID850QB. (Refer to "5.1 Setting Debugging Environment".)

Figure 6-7 Extended Option Dialog Box Extended Option Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Option ] menu -> [Extended Option ...] .

## Explanation of each area

**(1) Trace**

This area is used to set about trace (refer to "5.10 Trace Function (When IECUBE Is Connected)") (when the IECUBE is connected).

(a) Timetag Count Rate:

This area is used to set the division ratio of the counter used for time tag display in the Trace View Window.

This division ratio is selected from a drop-down list.

If the division ratio is set, the number of clocks necessary for counting up the counter displayed for time tag is changed.

The relationship between the time tag counter division ratio and maximum measurement time is as follows.

Table 6-6 The Relationship Between the Time Tag Counter Division Ratio and Maximum Measurement Time (Time tag counter (Trace))

| Division Ratio | Resolution (ns) | Measurable Maximum Time | Remark |
|---|---|---|---|
| TMCLK (1/1) | 20 | 1.4minutes | Time tag counter 32 bits, In case of 50 MHz external clock |
| TMCLK (1/2) | 40 | 2.8minutes | |
| TMCLK (1/4) | 80 | 5.7minutes | |
| TMCLK (1/8) | 160 | 11.4minutes | |
| TMCLK (1/16) | 320 | 22.8minutes | |
| TMCLK (1/32) | 640 | 45.6minutes | |
| TMCLK (1/64) | 1280 | 1.5hours | |
| TMCLK (1/128) | 2560 | 3hours | |
| TMCLK (1/256) | 5120 | 6hours | |
| TMCLK (1/512) | 10240 | 12.2hours | |
| TMCLK (1/1024) | 20480 | 24.4hours | |
| TMCLK (1/2048) | 40960 | 48.8hours | |
| TMCLK (1/4096) | 81920 | 97.6hours | |

(b) Memory Size

Set the size of the trace memory (buffer).

In other words, specify a memory size by dragging the knob. The sizes that can be specified are 8K (min.), 32K, 64K, 128K, and 256K (max).

**Caution:** The larger the value that is set, the greater the number of trace data that are recorded. However, the response when reading trace data becomes correspondingly slower.

(c) Clear Trace Memory Before Run

Select this checkbox to clear the trace memory prior to program execution.

(d) Trace Data

Select the trace data to be collected.

**Remark:** When [Options] menu -> [RRM Function] is selected, the setting is fixed to Branch PC.
When [Option] menu -> [Coverage Function] is selected, the setting is fixed to All PC.

Table 6-7 Relationship Between Meaning of Trace Data to Be Collected and Trace Collection Mode

| Item | Traced Range | | | |
|---|---|---|---|---|
| | **Branch PC** Collects PC values of branch origin and branch destination instructions | **All PC** Collects PC values of all instructions | **Access PC** Collects PC values of instructions that caused access | **Access Data** Collects access address and access data |
| Branch PC | Traced | - | - | - |
| All PC | Traced*1 | | | - |
| Access Data | - | - | - | Traced*2 |
| Branch PC + Access Data | Traced | - | - | Traced*2 |
| All PC + Access Data | Traced*1 | | | Traced*2 |
| Access Data + Access PC | - | - | Traced | Traced*2 |
| Branch PC + Access Data + Access PC | Traced | - | Traced | Traced*2 |

***1** When trace data that contains "All PC" is selected, the unconditional trace mode is enabled. In this case, the qualify trace mode and the section trace mode cannot be set at the same time. (The conditional trace setting is disabled.)

***2** When trace data that contains "Access Data" is selected, when the high-speed priority mode (select Speed Priority in (e) Mode:) is selected, and when access to the internal RAM area is performed 32 times in succession, data may be missed.

(e) Mode:

Specify the trace collection mode.

**Remark:** When [Option] menu -> [RRM Function / Coverage Function] is selected, the setting is fixed to Speed Priority.

| | |
|---|---|
| Speed Priority | This mode performs tracing by prioritizing speed (real-time operation).<br>In this mode, data may be missed depending on the trace data to be collected (refer to "Table 6-7 Relationship Between Meaning of Trace Data to Be Collected and Trace Collection Mode"). |
| Trace Priority | This mode performs tracing by prioritizing data collection (non-real-time).<br>Since, in order to reliably collect all the trace data, the CPU's execution pipeline is momentarily stopped when data is likely to be missed, the real-time characteristic of operation in relation to the user program is lost.<br>Trace Priority cannot be selected when Branch PC or All PC is selected in (d) Trace Data. |

(f) Complement Data

Select this checkbox to perform complementary display of trace data (default:checked).

(g) Add DMA Point

Check this checkbox to perform the DMA point trace function. (The checkbox is checked in default.)

When the checkbox is checked, the DMA start and end frames are marked.

**(2) Timer**

Set the rate value for the timer counter.

The values set in this area are displayed in (4) Count Rate of the Timer Dialog Box.

The rate value is selected from a drop-down list.

The relationship between the timer counter division ratio and maximum measurement time is as follows.

Table 6-8 The Relationship Between the Timer Count Division Ratio and Maximum Measurement Time (Timer counter (Timer))

| Division Ratio | Resolution (ns) | Measurable Maximum Time | Remark |
|---|---|---|---|
| TMCLK (1/1) | 20 | 2.8minutes | Timer counter 33 bits, In case of 50 MHz external clock |
| TMCLK (1/2) | 40 | 5.7minutes | |
| TMCLK (1/4) | 80 | 11.4minutes | |
| TMCLK (1/8) | 160 | 22.8minutes | |
| TMCLK (1/16) | 320 | 45.6minutes | |
| TMCLK (1/32) | 640 | 1.5hours | |
| TMCLK (1/64) | 1280 | 3hours | |
| TMCLK (1/128) | 2560 | 6hours | |
| TMCLK (1/256) | 5120 | 12.2hours | |
| TMCLK (1/512) | 10240 | 24.4hours | |
| TMCLK (1/1024) | 20480 | 48.8hours | |
| TMCLK (1/2048) | 40960 | 97.6 時 hours | |
| TMCLK (1/4096) | 81920 | 195.2hours | |

**(3) RAM Monitor**

(a) Break When Readout:

Select this item to specify the target range of RAM sampling by instantaneously generating a break in the user program execution (refer to "5.13.2 Pseudo real-time RAM monitor function (Break When Readout)").

| | |
|---|---|
| Whole | Whole memory space.Note<br>**Remark:**<br>The user program execution is stopped for a long time when a large number of windows are opened because the range from which memory is read out is wid. |
| IRAM + IOR | Internal RAM area and IOR area |
| Off | Disables the Pseudo real-time RAM monitor function (Break When Readout)(default). |

**Note:** The range specified in the RRM Setting Dialog Box is excluded.

(b) Redraw Interval:

Specify the sampling time (ms) of the RAM sampling.

It can be specified in 100-ms units from 0 to 65500.

If 0 is specified, or if this area is blank, the data is not displayed in real time.

**(4) On Mouse Click**

This area is used to select whether a software breakpoint or hardware breakpoint is set as the default breakpoint if a breakpoint is set in the point mark area by clicking the mouse button in the Source Window or Assemble Window (refer to "5.4.2 Breakpoint setting").

| | |
|---|---|
| Soft break | Sets a software breakpoint (default).<br>The mark of breakpoint is displayed in blue. |
| Hard break | Sets a hardware breakpoint.<br>The mark of breakpoint is displayed in green (before execution). |

**(5) Break Sound**

If the check box is checked, a beep sound is issued when a break occurs.

**(6) Verify Check**

This area is used to specify whether a verify check is performed when data has been written to memory.

A verify check is performed when download, memory fill, or memory copy is executed. A verify check is also performed when a variable or data is changed in the Watch Window or Memory Window and is written to memory.

**Caution:** When N-Wire CARD, MINICUBE is connected, during write to the internal flash memory (including download), verify check is not performed whether or not the checkbox in this area is selected, and internal verify of flash self-write is always performed (read verify is not performed).

**(7) Clear Register When Reset**

Select this checkbox in order to clear the program registers (r1 to r31) and registers EIPC, EIPSW, FEPC, FEPSW, CTPC, CTPSW, and CTBP upon CPU reset.

Under the default setting, the registers are not cleared.

## Function buttons

| | |
|---|---|
| OK | Validates the settings and closes this dialog box. |
| Cancel | Cancels the changes and closes this dialog box. |
| Restore | Restores the previous settings before this dialog box was opened. |
| Help | Displays the help window of this window. |

# Fail-safe Break Dialog Box

IECUBE

This dialog box is used to perform the fail-safe break settings. (Refer to "5.1 Setting Debugging Environment".)
When a project file is read, the results obtained by reading this project file are reflected in this dialog box.

**Remark:** For details on the fail-safe break function, refer to the user's manuals of the in-circuit emulator and the emulation board that are used.

Figure 6-8 Fail-safe Break dialog box


- Opening
- Explanation of each area
- Function buttons

## Opening

Click the <Detail...> button in the Configuration Dialog Box.

## Explanation of each area

**(1) Protect**

The fail-safe break protect settings are performed in this area.
The fail-safe breaks corresponding to the selected checkboxes are protected.
Under the default setting, all checkboxes are selected.

(a) Internal ROM:

This area is used to perform the protect settings for the internal ROM area.

| | |
|---|---|
| Non Map | Access to access prohibited area |
| Write | Write to write prohibited area |

(b) Internal RAM:

This area is used to perform the protect settings for the internal RAM area.

| | |
|---|---|
| Non Map | Access to access prohibited area |

(c) I/O Register:

This area is used to perform the protect settings for the peripheral I/O registers area.

| | |
|---|---|
| Non Map | Access to access prohibited area |
| Read | Read to read prohibited area |
| Write | Write to write prohibited area |

(d) External Memory:

This area is used to perform the protect settings for the external memory area.

| | |
|---|---|
| Non Map | Access to access prohibited area |
| Write | Write to write prohibited area |

**(2) Verify**

This area is used to perform the verify settings for fail-safe breaks.

Verify check is performed during access to the items corresponding to the selected checkboxes.

(a) Internal RAM:

This area is used to perform the verify check setting in the internal RAM area.

| | |
|---|---|
| Non Map Write | Write to write prohibited area |

## Function buttons

| | |
|---|---|
| OK | Validates the settings and closes this dialog box. |
| Cancel | Closes this dialog box. |
| Restore | Restores the previous settings before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# RRM Setting Dialog Box

IECUBE

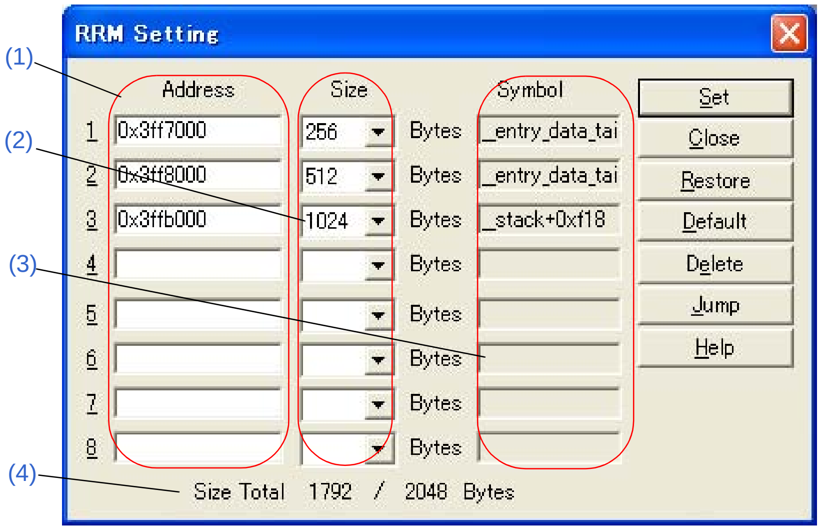
This dialog box is used to set the sampling range for the RRM function. (Refer to "5.13 RRM Function".

Up to 8 locations can be specified in 256-byte units as the sampling range.

The total of the sizes specified for the 8 locations cannot exceed 2048 bytes.

**Caution:** This dialog box cannot be opened when an item other than [RRM Function] has been selected in the [Option] menu.

Figure 6-9 RRM Setting Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

The settings of this dialog box when it is opened differ depending on the opening method.

(a) When settings are performed from RRM Setting Dialog Box

The dialog box is opened by selecting [Option] menu -> [RRM Setting...].

In this case, the data in (1) Address and (2) Size are input manually.

(b) When settings are performed from the Memory Window

This dialog box is opened by opening the Memory Window, selecting an address in the window, and then selecting [RRM Setting...] from the context menu.

In this case, the selected address is displayed in an empty row in (1) Address, "256" is displayed in an empty row in (2) Size, and the value obtained by converting the address to a symbol is displayed in an empty row in (3) Symbol.

**Remark:** If the total of the sizes specified for the 8 locations already exceeds 2048 bytes, the dialog box opens but no value can be set.

(c) When settings are performed from the Watch Window

This dialog box is opened by opening the Watch Window, selecting a variable in the window, and then selecting [RRM Setting...] from the context menu.

In this case, the value obtained by converting the variable into an address is displayed in an empty row in (1) Address, "256" is displayed in an empty row in (2) Size, and the value obtained by converting the variable to a symbol is displayed in an empty row in (3) Symbol.

**Remark:** If the total of the sizes specified for the 8 locations already exceeds 2048 bytes, the dialog box opens but no value can be set.

## Explanation of each area

### (1) Address

This area is used to specify the sampling start address for the RRM function.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)

Following input, click the <OK> button to enable the settings.

**Remark:** When the settings are enabled, the addresses are aligned in 256-byte units, but if an address is duplicate, it is not enabled.

**(2) Size**

This area is used to specify the sampling range from (1) Address.

The values that can be selected are 256, 512, 768, 1024, 1280, 1536, 1792, 2048.

However, the total of the sizes specified for the 8 locations cannot exceed 2048 bytes.

**(3) Symbol**

This area displays the symbols of the addresses specified in (1) Address.

The specified addresses are displayed as a symbol or as a symbol + offset.

If the address has not been set, nothing is displayed.

**(4) Size Total**

This area displays the total of the sizes specified in (2) Size. If the total exceeds 2048 bytes, it is displayed in red.

## Function buttons

| | |
|---|---|
| Set | Determine the specified sampling range. |
| Close | Closes this dialog box. |
| Restore | Restores the previous settings before this dialog box was opened. |
| Default | Clears the current setting and sets the internal RAM start address in the first row in (1) Address, and "2048" in the first row in (2) Size. |

| | |
|---|---|
| Delete | Deletes the setting for the numbers with a focus. |
| Jump | Opens the Memory Window and displays the addresses in (1) Address whose numbers have a focus.<br>Jump is performed for Memory Window that are in the active status.<br>If multiple memory windows are to be opened, they must be set in the static status.<br>(Refer to "5.16.1 Active status and static status") |
| Help | Displays this dialog box online help files. |

# Flash Option Dialog Box

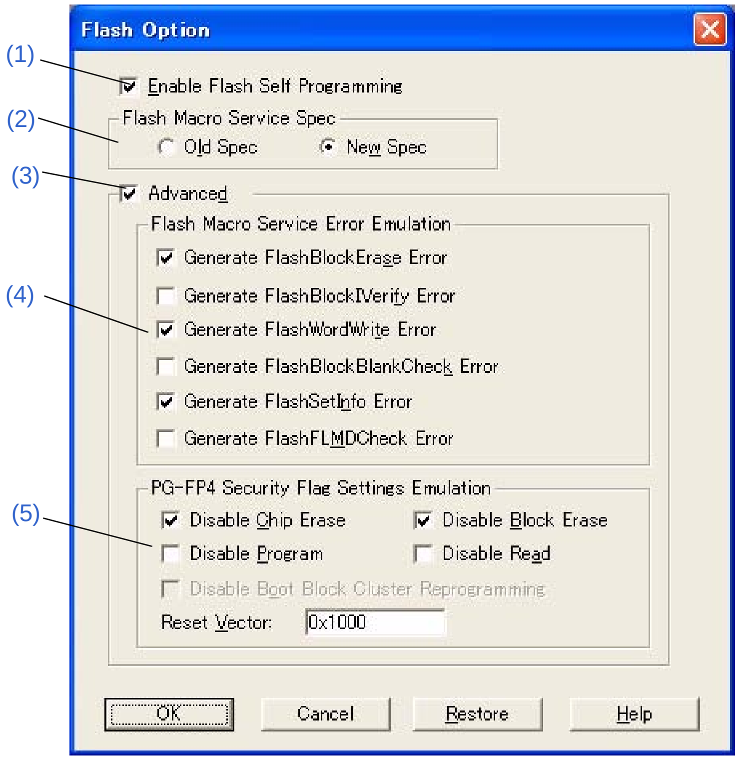
IECUBE

This dialog box is used to make the flash self programming emulation settings.

This dialog box cannot be opened during user program execution.

**Caution:** This dialog box cannot be opened when using a device that does not incorporate the flash memory.

Figure 6-10 Flash Option Dialog Box




- Opening
- Explanation of each area
- Cautions
- Function buttons

## Opening

Select [Option] menu -> [Flash Option...].

## Explanation of each area

### (1) Enable Flash Self Programming

If this item is selected, flash self emulation is enabled.

This enables setting of (3) Advanced.

This item is not selected by default.

### (2) Flash Macro Service Spec

Select the flash macro service specification type in this area when flash self emulation is enabled.

This area is valid only when the flash memory process is "Type1".

If "New Spec" is selected, "Disable Read" and "Reset Vector" become selectable in (5) PG-FP4 Security Flag Settings Emulation. Input an address within the address range of the internal flash memory in the "Reset vector" field for the reset vector address.

| | |
|---|---|
| Old Spec | Flash memory self programming Ver. 2.00 or earlier (default) |
| New Spec | Flash memory self programming Ver. 3.00 or later |

### (3) Advanced

If this item is selected, detailed settings for flash self emulation are enabled.

This enables setting of (4) Flash Macro Service Error Emulation and (5) PG-FP4 Security Flag Settings Emulation.

This item is not selected by default.

### (4) Flash Macro Service Error Emulation

These items set the operation of the self library function.

When this checkbox is checked, an error value that is returned when flash memory is damaged can forcibly be returned. (The error values that are returned when flash memory is damaged are not returned in normal emulation.)

The settable items are as follows: All items are not selected by default.

| | |
|---|---|
| Generate FlashBlockErase Error | Returns error values in FlashBlockErace functions. |
| Generate FlashBlockIVerify Error | Returns error values in FlashBlockIVerify functions. |
| Generate FlashWordWrite Error | Returns error values in FlashWordWrite functions. |
| Generate FlashBlockBlankCheck Error | Returns error values in FlashBlockBlankCheck functions. |
| Generate FlashSetInfo Error | Returns error values in FlashSetInfo functions. |
| Generate FlashFLMDCheck Error[Note] | Returns error values in FlashFLMDCheck functions. |

**Note:** IECUBE cannot read the values of the FLMD0 pin. (Normally, with an assumption that a high level is input to the FLMD0 pin, the FlashFLMDCheck functions always return a normal completion.) Check the "GenerateFlashFLMDCCheck Error" checkbox when errors, which occur when a low level is input to the FLMD0 pin in the case of the FlashFLMDCheck functions, need to forcibly be returned.

**(5) PG-FP4 Security Flag Settings Emulation**

The initial value of the security flag is emulated when the security has been set to the flash memory using flash memory programmer PG-FP4.

The settable items are as follows: All items are not selected by default.

| | |
|---|---|
| Disable Chip Erase | Prohibits chip erase |
| Disable Block Erase | Prohibits block erase |
| Disable Program | Prohibits write |
| Disable Read[Note1] | Prohibits read |
| Disable Boot block cluster reprogramming[Note2] | Prohibits boot area rewrite |
| Reset vector[Note1] | Reset vector address setting |

**Note1:** This item is selectable only when the flash memory process is "Type1" and flash memory self programming Ver. 3.00 or later is used.

**Note2:** This item is selectable only when the flash memory process is "Type2" or "Type3".

## Cautions

**(1) Flash self emulation in the following case.**

(a) The internal ROM size is not set to the default size

Workaround: Set the default size to the internal ROM size.

(b) When two breaks before execution are used

Workaround: Disable or delete one of the breaks before execution.

**(2) The following restriction applies if flash self emulation is enabled.**

(a) The internal ROM size and internal RAM size cannot be changed.

(b) The DMM function and pseudo RRM functions cannot be used.

(c) When the SP register is 0 (the SP register does not indicate the internal RAM area), the user program illegally breaks.

(d) When the "Clear register when reset" checkbox is checked in the Extended Option Dialog Box, the value of the r3 register that the flash macro macro service changed in the reset emulation is cleared.

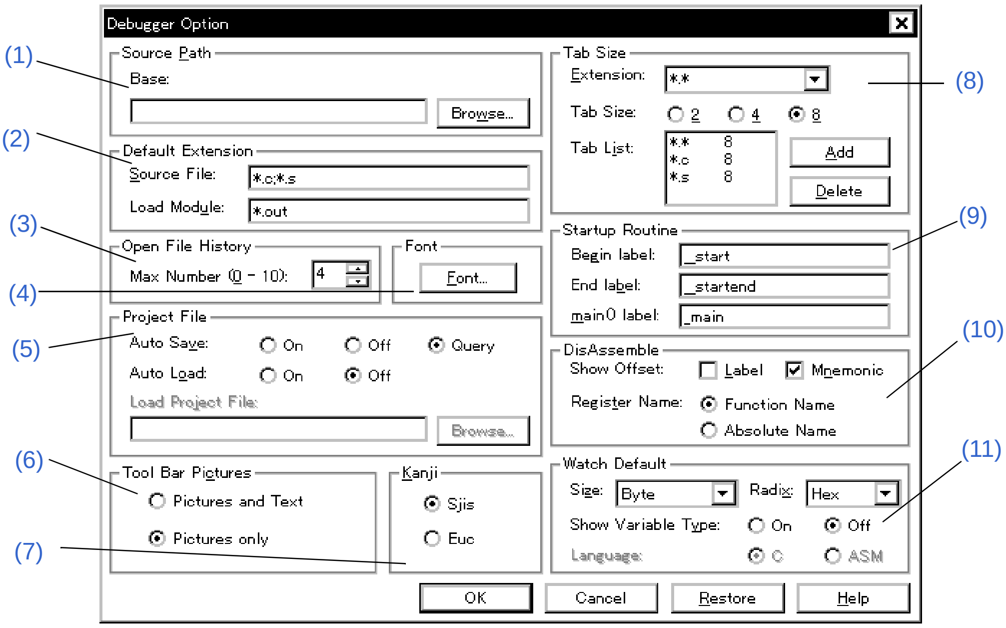
## Function buttons

| | |
|---|---|
| OK | Validates the settings and closes this dialog box. |
| Cancel | Closes this dialog box. |
| Restore | Restores the previous settings before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Debugger Option Dialog Box

This dialog box is used to display and set the various options of the ID850QB.

Figure 6-11 Debugger Option Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Option] menu -> [Debugger Option...].

## Explanation of each area

**(1) Source Path**

This area is used to specify the directory in which a source file or text file is searched.

(a) Base:

The directory is the basis of a relative path is displayed. The base directory is determined in the following sequence:

(i) Directory to which the project file has been loaded

(ii) Directory to which a load module or hex file has been loaded last

(iii) Current directory of Windows

(b) Text box

This area is used to specify the directory searched.

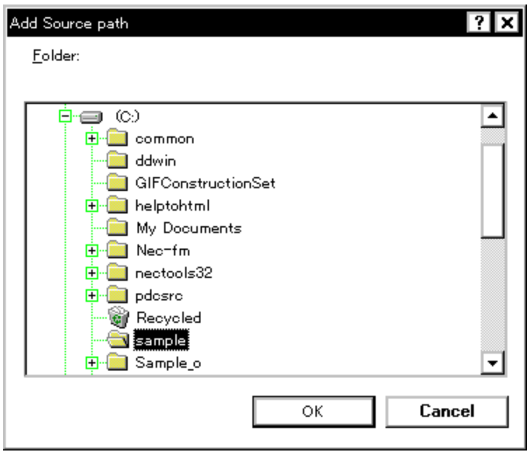
To specify a directory, either directly input one to the text box, or click the <Browse...> button. A relative path can also be specified.

Opens the Add Source Path Dialog Box by clicking the <Browse...> button. To delimit paths, use ";" (semicolon) or "," (comma).

**Remarks1:** Directories that contain ";" and/or "," in the source path can be specified. Non-existent directories cannot be specified.

**Remarks2:** Immediately after this dialog box has been opened, the base directory is selected and opened. If the selected directory has already been set for the source path, a source path is not added.

Figure 6-12 Add Source Path Dialog Box



**(2) Default Extension**

This area is used to specify the default extension.

Delimit extensions with " " (blank),";" (semicolon) or "," (comma).

| | |
|---|---|
| Source File: | Set the extension of a source file that is displayed when the Browse Dialog Box is opened by selecting [File] menu -> [Open...].<br>The default extension is " *.c, *.s ". |
| Load Module: | Set the extension of a load module that is displayed when the Download Dialog Box is opened.<br>The default extension is " *.out ". |

**(3) Open File History**

This area is used to set the number of histories of the open file displayed in the bottom field of the [File] menu. The default value is 4. If 0 is set, no history is displayed on the menu.

**(4) Font**

This area is used to specify the font displayed on the Source Window, Watch Window, Quick Watch Dialog Box, Local Variable Window, and Stack Window. Clicking the <Font...> button opens the Font Dialog Box in which the font to be displayed and its size can be set.

**(5) Project File**

This area is used to set automatic saving and loading of the project file. (Refer to "5.15.1 Debugging environment (project file)".)

(a) Auto Save:

Sets whether the project file is automatically saved at the ID850QB termination.

| | |
|---|---|
| On | Automatically saves the project file at the ID850QB termination. |
| Off | Does not automatically save the project file at the ID850QB termination. |
| Query | Displays the Exit Debugger Dialog Box at the ID850QB termination (default). |

(b) Auto Load:

Sets whether the project file is automatically loaded at the ID850QB start up.

| | |
|---|---|
| On | Automatically loads the project file at the ID850QB start up. Specify the project file to be loaded in (c) Load Project File:. |
| Off | Does not automatically loads the project file at the ID850QB start up. (default) |

(c) Load Project File:

Specify the project file to be loaded automatically.

Set a project file name by inputting from the keyboard or clicking the <Browse...> button.

Clicking the <Browse...> button displays the Browse Dialog Box.

**(6) Tool Bar Pictures**

This area sets the buttons to be displayed on the tool bar. (Refer to " Toolbar".)

| | |
|---|---|
| Pictures and Text | Displays a button on which a graphic and character are displayed. |
| Pictures only | Displays a button with only graphic (default). |

**(7) Kanji**

Cannot be selected in this area.

**(8) Tab Size**

This area is used to set the tab size for each extension when files are displayed.

(a) Extension:

Set an extension. Input an extension from the keyboard, or select one from the drop-down list.

(b) Tab Size:

Select the tab size. Select how many spaces are displayed as a tab code (2, 4, or 8).

(c) Tab List:

Displays the tab size set for each extension.

(d) <Add> button

To change the tab size setting, select (a) Extension: and (b) Tab Size: and click the <Add> button.

(e) <Delete> button

To delete the tab size setting, select the setting to be deleted from Tab List and click <Delete> button.

**(9) Startup Routine**

This area is used to specify the first address , end address, and display start symbol of the text area (code area) of the start-up routine by symbols.

The source file can be opened if an object file in the load module format is downloaded in the Download Dialog Box.

| | |
|---|---|
| Begin label: | Specifies the symbol of the first address (default: _start) |
| End label: | Specifies the symbol of the end address (default: _startend) |
| main() label: | Specifies the display start symbol (default _main) |

**Cautions1:** If the specified symbol is not correct, the source file cannot be opened until the PC reaches the address range of the corresponding source file. In addition, the start-up routine cannot be skipped by step execution.

**Cautions2:** Be sure to specify this area. If this area is blank, the dialog box cannot be closed.

**(10) DisAssemble**

This area is used to set for disassemble display.

(a) Show Offset:

Specifies whether an offset (symbol + offset) is displayed during disassemble display.

When the offset is not displayed, only a symbol that matches a numeric value is displayed, if any. If no matching symbol is found, the numeric value is displayed as a hexadecimal number unchanged.

| | |
|---|---|
| Label | Specifies whether the offset is displayed in the Label field. In the default condition, the offset is not displayed. |
| Mnemonic | Specifies whether the offset is displayed in the Mnemonic field. In the default condition, the offset is displayed. |

(b) Register Name:

This area is used to select the method of displaying register names in mnemonics during disassemble display.

| | |
|---|---|
| Function Name | Displays register names as function names or nicknames (default). |
| Absolute Name | Displays register names as absolute names. |

**(11) Watch Default**

This area is used to specify a symbol to be watched in the Watch Window etc. .

(a) Size:

Sets the default display size of data if [Adaptive] is specified.

| | |
|---|---|
| Byte | 8-bit display(default) |
| Half Word | 16-bit display |
| Word | 32-bit display |

(b) Radix:

Sets the default radix in which data is to be displayed if [Proper] is specified.

| | |
|---|---|
| Hex | Displays data in hexadecimal numbers (default). |
| Dec | Displays data in decimal numbers. |
| Oct | Displays data in octal numbers. |
| Bin | Displays data in binary numbers. |
| String | Displays data in character strings. |

(c) Show Variable Type:

Select the display/non-display of variable type is specified.

| | |
|---|---|
| On | Displays the type of a variable. |
| Off | Does not display the type of a variable (default). |

(d) Language:

Select the display/non-display of type of variable is specified.

| | |
|---|---|
| C | Displays a C-like base number (default). |
| ASM | Cannot be selected. |

## Function buttons

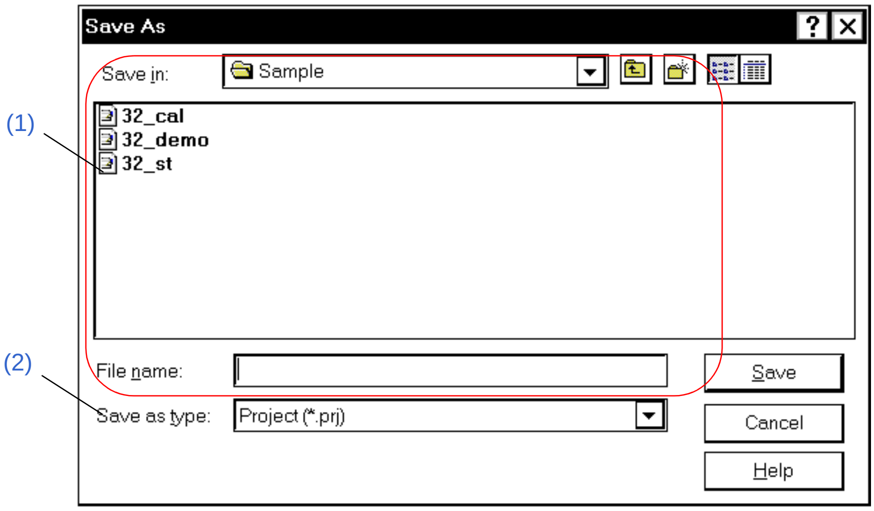
| | |
|---|---|
| OK | Validates the settings and closes this dialog box. |
| Cancel | Cancels the changings and closes this dialog box. |
| Restore | Restores the previous settings before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Project File Save Dialog Box

This dialog box is used to save the current debugging environment to a project file. (Refer to "5.15.1 Debugging environment (project file)".)

Project files can be newly saved or saved under an existing file name in this dialog box.

Figure 6-13 Project File Save Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [File] menu -> [Project] -> [Save As...].

(To save a file of same name as a project file previously loaded or saved, select [File] menu -> [Project] -> [Save].)

## Explanation of each area

**(1) Save in:, File name:**

This area is used to specify a file name. A file name can be directly input, or selected from the list at the upper part of this area.

Up to 257 characters string with a extension can be specified.

**(2) Save as type:**

This area is used to specify the extension (*.prj) of the project file to be saved.

If the extension is omitted, "*.prj" is appended as the default extension.

## Function buttons

| | |
|---|---|
| Save | Saves the debugging environment to the selected file. After saving, the dialog box is closed. |
| Cancel | Closes this dialog box without saving the file. |
| Help | Displays this dialog box online help files. |

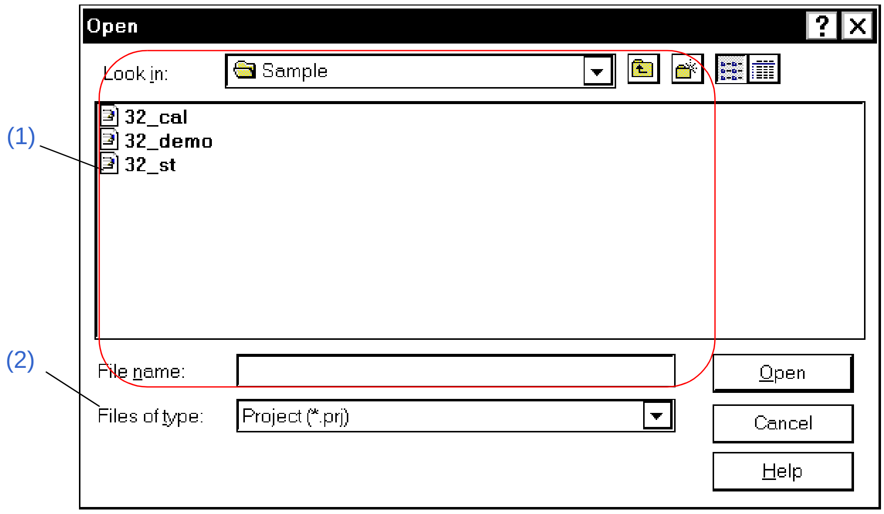
# Project File Load Dialog Box

This dialog box is used to restore the debugging environment to the debugging environment saved to the project file. (Refer to "5.15.1 Debugging environment (project file)".)

If there is an active Source Window after a project file has been loaded, it is displayed at the top.

**Caution:** Following ID850QB startup, if a project file with settings that differ from those of the target device at startup has been loaded, the target device specified at startup is used.

Figure 6-14 Project File Load Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Click the **Proj** button, or select [File] menu -> [Project] -> [Open...].

## Explanation of each area

**(1) Look In:, File name:**

This area is used to specify the file name to be loaded. A file name can be directly input from the keyboard, or selected from the list.

Up to 257 characters string with a extension can be specified.

**(2) Files of type:**

This area is used to specify the extension (*.prj) of the file to be loaded.

## Function buttons

| | |
|---|---|
| Open | Loads the selected file. After loading the file, this dialog box is closed. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |

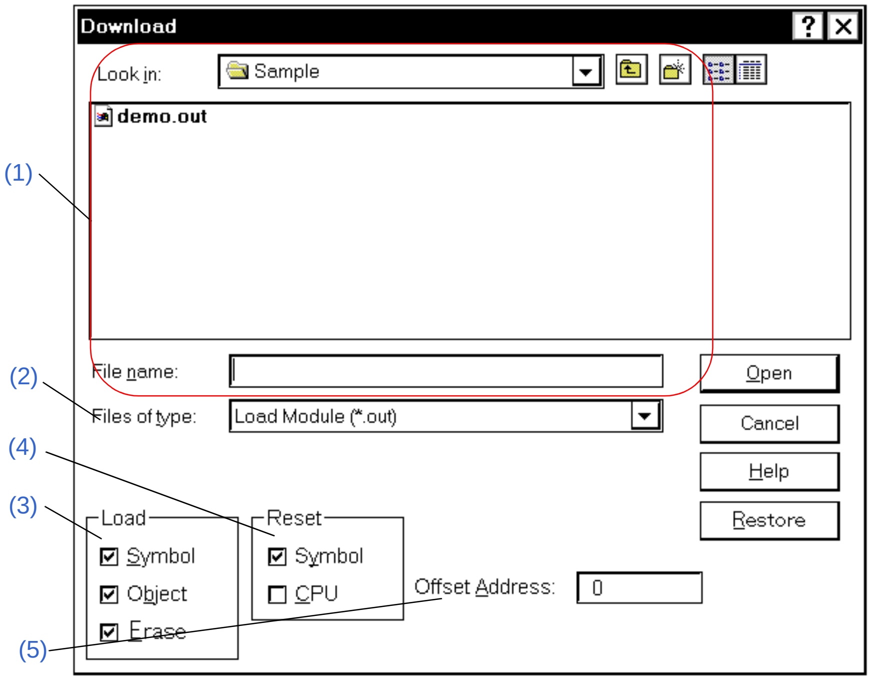
# Download Dialog Box

This dialog box is used to select the name and format of a file to be downloaded, and downloads memory contents to the in-circuit emulator and the target system. (Refer to "5.2 Download Function, Upload Function".)

If a load module file has been downloaded, the corresponding source file is searched, and the Source Window is automatically opened.

**Caution:** If a file other than a load module file is loaded, source debugging cannot be executed.

Figure 6-15 Download Dialog Box




**Remark:** The following dialog box appears while downloading and the downloading can be can celled at any time. This dialog box is closed automatically after completing downloading.

Figure 6-16 The Progress of Download

- Opening
- Explanation of each area
- Function buttons

## Opening

Click the **Load** button, or select [File] menu -> [Download...].

## Explanation of each area

**(1) Look In:, File name:**

This area is used to specify a file name. A file name can be directly input from the keyboard, or selected from the list at the upper part of this area.

Up to 257 characters string with a extension can be specified.

**Remark:** Two or more files can be specified in this dialog box. To specify two or more files, delimit each file name with " " (double quotation mark). Files can also be specified by clicking the mouse button while holding down the Shift or Ctrl key.
Up to 20 load module files can be downloaded.

**(2) Files of type:**

This area is used to specify the type (extension) of the file to be downloaded. (Refer to "Table 5-3 Type of File That Can Be Downloaded".)

**Remark:** These are default extensions; other extensions can also be used.
The default extension of the displayed load module can also be specified in the Debugger Option Dialog Box.

**(3) Load**

This area is used to set a load condition. This setting is valid only if a file in the load module format is specified.

This setting is valid only if a file in the load module format is specified.

| | |
|---|---|
| Symbol | Specifies whether symbol information is read or not. |
| Object | Specifies whether object information is read (when checked, default) or not.<br>(The object information is read even if this button is not checked when a HEX file is loaded.) |
| Erase | Specifies whether the contents of the internal flash memory are erased all before download (when checked, default) or not (when N-Wire CARD, MINICUBEconnected). |

**Remark:** Specifies whether symbol information is read (when checked, default) or not. The memory capacity can be saved by not reading symbol information when a program consisting of two or more load module files is to be debugged and if the symbol information of some modules does not have to be read.

**(4) Reset**

This area is used to set a reset condition.

This setting is valid only if a file in the load module format is specified.

| | |
|---|---|
| Symbol | Specifies whether symbol information is reset (when checked, default) or not. [Note] |
| CPU | Specifies whether the CPU is reset or not. (Not checked, default.) |

**Note:** When debugging a program consisting of two or more load module files, load each one of the load module files without resetting the symbol information. When downloading two or more load module files, take care that location addresses do not overlap.

**(5) Offset Address:**

This area is used to specify the offset address that is used when a file is loaded (for binary data, specify the start address). An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".) The default radix for inputting a numeric value is hexadecimal.

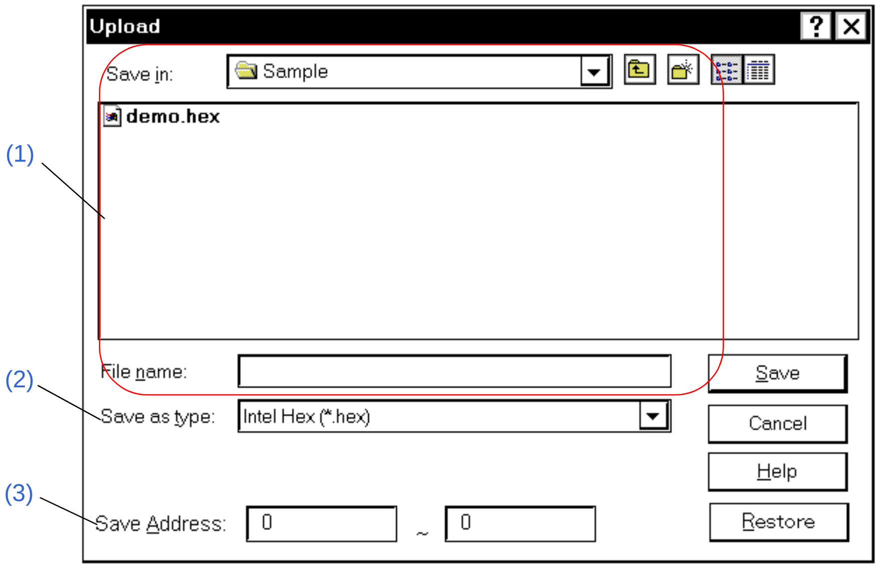
## Function buttons

| | |
|---|---|
| Open | Loads the selected file. After loading the file, this dialog box is closed. |
| Cancel | Closes this dialog box without loading the file. |
| Help | Displays this dialog box online help files. |
| Restore | Restores the input data to the original status. |

# Upload Dialog Box

This dialog box is used to set the name and format of the file to be saved, and save the set memory contents, etc., to that file. (Refer to "5.2 Download Function, Upload Function".)

Figure 6-17 Upload Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [File] menu -> [Upload...].

## Explanation of each area

**(1) Save in:, File name:**

This area is used to specify the file name to be saved. A file name can be directly input from the keyboard, or selected from the list.

Up to 257 character string with a extension can be specified.

**(2) Save as type:**

This area is used to specify the type (extension) of the file to be saved.

The format of the data to be saved is determined by the extension. (Refer to "Table 5-4 Type of File That Can Be Uploaded")

However, if "All (*.*) " is selected, the data is saved in the default Intel extended Hex format.

**Remark:** Extensions other than those listed can also be used.

**(3) Save Address:**

This area is used to specify the range of address to be saved.

All the ranges are saved (this area cannot be set) when coverage data (*.cvb) is selected in (2) Save as type:.

An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)

The default radix for inputting a numeric value is hexadecimal.

## Function buttons

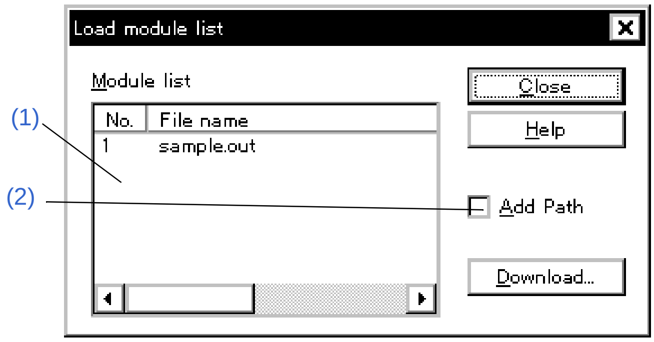
| | |
|---|---|
| Save | Saves the file according to the setting. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |
| Restore | Restores the status before this dialog box was opened. |

# Load Module List Dialog Box

This dialog box displays the list of the files and file paths that have been downloaded from Download Dialog Box (refer to "5.2 Download Function, Upload Function").

The listed files (excluding the coverage data files) are saved in the project file; they are downloaded when the project file is opened next. By using the <Download...> button, the Download Dialog Box can be opened and a file can be downloaded.

Figure 6-18 Load Module List Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [File] menu -> [Load Module...].

## Explanation of each area

**(1) Module list**

This area displays the names of the files that have already been downloaded.

| | |
|---|---|
| No. | The numbers displayed indicate the sequence in which the load module file names were read. |
| File name | The file names are displayed with the full path if (2) Add Path is checked; otherwise, only the file names will be displayed. |

**Caution:** If symbol information has been reset in the Reset Debugger Dialog Box, or if symbol information has been reset in the Download Dialog Box, the file names downloaded before that are cleared.

**(2) Add Path**

This should be checked to specify file names are displayed with the path in (1) Module list.

## Function buttons

| | |
|---|---|
| Close | Closes this dialog box. |
| Help | Displays this dialog box online help files. |
| Download... | Opens the Download Dialog Box.<br>A new load module can be downloaded. The file name of the newly file will be added to the file name display area when the Download dialog box is closed. |

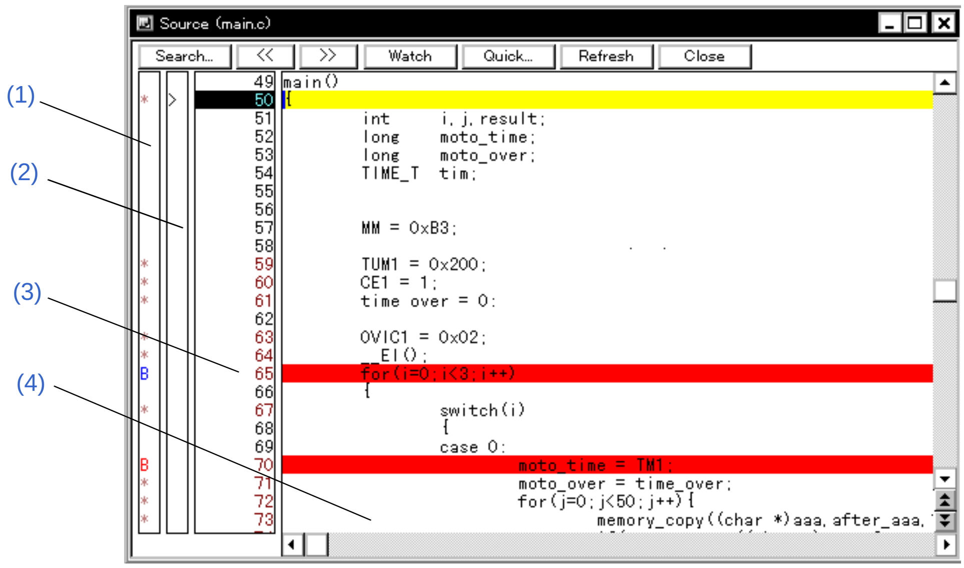
# Source Window

This window is used to displays source files or text files. (Refer to "5.3 Source Display, Disassemble Display Function".) In addition to Breakpoint setting , Display of locations for which coverage measurement is executed and Mixed display mode (Source Window), a number of other operations using Context menu, Function buttons, etc., can be performed in this window. Moreover, there are two statuses, Active status and static status, for this window. When the window is in the active status, it has the Trace Result with Linking Window (when IECUBE is connected). Moreover, the items selected in the window with Drag & drop function can be used in another window. (Refer to "5.16 Functions Common to Each Window".)

**Caution:** If program codes is described in an include file and these codes are included in multiple files, the line numbers and addresses do not correspond on a one-to-one bases. In such an include file, function that indicates the correspondence relationship between line numbers and addresses dose not correctly operate.

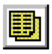
**Remark:** Up to 65,535 lines of C and assembly language source files can be displayed. If the source files exceed 65,535 lines, partition them.

Figure 6-19 Source Window




- Opening
- Explanation of each area
- [View] menu (Source Window-dedicated items)
- Context menu
- Function buttons

## Opening

Click the **Src** button, or select [Browse] menu -> [Source Text].

(This window is automatically opened if the corresponding source file exists after the download module file has been downloaded. )

## Explanation of each area

**(1) Point mark area**

This area is used for the Event Setting Status (Event Mark) and program codes (*) display, as well as Breakpoint setting.

**Remark:** The program code is displayed only when the symbol information downloaded by the load module file is read. Breakpoints can be set or deleted by clicking with the mouse on this program code. (if "*" is not displayed for the line, the breakpoint is set on the line above or below the line, which-ever has "*" displayed.)

If an event has been set for the corresponding line, one of the marks listed in the following table is displayed. The color of the "B" mark differs according to the breakpoint type and status. (When a breakpoint is set in this area, it is enabled at the same time that it is set.)

Table 6-9 Event Setting Status (Event Mark)

| Mark | Meaning |
|---|---|
| B (blue) | Software breakpoint is set. |
| B (red) | Valid hardware breakpoint (after execution) is set. |
| B (green) | Valid hardware breakpoint (before execution) is set.<br>**Note**: Breaks before execution are set with priority. |
| B (black) | Invalid hardware breakpoint is set.<br>This hardware breakpoint can be validated on the Event Manager or in the Break Dialog Box. |
| E | Event condition is set. |
| L | Event link condition is set. |
| T | Trace event is set.(IECUBE) |
| Ti | Timer event is set. (IECUBE) |
| A | Multiple events are set. |

**Remark:** If an address range is specified as the address condition of the event, the lower addresses of the range are displayed. The mask specification of the address condition is not reflected.

**(2) Current PC mark area**

The mark ">", which indicates the current PC value (PC register value), is displayed in this area.

Clicking this mark with the mouse displays a pop-up window that shows the PC register value.

By double-clicking the current PC mark area, the program can be executed up to a specified line. (Refer to "[Come Here]".)

**(3) Line number/address display area**

This area displays the line numbers of a source file or text file.

Red indicates line numbers for which corresponding program code exists, and black indicates line numbers for which corresponding program code does not exist. In the Mixed display mode (Source Window), disassemble display addresses are displayed in gray.

In addition, executed addresses are highlighted based on code coverage measurement information (refer to "5.11.3 Display of locations for which coverage measurement is executed").

**(4) Source text display area**

This area displays source files and text files.

Yellow indicates the current PC line, and red indicates lines where a valid breakpoint is set. In the Mixed display mode (Source Window), source lines are displayed in the regular color.

Moreover, this area also provides the following functions for lines (start address of program code) and addresses where the cursor has been placed.

- [Come Here], [Start From Here] (Refer to "Table 5-9 Type of Execution")
- Drag & drop function
- Context menu

**Caution:** If a Program code does not exist on the source line, the top address of the line above or below the line on which a program code exists is manipulated by these functions.
These functions cannot be performed in the following cases. The corresponding menu will be dimmed and cannot be selected.
- If a file other than a source file is displayed
- While the user program is being executed

## [View] menu (Source Window-dedicated items)

The following items are added in the [View] menu, when the Source Window is active.

| | | |
|---|---|---|
| Create Break Event | | Sets a break event that occurs if the selected variable is accessed. |
| | Break when Access to this Variable | Sets a break event that occurs if the selected variable is accessed for read/ write. |
| | Break when Write to this Variable | Sets a break event that occurs if the selected variable is accessed for write. |
| | Break when Read from this Variable | Sets a break event that occurs if the selected variable is accessed for read. |
| | Clear | Deletes a break event corresponding to the selected variable. |
| Event? | | Displays the event information of a line at the cursor position or a selected variable name. If an event is set, the Event Dialog Box is opened. |
| Mix | | Turns on/off Mixed display mode (Source Window). |

## Context menu

| | |
|---|---|
| Move... | Moves the display position.<br>Opens the Source Text Move Dialog Box. |
| Mix | Turns on/off Mixed display mode (Source Window). |
| Add Watch... | Adds the specified data to the Watch Window.<br>Opens the Add Watch Dialog Box. |
| Symbol... | Displays the address of the specified variable or function, or the value of the specified symbol.<br>Opens the Symbol To Address Dialog Box. |
| Break when Access to this Variable | Sets a break event that occurs if the selected variable is accessed for read/ write. |
| Break when Write to this Variable | Sets a break event that occurs if the selected variable is accessed for write. |
| Break when Read from this Variable | Sets a break event that occurs if the selected variable is accessed for read. |
| Clear | Deletes a break event corresponding to the selected variable. |
| Event? | Displays the event information of a line at the cursor position or a selected variable name.<br>If an event is set, the Event Dialog Box is opened. |
| Come Here | Executes the program from the current PC to the cursor position. (Refer to "Table 5-7 Break Types".) |
| Change PC | Sets the address at the cursor position to the PC. |
| Break Point | Sets or deletes a hardware breakpoint at the cursor position.<br>**Note:** Breaks before execution (B) are set with priority. |
| Software Break Point | Sets or deletes a software breakpoint at the cursor position. |

| | |
|---|---|
| Assemble | Disassembles and displays starting from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>Opens the Assemble Window.<br>If an active Assemble Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Memory | Displays the memory contents starting from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>Opens the Memory Window.<br>If an active Memory Window is open, that window is displayed in the forefront (so that it can be manipulated). |

## Function buttons

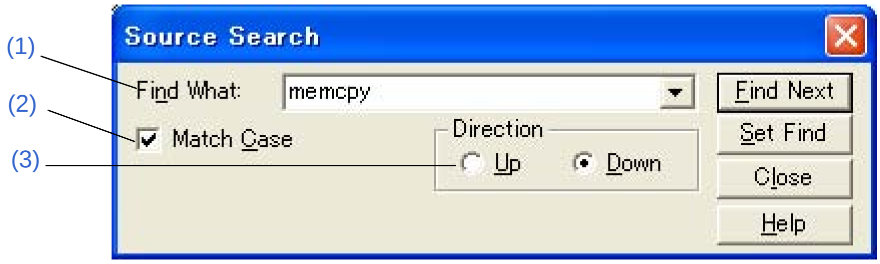
| | |
|---|---|
| Search... | Opens the Source Search Dialog Box and searches a character string of the source text. If a character string is selected in the source text display area, the Source Search Dialog Box is opened to search the character string.<br>If no character string is selected, the Source Search Dialog Box is opened with nothing specified to be searched.<br>Specify a search method in the Source Search Dialog Box.<br>The results of search is highlighted in the Source window.<br>This is the same operation as selecting [View] menu -> [Search...]. |
| << | Searches forward (upward on screen) for the text that satisfies the search condition set in the Source Search Dialog Box, starting from the address at the cursor position.<br>This button is displayed as the <Stop> button during a search. |
| >> | Searches backward (downward on screen) for the text that satisfies the search condition set in the Source Search Dialog Box, starting from the address at the cursor position.<br>This button is displayed as the <Stop> button during a search. |
| Stop (during a search) | Stops searching. |
| Watch | Adds the variables selected in the source text display area to the Watch Window.<br>If the Watch Window is not opened, it is opened.<br>If no text is selected in the source text display area, the Watch Window is only opened.<br>This is the same operation as selecting [View] menu -> [View Watch]. |
| Quick... | Temporarily displays the contents, such as a variable, selected in the source text display area in the Quick Watch Dialog Box. If no text is selected in the source text display area, the Quick Watch Dialog Box is only opened.<br>This is the same operation as selecting [View] menu -> [Quick Watch...]. |
| Refresh | Updates the contents of the window with the latest data. |
| Close | Closes this window. |

# Source Search Dialog Box

This dialog box is used to search the contents of a file in the Source Window. (Refer to "5.3.1 Source display".)

By setting each item and then clicking the <Find Next> button, searching can be started. By clicking the <Set Find> button, the direction buttons ("<<" and ">>") in the Source Window can be used for the search.

Figure 6-20 Source Search Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Source Window is the current window, select [View] menu -> [Search...], or click the <Search...> button in the same window.

## Explanation of each area

**(1) Find What**

This area is used to specify the data to be searched. (Up to 256 character.)

In the default condition, the string selected in the window that called this dialog box is displayed. As necessary, the character string displayed can be changed.

Up to 16 input histories can be recorded.

**(2) Match Case**

This should be checked to distinguish between uppercase and lowercase.

**(3) Direction**

This area is used to specify the direction of the search.

| | |
|---|---|
| Up | Forward search. Searches data forward (upward on screen) from the current position of the cursor. |
| Down | Backward search. Searches data backward (downward on screen) from the current position of the cursor (default). |

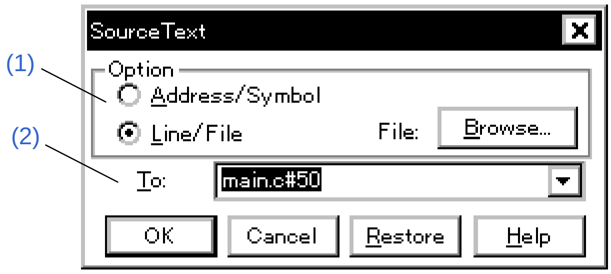
## Function buttons

| | |
|---|---|
| Find Next | Searches the specified data in accordance with a given condition.<br>If the specified character string is found as a result of a search, it is highlighted.<br>To continue searching, click this button again. |
| Set Find | Sets the specified condition as the search condition and closes this dialog box. |
| Stop (during searching) | Stops searching. |
| Close | Closes this dialog box.<br>(During searching, this button is replaced by the <Stop> button.) |
| Help | Displays this dialog box online help files. |

# Source Text Move Dialog Box

This dialog box is used to specify a file to be displayed in the Source Window and the position from which displaying the file is to be started. (Refer to "5.3.1 Source display".)

Figure 6-21 Source Text Move Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Source Window is the current window, select [View] menu -> [Move...].

## Explanation of each area

**(1) Option**

This area is used to select the input mode when the display start position is specified.

(a) Address/Symbol

This should be selected to specify by an address (or symbol).

(b) Line/File

This should be selected to specify by a line number (or file name). To search the file name, use the <Browse...> button.

**(2) To:**

This area is used to specify the file name or address to be displayed.

Up to 16 input histories can be recorded.

- When the (a) Address/Symbol is selected

  Specifies the address from which display is to be started.

  The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or a expression. (Refer to "Table 5-6 Specifying Symbols".)

  Clicking the <OK> button displays the source text so that the source line corresponding to the specified address value can be viewed.

- When the (b) Line/File is selected

  Specifies the line number (or a file name) from which display is to be started.

  The line number is specified by **[[path name] file name]# line number**.

  The default radix for inputting a numeric value is decimal.

  The file name can be specified just bythe file name, or using the absolute path and relative path.

  If just the file name or the relative path was specified, the file in the source path specified in the Debugger Option Dialog Box is searched.

  The file whose specified line number was specified as the first line is displayed by clicking the <OK> button. When the file name is omitted, the currently displayed file is displayed from the specified line. If the line number is omitted, the file is displayed from the first line.

## Function buttons

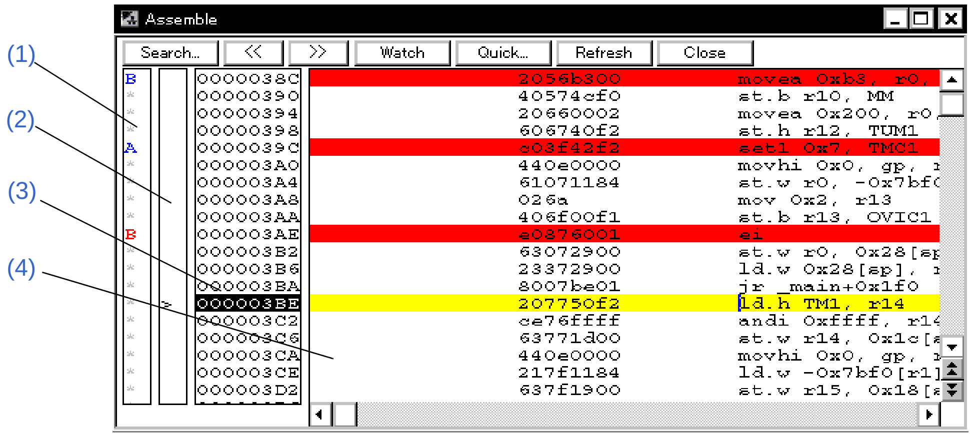
| | |
|---|---|
| OK | Starts displaying the source text from the specified position. |
| Cancel | Closes this dialog box. |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Assemble Window

This window is used to disassemble and display programs. It is also used to execute Online assembly. (Refer to "5.3 Source Display, Disassemble Display Function".) The results of online assembly are also reflected in the Memory Window.

In addition to Breakpoint setting and Display of locations for which coverage measurement is executed, a number of other operations using Context menu, Function buttons, etc., can be performed in this window. Moreover, there are two statuses, Active status and static status, for this window. When the window is in the active status, it has the Trace Result with Linking Window (when IECUBE is connected). Moreover, the items selected in the window with the Drag & drop function can be used in another window. (Refer to "5.16 Functions Common to Each Window".)

Figure 6-22 Assemble Window




- Opening
- Explanation of each area
- [View] menu (Assemble Window-dedicated items)
- Context menu
- Function buttons
- Related operations

## Opening

Click the **Asm** button, or select [Browse] menu -> [Assemble].

## Explanation of each area

**(1) Point mark area**

This area is used for Event Setting Status (Event Mark) and Breakpoint setting.

**(2) Current PC mark area**

The mark ">", which indicates the current PC value (PC register value), is displayed in this area.

By double-clicking the current PC mark area, the program can be executed up to a specified line. (Refer to "[Come Here]".)

**(3) Address specification area**

This area displays the disassembly start address.

In addition, executed addresses are highlighted based on code coverage measurement information (refer to "5.11.3 Display of locations for which coverage measurement is executed").

**Caution:** The end address is not display. The end address is 0xFFFFFFFE

**(4) Disassemble display area**

This area displays the labels and code data of addresses, and disassembled mnemonics.

This area displays source files and text files. Yellow indicates the current PC line, and red indicates lines where a valid breakpoint is set.

It can be Online assembly in the mnemonic field.

This area also provides the following functions:

- [Come Here], [Start From Here] (Refer to "Table 5-9 Type of Execution")
- Drag & drop function
- Context menu

## [View] menu (Assemble Window-dedicated items)

The following items are added in the [View] menu, when the Assemble Window is active.

| | |
|---|---|
| Event? | Displays the event information of the address at the cursor position.<br>If an event is set, the Event Dialog Box is opened. |

## Context menu

The menu items are effective for the selected line or item, not the position where the mouse pointer was clicked (same operation as when selecting the main menu with the same name).

| | |
|---|---|
| Move... | Moves the display position.<br>Opens the Address Move Dialog Box. |
| Add Watch... | Adds the specified data to the Watch Window.<br>Opens the Add Watch Dialog Box. |
| Symbol... | Displays the address of the specified variable or function, or the value of the specified symbol.<br>Opens the Symbol To Address Dialog Box. |
| Come Here | Executes the program from the current PC to the cursor position. (Refer to "Table 5-7 Break Types".) |
| Change PC | Sets the address at the cursor position to the PC. |
| Break Point | Sets or deletes a hardware breakpoint at the cursor position.<br>**Remark:** Breaks before execution (B) are set with priority (Nx85ET). |
| Software Break Point | Sets or deletes a software breakpoint at the cursor position. |
| Source Text | Displays the corresponding source text and source line, using the data value at the cursor position as the jump destination address. (Refer to "5.16.2 Jump function".)<br>If no line information exists at the jump destination address, however, you cannot jump.<br>Opens the Source Window.<br>If an active Source Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Memory | Displays the memory contents starting from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>Opens the Memory Window.<br>If an active Memory Window is open, that window is displayed in the forefront (so that it can be manipulated). |

## Function buttons

| | |
|---|---|
| Search... | Opens the Assemble Search Dialog Box and searches for a character string of mnemonics.<br>Specify a search method in the Assemble Search Dialog Box. The results of search is highlighted in the Assemble Window.<br>This is the same operation as selecting [View] menu -> [Search...]. |
| << | Searches forward (upward on screen) for the contents that satisfy the search condition set in the Assemble Search Dialog Box, starting from the address at the cursor position.<br>This button is displayed as the <Stop> button during a search. |
| >> | Searches backward (downward on screen) for the contents that satisfy the search condition set in the Assemble Search Dialog Box, starting from the address at the cursor position.<br>This button is displayed as the <Stop> button during a search. |
| Stop(during a search) | Stops searching. |

| | |
|---|---|
| Watch | Adds the symbols selected in (4) Disassemble display area to the Watch Window. If the Watch Window is not opened, it is opened.<br>If no text is selected in (4) Disassemble display area, the Watch Window is only opened.<br>This is the same operation as selecting [View] menu -> [View Watch]. |
| Quick... | Temporarily displays the contents, such as symbols, selected in (4) Disassemble display area on the Quick Watch Dialog Box. Opens the Quick Watch Dialog Box. If no text is selected in the disassemble display area, the Quick Watch Dialog Box is only opened.<br>This is the same operation as selecting [View] menu -> [Quick Watch...]. |
| Refresh | Updates the contents of the window with the latest data. |
| Close | Closes this window. |

## Related operations

**(1) Online assembly**

To change the disassembled contents, move the cursor to the mnemonic field (the overwrite and insertion modes are alternately selected by pressing the Insert key).

If an attempt is made to move the cursor to another line after the disassembled contents have been changed in the mnemonic field, the new contents are checked. If the new contents are illegal, the code data on the line where the contents have been changed is indicated as "*".

The contents changed in the mnemonic field are written into the memory by pressing the Enter key. By pressing the Enter key, the new contents are checked. If even one line is illegal, the new contents are not written into the memory. To discard the contents, press the ESC key.

If the contents are correct and if the Enter key is pressed, the contents are written to the memory, and then the cursor moves to the next line in the mnemonic field, so that the data on the next line can be changed.

**Caution:** If the number of new instruction bytes is less than the number of previous instruction bytes as a result of changing, as many 'nop' instructions as necessary are inserted. If the number of new instruction bytes is more than the number of previous instruction bytes, the next instruction is overwritten. In this case also, as many 'nop' instructions as necessary are inserted. The same applies to instructions that straddle over source lines.

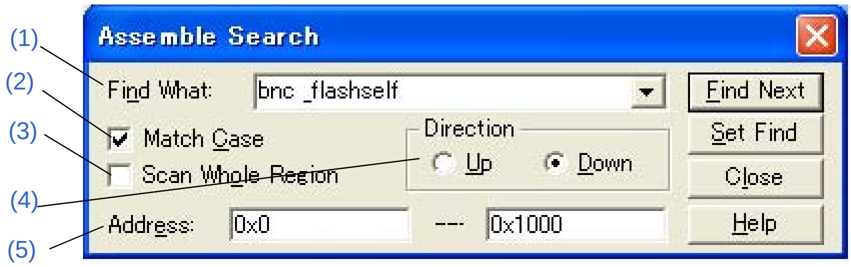
# Assemble Search Dialog Box

This dialog box is used to search the contents in the Assemble Window. (Refer to "5.3.2 Disassemble display".)

Successive character strings included in an input character string and disassembler character string are compared as one blank character.

By setting each item and then clicking the <Find Next> button, searching can be started. By clicking the <Set Find> button, the direction buttons ("<<" and ">>") in the Assemble Window can be used for the search.

Figure 6-23 Assemble Search Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Assemble Window is the current window, select [View] menu -> [Search...], or click the <Search...> button in the same window.

## Explanation of each area

**(1) Find What:**

This area is used to specify the data to be searched. (Up to 256 character.)

In the default condition, the string selected in the window that called this dialog box is displayed. As necessary, the character string displayed can be changed. Up to 16 input histories can be recorded.

**(2) Match Case**

This should be checked to distinguish between uppercase and lowercase.

**(3) Scan Whole Region**

This should be checked to search the entire specified range.

**(4) Direction**

This area is used to specify the direction of the search.

| | |
|---|---|
| Up | Forward search. Searches data forward (upward on screen) from the current position of the cursor. |
| Down | Backward search. Searches data backward (downward on screen) from the current position of the cursor (default). |

**(5) Address:**

This area is used to specify the address to be searched.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)

## Function buttons

| | |
|---|---|
| Find Next | Searches the specified data in accordance with a given condition.<br>If the specified character string is found as a result of a search, it is highlighted.<br>To continue searching, click this button again. |
| Set Find | Sets the specified condition as the search condition and closes this dialog box. |
| Stop (searching) | Stops searching. |
| Close | Closes this dialog box. (During searching, this button is replaced by the <Stop> button.) |
| Help | Displays this dialog box online help files. |

# Address Move Dialog Box

This dialog box is used to specify the start address from which displaying, as follows.

- Memory Window
- Assemble Window
- IOR Window

Figure 6-24 Address Move Dialog Box (Example: When Memory Window Is Open)


- Opening
- Explanation of each area
- Function buttons

## Opening

When the target window is the current window, select [View] menu -> [Move...].

## Explanation of each area

**(1) To:**

This area is used to specify an address.

In the default condition, the string selected in the window that called this dialog box, or the current PC value etc. is displayed. As necessary, the character string displayed can be changed.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".) Up to 16 input histories can be recorded.

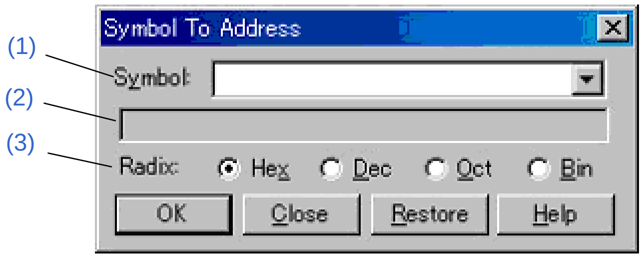
## Function buttons

| | |
|---|---|
| OK | The corresponding window is displayed from the address. |
| Cancel | Closes this dialog box. |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Symbol To Address Dialog Box

This dialog box is used to display the address of the specified variable or function, or the value of the specified symbol. (Refer to "5.3.4 Convert symbol (symbol to address)".)

Figure 6-25 Symbol To Address Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [View] menu -> [Symbol...].

## Explanation of each area

**(1) Symbol:**

This area is used to specify the variable, function name, symbol name, or line number to be converted. (Refer to "Table 5-6 Specifying Symbols".)

The default radix for inputting a numeric value is decimal. Up to 16 input histories can be recorded.

To change the contents of this area, click the <OK> button. The conversion result will be displayed in (2) Conversion result display area.

**(2) Conversion result display area**

If bit symbol have been specified, they are converted to the Address.bit format. Also, equations that include bit symbols cannot be specified.

The variable, address of the function, value of the symbol, address of the line number, or value of the expression specified in (1) Symbol: is displayed. The address value of an I/O port name or peripheral I/O registers name, the register contents of a register name, or flag value of a PSW flag name is displayed.

**(3) Radix:**

This area is used to select the radix of the data to be displayed in (2) Conversion result display area.

| | |
|---|---|
| Hex | Hexadecimal number (default) |
| Dec | Decimal number |
| Oct | Octal number |
| Bin | Binary number |

## Function buttons

| | |
|---|---|
| OK | If the contents of (1) Symbol: have been changed, converts the symbol. After conversion, closes the dialog box if the contents of (1) Symbol: have not been changed. |
| Close | Closes this dialog box. |
| Restore | Restores the input data to the original status.<br>If the <OK> button has already been clicked, the data is restored to the status immediately after the <OK> button was clicked. |
| Help | Displays this dialog box online help files. |

# Watch Window

This window is used to display and change specified watch data. (Refer to "5.6 Watch Function".)

This window can also display wide-ranging watch data (such as global variables and public symbols) in real time even during program execution, in the same way as the Memory Window.

The results of updating and rewriting data in this window will be reflected in the Memory Window.

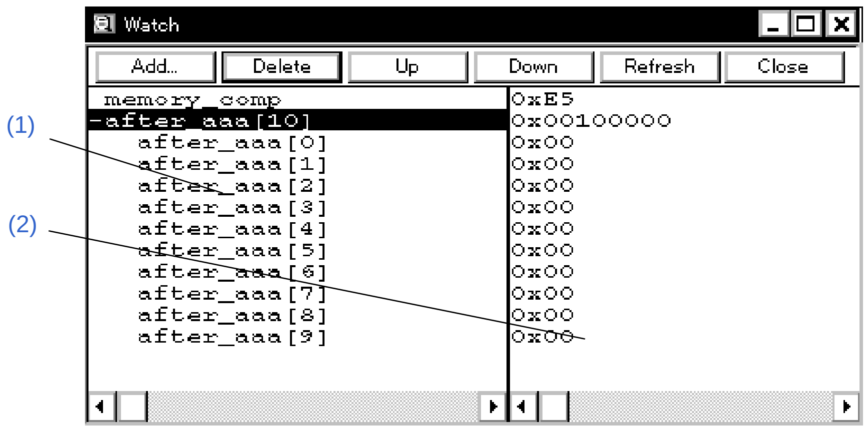
Watch data is registered by clicking the <Watch...> button in the Source Window or Assemble Window. (Refer to "5.6.3 Registering and deleting watch data".)

This window allows easy setting of breakpoints to variables via a Context menu.

**Remarks1:** If a local variable and a global variable exist with the same name, the local variable takes priority.

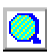
**Remarks2:** A maximum of 10,000 lines can be displayed in the Watch Window.

Figure 6-26 Watch Window




- Opening
- Explanation of each area
- [View] menu (Watch Window-dedicated items)
- Context menu
- Function buttons

## Opening

 Click the **Wch** button, or select [Browse] menu -> [Watch].

## Explanation of each area

### (1) Symbol name display area

This area is used to display variable names, symbol names and types, and tag names of structures or unions.

'+' is prefixed to the displayed arrays, pointer variables, and structures or unions. These variables are expanded and displayed as follows when they are double-clicked:

Table 6-10 Watch Window Display Format (Symbol)

<table>
<tr><th>First character</th><th colspan="2">Meaning</th></tr>
<tr><td rowspan="4">+</td><td colspan="2">Array, pointer variable, or structure/union<br>Expanded display is performed by double-clicking "+" (first character changes from "+" to "-").</td></tr>
<tr><td>Array</td><td>By double-clicking the "+", all the elements of the variable are displayed in accordance with the type of the array variable.</td></tr>
<tr><td>Pointer variable</td><td>By double-clicking the "+", the data indicated by the pointer is displayed.</td></tr>
<tr><td>Structure/union</td><td>By double-clicking the "+", all the members of the structure/union are displayed in accordance with the type of the member variable.<br>If a structure or union is defined in the structure or union, the structure name or union name of the internal structure or union is also displayed.<br>The internal structure or union can be also expanded by using "+".</td></tr>
<tr><td>-</td><td colspan="2">Expanded display variable<br>Expanded display is canceled by double-clicking "-" (first character changes from "-" to "+").</td></tr>
</table>

**Remark:** If an array has too many variables and takes too long to expand, a warning message is displayed.

Registered watch data changes are performed in the Change Watch Dialog Box opened by selecting the item to be changed and then selecting Context menu -> [Change Watch...]. A line with an expanded hierarchy, such as the elements of an array, and members of structures and unions cannot be deleted.

If an access breakpoint is set for a variable or a symbol in the Watch Window, the symbol name display area is highlighted in gold.

### (2) Data value display/setting area

This area is used to display and change watch data values. A value is updated when execution is stopped.

To save a value, select [File ] menu -> [Save As...]. This area is blank if getting data has failed.

Values are changed through direct input. The location to be changed is displayed in red and the contents of the change are written into the target memory when the Enter key is pressed. The previous value can be canceled by the ESC key.

The display format is as follows:

Table 6-11 Watch Window Display Format (Data)

| Display Data | Contents |
|---|---|
| Integer | Hexadecimal (**0x***xxxx*)<br>Decimal (*xxxx*)<br>Octal (**0***xxxx*)<br>Binary (**0b***xxxx*) |
| Character | "Character" |
| Enumeration type | Member name |
| If scope is specified | Displayed in accordance with specified scope. |
| Floating-point type | Single precision/double precision supported<br>The input/display format is as follows:<br>[ + \| - ] inf<br>[ + \| - ] nan<br>[ + \| - ] integer e [ + \| - ]exponent<br>[ + \| - ] integer.fraction[ e [ + \| - ]exponent |
| "?" | Data that has been invalidated because of a change in the scope or optimized compiling |

**Remark:** The radix of a data value can be changed on the Context menu for each variable. The display format of "integer" can be changed on the Debugger Option Dialog Box.

## [View] menu (Watch Window-dedicated items)

When this window is the current window, The following items are added on [View] menu.

Only the selected item is subject to this manipulation.

| | | |
|---|---|---|
| Create Break Event | | Creates a break event by using the selected item as follows. |
| | Beak when Access to this Variable | Creates a break event that can be accessed for read/write. |
| | Break when Write to this Variable | Creates a break event that can be accessed for write. |
| | Break when Read from this Variable | Creates a break event that can be accessed for read. |
| | Clear | Deletes a break event corresponding to the selected item. |
| Event? | | Displays the event information of the variable selected. If an event is set, the Event Dialog Box is opened. |
| Bin | | Displays binary numbers. |
| Oct | | Displays octal numbers. |
| Dec | | Displays decimal numbers. |
| Hex | | Displays hexadecimal numbers. |
| String | | Displays character strings. |

| | |
|---|---|
| Proper | Displays the default value of each variable.<br>Symbols are displayed in accordance with the setting of the Debugger Option Dialog Box (default). |
| Byte | Displays in 8-bit units. |
| Half Word | Displays in 16-bit units. |
| Word | Displays in 32-bit units. |
| Adaptive | Displays the default value of each variable (default).<br>Only this item is valid for a symbol in C language.<br>Symbols in assembly language are displayed in accordance with the setting of the Debugger Option Dialog Box. |
| Up | Moves up one line. |
| Down | Moves down one line. |
| Compulsion Read | Forcibly reads peripheral I/O registers that are disabled from being read because their values will be changed, or the data of the I/O ports and I/O protect area added in the Add I/O Port Dialog Box. |

## Context menu

The menu items are effective for the selected line or item, not the position where the mouse pointer was clicked (same operation as when selecting the main menu with the same name).

| | |
|---|---|
| Beak when Access to this Variable | Creates a break event that can be accessed for read/write by using the selected item. |
| Break when Write to this Variable | Creates a break event that can be accessed for write by using the selected item. |
| Break when Read from this Variable | Creates a break event that can be accessed for read by using the selected item. |
| Clear | Deletes a break event corresponding to the selected item. |
| RRM Setting... | Sets the sampling range of the RRM function.<br>Opens the RRM Setting Dialog Box (when IECUBE is connected). |
| Event? | Displays the event information of the variable selected. If an event is set, the Event Dialog Box is opened. |
| Change Watch... | Changes the selected watch data.<br>Opens the Change Watch Dialog Box. |
| Delete Watch | Deletes the selected watch data from the window. |
| Bin | Displays the selected line in binary numbers. |
| Oct | Displays the selected line in octal numbers. |
| Dec | Displays the selected line in decimal numbers. |
| Hex | Displays the selected line in hexadecimal numbers. |
| String | Displays the selected line as a character string. |
| Proper | Displays the selected line as the default value of each variable.<br>Symbols are displayed in accordance with the setting of the Debugger Option Dialog Box (default). |
| Byte | Displays the selected line in 8-bit units. |

| | |
|---|---|
| Half Word | Displays the selected line in 16-bit units. |
| Word | Displays the selected line in 32-bit units. |
| Adaptive | Displays the selected line as the default value of each variable (default).<br>Only this item is valid for a symbol in C language.<br>Symbols in assembly language are displayed in accordance with the setting of the Debugger Option Dialog Box. |
| Up | Moves the selected line one line up. |
| Down | Moves the selected line one line down. |

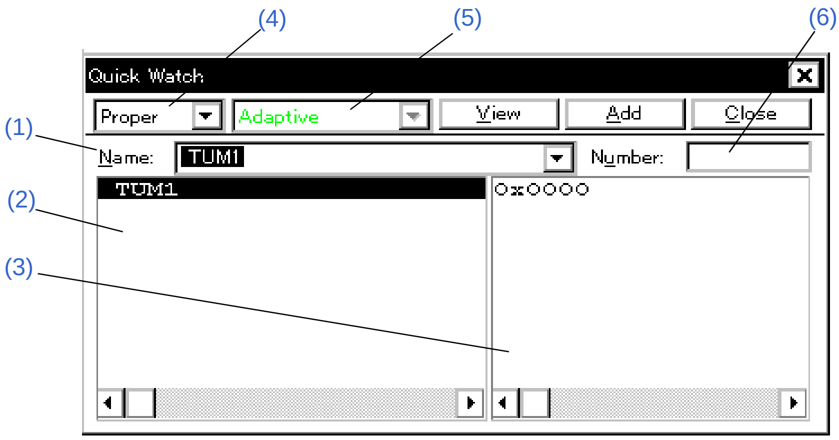
## Function buttons

| | |
|---|---|
| Add... | Opens the Add Watch Dialog Box.<br>If watch data is specified and the <Add...> button is clicked in the Add Watch Dialog Box, the specified watch data is added to the Watch Window. |
| Delete | Deletes the selected watch data from the window. |
| Up | Moves the selected line one line up. |
| Down | Moves the selected line one line down. |
| Refresh | Updates the contents of this window with the latest watch data. |
| Close | Closes this window. |

# Quick Watch Dialog Box

This dialog box is used to temporarily display or change specified watch data. (Refer to "5.6 Watch Function".)

Figure 6-27 Quick Watch Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Source Window or Assemble Window is the current window, select [View] menu -> [Quick Watch...], or click the <Quick...> button in same window.

## Explanation of each area

**(1) Name:**

This area is used to specify the watch data to be displayed.

In the default condition, the string selected in the window that called this dialog box is displayed. As necessary, the character string displayed can be changed. Up to 16 input histories can be recorded.

If the contents of this area have been changed, the data specified can be displayed in the field below by clicking the <View> button.

**(2) Symbol name display area**

This area is used to display watch data (variable names, symbol names and types, and tag names of structures or unions). (Refer to "(1) Symbol name display area" in the Watch Window.)

This area cannot be edited.

**(3) Data value display/setting area**

This area is used to display and change data values. (Refer to "(2) Data value display/setting area" in the Watch Window.)

**(4) Display radix selection area**

This area is used to select the display radix.

| | |
|---|---|
| Proper | Variable: Displays the default value of each variable.<br>Symbol: Displays data with the radix set in the Debugger Option Dialog Box. |
| Hex | Displays in hexadecimal numbers. |
| Dec | Displays in decimal numbers. |
| Oct | Displays in octal numbers. |
| Bin | Displays in binary numbers. |
| String | Displays as a character string. |

**(5) Display size selection area**

This area is used to select the display size. If the display size is fixed, such as when a variable in C language or register is to be displayed, it cannot be changed.

| | |
|---|---|
| Adaptive | Variable: Displays the default value of each variable.<br>Symbol: Displays data with the size set in the Debugger Option Dialog Box. |
| Byte | Displays in 8-bit units. |
| Half Word | Displays in 16-bit units. |
| Word | Displays in 32-bit units. |

**(6) Number:**

This area is used to specify the number of data to be displayed (blank or a number of 1 to 256).

If this area is blank, data is displayed as a simple variable. If a number of 1 or more is specified, data is displayed as an array variable in the Watch Window.

If an array variable is displayed, "+" is prefixed to the data. By double-clicking this "+", all the elements of the data are expanded and displayed in accordance with the type of the data ( "-" is prefixed to the expanded data. If this "-" is double-clicked, the expanded display is canceled).

If the number of data to be displayed is fixed, such as when a variable in C language or register is to be displayed, the specified number of data is invalid.

## Function buttons

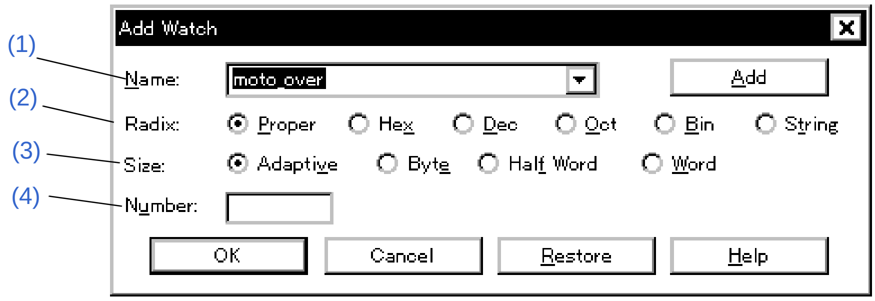
| | |
|---|---|
| View | Displays the data specified in (1) Name: in the field below. |
| Add | Adds the data specified in (1) Name: to the Watch Window. |
| Close | Closes this dialog box.<br>Data that has not actually been written to the target memory will be canceled. |

# Add Watch Dialog Box

This dialog box is used to register watch data to be displayed in the Watch Window. (Refer to "5.6 Watch Function".)

Multiple data with the same symbol name can be registered.

Figure 6-28 Add Watch Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [View] menu -> [Add Watch...], or click the <Add...> button in the Watch Window.

## Explanation of each area

**(1) Name:**

This area is used to specify symbol to be added to the Watch Window.

In the default condition, the string selected in the window that called this dialog box is displayed. As necessary, the character string displayed can be changed. This area is blank if no character string is selected. Up to 16 input histories can be recorded.

The input format is as follows:

Table 6-12 Watch Window Input Format

| | | |
|---|---|---|
| - Variable Name of C language | | |
| | Variable expression : Variable Name | |
| | \|Variable expression [Constant value \| Variable Name] | Elements of array |
| | \|Variable expression . Member name | Entity members of structure/union |
| | \|Variable expression -> Member name | Members of structure/union indicated by pointer |
| | \|*Variable expression | Value of pointer variable |
| | \|&Variable expression | Address where variable is located |
| - Register name | | |
| - peripheral I/O registers name, peripheral I/O registers bit name | | |
| - Label and address of immediate value | | |
| - Register name.bit | | |
| - peripheral I/O registers name. bit | | |
| - Label name.bit , address of immediate value.bit | | |
| - Specification of scope | | |

How a variable is handled when a scope Is specified is as follows:

Table 6-13 How a Variable Is Handled When a Scope Is Specified

| Scope Specification | Program nNme | File Name | Function Name | Variable Name |
|---|---|---|---|---|
| **prog$file#func#var** | prog | file | func | var |
| **prog$file#var** | prog | file | global | var |
| **prog$func#var** | prog | global | func | var |
| **prog$var** | prog | global | global | var |
| **file#func#var** | current | file | func | var |
| **file#var** | current | file | global | var |
| **func#var** | current | current | func | var |
| **var** | current | current | current | var |

**(2) Radix:**

This area is used to select the display radix. (Refer to "(4) Display radix selection area" in the Quick Watch Dialog Box.)

**(3) Size:**

This area is used to select the display size. (Refer to "(5) Display size selection area" in the Quick Watch Dialog Box.)

**(4) Number:**

This area is used to specify the number of data to be displayed. (Refer to "(6) Number:" in the Quick Watch Dialog Box.)

## Function buttons

| | |
|---|---|
| Add | Adds the specified data to the Watch Window.<br>The dialog box remains open. |
| OK | Adds the specified data to the Watch Window.<br>Closes this dialog box. |
| Cancel | Closes this dialog box. |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Change Watch Dialog Box

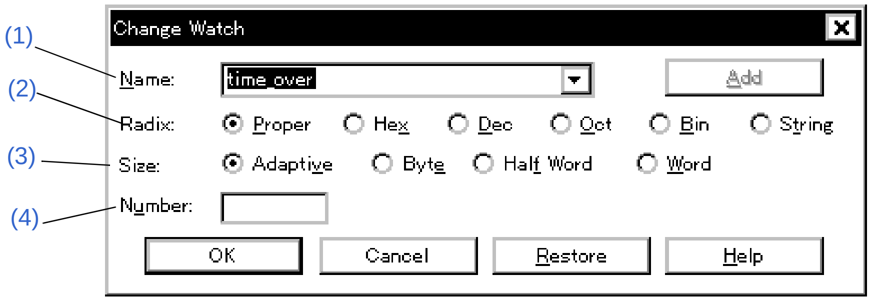
This window is used to change the data on a line selected in the Watch Window. (Refer to "5.6 Watch Function".)

A line with an open hierarchy, such as the elements of an array, and members of structures and unions cannot be changed.

When watch data is changed, the contents of the selected line are replaced with the new data.

The symbol name can be changed even if it results in duplication of a name already in use with existing data.

Figure 6-29 Change Watch Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Watch Window is the current window, select [View] menu -> [Change Watch...].

## Explanation of each area

**(1) Name:**

This area is used to change a symbol name on a line selected in the Watch Window. (Refer to "(1) Name:" in the Add Watch Dialog Box.)

**(2) Radix:**

This area is used to change the display radix on a line selected in the Watch Window. (Refer to "(4) Display radix selection area" in the Quick Watch Dialog Box.)

**(3) Size:**

This area is used to change the display size on a line selected in the Watch Window. (Refer to "(5) Display size selection area" in the Quick Watch Dialog Box.)

**(4) Number:**

This area is used to change the number of data to be displayed on a line selected in the Watch Window. (Refer to "(6) Number:" in the Quick Watch Dialog Box.)

## Function buttons

| | |
|---|---|
| Add | Cannot be selected. |
| OK | Replaces the data on a line selected in the Watch Window with the specified data, and then closes this dialog box. |
| Cancel | Closes this dialog box. |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

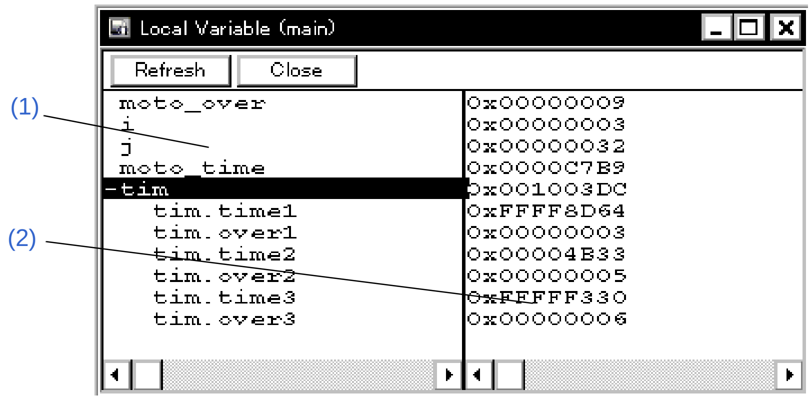
# Local Variable Window

This window is used to display the local variable in the current function and change the local variable values. (Refer to "5.6 Watch Function".)

It is linked with the Jump function of the Stack Window, and displays the local variable in the function jumped when jumping to the Source Window.

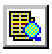
A number of other operations using Context menu, Function buttons, etc., can be performed in this window.

Figure 6-30 Local Variable Window




- Opening
- Explanation of each area
- Context menu
- Function buttons

## Opening

Click the **Loc** button, or select [Browse] menu -> [Local Variable].

## Explanation of each area

**(1) Local variable name display area**

This area displays local variable name. (Refer to "(1) Symbol name display area" in the Watch Window.)

Auto, Internal Static, and Register variables can be displayed. This area cannot be edited.

**(2) Local variable value display/setting area**

This area is used to display and change local variable values. (Refer to "(2) Data value display/setting area" in the Watch Window.)

## [View] menu (Local Variable Window-dedicated items)

When this window is the current window, the following items are added on [View] menu.

| | |
|---|---|
| Bin | Displays binary numbers. |
| Oct | Displays octal numbers. |
| Dec | Displays decimal numbers. |
| Hex | Displays octal numbers. |
| String | Displays character strings. |
| Proper | Displays the default value of each variable (default). |

## Context menu

The menu items are effective for the selected line or item, not the position where the mouse pointer was clicked (same operation as when selecting the main menu with the same name).

| | |
|---|---|
| Add Watch... | Opens the Add Watch Dialog Box. |
| Bin | Displays the selected line in binary numbers. |
| Oct | Displays the selected line in octal numbers. |
| Dec | Displays the selected line in decimal numbers. |
| Hex | Displays the selected line in hexadecimal numbers. |
| String | Displays the selected line as a character string. |
| Proper | Displays the selected line as the default value of each variable.<br>Symbols are displayed in accordance with the setting of the Debugger Option Dialog Box (default). |

## Function buttons

| | |
|---|---|
| Refresh | Updates the contents of this window with the latest watch data. |
| Close | Closes this window. |

# Stack Window

This window is used to display or change the current stack contents of the user program. (Refer to "5.6.7 Stack trace display function".)

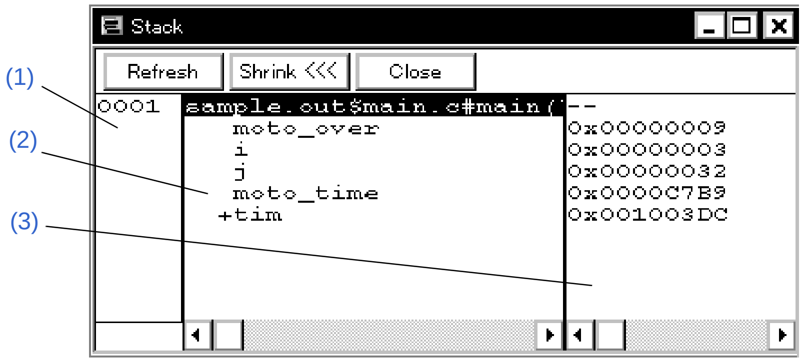
The window corresponding to the stack contents can be jumped to using the Jump function.

A number of other operations using Context menu, Function buttons, etc., can be performed in this window.

**Caution:** The stack trace display function may not operate correctly if there is a function that does not create a stack frame.

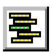
**Remark:** [ERROR] may be displayed during prologue or epilogue processing of a function.

Figure 6-31 Stack Window




- Opening
- Explanation of each area
- [View] menu (Stack Window-dedicated items)
- Context menu
- Function buttons

## Opening

Click the **Stk** button, or select [Browse] menu -> [Stack Trace].

## Explanation of each area

**(1) Stack frame number display area**

This area assigns numbers to and displays the stack contents.

A stack frame number is a natural number starting from 1. The shallower the nesting of the stack, the higher the number. This means that a function having stack number one higher than that of a certain function is the function that calls the certain function.

**(2) Stack frame contents display area**

This area displays the stack frame contents.

It displays function names or local variable names. Note, however, that this area cannot be edited.

(a) If the stack contents consist of a function

They are displayed as follows:

**[program name$file name#function name (argument list) #line number]**

If this line is double-clicked, the operation will be the same as jumping to the Source Window of the Jump function (i.e., the local variable in the function to which execution has jumped will be displayed in the Local Variable Window).If the function has a local variable, the local variable will be displayed on the next and subsequent lines.

(b) If the stack contents consist of a local variable

Its type and name are displayed. (Refer to "Table 6-10 Watch Window Display Format (Symbol)".)

Note that the internal Static and Register variables are not displayed.

**(3) Stack contents display/setting area**

This area is used to display or change the stack contents.

(a) If the stack contents are a function

"--" is displayed and the function cannot be changed.

(b) If the stack contents are a local variable

The variable value is displayed. (Refer to "Table 6-11 Watch Window Display Format (Data)".)

## [View] menu (Stack Window-dedicated items)

When this window is the current window, The following items are added on [View] menu.

| | |
|---|---|
| Bin | Displays binary numbers. |
| Oct | Displays octal numbers. |
| Dec | Displays decimal numbers. |
| Hex | Displays octal numbers. |
| String | Displays character strings. |
| Proper | Displays the default value of each variable (default). |

## Context menu

| | |
|---|---|
| Bin | Displays the selected line in binary numbers. |
| Oct | Displays the selected line in octal numbers. |
| Dec | Displays the selected line in decimal numbers. |
| Hex | Displays the selected line in hexadecimal numbers. |
| String | Displays the selected line as a character string. |
| Proper | Displays the selected line as the default value of each variable.<br>Symbols are displayed in accordance with the setting of the Debugger Option Dialog Box (default). |
| Source Text | Displays the corresponding source text and source line from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>If no line information exists at the jump destination address, however, you cannot jump.<br>Opens the Source Window.<br>If an active Source Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Assemble | Disassembles and displays starting from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>Opens the Assemble Window.<br>If an active Assemble Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Memory | Displays the memory contents starting from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>Opens the Memory Window.<br>If an active Memory Window is open, that window is displayed in the forefront (so that it can be manipulated). |

## Function buttons

| | |
|---|---|
| Refresh | Updates the contents of this window with the latest watch data. |
| Shrink <<< | Collapses the local variable list of the selected function. |
| Expand >>><br>(when the <Shrink<<<> button is clicked) | Displays the local variable list of the selected function. |
| Close | Closes this window. |

# Memory Window

This window is used to display and change the memory contents. (Refer to "5.7 Memory Manipulation Function".) Other operations using Context menu, Function buttons, etc., can be performed in this window.

Moreover, there are two statuses, Active status and static status, for this window. When the window is in the active status, it has the Trace Result with Linking Window (when IECUBE is connected), Jump function. (Refer to "5.16 Functions Common to Each Window".)

**Remarks1:** The memory access status (read, write, read & write) can be displayed using different colors (refer to "5.7.3 Access monitor function (when IECUBE is connected)").

**Remarks2:** The display start position when the this window is opened is as follows:

First time: Display starts from the first address of the RAM area.

Second and subsequent times: Display starts from the address at which an active status window was closed. (if an active status window has never been closed, display starts from the first display start position).

Figure 6-32 Memory Window

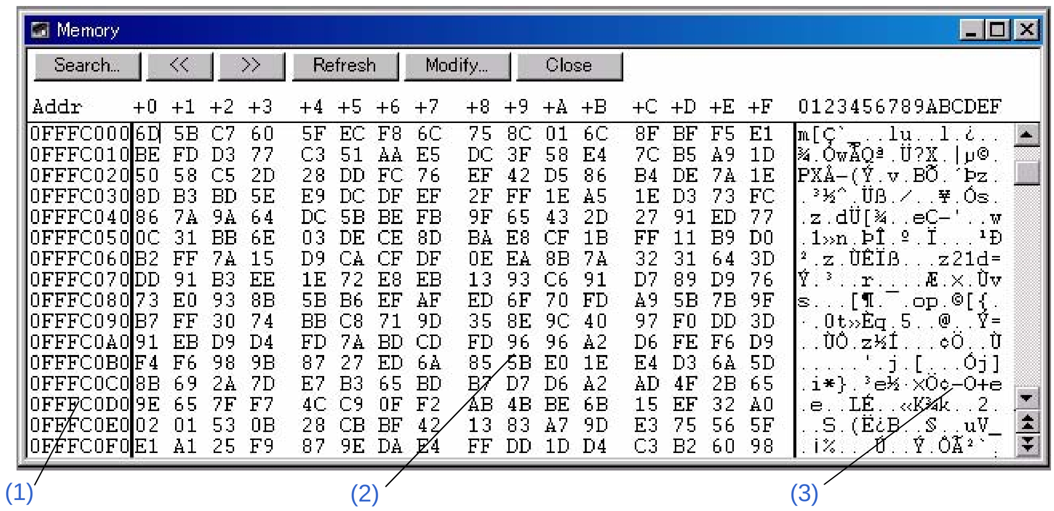
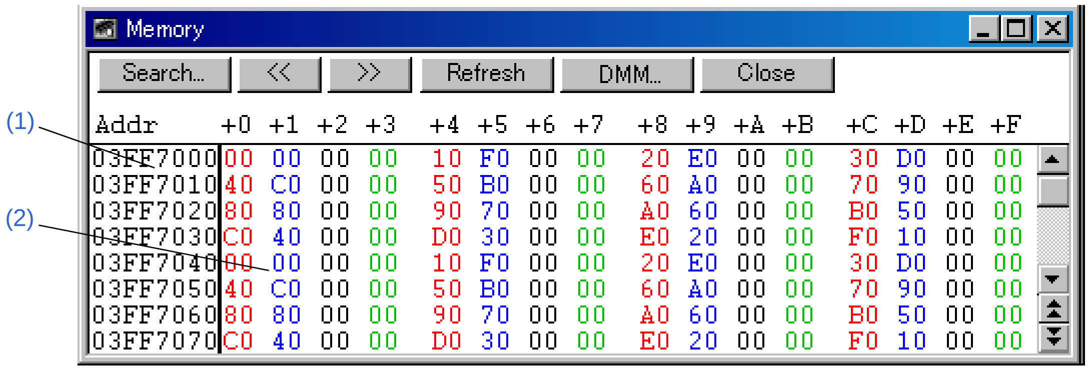



Figure 6-33 Memory Window (When RRM Function Is Selected)



- Opening
- Explanation of each area
- [View] menu (Memory Window-dedicated items)
- Context menu
- Function buttons

## Opening

Click the **Mem** button, or select [Browse] menu -> [Memory].

## Explanation of each area

**(1) Addr**

This area displays memory addresses.

**(2) +0 +1 +2....**

This area is used to display and change memory contents, and to display the access status (refer to "5.7.3 Access monitor function (when IECUBE is connected)"

| | |
|---|---|
| Black | No access |
| Green | Read |
| Red | Write |
| Blue | Read & write |

Memory contents are changed through direct input. The location to be changed is displayed in red and the contents of the contents of the change are written into the target memory when the Enter key is pressed. The previous value can be canceled by the ESC key. Up to 256 bytes can be specified at one time.

**Remark:** To change the memory contents during user program execution, open the DMM Dialog Box by clicking the <DMM...> button.

**(3) 0 1 2 3....**

This area is used to display and change the memory contents in ASCII characters.
This area is displayed when [View] menu -> [Ascii] is selected.
Data can be changed in this area in the same manner as in the memory display area.
The changing method is the same as in (2) +0 +1 +2.....

**Remark:** When the display address is changed, the position of the cursor in the ASCII display area is not synchronized.

## [View] menu (Memory Window-dedicated items)

The following items are added in the [View] menu , when the Memory Window is active.

| | | |
|---|---|---|
| Bin | | Displays binary numbers. |
| Oct | | Displays octal numbers. |
| Dec | | Displays decimal numbers. |
| Hex | | Displays hexadecimal numbers (default). |
| Nibble | | Displays in 4-bit units. |
| Byte | | Displays in 8-bit units (default). |
| Half Word | | Displays in 16-bit units. |
| Word | | Displays in 32-bit units. |
| Ascii | | Selects whether ASCII characters are displayed or not.<br>Checked: Displayed<br>Not checked: No display (default) |
| Little Endian | | Displays in little endian (default). |
| Big Endian | | Displays in big endian. |
| Access Monitoring | | This area is used to set about Access monitor function (when IECUBE is connected). |
| | Clear | Clears the display color through the access monitor function. |
| | Accumulative | Enables/disables cumulative display of access status (memory content change).<br>Checked : Cumulative display of memory contents changes<br>Not checked : Display of only memory contents changes from previous update. |

## Context menu

The menu items are effective for the selected line or item, not the position where the mouse pointer was clicked (same operation as when selecting the main menu with the same name).

| | |
|---|---|
| Move... | Moves the display position.<br>Opens the Address Move Dialog Box. |
| RRM Setting... | Opens the RRM Setting Dialog Box. |
| Bin | Displays binary numbers. |
| Oct | Displays octal numbers. |
| Dec | Displays decimal numbers. |
| Hex | Displays hexadecimal numbers (default). |
| Nibble | Displays in 4-bit units. |
| Byte | Displays in 8-bit units (default). |
| Half Word | Displays in 16-bit units. |
| Word | Displays in 32-bit units. |

| | |
|---|---|
| Ascii | Selects whether ASCII characters are displayed or not.<br>Checked: Displayed<br>Not checked: No display (default) |
| Clear Access Monitor | Clears the display color through the access monitor function (when IECUBE is connected). |
| Accumulative | Enables/disables cumulative display of access status (memory content change) (when IECUBE is connected).<br>Checked : Cumulative display of memory contents changes<br>Not checked : Display of only memory contents changes from previous update |

## Function buttons

| | |
|---|---|
| Search... | Opens the Memory Search Dialog Box and searches for character strings from the displayed memory contents, or memory contents.<br>Selected data (a memory value) is displayed in the Memory Search Dialog Box as data to be searched.If the Memory Search Dialog Box is opened without data specified, specify data from the keyboard.<br>The results of the search is highlighted in the Memory window. |
| << | Searches the memory contents satisfying the search condition set in the Memory Search Dialog Box, forward (upward on screen) from the address at the cursor position.<br>This button is displayed as the <Stop> button during a search. |
| >> | Searches the memory contents satisfying the search condition set in the Memory Search Dialog Box, backward (downward on screen) from the address at the cursor position.<br>This button is displayed as the <Stop> button during a search. |
| Stop(searching) | Stops searching. |
| Refresh | Updates the contents of the window with the latest data. |
| DMM... | Opens the DMM Dialog Box. |
| Close | Closes this window. |

# Memory Search Dialog Box

This dialog box is used to search the memory contents of the part of the Memory Window at which the cursor is located. (Refer to "5.7 Memory Manipulation Function".)

If the cursor is placed in (2) +0 +1 +2.... in the Memory Window, the specified data is treated as a binary data string, and if the cursor is placed in (3) 0 1 2 3...., the specified data is treated as an ASCII character string, and the contents of these respective areas are searched.

By setting each item and then clicking the <Find Next> button, searching can be started. By clicking the <Set Find> button, the direction buttons ("<<" and ">>") in the Memory Window can be used for the search.

**Caution:** Non-mapped, peripheral I/O registers, and I/O protect areas are not searched.

Figure 6-34 Memory Search Dialog Box


- Opening
- Explanation of each area
- Function buttons

## Opening

When the Memory Window is the current window, select [View] menu -> [Search...], or click the <Search...> button in the same window.

## Explanation of each area

**(1) Find What:**

This area is used to specify the data to be searched.

In the default condition, the string selected in the window that called this dialog box is displayed. As necessary, the character string displayed can be changed. Up to 16 input histories can be recorded.

(a) When searching in (2) +0 +1 +2....

Up to 16 data items can be specified. Delimit each data with a "blank character".

(b) When searching in (3) 0 1 2 3....

Up to 256 characters can be specified. A "blank character" in the data is treated as a blank character.

**(2) Unit:**

This area is used to specify the number of bits of the data to be searched in (2) +0 +1 +2.....

| | |
|---|---|
| Byte | Searches the data as 8-bit data (default). |
| Half Word | Searches the data as 16-bit data. |
| Word | Searches the data as 32-bit data. |

**(3) Scan Whole Region**

This should be checked to search the entire specified range.

**(4) Direction**

This area is used to specify the direction of the search.

| | |
|---|---|
| Up | Forward search. Searches data forward (upward on screen) from the current position of the cursor. |
| Down | Backward search. Searches data backward (downward on screen) from the current position of the cursor (default). |

**(5) Address:**

This area is used to specify the address to be searched.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)

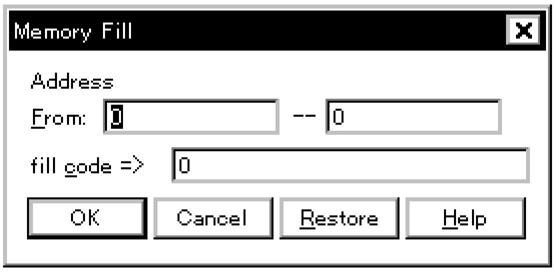
## Function buttons

| | |
|---|---|
| Find Next | Searches the specified data in accordance with a given condition.<br>If the specified character string is found as a result of a search, it is highlighted.<br>To continue searching, click this button again. |
| Set Find | Sets the specified condition as the search condition and closes this dialog box. |
| Stop (searching) | Stops searching. |
| Close | Closes this dialog box.(During searching, this button is replaced by the <Stop> button.) |
| Help | Displays this dialog box online help files. |

# Memory Fill Dialog Box

This dialog box is used to fill the memory contents in the Memory Window with specified codes (fill code). (Refer to "5.7 Memory Manipulation Function".)

Figure 6-35 Memory Fill Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Edit] menu -> [Memory] -> [Fill...].

## Explanation of each area

### (1) Address

This area is used to specify the filling range and fill code.

| | |
|---|---|
| From: | Specifies the filling range (start address -- end address).<br>The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".) |
| Fill code => | Specify the data (fill code) used when filling the range specified in "From:".<br>Up to 16 binary data strings (byte data strings) can be specified. Delimit each data with a "blank character". |

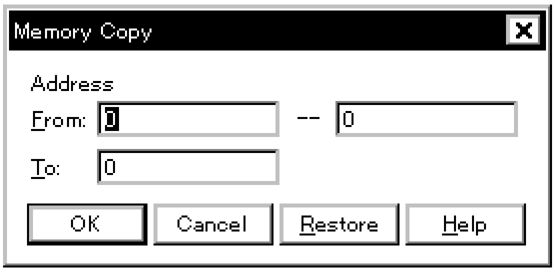
## Function buttons

| | |
|---|---|
| OK | Fills the specified data in accordance with a given condition. |
| Stop (filling) | Stops filling. |
| Cancel | Closes this dialog box. (During filling, this button is replaced by the <Stop> button.) |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Memory Copy Dialog Box

This dialog box is used to copy the memory contents in the Memory Window. (Refer to "5.7 Memory Manipulation Function".)

Figure 6-36 Memory Copy Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Edit] menu -> [Memory] -> [Copy...].

## Explanation of each area

### (1) Address

This area is used to specify the copy source and copy destination addresses.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)

| | |
|---|---|
| From: | Specify the address range (start address -- end address) of the copy source. |
| To: | Specify start address of the copy destination. |

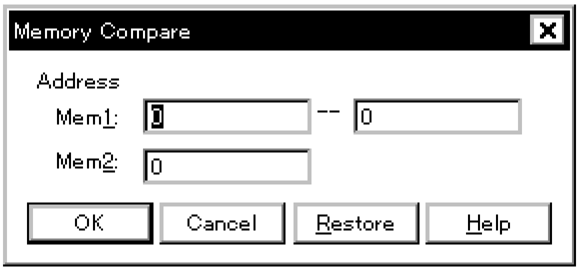
## Function buttons

| | |
|---|---|
| OK | Copies the memory contents in accordance with a given condition. |
| Stop (copying) | Stops copying. |
| Cancel | Closes this dialog box.(During copying, this button is replaced by the <Stop> button.) |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Memory Compare Dialog Box

This dialog box is used to compare the memory contents in the Memory Window. (Refer to "5.7 Memory Manipulation Function".)

Figure 6-37 Memory Compare Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Edit] menu -> [Memory] -> [Compare...].

## Explanation of each area

**(1) Address**

This area is used to specify the comparison source address and comparison destination address.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)

| | |
|---|---|
| Mem1: | Specify the address range (start address -- end address) of the comparison source. |
| Mem2: | Specify the start address of the comparison destination. |

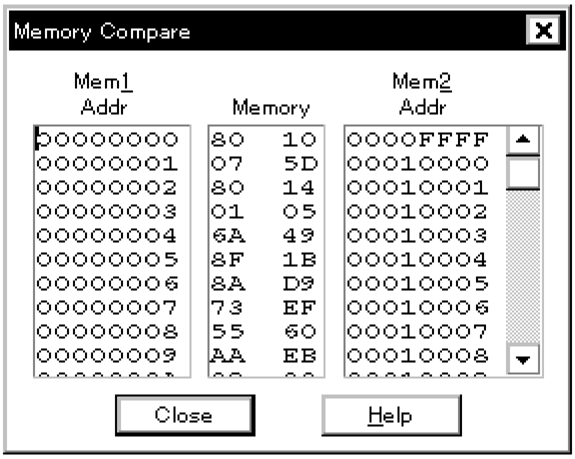
## Function buttons

| | |
|---|---|
| OK | Compares the memory contents in accordance with a given condition. If no difference is found as a result of comparison, " Wf200: No difference encountered." is displayed. If a difference is found, the Memory Compare Result Dialog Box is opened. |
| Stop (comparison) | Stops memory comparison. |
| Cancel | Closes this dialog box.(During comparison, this button is replaced by the <Stop> button.) |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Memory Compare Result Dialog Box

This dialog box is displayed if any difference is found in the memory contents when the memory has been compared in the Memory Compare Dialog Box. (Refer to "5.7 Memory Manipulation Function".)

Figure 6-38 Memory Compare Result Dialog Box




- Explanation of each area
- Function buttons

## Explanation of each area

**(1) Comparison result display area**

This area displays the results of comparing the memory. Only differences that have been found as a result of comparison are displayed.

| | |
|---|---|
| Mem1 Addr | Displays a comparison source address in which a difference has been found. |
| Memory | Displays the data in which a difference has been found.<br>(Left: Comparison source data, Right: Comparison destination data). |
| Mem2 Addr | Displays the comparison destination address at which a difference has been found. |

## Function buttons

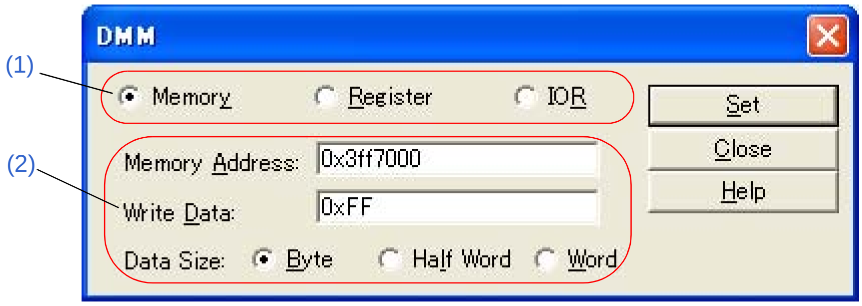
| | |
|---|---|
| Close | Closes this dialog box. |
| Help | Displays this dialog box online help files. |

# DMM Dialog Box

This dialog box is used to set addresses and data for DMM (Dynamic Memory Modification). (Refer to "5.14 DMM Function".) The memory contents are rewritten via the DMM function in real time during user program execution.

**Caution:** The pseudo DMM is enabled when IECUBE is connected, a break occurs instantaneously upon a write during the user program execution.

Figure 6-39 DMM Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Edit] menu -> [DMM...], or click the <DMM...> button in the Memory Window / Register Window / IOR Window.

## Explanation of each area

**(1) DMM target selection area**

This area is used to select the target for DMM. The items displayed in (2) DMM setting area change by selecting the option button.

| | |
|---|---|
| Memory | Memory |
| Register | Register |
| IOR | Peripheral I/O register |

**Remark:** If the DMM Dialog Box is opened via the Memory Window, Register Window, or IOR Window, the corresponding option button has already been selected.

**(2) DMM setting area**

The items displayed vary as follows, depending on the selection in (1) DMM target selection area.

(a) When Memory is selected

| | |
|---|---|
| Memory Address: | This area is used to specify the memory address to which data is to be written.<br>The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".) |
| Write Data: | This area is used to specify the data to be written to the memory address specified in "Memory Address:". |
| Data Size: | This area is used to specify the size of the data specified in "Write Data:" to be written.<br>Byte:Writes the data as 8-bit data.<br>Half Word:Writes the data as 16-bit data.<br>Word: Writes the data as 32-bit data. |

(b) When Register is selected

| | |
|---|---|
| Register Name: | This area is used to specify the register name to which data is to be written. The case is distinguished. Both functional and absolute names can be used for specification. |
| Write Data: | This area is used to specify the data to be written to the Register specified in "Register Name:". |

(c) When IOR is selected

| | |
|---|---|
| IOR Name: | This area is used to specify the IOR name to which data is to be written. The case is distinguished. The read-only SFRs cannot be specified. |
| Write Data: | This area is used to specify the data to be written to the register specified in "IOR Name:". |

## Function buttons

| | |
|---|---|
| Set | Writes the data in accordance with a given condition. |
| Close | Closes this dialog box. |
| Help | Displays this dialog box online help files. |

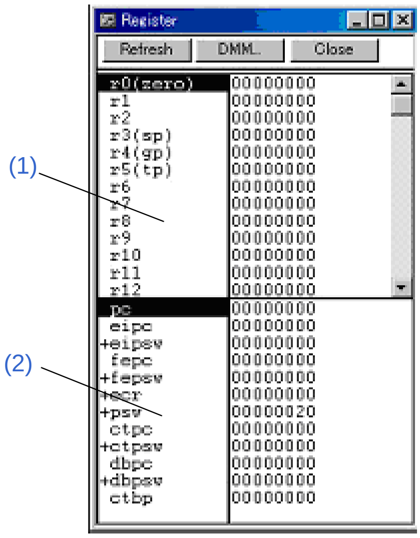
# Register Window

This window is used to display and change registers (program register/system register). (Refer to "5.8 Register Manipulation Function".)

Other operations using Context menu, Function buttons, etc., can be performed in this window.

Each area in this window are the jump pointer of the Jump function.

Figure 6-40 Register Window




- Opening
- Explanation of each area
- [View] menu (Register Window-dedicated items)
- Context menu
- Function buttons

## Opening

Click the **Reg** button, or select [Browse] menu -> [Register].

## Explanation of each area

### (1) The program register display area

This area is used to display and change the program register.

Register values are changed through direct input. The location to be changed is displayed in red and the contents of the contents of the change are written into the target memory when the Enter key is pressed. The previous value can be canceled by the ESC key.

**Caution:** When overflow of a register occurs due to an illegal value entered by the user, the register will be updated with a value of 0xFFFFFFFF.

### (2) The system register display area

This area is used to display and change the system register.

By double-clicking "+", flag name and flag value are displayed (first character changes from "+" to "-"). Expanded display is canceled by double-clicking "-" (first character changes from "-" to "+").

Register values are changed through direct input. The location to be changed is displayed in red and the contents of the contents of the change are written into the target memory when the Enter key is pressed. The previous value can be canceled by the ESC key.

**Caution:** When overflow of a register occurs due to an illegal value entered by the user, the register will be updated with a value of 0xFFFFFFFF.

## [View] menu (Register Window-dedicated items)

The following items are added in the [View] menu , when the Register Window is active.

| | |
|---|---|
| Bin | Displays binary numbers. |
| Oct | Displays octal numbers. |
| Dec | Displays decimal numbers. |
| Hex | Displays hexadecimal numbers (default). |
| Pick Up | Displays only the registers selected in the Register Select Dialog Box. |
| Select... | Opens the Register Select Dialog Box. |

## Context menu

| | |
|---|---|
| Add Watch... | Registers a selected character string to the Watch window.<br>Opens the Add Watch Dialog Box. |
| Bin | Displays the selected line in binary numbers. |
| Oct | Displays the selected line in binary numbers. |
| Dec | Displays the selected line in decimal numbers. |
| Hex | Displays the selected line in hexadecimal numbers (default). |
| Pick Up | Displays only the registers selected in the Register Select Dialog Box. |
| Select... | Selects the register to be displayed<br>Opens the Register Select Dialog Box. |

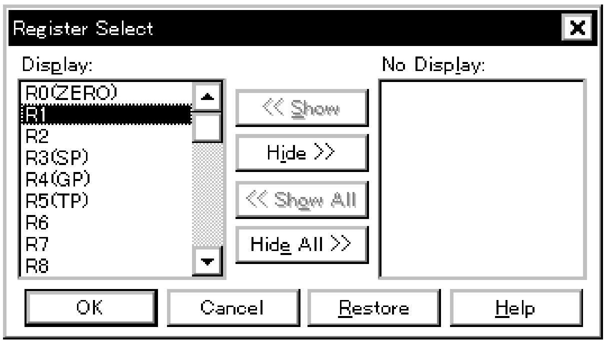
## Function buttons

| | |
|---|---|
| Refresh | Updates the contents of the window with the latest data. |
| DMM... | Opens the DMM Dialog Box. |
| Close | Closes this window. |

# Register Select Dialog Box

This dialog box is used to select registers that are not displayed in the Register Window. (Refer to "5.8 Register Manipulation Function".)

Figure 6-41 Register Select Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Register Window is the current window, select [View] menu -> [Select...].

## Explanation of each area

**(1) Display:, No Display:**

This area is used to select registers that are displayed in the Register Window, and those that are not.

(a) Display:

Registers displayed in the Register Window.

(b) No Display:

Registers not displayed in the Register Window.

(c) Button

The following buttons are used to change register to be displayed in the Register Window.

| | |
|---|---|
| << Show | Moves the register selected from the (b) No Display: list to (a) Display:. |
| Hide >> | Moves the register selected from the (a) Display: list to (b) No Display:. |
| << Show All | Moves all registers to (a) Display:. |
| Hide All >> | Moves all registers to (b) No Display:. |

Two or more registers can be selected by clicking any of the above buttons while holding down the Ctrl or Shift key.

## Function buttons

| | |
|---|---|
| OK | Reflects the selection in this dialog box in the Register Window and closes this dialog box. |
| Cancel | Cancels the changes and closes this dialog box. |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# IOR Window

This window is used to display and change the contents of peripheral I/O registers and the I/O ports that have been registered in the Add I/O Port Dialog Box. (Refer to "5.8 Register Manipulation Function".)

A number of other operations using Context menu, Function buttons, etc., can be performed in this window.

**Cautions1:** However, that the values of read-only peripheral I/O registers and I/O ports cannot be changed. In addition, the peripheral I/O registers and I/O ports that cause the device to operate when they are read are read-protected and therefore cannot be read. To read these registers, select a register, and select and execute [Compulsion Read] from the Context menu.

**Cautions2:** During user program execution, the IOR contents are updated at every sampling time set in the Extended Option Dialog Box (refer to "5.13 RRM Function"). However, updating is not performed when the RAM monitor function is OFF.

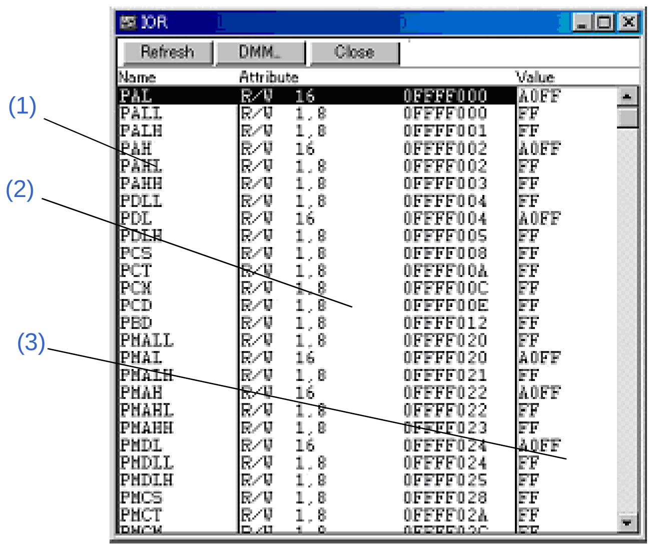
**Remarks1:** The display start position when the window is opened is as follows.

First time: Display from peripheral I/O registers of minimum address

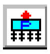
Second and subsequent times: Display from first peripheral I/O registers when window was last closed

**Remarks2:** If the device supports programmable I/O registers and a programmable I/O area has been set in the Configuration Dialog Box, the programmable I/O registers and expansion peripheral I/O registers are also displayed. If the value of an I/O port address is defined, the I/O port name is displayed in light color.

Figure 6-42 IOR Window



## Opening

Click the **IOR** button, or select [Browse] menu -> [IOR].

## Explanation of each area

**(1) Name**

This area displays the names of peripheral I/O registers and I/O ports.

If the value of an I/O port address is not defined, the I/O port name displayed in light color.

**(2) Attribute**

This area displays the attributes of peripheral I/O registers and I/O ports.

This area displays the read/write attributes, access types, and displays and absolute addresses from the left side. When the bit peripheral I/O registers is displayed, bit-offset value is also displayed.

It can be specified whether this area is displayed or not, by selecting [View] menu -> [Attribute].

| Read/Write Attribute | |
|---|---|
| R | Read only |
| W | Write only |
| R/W | Read/write |
| * | Register that is read via an emulation register to prevent the device from operating when this register is read.<br>To read this attribute directly from a peripheral I/O registers, execute [View] menu -> [Compulsion Read].<br>Even a write-only peripheral I/O registers can also be read via an emulation register.<br>However, some devices do not support this function. |
| Access Type | |
| 1 | Can be accessed in Bit units. |
| 8 | Can be accessed in Byte units. |
| 16 | Can be accessed in Half Word units. |
| 32 | Can be accessed in Word units. |

**(3) Value**

This area is used to display and change the contents of a peripheral I/O registers and I/O port.

The contents are displayed differently as follows, depending on the attribute:

| | |
|---|---|
| Black Display | Read only or read/write |
| -- | Write only |
| ** | Value changes if read |

Values are changed through direct input. The location to be changed is displayed in red and the contents of the contents of the change are written into the target memory when the Enter key is pressed. The previous value can be canceled by the ESC key.

Note that the values of read-only peripheral I/O registers and I/O ports cannot be changed.

The value of read-protected peripheral I/O registers and I/O ports can be read by selecting Context menu -> [Compulsion Read].

## [View] menu (IOR Window-dedicated items)

When this window is the current window, the following items are added on [View] menu.

| | |
|---|---|
| Bin | Displays binary numbers. |
| Oct | Displays octal numbers. |
| Dec | Displays decimal numbers. |
| Hex | Displays octal numbers (default). |
| Sort By Name | Displays in alphabetical order. |
| Sort By Address | Displays in address order (default). |
| Unsort | Does not sort. |
| Attribute | Switches on/off display of (2) Attribute. |
| Pick Up | Displays only the registers selected in the IOR Select Dialog Box. |
| Select... | Opens the IOR Select Dialog Box. |
| Compulsion Read | Forcibly reads the peripheral I/O registers that are disabled from being read because their values will be changed, or the data of the I/O ports and I/O protect area added in the Add I/O Port Dialog Box. |

## Context menu

| | |
|---|---|
| Move... | Opens the Address Move Dialog Box. |
| Add Watch... | Opens the Add Watch Dialog Box. |
| Add I/O Port... | Opens the Add I/O Port Dialog Box. |
| Bin | Displays binary numbers. |
| Oct | Displays octal numbers. |
| Dec | Displays decimal numbers. |
| Hex | Displays octal numbers (default). |
| Sort By Name | Displays in alphabetical order. |
| Sort By Address | Displays in address order (default). |
| Unsort | Does not sort. |
| Attribute | Switches on/off display of (2) Attribute. |
| Pick Up | Displays only the registers selected in the IOR Select Dialog Box. |
| Select... | Opens the IOR Select Dialog Box. |
| Compulsion Read | Forcibly reads the peripheral I/O registers that are disabled from being read because their values will be changed, or the data of the I/O ports and I/O protect area added in the Add I/O Port Dialog Box. |

## Function buttons

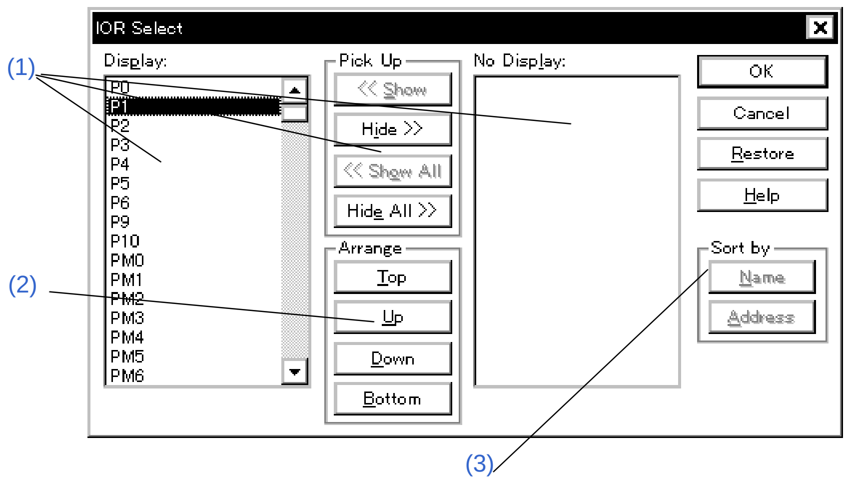
| | |
|---|---|
| Refresh | Updates the contents of this window with the latest watch data. |
| DMM... | Opens the DMM Dialog Box. |
| Close | Closes this window. |

# IOR Select Dialog Box

This dialog box is used to select peripheral I/O registers and I/O ports that are not displayed the IOR Window. (Refer to "5.8 Register Manipulation Function".)

It is also used to specify the sequence in which registers and ports are displayed.

Figure 6-43 IOR Select Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the IOR Window is the current window, select [View] menu -> [Select...].

- Explanation of each area

**(1) Display:, Pick Up, No Display:**

This area is used to select peripheral I/O registers or I/O ports that are displayed in the IOR Window, and those that are not.

(a) Display:

The peripheral I/O registers or I/O ports displayed in the IOR Window.

(b) No Display:

The peripheral I/O registers or I/O ports not displayed in the IOR Window.

(c) Pick Up

The following buttons are used to change peripheral I/O registers or I/O ports to be displayed in the IOR Window.

Two or more registers can be moved by clicking any of the above buttons while holding down the Ctrl or Shift key.

| | |
|---|---|
| << Show | Moves peripheral I/O registers or I/O ports selected from (b) No Display: list to (a) Display:. |
| Hide >> | Moves peripheral I/O registers or I/O ports selected from (a) Display: list to (b) No Display:. |
| << Show All | Moves all peripheral I/O registers or I/O ports to (a) Display:. |
| Hide All >> | Moves all peripheral I/O registers or I/O ports to (b) No Display:. |

**(2) Arrange**

The following buttons are used to change the display sequence in (a) Display:.

If the display arrangement is changed, multiple lines cannot be selected.  Select one line at a time.

| | |
|---|---|
| Top | Moves the selected peripheral I/O registers or I/O port to the top of the list. |
| Up | Moves the selected peripheral I/O registers or I/O port one line up. |
| Down | Moves the selected peripheral I/O registers or I/O port one line down. |
| Bottom | Moves the selected peripheral I/O registers or I/O port to the bottom of the list. |

**(3) Sort by**

The following buttons are used to change the display sequence in (b) No Display:.

| | |
|---|---|
| Name | Displays in alphabetical order. |
| Address | Displays in address order. |

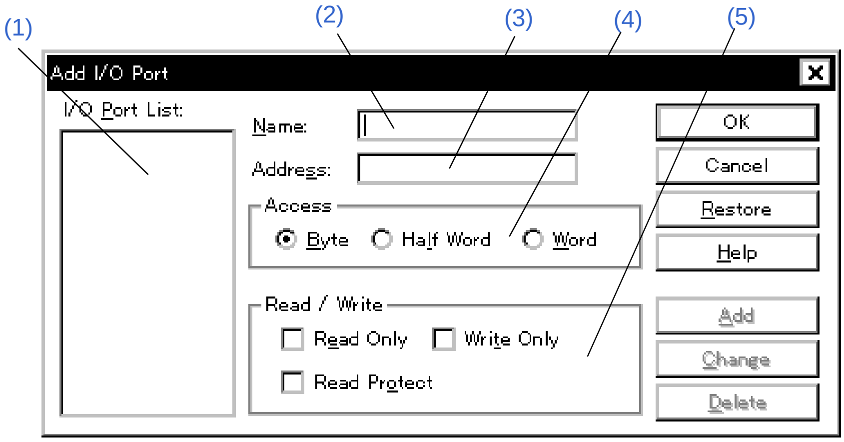
## Function buttons

| | |
|---|---|
| OK | Reflects the selection in this dialog box in the IOR Window and closes this dialog box. |
| Cancel | Cancels the changes and closes this dialog box. |
| Restore | Restores the status before this dialog box was opened. |
| Help | Displays this dialog box online help files. |

# Add I/O Port Dialog Box

This dialog box is used to register an I/O port to be added to the IOR Window. (Refer to "5.8 Register Manipulation Function".)

Figure 6-44 Add I/O Port Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Option] menu -> [Add I/O Port...].

## Explanation of each area

**(1) I/O Port List:**

This area lists the I/O ports currently registered.

If a new I/O port is registered, it is added to this list. An I/O port already registered can be selected and changed or deleted by Function buttons.

**(2) Name:**

This area is used to specify an I/O port name to be added (up to 15 characters long).

**(3) Address:**

This area is used to specify the address of the I/O port to be added.

The default radix for inputting a numeric value is hexadecimal. An address can be also specified by a symbol. (Refer to "Table 5-6 Specifying Symbols".)

The address that can be set in this area is either a Target area address or peripheral I/O registers area address.

### (4) Access

This area is used to select the access size of the I/O port to be added.

| | |
|---|---|
| Byte | 8-bit unit (default) |
| Half Word | 16-bit unit |
| Word | 32-bit unit |

### (5) Read / Write

This area is used to specify the access attribute of the I/O port to be added.

In the default condition, all the attributes are not checked (i.e., the I/O port can be both read and written).

## Function buttons

| | |
|---|---|
| OK | Reflects the results of addition in the IOR Window and closes this dialog box. |
| Cancel | Cancels the changing, closes this dialog box. |
| Restore | Restores the original status. |
| Help | Displays this dialog box online help files. |
| Add | Adds an I/O port of the specified address. |
| Change | Changes the setting of the I/O port selected in (1) I/O Port List:. |
| Delete | Deletes the I/O port selected in (1) I/O Port List:. |

# Timer Dialog Box

IECUBE

This dialog box is used to register and set timer event conditions, and display execution time measurement results. (Refer to "5.12 Event Function" and "5.9 Timer Function (When IECUBE Is Connected)".)

The (5) Execution time display area can be constantly displayed as the Timer Result Dialog Box by clicking the <View Always> button.

Registration and setting of timer event conditions is done by setting each item (256 items max.) in this dialog box and then pressing the <OK> button.  The registered timer event conditions are managed by the Event Manager.

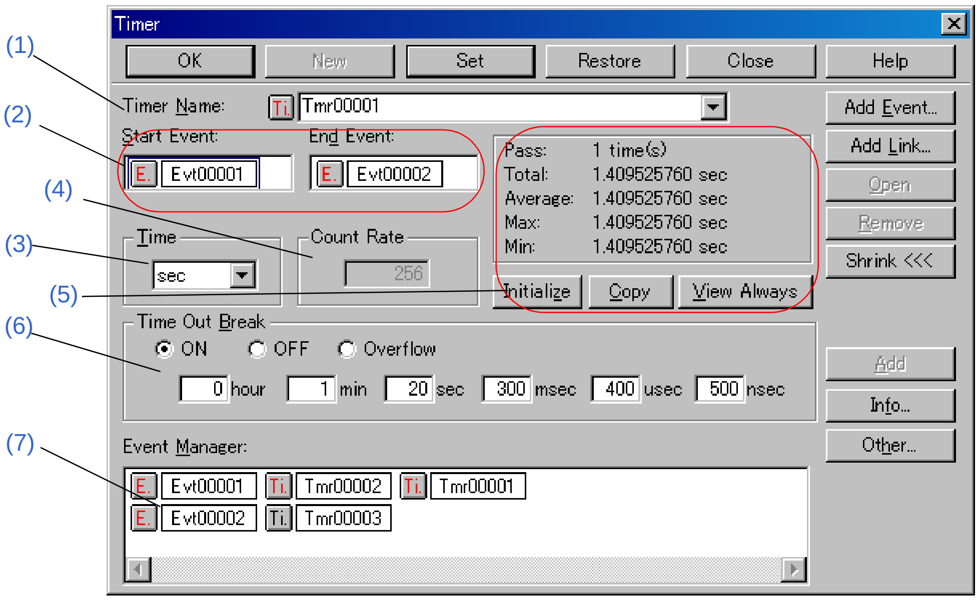
The number of timer event conditions that can be simultaneously used (validated) is limited (refer to "5.12.4 Number of enabled events for each event condition").

The execution time measurement result is displayed when the set timer event condition is selected.

The timer event conditions can be set, deleted, validated, or invalidated even during user program execution.

**Remark:** The measurement result display contents are updated at each sampling time of the RRM Function, even during user program execution.

Figure 6-45 Timer Dialog Box




- Opening
- Explanation of each area
- Cautions
- Function buttons (Related event function)

## Opening

Click the **Tim** button, or select [Event] -> [Timer...] on the menu.

## Explanation of each area

**(1) Timer Name:**

This area is used to set a timer event name.

Directly input an alphanumeric string of up to eight characters as a name.

To display the contents of an already created event condition, select from the drop-down list.

To display from user program execution until break, specify "Run-Break" (refer to "5.9.2 Run-Break event").

The mark on the left of this area indicates the utilization status of events (refer to "Table 5-18 Event Icon"). The gray mark indicates that an event condition is being edited and has not been registered yet.

By clicking the left mark, an event condition can be validated or invalidated.

**(2) Start Event:, End Event:**

This area is used to set an event condition for the timer.

The number of event conditions that can be registered in this area is one for each of the start and end conditions.

Setting of event conditions is easily done by dragging the icon of the event to be set from the event manager area and dropping it in this area. For details, refer to "5.12.3 Setting event conditions".

**(3) Time**

This area is used to select the unit in which the timer measurement result is to be displayed.

| | |
|---|---|
| nsec | Nanoseconds (default) |
| usec | Microseconds |
| msec | Milliseconds |
| sec | Seconds |
| min | Minutes |

**(4) Count Rate**

This area displays timer count rate values (the values set in (2) Timer in the Extended Option Dialog Box). (The timer count rate value at the time of the event creation is displayed when the contents of the existing timer event conditions are displayed. The current timer count rate value is displayed when the event is newly created, or when the contents of the Run-Break event conditions are displayed.)

This area cannot be edited.

When the timer count rate value of the existing timer event conditions needs to be changed, first, change the value of (2) Timer in the Extended Option Dialog Box. Then, reset the timer event conditions. When the timer count rate value at the time of the event creation is different from the value set in (2) Timerin the Extended Option Dialog Box), the timeout time is affected. Therefore, the value in this area is displayed in red (refer to " Cautions").

### (5) Execution time display area

This area displays the result of measuring the execution time of the program (refer to "Table 6-14 Measurable Values").

**Note:** Measurement results that cannot be trusted due to counter overflow are displayed in red.

| | |
|---|---|
| Pass | Number of passes |
| Total | Total execution time in the measurement zone specified by start event and end event conditions |
| Average | Average execution time |
| Max | Maximum execution time |
| Min | Minimum execution time |

Table 6-14 Measurable Values

| Connected IE | Measurable Execution Time | Measurable Execution Count |
|---|---|---|
| IECUBE | Approx. 2.8 minutes max.<br>(External clock = 50 MHz, 1 division, resolution = 20 nsec)<br>Approx. 195.2 hours max.<br>(External clock = 50 MHz, 4K division, resolution = 81920 nsec)<br>**Note:**<br>The measurable execution time differs according to "Table 6-8 The Relationship Between the Timer Count Division Ratio and Maximum Measurement Time (Timer counter (Timer))". | 65535 times max.<br>(16 bit) |
| N-Wire CARD, MINICUBE | Approx. 3 minutes and 30 seconds max. (Resolution = 100 nsec)<br>**Note:**<br>Only during display to the status bar in the Main window | - |

(a) Button

| | |
|---|---|
| Initialize | Clears the measurement results. |
| Copy | Copies the measurement result to the clipboard in text format. |
| View Always | Opens the Timer Result Dialog Box. |

**(6) Time Out Break**

This area is used to set the timeout break for the section measurement time specified in (2) Start Event:, End Event: (time from the establishment of timer start event to the establishment of timer end event).

| | |
|---|---|
| ON | A timeout break occurs (execution is terminated) if the section measurement time exceeds the specified timeout time. |
| OFF | No timeout break occurs (default). |
| Overflow | A timeout break occurs (execution is terminated) if the section measurement time exceeds the maximum measurable time (refer to "Table 6-14 Measurable Values"). |

**Remark:** In the case of a Run-Break event, this area is fixed to the "OFF" position and disabled.

**(7) Event Manager:**

This area is used to display the list of the events registered. (Refer to "Table 5-18 Event Icon", "(4) Manipulation in event manager area".)

## Cautions

For IECUBE, the same timer count rate value is applied to all timer events. Therefore, the time set in (6) Time Out Break changes when the timer count rate at the time of the event creation (the value displayed in "(4) Count Rate" is different from the current timer count rate value (the value set in (2) Timer in the Extended Option Dialog Box).

**Example:**

In the case of a timer event whose timer count rate value at the time of the event creation is set to "16" and whose timeout time is set to "1 sec"

(1) When the current timer count rate value is "32"

->The current timer count rate value is two times higher than the value at the time of the event creation. This leads to the occurrence of a Timer Over Break in 2 seconds.

(2) When the current timer count rate value is "4"

->The current timer count rate value is a quarter of the value at the time of the event creation. This leads to the occurrence of a Timer Over Break in 250 m seconds.

Therefore, do not set the timeout break in the above case. (No problems occur when the timeout break is set to "OFF" or "Overflow.")

To set the timeout break, change the timer count rate value in (2) Timer in the Extended Option Dialog Box, and reset the timer event.

## Function Buttons

Refer to " Function buttons (Related event function)" in the Event Manager.

# Timer Result Dialog Box

IECUBE

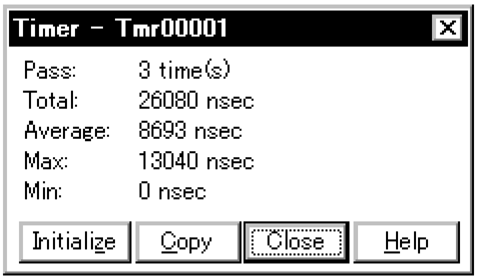
This dialog box displays the results of measuring the execution time. (Refer to "5.9 Timer Function (When IECUBE Is Connected)".)

By clicking the <View Always> button in the Timer Dialog Box, this dialog box is opened corresponding to a timer event condition on a one-to-one basis. Two or more of this dialog box can be simultaneously opened.

Up to 256 + 1 (Run-Break event) Timer Result Dialog Boxes can be opened, the number of events that can be measured at the same time is the number of valid events described in "5.12.4 Number of enabled events for each event condition" + 1 (Run-Break event).

**Remark:** The measurement result display contents are updated at each sampling time of the RRM Function, even during user program execution.

Figure 6-46 Timer Result Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select a timer event condition in the Timer Dialog Box , click the <View Always> button.

## Explanation of each area

**(1) Execution time display area**

Same area is Timer Dialog Box. Refer to "(5) Execution time display area".

## Function buttons

| | |
|---|---|
| Initialize | Clears the measurement results. |
| Copy | Copies the measurement result to the clipboard in text format. |
| Close | Closes this dialog box. |
| Help | Displays this dialog box online help files. |

# Trace View Window

IECUBE

This window used to display the trace results. (Refer to "5.10 Trace Function (When IECUBE Is Connected)".)

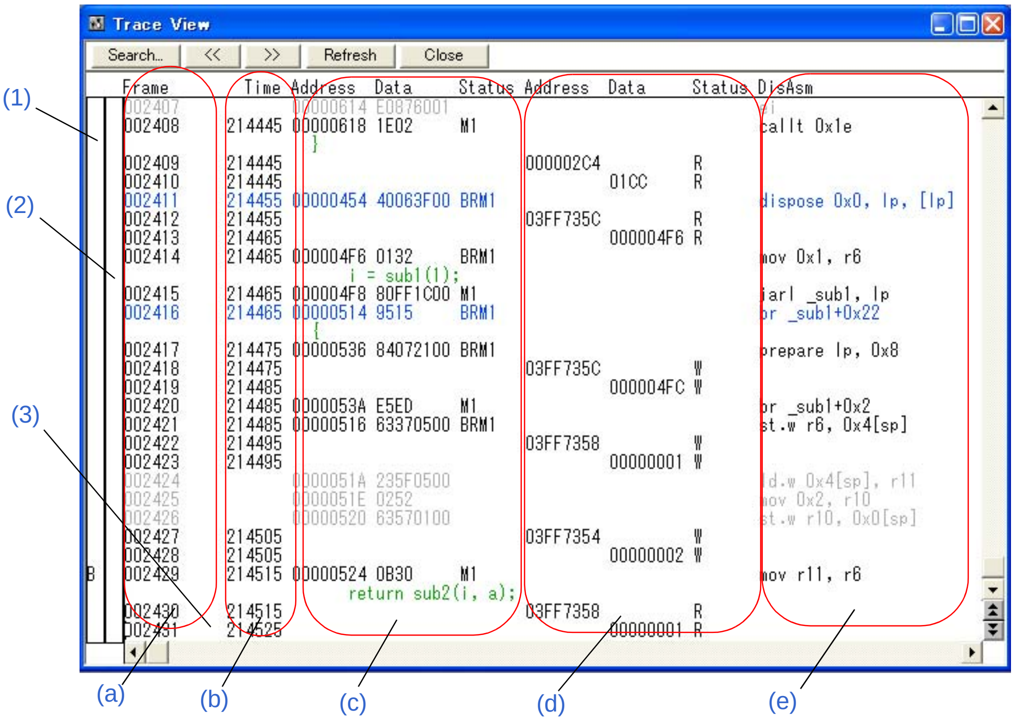
Display updates are performed during breaks or during step execution.

This window has Mixed display mode (Trace View Window).

Also, It has "5.16.3 Trace Result with Linking Window (when IECUBE is connected)".

Context menu, A number of other operations using Function buttons, etc., can be performed in this window.

Figure 6-47 Trace View Window




- Opening
- Explanation of each area
- [View] menu (Trace View Window-dedicated items)
- Context menu
- Function buttons

## Opening

Click the **TrW** button, or select [Browse] -> [Trace] on the menu.

## Explanation of each area

**(1) Point mark display area**

This area displays the Event Setting Status (Event Mark).

If an execution event or access event is set at the corresponding trace address, the mark corresponding to the type of the event is displayed.

The mark displayed is not that during trace but an event mark that is set when the trace result is displayed.

**(2) Trace mode display area**

This area displays the type of tracer mode.

| | |
|---|---|
| A | a start or an end frame (section trace or qualify trace) |
| T | Delay trigger frame |
| M | DMA point access frame (DMA start point and end point) (Refer to "5.10.7 DMA point trace function".) |
| N | Frame for which not all the trace data was fetched |

**(3) Trace result display area**

This area displays the trace results.

Complement frames are displayed in gray. In the case of complement frames, only (a) Frame , (c) Address Data Status (fetch access display) and (e) DisAsm (mnemonic display) are displayed.

Whether each of the following sub-areas is displayed or not can be selected in the Trace Data Select Dialog Box.

(a) Frame

This area displays the trace flame number.

(b) Time

This area displays the number of clocks required for the target chip from the execution start of the program to the execution start of an instruction of each frame or generation of memory access cause.

In complement frames, this area is not displayed.

The display contents can be switched between clock count display and time display in the Trace Data Select Dialog Box.

For the relationship between the timer count division ratio and maximum measurement time, refer to "Table 6-6 The Relationship Between the Time Tag Counter Division Ratio and Maximum Measurement Time (Time tag counter (Trace))"). The timetag and time measurement are performed using an external clock .

**Caution:** If overflow occurs, the time tag maximum value is displayed in red.

(c) Address Data Status (fetch access display)

This area displays the result of fetching the program.

In the case of a complement frame, only (ii) Address Data is displayed.

(i) Status

The following types of statuses are available:

**Program fetch display**

| | |
|---|---|
| BRM1 | Fetching of first byte of first instruction after branch(The frame which is BRM1 and is also M1 is included.)<br>If the fetch address is the start of the symbol, the first line is highlighted in blue.<br>(The frame which is BRM1 and is also M1 is included.) |
| M1 | Fetching of first byte of instruction |
| *Brank* | Any of the following<br>- Data access frame (3 frames max. (2 frames max when RRM function is selected.))<br>- Frame of the second instruction (when two instructions are executed at the same time)<br>- Invalid flame |
| IF | Fetch address that entered interrupt and was canceled (The address where the breakpoint was set and this correspond.) |
| INFO | Trace start (delay trigger trace), traces start/trace end (section trace), Fetch address (qualify trace start/end) |

**Remark:** When a 6-byte or 8-byte instruction code is displayed, the first 4 bytes are displayed in the first frame, and the other bytes are displayed in the second and third frames. If two instructions are executed at the same time, one frame is displayed on two lines. The instruction code at the lower address is displayed on the first line and the instruction code at the higher address is displayed on the second line.

(ii) Address Data

This area displays the address and data.

| | |
|---|---|
| Address | Displays the fetch address |
| Data | Displays the fetch data |

(d) Address Data Status (data access display)

This area displays the result of accessing data.

In complement frames, this area is not displayed.

**Status**

| | |
|---|---|
| R | Data read |
| W | Data write |

(e) DisAsm (mnemonic display)

This area displays the disassemble results.

When (i) Status is only BRM1, M1or complement frames, this area is displayed.

**Remark:** To display instruction codes when two instructions are simultaneously executed, one instruction is displayed on the first line, and the other instruction is displayed on the second line.

## [View] menu (Trace View Window-dedicated items)

The following items are added in the [View] menu , when the Trace View Window is active.

| | | |
|---|---|---|
| Select... | | Selects the contents to be displayed.<br>Opens the Trace Data Select Dialog Box. |
| Mix | | Specifies whether the source file are displayed in mixed display mode, or not displayed.<br>Checked: Mixed display<br>Not checked: No display (default) |
| Window Synchronize | | Links the Trace View Window with the following windows: (Refer to "5.16.3 Trace Result with Linking Window (when IECUBE is connected)".)<br>A checked window is linked. |
| | Source Text | Links the Source Window. |
| | Assemble | Links the Assemble Window. |
| | Memory | Links the Memory Window. |

## Context menu

| | |
|---|---|
| Move... | Moves the display position.<br>Opens the Trace Move Dialog Box. |
| Trace Clear | Clears the trace data. |
| Select... | Selects the contents to be displayed.<br>Opens the Trace Data Select Dialog Box. |
| Mix | Specifies whether the source file are displayed in mixed display mode, or not displayed.<br>Checked: Mixed display<br>Not checked: No display (default) |

| | |
|---|---|
| Window Synchronize | Links the Trace Window with the following windows: (Refer to "5.16.3 Trace Result with Linking Window (when IECUBE is connected)".) |
| Source Text | Links the Source Window. |
| Assemble | Links the Assemble Window. |
| Memory | Links the Memory Window. |
| Source Text | Displays the corresponding source text and source line, using the data value at the cursor position as the jump destination address (refer to "5.16.2 Jump function").<br>If no line information exists at the jump destination address, however, you cannot jump.<br>Opens the Source Window.<br>If the Source Window in active is open, that window is displayed in the forefront (so that it can be manipulated). |
| Assemble | Disassembles and displays starting from the jump destination address specified by the data value at the cursor position (refer to "5.16.2 Jump function").<br>Opens the Assemble Window.<br>If the Assemble Window in active is open, that window is displayed in the forefront (so that it can be manipulated). |
| Memory | Displays the memory contents starting from the jump destination address specified by the data value at the cursor position (refer to "5.16.2 Jump function").<br>Opens the Memory Window.<br>If the Memory Window in active is open, that window is displayed in the forefront (so that it can be manipulated). |

## Function buttons

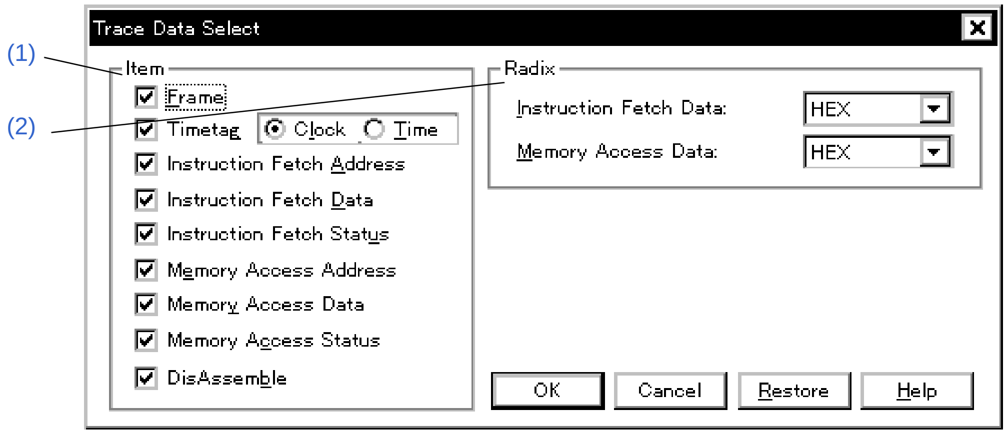
| | |
|---|---|
| Search... | Opens the Trace Search Dialog Box and searches trace results.<br>The searched result will be highlighted in the Trace Window.<br>Same function as [View] menu -> [Search...]. |
| << | Searches forward (upward on screen) for a trace result that satisfies the search condition set in the Trace Search Dialog Box. |
| >> | Searches backward (downward on screen) for a trace result that satisfies the search condition set in the Trace Search Dialog Box. |
| Refresh | Updates the contents of the window with the latest data. |
| Close | Closes this dialog box. |

# Trace Data Select Dialog Box

IECUBE

This dialog box is used to select items to be displayed in the Trace View Window. (Refer to "5.10 Trace Function (When IECUBE Is Connected)".)

Figure 6-48 Trace Data Select Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Trace View Window is the current window, select [View] -> [Select...] menu.

## Explanation of each area

**(1) Item**

This area is used to select items to be displayed in the Trace View Window. Displaying the following items may or may not be selected. The field checked is displayed.

| | |
|---|---|
| Frame | (a) Frame field |
| Timetag | (b) Time field Whether the clock or time is displayed can be selected. |
| Instruction Fetch Address | Address field in (c) Address Data Status (fetch access display) |
| Instruction Fetch Data | Data field in (c) Address Data Status (fetch access display) |
| Instruction Fetch Status | Status field in (c) Address Data Status (fetch access display) |
| Memory Access Address | Address field in (d) Address Data Status (data access display) |
| Memory Access Data | Data field in (d) Address Data Status (data access display) |
| Memory Access Status | Status field in (d) Address Data Status (data access display) |

| | |
|---|---|
| DisAssemble | (e) DisAsm (mnemonic display) field |

**(2) Radix**

This area is used to select the radix in which data is to be displayed. Displaying the following items may or may not be selected.

| | |
|---|---|
| Instruction Fetch Data | Data field in (c) Address Data Status (fetch access display) |
| Memory Access Data | Data field in (d) Address Data Status (data access display) |

| | |
|---|---|
| HEX | Displays hexadecimal numbers. (default) |
| DEC | Displays decimal numbers. |
| OCT | Displays octal numbers. |
| Bin | Displays binary numbers. |

## Function buttons

| | |
|---|---|
| OK | Reflects the results of selection in this dialog box in the Trace View Window. |
| Cancel | Closes this dialog box. |
| Restore | Restores the original status. |
| Help | Displays this dialog box online help files. |

# Trace Search Dialog Box

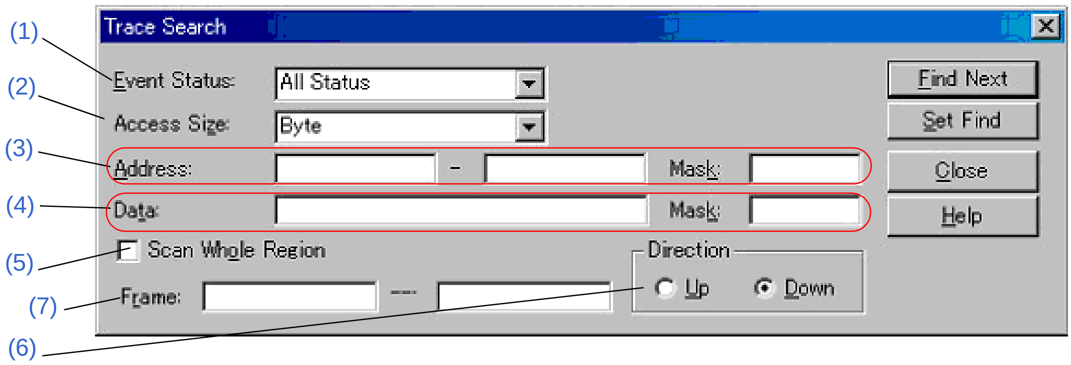
IECUBE

This dialog box is used to search in the Trace View Window. (Refer to "5.10 Trace Function (When IECUBE Is Connected)".)

By setting each item and then clicking the <Find Next> button, searching can be started.

By clicking the <Set Find> button, the direction buttons (<< and >>) in the Trace View Window can be used for the search.

Figure 6-49 Trace Search Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Trace View Window is the current window, select [View] menu -> [Search...], or click the <Search... > button in the same window.

## Explanation of each area

**(1) Event Status:**

This area is used to select a status condition.

If a status condition is omitted, all frames (All status) are searched.

**Remark:** The IF and INFO frames are not searched.

| | |
|---|---|
| All Status | All frames (default) |
| M1 Fetch | M1 fetch (including BRM1) |
| R/W | Data read/write (including R, W) |
| Read | Data read |
| Write | Data write |

**(2) Access Size:**

This area is used to select an access size condition.

By specifying an access size condition, the access width of a data condition to be detected by an access event is determined.

| | |
|---|---|
| Byte | Searches for a data condition with 8-bit width (only during 8-bit access). |
| Half Word | Searches for a data condition with 16-bit width (only during 16-bit access). |
| Word | Searches for a data condition with 32-bit width (only during 32-bit access). |
| No Condition | Does not search based on access size (nothing can be input to the Data area). |
| Bit | Searches for a data condition with 1-bit width (only during 8-bit access).<br>In this case, a search is made for a data condition with 1-bit width. Because of the operation of the simulator, access to a bit is not directly detected; the simulator searches a dummy bit access by internally setting address conditions and data conditions as follows:<br>**Input example)**<br>Address: FE20.1<br>Data: 1<br>\|<br>**Setting of trace search)**<br>Address: fe20<br>Data: 00000010B<br>Mask: 11111101B<br>If another bit of the same address is accessed or if all the 8 bits of the same address are accessed, therefore, an event is detected in accordance with the specified status if the address and bit match the specified value of [address.bit]. |

**Caution:** If an access event is specified as a status condition, the alternative of Bit is not displayed. If Bit or 1 is specified, an error occurs.

**Remark:** If no access size condition is specified, a judgment is automatically made from the address condition and data condition, and the following is set:

-Bit if the address condition is set in bit units

-Byte if the data condition is set in 8-bit units

-Half Word if the data condition is set in 16-bit units

-Word if the address condition is set in 32-bit units

-No Condition if no data condition is specified

**(3) Address:, Mask:(Address setting area)**

This area is used to specify an address condition.

The default radix for inputting a numeric value is hexadecimal. A symbol can be also specified by a symbol or expression (refer to "Table 5-6 Specifying Symbols").

The following can be set:

Table 6-15 Settable Range of Address Condition (Trace)

| Settable Range | Condition |
|---|---|
| 0 <= address value <= 0xFFFFFFFF | None |
| 0 <= mask value <= 0xFFFFFFFF | None |

(a) Address:

Set an address condition (lower address - higher address) (may be omitted).

The following can be set:

(i) Setting as a point

Set a value to only the lower address, or set the same value to the lower address and the higher address.

(ii) Setting as a range

Set a value to only the lower address, or set the same value to the lower address and the higher address.

(iii) Setting as a bit

Set a value to only the lower address, or set the same value to the lower address and the higher address.

Specify a value in the form of "address.bit". Mask cannot be set.

The value of bit, which indicates the bit position, must be 0 <= bit<= 7.

(b) Mask:

Set a mask value for an address value (only when (i) Setting as a point).

Mask may be omitted.

The address value of a bit whose mask value is 1 may be 0 or 1.

**Example1:**

| | |
|---|---|
| Address | 0x4000 to 0x4000 |
| Mask | 0xFF |

With this setting, addresses 0x4000 to 0x40FF satisfy the condition.

**Example2:**

| | |
|---|---|
| Address | 0x4000 to 0x4000 |
| Mask | 0x101 |

With this setting, addresses 0x4000, 0x4001, 0x4100, and 0x4101 satisfy the condition.

**(4) Data:, Mask:(Data setting area)**

This area is used to set data conditions.

The default radix for inputting a numeric value is hexadecimal.

The settable range differs as follows depending on the access size condition specified in (2) Access Size:. (Refer to "Table 6-22 Settable Range of Data Condition".)

(a) Data:

Set a data value as data conditions. A data can be also specified by a symbol (refer to "Table 5-6 Specifying Symbols").

(a) Mask:

Set a mask value for the data value.

When a mask is set, the data value for the bit whose mask value is 1 may be 0 or 1.

**Example1:**

| | |
|---|---|
| Data | 0x4000 |
| Mask | 0xFF |

With this setting, addresses 0x4000 to 0x40FF satisfy the condition.

**Example2:**

| | |
|---|---|
| Data | 0x4000 |
| Mask | 0x101 |

With this setting, addresses 0x4000, 0x4001, 0x4100, and 0x4101 satisfy the condition.

**(5) Scan Whole Region**

This should be checked to search the entire specified range.

**(6) Direction**

This area is used to specify the direction of the search.

| | |
|---|---|
| Up | Forward search.<br>Searches data forward (upward on screen) from the current position of the cursor. |
| Down | Backward search (default).<br>Searches data backward (downward on screen) from the current position of the cursor. |

**(7) Frame:**

This area is used to specify a frame number to be searched.

The default radix for inputting a numeric value is decimal. A symbol can be also specified by Frame Number Specification Format.

## Function buttons

\

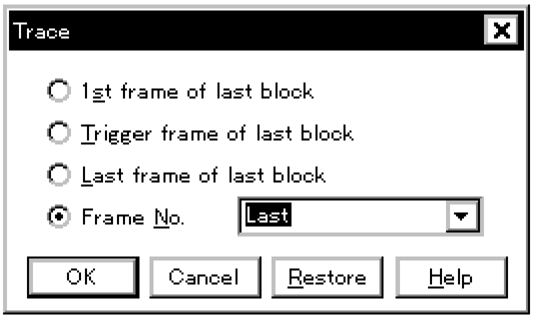
| | |
|---|---|
| Find Next | Searches the specified data in accordance with a given condition.<br>If the specified frame is found as a result of a search, it is highlighted. To continue searching, click this button again. |
| Set Find | Sets the specified condition as the search condition and closes this dialog box. |
| Close | Closes this dialog box. |
| Help | Displays this dialog box online help files. |

# Trace Move Dialog Box

IECUBE

This dialog box is used to specify the position from which displaying the Trace View Window is started. (Refer to "5.10 Trace Function (When IECUBE Is Connected)".)

Figure 6-50 Trace Move Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the Trace View Window is the current window, select [View] menu -> [Move...].

## Explanation of each area

**(1) Frame selection area**

This area is used to specify the frame at the destination.

| | |
|---|---|
| 1st frame of last block | Moves the display start position to the first frame in the newest block frame of trace data.<br>The display start position is moved to the first frame of the trace data when using an in-circuit emulator without a block frame. |
| Trigger frame of last block | Moves the display start position to the trigger frame in the newest block frame of trace data. |
| Last frame of last block | Moves the display start position to the last frame of trace data. |
| Frame No. | Moves the display start position to the specified frame number.<br>In the default condition, the character string selected in the window that called this dialog box or "Last" is selected.<br>The default radix for inputting a numeric value is decimal.<br>If 0 is specified, the display start position is moved to the first frame of trace data.<br>In addition, frame number can also be specified in the following format.<br>Up to 16 input histories can be recorded. |

Table 6-16 Frame Number Specification Format

| Specification | Abbreviation | Contents |
|---|---|---|
| +numeric value | None | Moves backward (downward on screen) the display start position from the frame at the cursor by the specified number of frames (numeric value). |
| -numeric value | None | Moves forward (upward on screen) the display start position from the frame at the cursor by the specified number of frames (numeric value). |
| Top | O | Moves the display start position to the first frame |
| First | S | Same as "1st frame of last block" |
| Trigger | T | Same as "Trigger frame of last block" |
| Last | L | Same as "Last frame of last block" |
| Bottom | B | Moves the display start position to the last frame |

## Function buttons

| | |
|---|---|
| OK | Starts trace display from the specified position. |
| Cancel | Closes this dialog box. |
| Restore | Restores the input data to the original status. |
| Help | Displays this dialog box online help files. |

# Trace Dialog Box

IECUBE

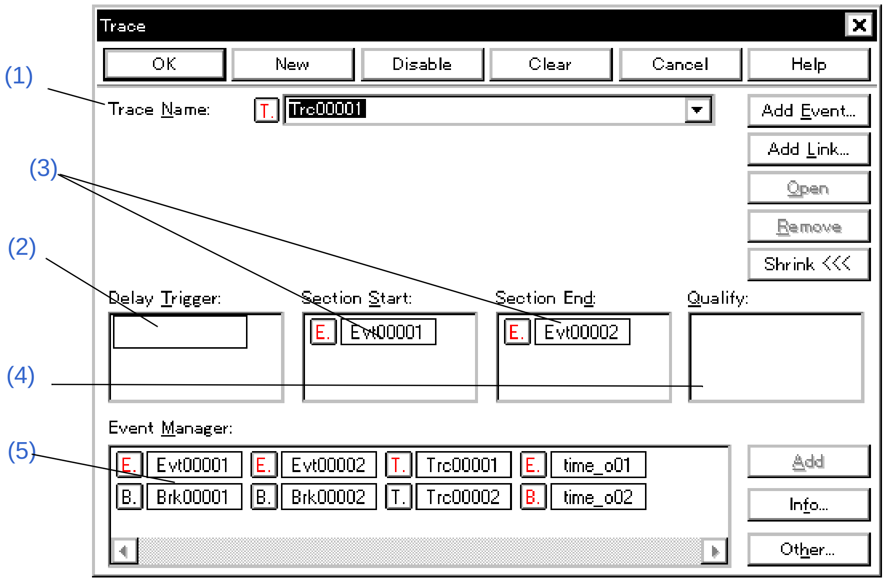
This dialog box is used to register, set, and display trace event conditions. (Refer to "5.12 Event Function", "5.10 Trace Function (When IECUBE Is Connected)".)

The trace event conditions for when performing conditional trace are specified in this dialog box (refer to "Table 5-13 Types of Conditional Trace" ).

Registration and setting of trace event conditions is done by setting each item (256 items max.) in this dialog box and then clicking the <OK> button. The registered trace event conditions are managed by the Event Manager.

The number of trace event conditions that can be simultaneously used (validated) is limited (refer to "5.12.4 Number of enabled events for each event condition").

Figure 6-51 Trace Dialog Box




- Opening
- Explanation of each area
- Function buttons (Related event function)

## Opening

Click the **Trc** button, or select [Event] menu -> [Trace...].

## Explanation of each area

**(1) Trace Name:**

This area is used to set a trace event name.

Directly input an alphanumeric string of up to eight characters as a name.

To display the contents of an already created event condition, select from the drop-down list.

The mark on the left of this area indicates the utilization status of events (refer to "Table 5-18 Event Icon"). The gray mark indicates that an event condition is being edited and has not been registered yet.

By clicking the left mark, an event condition can be validated or invalidated.

**(2) Delay Trigger:**

This area is used to set an event condition for a delay trigger (refer to "5.10.6 Setting conditional trace").

For the number of items that can be set to this area, refer to "Table 6-17 Number of Events Settable".

Event conditions are easily set by dragging the icon of the event to be set from the event manager area and dropping it in this area. For details, refer to "5.12.3 Setting event conditions".

**(3) Section Start:, Section End:**

This area is used to set event conditions for starting and stopping a section trace (refer to "5.10.6 Setting conditional trace").

For the number of items that can be set to this area, refer to "Table 6-17 Number of Events Settable".

Event conditions are easily set by dragging the icon of the event to be set from the event manager area and dropping it in this area. For details, refer to "5.12.3 Setting event conditions".

**(4) Qualify:**

This area is used to set an event condition for a qualify trace (refer to "5.10.6 Setting conditional trace").

If two or more events are set, trace is performed when each event occurs.

The number of event conditions that can be set in this area is as follows:

Table 6-17 Number of Events Settable

| Connected IE | Event Conditions<br>Total (execution/access) | Event Link Conditions |
|---|---|---|
| IECUBE | 14 (8[*a]/6) | 1 |

***a** Can be used to events after execution.

Event conditions are easily set by dragging the icon of the event to be set from the event manager area and dropping it in this area. For details, refer to "5.12.3 Setting event conditions".

**(5) Event Manager:**

This area is used to display the list of the events registered. (Refer to "Table 5-18 Event Icon", "(4) Manipulation in event manager area".)

## Function Buttons

Refer to " Function buttons (Related event function)" in the Event Manager.

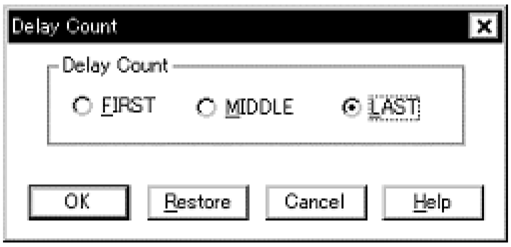
# Delay Count Dialog Box

IECUBE

This dialog box is used to set or display delay count values. (Refer to "5.10 Trace Function (When IECUBE Is Connected)".)

By setting a delay count value, a trace can be executed the number of times specified by the delay count value after the delay trigger event condition set in the Trace Dialog Box has been satisfied. (Refer to "5.10.6 Setting conditional trace".)

Figure 6-52 Delay Count Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Event ] menu -> [Delay Count...].

## Explanation of each area

**(1)Delay Count**

The following items can be selected.

| | |
|---|---|
| FIRST | Places the trigger pointer at the first of the trace data, traces all frames, and then stops the tracer. |
| MIDDLE | Places the trigger pointer at the center of the trace data, traces a half of all frames, and then stops the tracer. |
| LAST | Places the trigger pointer at the end of the trace data and immediately stops the tracer. |

**Caution:** The delay count value differs according to the trace memory size specified in the Extended Option Dialog Box.

Example: When the trace memory size is 256 K frames,

FIRST: 256K - 1frames

MIDDLE: (256K / 2) - 1frames

LAST: 5frames

## Function buttons

| | |
|---|---|
| OK | Validates the settings and closes this dialog box. |
| Restore | Restores the previous settings. |
| Cancel | Closes this dialog box. |
| Help | Displays this dialog box online help files. |

# Code Coverage Window

IECUBE

This dialog box displays the code coverage measurement result (C0 coverage) (refer to "5.11 Coverage Measurement Function (When IECUBE Is Connected)").

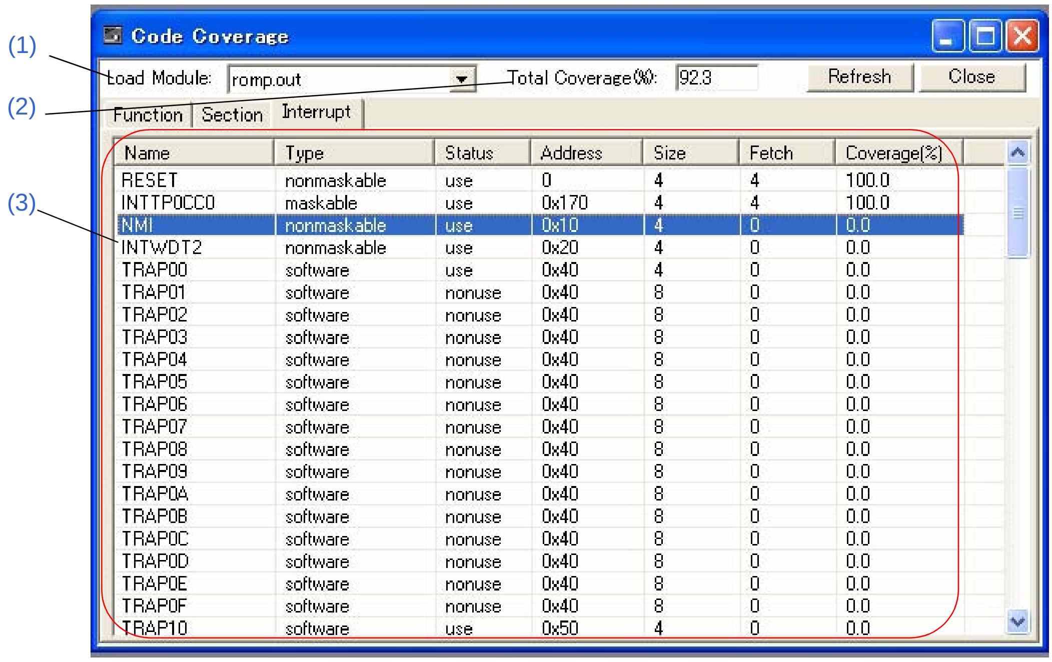
The lines where the user program has been executed or not yet executed can be checked in the Source Window or Assemble Window.

The coverage measurement result is updated at a break (it is not updated automatically during user program execution).

**Caution:** The coverage measurement result is inaccurate if the on-chip flash memory data is replaced via emulation of flash self programming (refer to " Flash Option Dialog Box").

Other operations using Context menu, Function buttons, etc., can be performed in this window.

Figure 6-53 Code Coverage Window




- Opening
- Explanation of each area
- Context menu
- Function buttons

## Opening

Click the **Cov** button, or select [Browse] menu -> [Code Coverage].

## Explanation of each area

**(1) Load Module:**

This area is used to select the load module file that has been downloaded.

This area is blank when no load module file has been downloaded.

**(2) Total Coverage (%):**

This area displays the coverage for the area for which code coverage has been measured.

**Total coverage = Total executed (fetched) function size/total function size**

(excluding sections outside the coverage measurement range)

This area is blank when no load module file has been downloaded.

**(3) Measurement result display area**

This area displays the code coverage per tab (function, section, interrupt handler).

This area is blank when no load module file has been downloaded.

**Remark:** The displayed items are sorted by clicking the title (on the label) in each column (ascending/descending order is switched each time the title is clicked).

(a) [Function] tab

| | |
|---|---|
| Name | Function name<br>(displayed as function in file units in case of assembler source file) |
| File | Name of file in which the function is defined |
| Address | Function start address |
| Size | Function size (unit: bytes) |
| Fetch | Number of bytes executed (fetched) |
| Coverage (%) | Coverage of the function (0 - 100%)<br>---- : When the function is outside the coverage measurement range |

(b) [Section] tab

| | |
|---|---|
| Name | Section name |
| Type | Section type (code, data) |
| Address | Section start address |
| Size | Section size (unit: bytes) |
| Fetch | Number of bytes executed (fetched) |
| Coverage (%) | Coverage of the section (0 - 100%)<br>---- : When the section is outside the coverage measurement range |

(c) [Interrupt] tab

| | |
|---|---|
| Name | Interrupt request name |
| Type | Interrupt type<br>(nonmaskable, maskable, software, security id, flash mask option) |
| Status | Utilization status in the program<br>---- : Unknown |
| Address | Starting address of the interrupt handler |
| Size | Size of the interrupt handler (unit: bytes)<br>Maximum size for statuses other than "use" |
| Fetch | Number of fetched bytes |
| Coverage (%) | Coverage of the interrupt handler (0 - 100%)<br>---- : When the interrupt handler is outside the coverage measurement range |

The display jumps from this tab to the Source Window or Assemble Window using the start address value of the selected line as a jump pointer. The jump destination window will be displayed from the jump pointer.

The jump function is executed by selecting a jump source line then selecting [Source Text/Assemble] in the [Jump] menu. Jump can also be performed by double-clicking the jump source line.

## Context menu

| | |
|---|---|
| Source Text | Displays the corresponding source text and source line, using the data value at the cursor position as the jump destination address. (Refer to "5.16.2 Jump function".)<br>If no line information exists at the jump destination address, however, you cannot jump.<br>Opens the Source Window.<br>If an active Source Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Assemble | Disassembles and displays starting from the jump destination address specified by the data value at the cursor position. (Refer to "5.16.2 Jump function".)<br>Opens the Assemble Window.<br>If an active Assemble Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Clear | Clears the coverage measurement results. |
| Select... | Selects the coverage measurement range as a space of 1 MB or more.<br>Opens the Coverage-Address Dialog Box. |

## Function buttons

| | |
|---|---|
| Refresh | Cannot be selected. |
| Close | Closes this window. |

# Coverage-Address Dialog Box

IECUBE

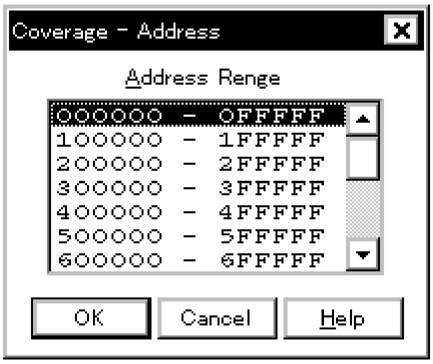
This dialog box is used to select the coverage measurement range to be displayed in the Code Coverage Window. (Refer to "5.11 Coverage Measurement Function (When IECUBE Is Connected)".)

The measurable ranges are given below. In this dialog box, any 1 MB range can be selected.

Table 6-18 Coverage Measurement Range (Detail)

| Connected IE | Code Coverage Measurement Range |
|---|---|
| IECUBE | - Any 1 MB space of addresses 0x100000 to 0x3FFFFFF<br>(Default:0x3FF0000 to 0x3FFFFFF)<br>- 1 MB space of addresses 0x000000 to 0x0FFFFF(fixed measurement areas) |

Figure 6-54 Coverage-Address dialog box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Option] menu -> [Coverage] -> [Select...]

## Explanation of each area

**(1) Address Range**

This is an area for selecting any 1 MB space that performs coverage measurement.

Changing the measurement range clears the measurement result (coverage data) in the previously selected range, but does not clear the coverage data in the fixed measurement areas 0 to 0x0FFFFF.

To determine the coverage measurement range, select the area, and click the <OK> button.

## Function buttons

| | |
|---|---|
| OK | Validates the coverage measurement range selected in (1) Address Range. |
| Cancel | Closes this dialog box. |
| Help | Displays this dialog box online help files. |

# Software Break Manager

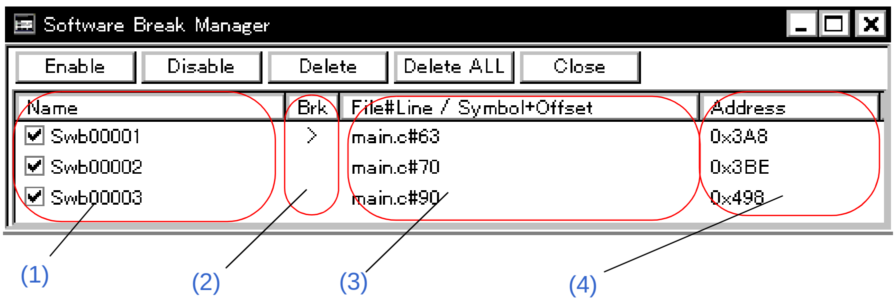
This window is used to display, enable or disable, and delete software breaks. (Refer to "5.4.4 Hardware break and software break".)

Software breakpoints cannot be set in this window; they can be set in the Source Window or Assemble Window. (Refer to "5.4.2 Breakpoint setting".)

**Caution:** Software breaks can be set or deleted while the user program is being executed. While the user program is being executed, software breaks can be set, deleted, enable or disable. The warning of a purport which makes a user program once take a break is displayed.

**Remark:** The displayed items are sorted by clicking the title (on the label) in each column (ascending/ descending order is switched each time the title is clicked).

Figure 6-55 Software Break Manager




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [Event] menu -> [Software Break Manager].

## Explanation of each area

### (1) Name

This area displays the names of registered events, and the check boxes that indicate whether each event is enabled or disabled.

An event name is displayed in the form of "Swb+[number]" in the default condition. It can be changed to an alphanumeric string of up to 256 characters. To change an event name, select and click a name. Then directly edit the name. To set the editing, press the Enter key.

When an event is enabled, the check box is checked. To be disable, the check box is not checked.

Furthermore, the name jumps to the Source Window by double-clicking an event name if the event name corresponds to the source line, whereas the name jumps to the Assemble Window if it does not correspond to the source line.

### (2) Brk

The ">" mark is displayed for a software break event that is set at the current PC position (so that the software break event that caused a break can be easily identified).

### (3) File#Line / Symbol+Offset

This area displays the location at which a software break event was set as follows:

- Program$file name#line number (If the event corresponds to the source line.)
- Program$file name#symbol+offset (If the event dose not correspond to the source line.)

Events are evaluated based on this when a symbol is re-downloaded.

### (4) Address

This area displays the address at which a software break event is set.

## Function buttons

| | |
|---|---|
| Enable | Enables the selected event. |
| Disable | Disables the selected event. |
| Delete | Deletes the selected event. |
| Delete ALL | Deletes all the set software break events. |
| Close | Closes this window. |

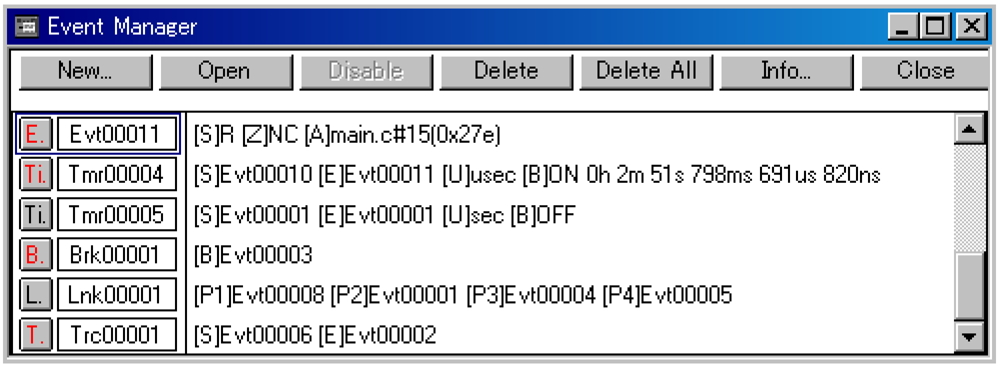
# Event Manager

This window is used to manage event conditions. This window allows display, enabling/disabling, and deletion of the Various Event Conditions. (Refer to "5.12 Event Function".)

Other operations using Context menu, Function buttons (Related event function), etc., can be performed in this window.

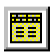
The event icon is the jump pointer of the Jump function.

Figure 6-56 Event Manager (In Detailed Display Mode)




- Opening
- Explanation of each area
- [View] menu (Event manager-dedicated items)
- Context menu
- Function buttons (Related event function)

## Opening

Click the **Mgr** button, or select [Event] -> [Event Manager] on the menu.

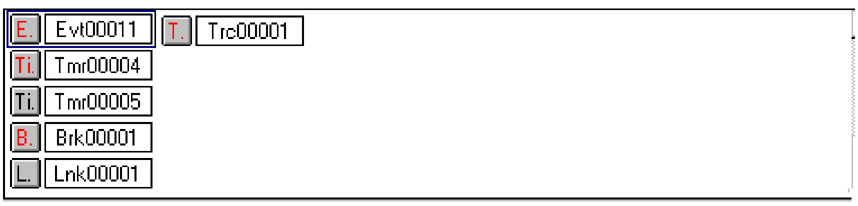
## Explanation of each area

### (1) Event display area

This area displays the icons (event icons) of the registered Various Event Conditions.

By selecting the context menu -> [Detail], the details can be displayed.

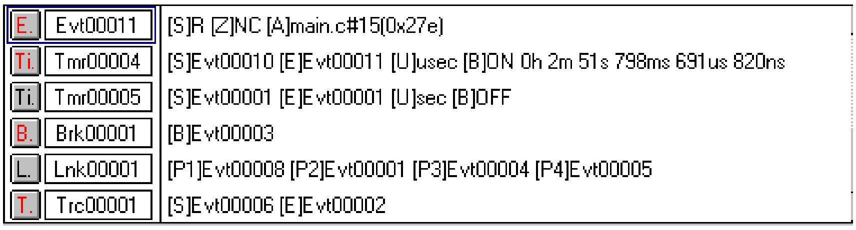
(a) [In list displayed]



Displays event icon (refer to "Table 5-18 Event Icon").

The event icon is the jump pointer (refer to "5.16.2 Jump function").

(b) In detailed display




Details of event contents are displayed by using the following key information as a separator.

Table 6-19 Separator for Displaying Event Details

| Key Information | Contents |
|---|---|
| **Event condition** | |
| [S] | Status condition |
| [Z] | Access size condition |
| [A] | Address condition<br>Symbol or expression: (actual address) |
| [D] | Data condition<br>Symbol or expression: (actual address) |
| [M] | Mask condition |
| **Event link condition** | |
| [P1] - [P4] | Event link condition on "n"th line |
| [D] | Disable condition |
| [P] | Pass count condition |
| **Break condition** | |
| [B] | Break condition |
| **Trace condition (when IECUBE is connected)** | |
| [M] | Tracer control mode |
| [D] | Delay Count |
| [S] | Trace start condition |
| [E] | Trace end condition |
| [Q] | Qualify trace condition |
| **Trace condition (when IECUBE is connected)** | |
| [S] | Timer measurement start condition |
| [E] | Timer measurement end condition |
| [U] | Timer measurement unit |
| [B] | Timeout break condition |

## [View] menu (Event manager-dedicated items)

The following items are added in the [View] menu , when the Event Manager is active.

| | |
|---|---|
| Select All Event | Selects all the registered events. |
| Delete Event | Deletes a selected event. |
| Sort By Name | Displays icons in the order of event names. |
| Sort By Kind | Displays icons in the order of event types. |
| Unsort | Does not sort icons (default). |
| Detail | Detail display |
| Overview | List display (default) |

## Context menu

| | |
|---|---|
| Sort By Name | Displays icons in the order of event names. |
| Sort By Kind | Displays icons in the order of event types. |
| Unsort | Does not sort icons (default). |
| Detail | Displays the details. |
| Overview | List display (default). |
| Source Text | Displays the corresponding source text and source line, using the position of the selected event as the jump destination address. (Refer to "5.16.2 Jump function".)<br>If no line information exists at the jump destination address, however, you cannot jump.<br>Opens the Source Window.<br>If an active Source Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Assemble | Displays the Assemble window from the position of the selected event, which is used as the jump destination address. (Refer to "5.16.2 Jump function".)<br>Opens the Assemble Window.<br>If an active Assemble Window is open, that window is displayed in the forefront (so that it can be manipulated). |
| Memory | Displays the memory contents from the position of the selected event, which is used as the jump destination address. (Refer to "5.16.2 Jump function".)<br>Opens the Memory Window.<br>If an active Memory Window is open, that window is displayed in the forefront (so that it can be manipulated). |

## Function buttons (Related event function)

Describes the all function buttons the related event dialogs (the Event Manager, Event Dialog Box and each various event setting dialog boxes (refer to "Table 5-16 Various Event Conditions").

| | |
|---|---|
| OK | (Event Dialog Box, Event Link Dialog Box)<br>Automatically registers the event condition being edited, if any, and closes this dialog box.<br>**In the select mode**<br>An event condition is selected and the setting dialog box (indicated on the title bar) that called the Event Link dialog box is displayed again. If the calling dialog box has already been closed, the select mode is returned to the normal mode, and the Event Dialog Box is not closed. Otherwise, this dialog box will be closed. |
| | (Other than above dialog boxes)<br>Automatically registers the event condition being edited, if any, and closes this dialog box. Each event condition becomes valid as soon as it has been registered. |
| New... | (Event Manager)<br>Opens the dialog box to create new event condition.<br>By clicking each button, the corresponding event setting dialog box can be opened with the new event name set. After the event setting dialog box has been opened, this dialog box is closed.<br>Returns to Event Manager by clicking the <Cancel> button. |
| | (Other than above dialog boxes)<br>Newly creates an event condition in this dialog box.<br>An event condition name is automatically created and a new event condition is prepared. |
| Set | (Event Dialog Box, Event Link Dialog Box)<br>Registers the various event conditions. Because the dialog box is not closed even after an event has been registered, new event conditions can be registered.<br>**In the select mode**<br>An event condition is selected. If there is an event being edited, it is automatically registered and selected. |
| | (Other than above dialog boxes)<br>Registers the various event conditions. Because the dialog box is not closed even after an event has been registered, new event conditions can be registered.<br>Each event condition becomes valid as soon as it has been registered. |
| Enable/Disable | Validates (enables) or invalidates (disables) the selected event condition.<br>However, event conditions and event link conditions cannot be enabled or disabled.<br>Same operation as the clicking the mark of event icon. |
| Clear | Clears the contents of the event condition. |
| Restore | Restores the contents of an edited event condition.<br>If an event condition not registered is displayed, all the fields other than the event name field are blank or the default values are set. |
| Cancel / Close | Closes this dialog box. Even if an event condition is being edited, it is not registered and the dialog box is closed. |
| Help | Displays the help window of this window. |
| Event Link... | Opens the Event Link Dialog Box |
| Break... | Opens the Break Dialog Box. |
| Trace... | Opens the Trace Dialog Box (when IECUBE is connected). |
| Timer... | Opens the Timer Dialog Box (when IECUBE is connected). |

| | |
|---|---|
| Manager | Opens the Event Manager. |
| Add Event... | Opens the Event Dialog Box in the select mode, and selects or newly creates an event condition to be set. The event condition will be added to the area selected when the < Add Event...> button is pressed. |
| Add Link... | Opens the Event Link Dialog Boxin the select mode, and selects or newly creates an event link condition. The event condition will be added to the area selected when the < Add Link...> button is pressed. |
| Open | Opens the various event setting dialog box corresponding to the selected event condition (one). Each setting dialog box displays the contents of the selected event condition.<br>Same operation as double-clicking the event icon or pressing the Enter key. |
| Remove / Delete | Deletes the selected event.<br>When an event condition or an event link condition is to be deleted, an error occurs and the event condition or event link condition cannot be deleted if the event is used as a various event condition. |
| Delete All | Deletes all event conditions except software break events |
| Expand >>> / Shrink <<< | Turns on (Expand>>>) or off ( Shrink<<<) display of the event manager area.<br>The size of the dialog box is expanded or reduced. |
| Add | The event condition and event link condition selected in Event Manager area add to setting area with a focus. |
| Info... | Opens the Select Display Information dialog box.<br>This dialog box is used to change the display mode and rearrange event names.<br>Figure 6-57 Select Display Information Dialog Box<br><br><br><Sort by Name> ......Sorts events into name order.<br><Sort by Kind>..........Sorts events into type order.<br><Unsort>..................Displays events in the order in which they have been registered without sorting the events.<br><Detail> ....................Sets the detailed display mode.<br><Overview> .............Sets the list display mode.<br><Cancel> .................Closes this dialog box (same as ESC key). |
| Other... | Opens the dialog box for selecting the event type.<br>By clicking each button, the corresponding event setting dialog box can be opened with the new event name set. After the event setting dialog box has been opened, this dialog box is closed.<br><Manager...> ...........Opens the Event Manager.<br><Cancel> ................Closes the dialog box to create event condition. |

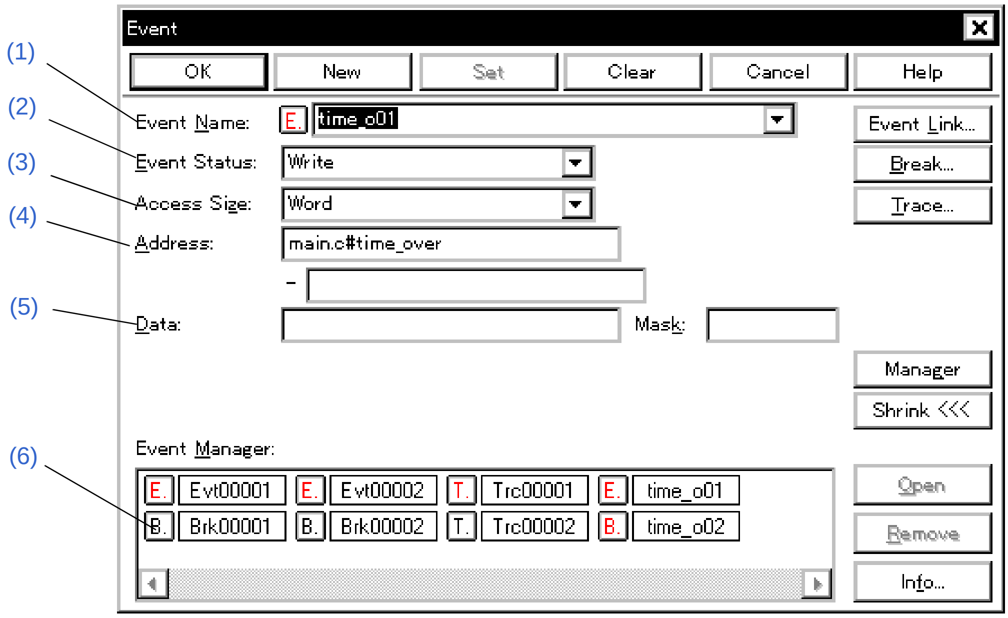
# Event Dialog Box

This dialog box is used to register and display event conditions. (Refer to "5.12 Event Function".)

Setting of event conditions is done by setting each item in this dialog box and then pressing the <OK> button. The registered event conditions are managed by the Event Manager.

One event condition can be set for multiple Various Event Conditions. However, the number of event conditions that can be simultaneously used is limited (refer to "5.12.4 Number of enabled events for each event condition").

Figure 6-58 Event Dialog Box

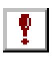



- Opening
- Explanation of each area
- Function buttons (Related event function)

## Opening

**In normal mode**

If the Event Dialog Box is opened as follows, an event condition can be registered without its purpose being specified.

Click the **Evn** button, or select [Event] -> [Event...] on the menu.

**In select mode**

If the <OK> button is pressed when the Event Dialog Box has been opened as follows, an event condition can be registered in the setting dialog box from which this dialog box was opened (the setting dialog box from which the this box was opened is displayed on the title bar.).

In each various event setting dialog box, click the <Add Event... > button.

## Explanation of each area

**(1) Event Name:**

This area is used to set an event name.

Directly input an alphanumeric string of up to eight characters as a name.

To display the contents of an already created event condition, select from the drop-down list.

In the select mode, the selected event condition can be set in the event condition setting area of the setting dialog box that called the Event Dialog Box.

The mark on the left of this area indicates the utilization status of events (refer to "Table 5-18 Event Icon"). The gray E. mark indicates that the event condition is being edited and has not been registered yet.

**(2) Event Status:**

This area is used to select a status condition.

By specifying a status condition, the type of the execution event and an access event is determined (if an execution event is specified, nothing can be input to the (3) Access Size: and (5) Data:, Mask:).

The status conditions that can be specified are listed below.

Table 6-20 Status Condition

| Status | Abbreviation | Meaning |
|---|---|---|
| **Execution event** | | |
| Execution | EX | Program execution |
| Before Execution | EX-B | Program execution (break before execution) Note |
| **Access event** | | |
| R/W | RW | Data read/write |
| Read | R | Data read |
| Write | W | Data write |

**Note:** Multiple items can be specified, but only two items, including access events, can be enabled. The address range cannot be specified. Can be used only for break event conditions.

**(3) Access Size:**

This area is used to select an access size condition.

By selecting an access size condition from the drop-down list, the access width of a data condition to be detected by an access event is determined.

| | |
|---|---|
| Byte | Detects data condition with 8-bit width (only during 8-bit access). |
| Half Word | Detects data condition with 16-bit width (only during 16-bit access). |
| Word | Detects data condition with 32-bit width (only during 32-bit access). |
| No Condition | Does not detect access size (nothing can be input to the Data area). |
| Bit | Detects data condition with 1-bit width (only during 8-bit access).<br>In this case, a data condition is detected with 1-bit width. Because of the operation of the in-circuit emulator, access to a bit is not directly detected; the ID850QB detects a dummy bit access by internally setting address conditions and data conditions as follows:<br><br>**Input example:**<br>Address: FE20.1<br>Data: 1<br>\|<br>**Setting of trace search:**<br>Address: FE20<br>Data: 00000010B<br>Mask: 11111101B<br><br>If another bit of the same address is accessed or if all the 8 bits of the same address are accessed, therefore, an event is detected in accordance with the specified status if the address and bit match the specified value of [address.bit].<br>When data is written to a bit, all the 8 bits are read/written. If read or read/write is specified as the status, an event occurs if a read operation is performed at this time if the value of the specified [Address.bit] matches. |

**Remark:** If no access size condition is specified, a judgment is automatically made from the address condition and data condition, and the following is set:

-Bit if the address condition is set in bit units

-Byte if the data condition is set in 8-bit units

-Half Word if the data condition is set in 16-bit units

-Word if the address condition is set in 32-bit units

-No Condition if no data condition is specified

**(4) Address:**

This area is used to specify an address condition (address value).

The default radix for inputting a numeric value is hexadecimal. A data can be also specified by a symbol (refer to "Table 5-6 Specifying Symbols").

Table 6-21 Settable Range of Address Condition (Event)

| Settable Range | Remark |
|---|---|
| 0 =< address value =< 0xFFFFFFFF | None |

(a) Address:

Set an address condition (lower address - higher address) (may be omitted).

The following can be set:

**Caution:** Specify a 28-bit address, since physical address and image space are distinguished in setting event.

(i) Setting as a point

Set a value to only the lower address, or set the same value to the lower address and the higher address.

(ii) Setting as a range

Set a value to the lower address and the higher address.

(iii) Setting as a bit

Set a value to only the lower address, or set the same value to the lower address and the higher address.

Specify a value in the form of address.bitî .

The value of bit, which indicates the bit position, must be 0 >= bit >= 7.

**(5) Data:, Mask:**

This area is used to specify an data condition (data value, mask value).

The default radix for inputting a numeric value is hexadecimal.

The settable range differs as follows depending on the access size condition specified in (3) Access Size:.

Table 6-22 Settable Range of Data Condition

| Access Size | Settable Range |
|---|---|
| Byte | 0 <= data value <= 0xFF<br>0 <= mask value <= 0xFF |
| Word | 0 <= data value <= 0xFFFFFFFF<br>0 <= mask value <= 0xFFFFFFFF |
| Bit | Data value = 0 or 1<br>Mask value = Cannot be specified. |

(a) Data:

Set a data value as data conditions. A data can be also specified by a symbol (refer to "Table 5-6 Specifying Symbols").

(b) Mask:

Set a mask value for the data value.

When a mask is set, the data value for the bit whose mask value is 1 may be 0 or 1.

**Example1:**

| | |
|---|---|
| Data | 0x4000 |
| Mask | 0xFF |

With this setting, addresses 0x4000 to 0x40FF satisfy the condition.

**Example2:**

| | |
|---|---|
| Data | 0x4000 |
| Mask | 0x101 |

With this setting, addresses 0x4000, 0x4001, 0x4100, and 0x4101 satisfy the condition.

**(6) Event Manager:**

This area is used to display the list of the events registered. (Refer to "Table 5-18 Event Icon", "(4) Manipulation in event manager area".)

## Function Buttons

Refer to " Function buttons (Related event function)" in the Event Manager.

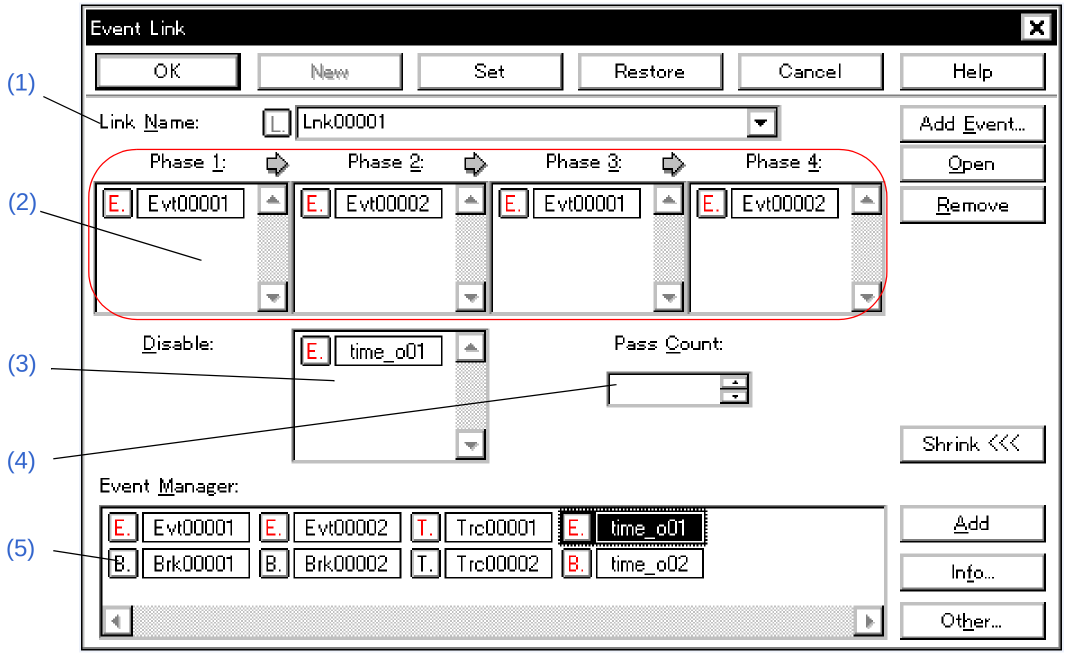
# Event Link Dialog Box

This dialog box is used to register and display event link conditions. (Refer to "5.12 Event Function".)

Registration of event link conditions is done by setting each item (256 items max.) in this dialog box and then pressing the <OK> button. The registered event link conditions are managed by the Event Manager.

However, the number of event link conditions that can be simultaneously used is limited ("5.12.4 Number of enabled events for each event condition").

Figure 6-59 Event Link Dialog Box




- Opening
- Explanation of each area
- Function buttons (Related event function)

## Opening

### In normal mode

If the Event Link Dialog Box is opened as follows, an event link condition can be registered without its purpose being specified.

Select [Event] -> [Event Link...] from the menu bar.

### In select mode

If the <OK> button is pressed when the Event Link Dialog Box has been opened as follows, an event link condition can be registered in the setting dialog box from which this dialog box was opened.

In each various event setting dialog box, click the <Add Link... > button.

(the setting dialog box from which the Event Link Dialog Box was opened is displayed on the title bar.)

## Explanation of each area

**(1) Link Name:**

This area is used to set a event link name.

Directly input an alphanumeric string of up to eight characters as a name.

To display the contents of an already created event link condition, select from the drop-down list.

In the select mode, the selected event condition can be set in the event link condition setting area of the setting dialog box that called the Event Link Dialog Box.

The mark on the left of this area indicates the utilization status of event link conditions ("Table 5-18 Event Icon"). The mark "L" in gray indicates that an event link condition is being edited and has not been registered yet.

**(2) Phase1:, Phase2:, Phase3:, Phase4:**

This area is used to specify the sequence in which event conditions and events are detected.

Up to four sequences can be specified. If a disable condition is detected while the program is being executed, however, the event conditions that have so far been satisfied are initialized, and the event conditions are detected again starting from the first event condition. If a link condition and a disable condition are detected at the same time, the disable condition takes precedence.

Set Phase 1 -> Phase 2 -> Phase 3 -> Phase 4, in that order.Phase 4 does not have to be set. In this case, an event occurs when the event condition set for the last phase has been detected. An event condition can be set for only Phase 1 or the same event condition can be set for two or more Phases.

The number of event conditions that can be set to each phase of this area and while the dialog box, is as follows:

Table 6-23 The Number of Event Conditions in Event Link Dialog Box

<table>
<tr><th colspan="2">Connected IE</th><th>Each Phase</th><th>Disable Area (Execution/ Access)</th><th>Total (Execution/ Access)</th></tr>
<tr><td colspan="2">IECUBE</td><td>10</td><td>10</td><td>14 (8/6)</td></tr>
<tr><td rowspan="2">N-Wire CARD, MINICUBE</td><td>Nx85ET （RCU0+TEU+TRCU）</td><td>10[*a]</td><td>12 (max)</td><td>12 (max)</td></tr>
<tr><td>Nx85E901（RCU0）, RCU1</td><td>1[*b]</td><td>--</td><td>2[*c]</td></tr>
</table>

***a** Can be set in Phase1-4 (expect before execution)

***b** Can be set only in Phase 1 and Phase 2 (must always be set in two stages)

***c** Only events before execution

**(3) Disable:**

This area is used to set an event condition that invalidates the event conditions that have so far been satisfied (refer to "Table 6-23 The Number of Event Conditions in Event Link Dialog Box").

Setting of event conditions is easily done by dragging the icon of the event to be set from the event manager area and dropping it in this area. For details, refer to "5.12.3 Setting event conditions".

**(4) Pass Count:**

This area is used to set a pass count condition (settable range: 1 - 4095).

A pass count condition specifies how many times an event condition must be satisfied during user program execution before a given condition is satisfied.

If no pass count is specified, 1 is assumed (the condition is satisfied as soon as the event condition is satisfied).

N-Wire CARD, MINICUBE (When Nx85E901（RCU0）, RCU1 is connected, this area does not have to be set.

**(5) Event Manager:**

This area is used to display the list of the events registered. (Refer to "Table 5-18 Event Icon", "(4) Manipulation in event manager area".)

## Function Buttons

Refer to " Function buttons (Related event function)" in the Event Manager.

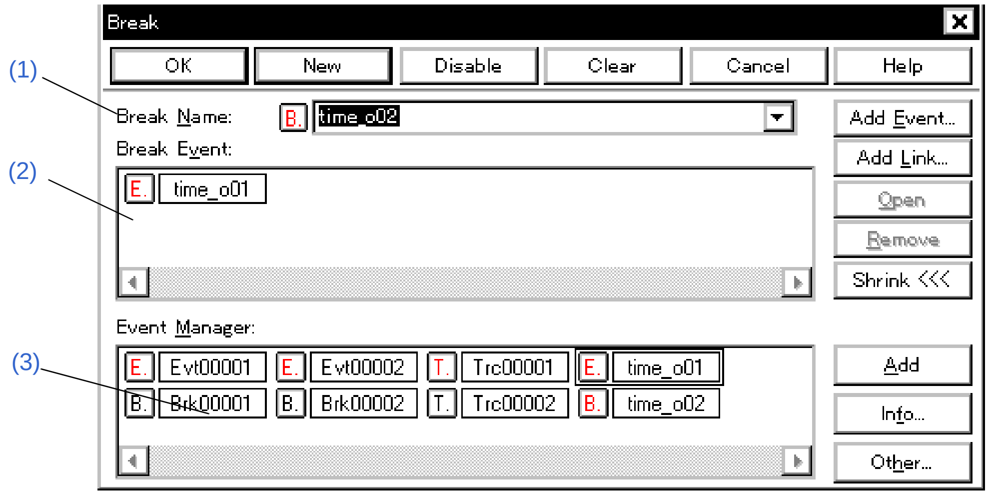
# Break Dialog Box

This dialog box is used to register; set, and display break event conditions. (Refer to "5.12 Event Function", "5.4 Break Function".)

Registration and setting of break event conditions is done by setting each item (256 items max.) in this dialog box and then clicking the <OK> button. The registered break event conditions are managed by the Event Manager.

There are restrictions on the number of break event conditions that can be simultaneously set (enabled). (Refer to "5.12.4 Number of enabled events for each event condition".)

Figure 6-60 Break Dialog Box




- Opening
- Explanation of each area
- Function buttons (Related event function)

## Opening

Click the **Brk** button, or select [Event] menu -> [Break...].

## Explanation of each area

**(1) Break Name:**

This area is used to set a break event name. Directly input an alphanumeric string of up to eight characters as a name.

To display the contents of an already created event condition, select from the drop-down list.

The mark on the left of this area indicates the utilization status of events (refer to "Table 5-18 Event Icon"). The gray mark indicates that an event condition is being edited and has not been registered yet. By clicking the left mark, an event condition can be validated or invalidated.

**(2) Break Event:**

This area is used to set an event condition for break.

The number of event conditions and event link conditions that can be set in this area is as follows:

Table 6-24 Number of Events Settable in Condition Setting Area

<table>
<tr><th colspan="2">Connected IE</th><th>Total Event Conditions (Before Execution / After Execution / Access)</th><th>Event Link Conditions</th></tr>
<tr><td colspan="2">IECUBE</td><td>16 (2 / 8 / 6)</td><td>1</td></tr>
<tr><td rowspan="2">N-Wire CARD, MINICUBE</td><td>Nx85ET<br>（RCU0+TEU+TRCU）</td><td>14 (2 / 8 / 4)</td><td>--</td></tr>
<tr><td>Nx85E901（RCU0），RCU1</td><td>2</td><td>--</td></tr>
</table>

Event conditions are easily set by dragging the icon of the event to be set from the event manager area and dropping it in this area. For details, refer to "5.12.3 Setting event conditions".

**(3) Event Manager:**

This area is used to display the list of the events registered. (Refer to "Table 5-18 Event Icon", "(4) Manipulation in event manager area".)

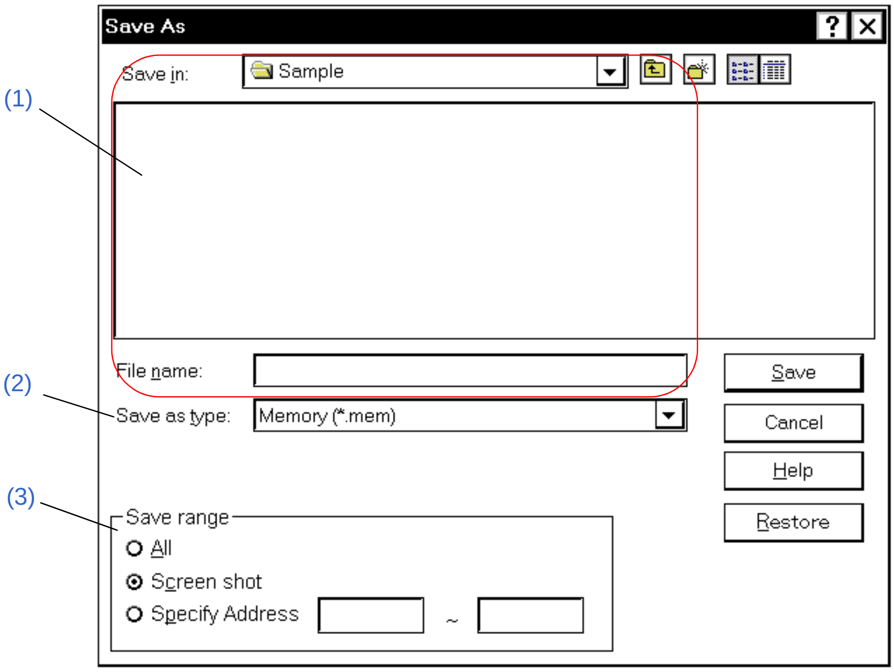
## Function Buttons

Refer to " Function buttons (Related event function)" in the Event Manager.

# View File Save Dialog Box

This dialog box is used to save the current display information of the current window to a view file. (Refer to "5.15.2 Window display information (view file)".)

Figure 6-61 View File Save Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the window to be saved is the current window, select [File] menu -> [Save As...].

## Explanation of each area

### (1) Save in:, File name:

This area is used to specify the file name to be saved. A file name can be directly input from the keyboard, or selected from the list.

Up to 257 character string with a extension can be specified.

### (2) Save as type:

This area is used to specify the type (extension) of the file to be saved. (Refer to "Table 5-21 Type of the View Files".) The extension of the file corresponding to the current window is displayed.

**(3) Save range**

Specify the range of data to be saved.

This area is displayed if the current window to be saved is the following.

- Assemble Window
- Memory Window
- Source Window
- Trace View Window

(a) All

This should be selected to save the entire range, from the first line to the last line.

(b) Screen shot

This should be selected to save the area visible on the screen, from the top line on the screen to the bottom line. If the Source Window is in the mixed display mode, however, the window contents are saved from the source line that includes the area visible on the screen.

(c) Specify Line / Specify Frame / Specify Address

This should be selected to specify the start line and end line of the area to be saved.

If the start line and end line are omitted, the first line and last line are assumed.

If a range of 100 lines / 100 frames / 256 bytes or more is specified, a message dialog box is displayed to indicate the progress of saving. To stop saving midway, click the <Stop> button in the message dialog box.

Display any of the following corresponding to the current window:

| | |
|---|---|
| Specify Line | Specify the range of the line numbers to be saved.<br>The default radix for inputting a numeric value is decimal.<br>If the Source Window is in the mixed display mode, the mixed displayed part on the specified line is also saved. |
| Specify Frame | Specify the range of trace frames to be saved. (Refer to "Table 6-16 Frame Number Specification Format".) The default radix for inputting a numeric value is decimal. |
| Specify Address | Specify the range of address to be saved.<br>An address can be also specified by a symbol or expression. (Refer to "Table 5-6 Specifying Symbols".)<br>The default radix for inputting a numeric value is hexadecimal. |

## Function buttons

| | |
|---|---|
| Save | Saves the display information of the current window to the selected file. After saving, this dialog box is closed. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |
| Restore | Restores the status before this dialog box was opened. |

# View File Load Dialog Box

This dialog box is used to read the view files. (Refer to "5.15.2 Window display information (view file)".)

When a view file is loaded, the reference window (Source Window in static status) opens and the display information at saving is displayed.

The window to be opened and its status differ as follows, depending on the file to be loaded.

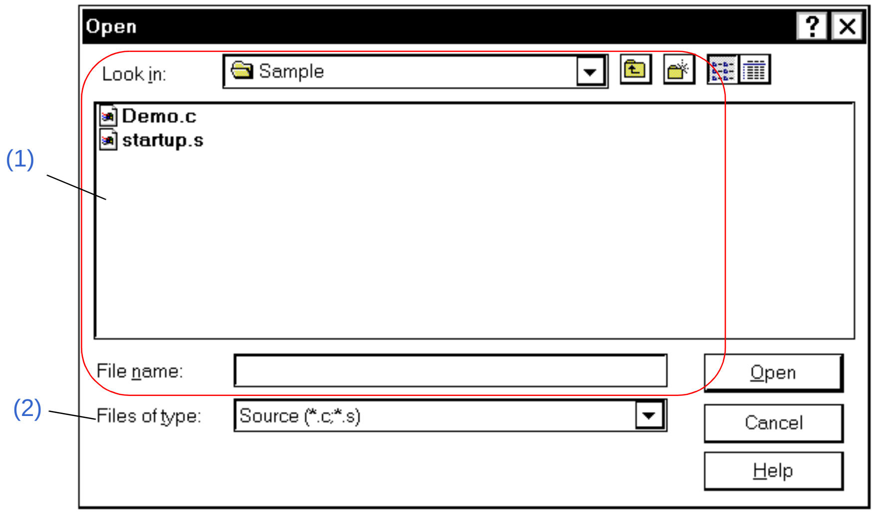
- **Loading source file to which symbol information has been read**

  If there is a Source Window in the active status, it is opened in the static status; otherwise, the Source Window is opened in the active status.

- **Loading source file to which symbol information has not been read, or view file**

  A window of text-format files is opened in the Source Window in the static status.

Figure 6-62 View File Load Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Click the **Open** button or select [File] menu -> [Open...].

## Explanation of each area

**(1) Look In:, File name:**

This area is used to specify the file name to be loaded.  A file name can be directly input from the keyboard, or selected from the list.

Up to 257 character string with a extension can be specified.

**(2) Files of type:**

This area is used to specify the type (extension) of the file to be loaded. (Refer to "Table 5-21 Type of the View Files".)

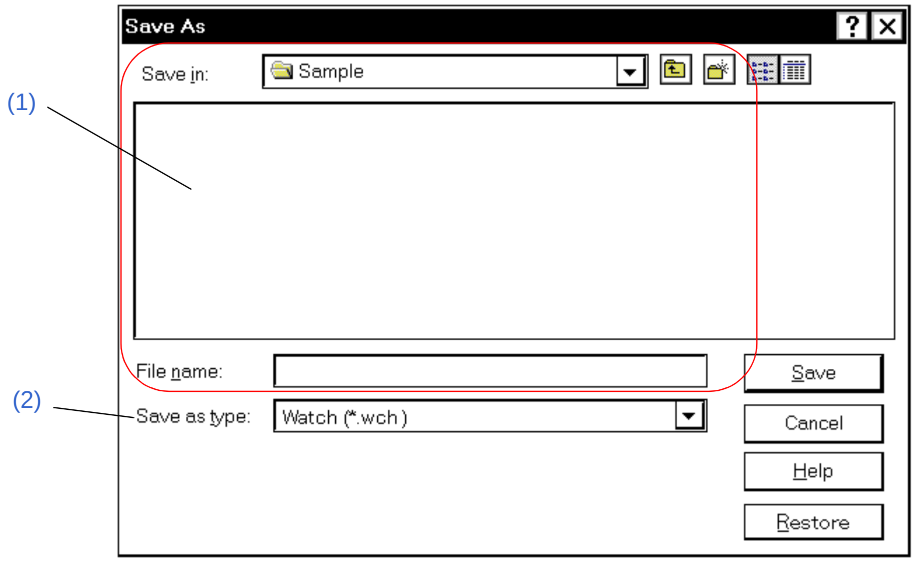
## Function buttons

| | |
|---|---|
| Open | Loads the selected file. After loading the file, this dialog box is closed. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |

# Environment Setting File Save Dialog Box

This dialog box is used to save the setting contents of the current window to a setting file. (Refer to "5.15.3 Window setting information (setting file)".)

Figure 6-63 Environment Setting File Save Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

When the window to be saved is the current window, select [File] menu -> [Environment] -> [Save As...].

## Explanation of each area

**(1) Save in:, File name:**

This area is used to specify the file name to be saved. A file name can be directly input from the keyboard, or selected from the list.

Up to 257 character string with a extension can be specified.

**(2) Save as type:**

This area is used to specify the type (extension) of the file to be saved. (Refer to "Table 5-22 Type of the Setting Files").

The extension of the file corresponding to the current window is displayed.

## Function buttons

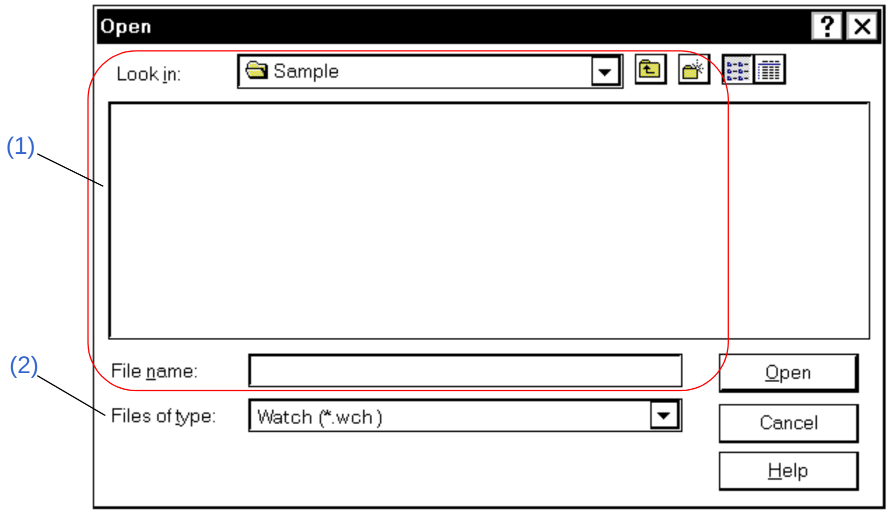
| | |
|---|---|
| Save | Saves the setting information of the current window to the selected file. After saving, this dialog box is closed. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |
| Restore | Restores the status before this dialog box was opened. |

# Environment Setting File Load Dialog Box

This dialog box is used to read the setting files. (Refer to "5.15.3 Window setting information (setting file)".)

When a setting file is loaded, the target window opens and the setting information at saving is restored.

Figure 6-64 Environment Setting File Load Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

select [File] menu -> [Environment] -> [Open...].

## Explanation of each area

**(1) Look In:, File name:**

This area is used to specify the file name to be loaded. A file name can be directly input from the keyboard, or selected from the list.

Up to 257 character string with a extension can be specified.

**(2) Files of type:**

This area is used to specify the type (extension) of the file to be loaded. (Refer to "Table 5-22 Type of the Setting Files".)

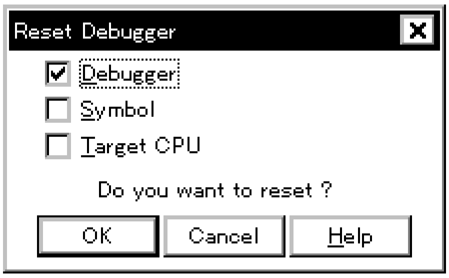
## Function buttons

| | |
|---|---|
| Open | Loads the selected file. After loading the file, this dialog box is closed. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |

# Reset Debugger Dialog Box

This dialog box is used to initialize the ID850QB, CPU, and symbol information.

Figure 6-65 Reset Debugger Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Select [File] menu -> [Debugger Reset...].

## Explanation of each area

**(1) Reset subject selection area**

This area is used to specify what is to be Initialized. Initializes the checked item.

| | |
|---|---|
| Debugger | Initializes the ID850QB (default). |
| Symbol | Initializes the symbol information. |
| Target CPU | Initializes the CPU. |

## Function buttons

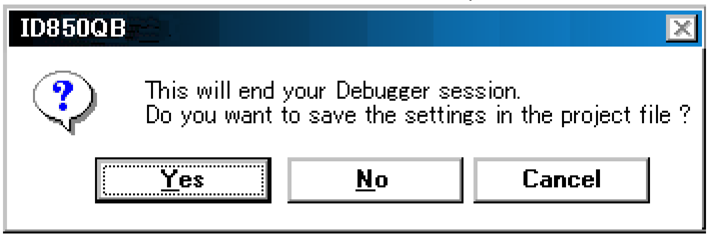
| | |
|---|---|
| OK | Initializes according to the setting. |
| Cancel | Cancels the changes and closes this dialog box. |
| Help | Displays this dialog box online help files. |

# Exit Debugger Dialog Box

This dialog box is used to select whether the current debug environment is saved to a project file or not before terminating the ID850QB. (Refer to "5.15.1 Debugging environment (project file)".)

It can be specified in the Debugger Option Dialog Box that the ID850QB is terminated without this confirmation dialog box being opened.

Figure 6-66 Exit Debugger Dialog Box




- Opening
- Function buttons

## Opening

- Select [File] menu -> [Exit].
- If forcible termination, such as to terminate the application, has been executed on the task list that terminates Windows.

## Function buttons

| | |
|---|---|
| Yes | Saves the current debug environment to a project file, closes all the windows, and terminates the ID850QB.<br>If a project file name is not specified, the Project File Save Dialog Box is opened. If the <Cancel> button is selected on the Project File Save Dialog Box, the environment is neither saved to a project file nor is the ID850QB terminated.<br>(If a project file is loaded or saved during debugger operation, this button has the default focus. ) |
| No | Closes all the windows and terminates the ID850QB.<br>(If a project file is not loaded or saved during debugger operation, this button has the default focus.) |
| Cancel | Closes this dialog box without executing anything. |

# About Dialog Box

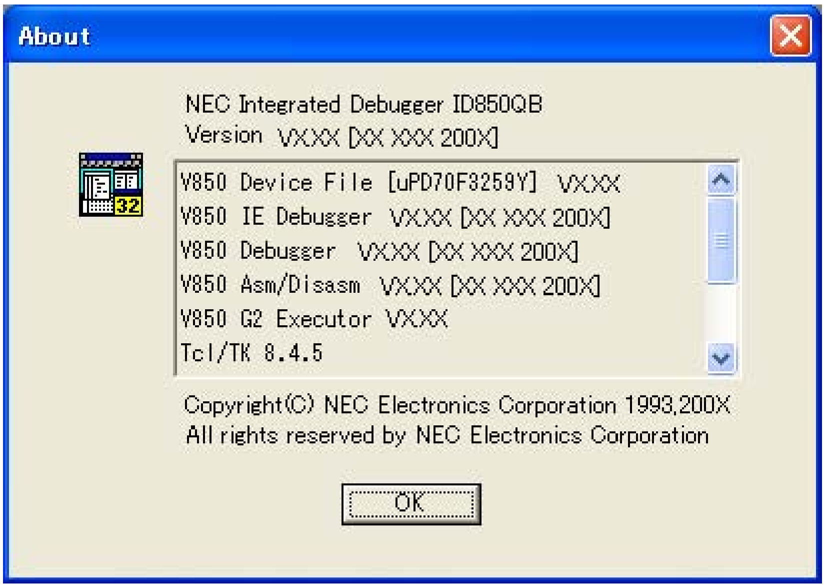
This dialog box displays the version information of the ID850QB (the year is displayed in 4 digits).

**Remark:** The version information can be copied to the clipboard by selecting [Select All and Copy (&C)] from the context menu in the dialog box.

The following version information is displayed:

- Product version of ID850QB
- Version of device file
- Version of GUI
- Version of debugger DLL
- Version of assembler DLL
- Version of executor
- Version of Tcl/Tk
- Product ID and product version of in-circuit emulator

Figure 6-67 About Dialog Box




- Opening
- Function buttons

## Opening

Select [Help] menu -> [About...], or click the <About...> button in the Configuration Dialog Box.

## Function buttons

| | |
|---|---|
| OK | Closes this dialog box. |

# Console Window

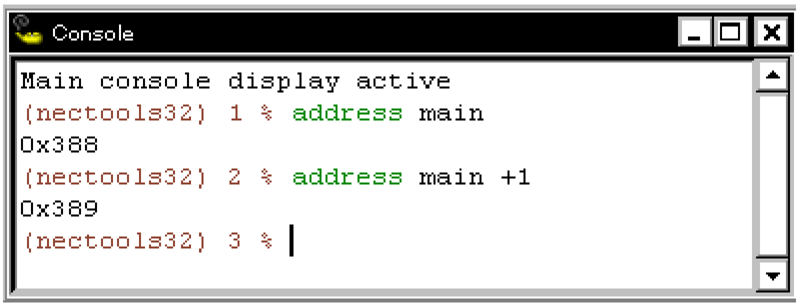
This window is used to input commands that control the ID850QB.

Because the key bind is Emacs-like, the accelerator key is not acknowledged if the Console Window is active. However, the F1 key displays the online help files of the Console Window.

While the Console Window is open, an error message window with only an <OK> button is displayed in the Console Window.

Refer to "CHAPTER 7 COMMAND REFERENCE" for details on the command specifications.

Figure 6-68 Console Window




- Opening

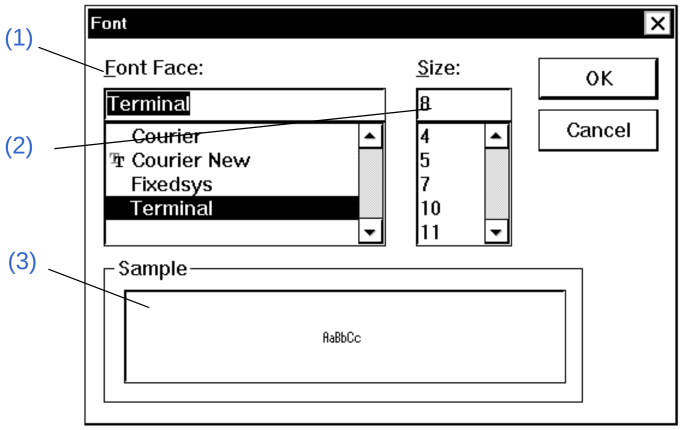
## Opening

Select [Browse] menu -> [Console].

# Font Dialog Box

This dialog box is used to select the font and font size to be displayed in the Source Window, Watch Window, Quick Watch Dialog Box, Local Variable Window, and Stack Window.

Figure 6-69 Font Dialog Box






## Opening

Click the <Font> button in the Debugger Option Dialog Box.

## Explanation of each area

**(1) Font Face:**

This area is used to select a font from the fonts currently usable.

Only fonts with equal width (fonts with a constant stroke width and a fixed pitch) are enumerated. When a font name is selected from the list, the font name is displayed in the text box, and the font size that can be used with that font is displayed under (2) Size:.

**(2) Size:**

This area is used to specify the font size (unit: pt.).

On the drop-down list, the font size usable for the font specified in (1) Font Face: is displayed in point units. When the font size is selected from the drop-down list, the selected font size is displayed in this area. The font size can also be directly input to the text box from the keyboard.

**(3) Sample**

This area displays a sample character string of the specified font and size.

## Function buttons

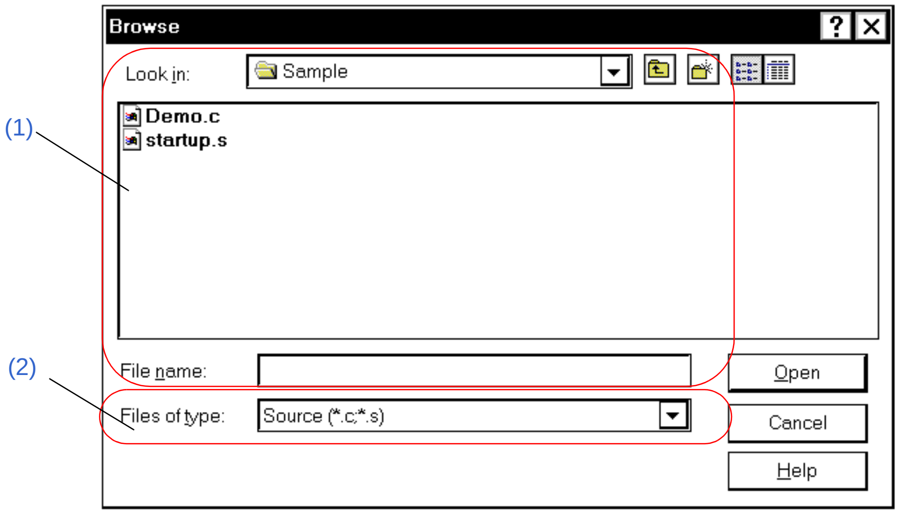
| | |
|---|---|
| OK | Validates the settings and closes this dialog box. |
| Cancel | Cancels the changes and closes this dialog box. |

# Browse Dialog Box

This dialog box is used to select the file to be set in the Source Text Move Dialog Box.

**Remark:** If this dialog box is opened for the first time after the system has been started up, the directory first specified by the source path is displayed. When the dialog box is opened the second and subsequent times, the previously displayed directory is recorded and displayed again. If the <Cancel> button is clicked, however, the previously displayed directory is not recorded.

Figure 6-70 Browse Dialog Box




- Opening
- Explanation of each area
- Function buttons

## Opening

Click the <Browse...> button in the target dialog box.

## Explanation of each area

**(1) Look In:, File name:**

This area is used to specify the file name to be opened. A file name can be directly input from the keyboard, or selected from the list.

Up to 257 character string with a extension can be specified.

**(2) Files of type:**

This area is used to specify the type (extension) of the file to be opened (refer to "Table 5-5 File Type Can Be Displayed").

## Function buttons

| | |
|---|---|
| Open | Sets the selected file. After setting the file, this dialog box is closed. |
| Cancel | Closes this dialog box without executing anything. |
| Help | Displays this dialog box online help files. |

# CHAPTER 7 COMMAND REFERENCE

This chapter explains the details of the command functions of the ID850QB.

## 7.1 Command Line Rules

The specification of command lines has the following rules:

- Command name, option, and argument are specified for command line.
- To divide words, a space (space key or tab key) is used.
- At the end of a line, a line feed character or a semicolon is used.
- When a command name and an option are entered to the point of identifiability, they are recognized.
- In script, command names have to be entered completely.

**Command format**

| |
|---|
| command *-options arg1 arg2 arg3 ...* |

## 7.2 Command List

Table 7-1 List of Debugger Control Commands

| Command name | Function |
|---|---|
| address | Evaluation of address expression |
| assemble | Disassemble/line assemble |
| batch | Executing batch (with echo) |
| breakpoint | Setting/deletion of breakpoint |
| cache | Setting of cache |
| dbgexit | Terminating ID850QB |
| dbgopt | Selecting debugger option |
| download | Download of files |
| erase | Deletion of the internal flash memory |
| extwin | Creation of expansion window |
| finish | Returning from function |
| flop | Manipulation related to internal flash memory |
| go | Continuous execution |
| help | Display of help |
| hook | Setting of hook |
| ie | Display/setting of IE register |
| inspect | Symbol inspect |
| jump | Jump to window |
| map | Setting / deleting memory mapping |
| mdi | Setting of expansion window |
| memory | Display/setting of memory |
| module | Display of the list of files and functions |
| next | Procedure step |
| refresh | Redrawing of window |
| register | Display/setting of register value and IOR value |
| reset | Reset |
| run | Reset and execution of CPU |
| step | Step execution |
| stop | Stop execution |
| upload | Upload |
| version | Display of the version information |
| watch | Display/setting of variables |

| Command name | Function |
|---|---|
| where | Stack trace |
| wish | Start of Tclet |
| xcoverage | Operation of coverage (IECUBE) |
| xtime | Operation of timer (IECUBE) |
| xtrace | Operation of tracer (IECUBE) |

Table 7-2 List of Console/Tcl Commands

| Command Name | Function |
|---|---|
| alias | Creation of another name |
| cd | Change of directory |
| clear | Clears the screen |
| echo | Echo |
| exit | Close/end |
| history | Display of history |
| ls | Display of files |
| pwd | Check of the directory |
| source | Execution of batch |
| time | Measurement of time for commands |
| tkcon | Console control |
| unalias | Deletion of another name |
| which | Display of the command path or another name |
| Other | Based on Tcl/Tk 8.4 |

## 7.3 List of Variables

- dcl (chip) - Chip name read only
- dcl (prjfile) - Project file name read only
- dcl (srcpath) - Source path read only
- dcl (ieid) - IE type read only
- dcl (iestat) - IE status read only
- dcl (bkstat) - Break status read only
- env (LANG) - Language
- dcl_version - Dcl version read only

## 7.4 List of Packages

- tcltest - Restoration test
- cwind - Automatic window control
- BWidget - Toolkit
- tcllib - Tcl library
- mclistbox - Multi-column list box
- combobox - Combo box

## 7.5 Key Bind

- tcsh + Emacs like
- Complement of command name [Tab]
- Complement of file name [Tab]
- HTML help [F1]

## 7.6 Expansion Window

The expansion windows can be created using Tk.

In the expansion windows, Widget is allocated with '.dcl' as a root instead of '.'.

When the following script files are allocated in bin/idtcl/tools/, an expansion window is added on selecting [Browse] menu -> [Others].

The mdi command, an exclusive command for expansion windows, has been added.

```
# Sample.tcl
wm protocol .dcl WM_DELETE_WINDOW { exit }
mdi geometry 100 50
button .dcl.b -text Push -command exit
pack .dcl.b
```

**Caution:** In the expansion windows, Tk menu commands cannot be used because of the restrictions of MDI windows.

## 7.7 Callback Procedure

Expansion windows can hold dcl_asyncproc procedures called by asynchronous messages.

```
proc dcl_asyncproc {mid} {
    if {$mid == 19} {
        redraw
    }
}
```

The asynchronous message ID is passed for the argument of the dcl_asyncproc procedure

The message IDs are shown below:

Table 7-3 Message ID

| Message ID | Meaning |
|---|---|
| 9 | After changing configuration |
| 10 | After registering event |
| 11 | After deleting event |
| 12 | Before executing |
| 13 | After breaking |
| 14 | After resetting CPU |
| 15 | After resetting ID850QB |
| 17 | After changing extended option |
| 18 | After changing debugger option |
| 19 | After downloading |
| 20 | After changing memory or register |
| 36 | Before starting tracer(IECUBE) |
| 37 | After stopping tracer (IECUBE) |
| 40 | Before starting timer (IECUBE) |
| 41 | After stopping timer (IECUBE) |
| 42 | After clearing trace (IECUBE) |
| 45 | After resetting symbol |
| 46 | After change from RRM function to trace function, and change from trace function to RRM function.(IECUBE) |

## 7.8 Hook Procedure

A hook can be set in the ID850QB using the hook procedure.

The hook procedures are shown below:

- BeforeDownload(Hook before downloading)
- AfterDownload(Hook after downloading)
- AfterCpuReset(Hook after CPU reset during break)
- BeforeCpuRun(Hook before starting execution)
- AfterCpuStop(Hook after breaking)

By using hook procedures, register values can be changed before downloading programs or after resetting the CPU.

An actual example of the procedure is shown below.  A hook is valid till the ID850QB is closed.

**(1)  [When hook is set with ID850QB control command]**

1) Create script file a. with an editor.
2) Start up the ID850QB, select [Browse] menu -> [Console], and open the Console Window.
3) If the script file is executed in the window as below, the hook in the script file is set.

   %hook test.tcl

**(2)  [When hook is set on downloading of project file]**

1) Create script file a. with an editor.[Note]
2) Start up the ID850QBand read test.prj.  The hook in the script file is set.

```
proc BeforeDownload {} {
    register MM 0x7
    register PMC8 0xff
    register PMC9 0xff
    register PMCX 0xe0
}

proc AfterCpuReset {} {
    register MM 0x7
    register PMC8 0xff
    register PMC9 0xff
    register PMCX 0xe0
}
```

**Note:** Be sure that the script file name is the same as the project file.

Example:

The script file corresponding to test.prj is test.tcl.

Allocate test.prj, test.pri, and test.tcl in the same directory.

## 7.9 Related Files

- Executes when the aliases.tcl console is opened.
  Sets the default alias etc.
- Executes when the project file name.tcl project is opened.  The following hooks can be used.
  BeforeDownload, AfterDownload, AfterCpuReset, BeforeCpuRun, AfterCpuStop
- Executes when the load module name.tcl load module is downloaded.  The following hooks can be used.
  BeforeDownload, AfterDownload, AfterCpuReset, BeforeCpuRun, AfterCpuStop

## 7.10 Cautions

- The separator for file and path is a slash (/).
- When a console is open, error messages are output to the console.
- To terminate the command forcibly, close the console.
- The execution of external commands (DOS commands) is OFF by default.

## 7.11 Explanation Of Commands

In this section, each command is explained using the format shown below.

## Command name

Describes the command name.

### Input format

Describes the input format of the command.

In the following explanation, italics indicate an Argument to be supplied by the user, while the argument enclosed in "?" may be omitted.

When a command name and an option are entered to the point of identifiability, they are recognized.

### Functions

Explains the functions of the command.

### Usage example

Shows an example of the usage of the command.

# address

address - Evaluation of address expression

## Input format

**address** *expression*

## Functions

Converts the address expression specified by *expression* into address.

## Usage example

```
(IDCON) 1 % address main
0xaa
(IDCON) 2 % address main+1
0xab
```

# assemble

assemble - Disassemble/line assemble

## Input format

**assemble** *?options? address ?code?*

## Functions

Assembles the character strings specified by *code*from the *address* specified by address.

When '.' is specified for address, it is understood as an *address* continuing from the immediately previous assemble.

When *code* is omitted, it is assembled from the *address* specified by address.

The following are *options*: They are ignored for assembly.

| | |
|---|---|
| **-code** | Command code is also displayed.It is ignored for assembly. |
| **-number** *number* | *Number* line is displayed. It is ignored for assembly. |

## Usage example

```
(IDCON) 1 % assemble -n 5 main
0x000000aa  B7          PUSH  HL
0x000000ab  B1          PUSH  AX
0x000000ac  891C        MOVW  AX,SP
0x000000ae  D6          MOVW  HL,AX
0x000000af  A100        MOV   A,#0H
(IDCON) 2 % assemble main mov a,b
(IDCON) 3 % assemble . mov a,b
```

# batch

batch - Executing batch (with echo)

## Input format

**batch** *scriptname*

## Functions

Executes in batch with displaying files specified by *scriptname* on the screen.

Nesting is possible.

## Usage example

```
(IDCON) 1 % clear
(IDCON) 2 % batch bat_file.tcl
(IDCON) 3 % tkcon save a:/log.txt
```

# breakpoint

breakpoint - Setting/deletion of breakpoint

## Input format

**breakpoint** *?options? ?address1? ?address2?*

**breakpoint** -delete *brkno*

**breakpoint** -enable *brkno*

**breakpoint** -disable *brkno*

**breakpoint** -information

## Functions

Operates the breakpoint specified by *options* and *address*.

If a breakpoint can be set correctly, the breakpoint number is returned.

The following are *options*:

| | |
|---|---|
| **-software** | A software break is specified. |
| **-hardware** | A hardware break is specified (default). |
| **-execute** | The *address* execution break is set (default). |
| **-beforeexecute** | The break before *address* execution is set. |
| **-read** | An *address* data read break is set. |
| **-write** | An *address* data write break is set. |
| **-access** | An *address* data access break is set. |
| **-size** *size* | The access size is set (8, 16 or 32bit). |
| **-data** *value* | The data condition is set. |
| **-datamask** *value* | The data mask is set. |
| **-information** | The list of breakpoints is displayed. |
| **-delete** | The breakpoint whose number is specified is deleted. |
| **-disable** | The breakpoint whose number is specified is disabled. |
| **-enable** | The breakpoint whose number is specified is enabled. |

## Usage example

```
(IDCON) 1 % breakpoint main
1
(IDCON) 2 % breakpoint -i
1 Brk00001 enable rammon.c#17

(IDCON) 3 % breakpoint -software sub
2
(IDCON) 4 % breakpoint -i
1 Brk00001 enable rammon.c#17
2 Brk00001 enable rammon.c#8

(IDCON) 5 % breakpoint -disable 2
(IDCON) 6 % breakpoint -i
1 Brk00001 enable rammon.c#17
2 Brk00001 disable rammon.c#8

(IDCON) 7 % breakpoint -delete 1
2 Brk00001 disable rammon.c#8
```

# cache

When N-Wire CARD, MINICUBE is connected

cache - Setting of cache

## Input format

cache

**cache** *config ?-icache itype? ?-dcache dtype?*

**cache** *clear ?-icache bool? ?-dcache bool?*

## Functions

When *config* is specified for the subcommand, the cache type is set.

When *clear* is specified for the subcommand, whether cache clear is to be done by EXEC, or not, is set (default is clear)

When subcommand or below is omitted, the current status is displayed.

*itype* is selected from the following:

| | |
|---|---|
| NB85E212 | NB85E212 is used. |
| NB85E213 | NB85E213 is used. |
| nouse | Not used (default). |

*dtype* is selected from the following:

| | |
|---|---|
| NB85E252 | NB85E252 is used. |
| NB85E263 | NB85E263 is used. |
| nouse | Not used (default). |

*bool* is selected from the following:

| | |
|---|---|
| 0, false, or off | Off |
| 11, true, or on | On |

## Usage example

```
(IDCON) 1 % cache config -i NB85E212 -d NB85E252
(IDCON) 2 % cache
i-cache: NB85E212
d-cache: NB85E252
(IDCON) 3 % cache clear -i false
(IDCON) 4 % cache
i-cache: NB85E212 (persist)
d-cache: nouse
```

# dbgexit

dbgexit - Terminating ID850QB

## Input format

**dbgexit** *?options?*

## Functions

Terminate the ID850QB.

The following are *options*:

| | |
|---|---|
| **-saveprj** | Project is saved on terminating ID850QB. |

## Usage example

(IDCON) 1 % dbgexit -saveprj

# dbgopt

dbgopt - Selecting the debugger option

## Input format

**dbgopt** *options ?value?*

## Functions

Selects the option of the debugger.

The following are *options*:

| | |
|---|---|
| **-function *?func?*** | Switches the RRM function, trace function, or coverage function (when IECUBE is connected).<br>When *func* is omitted, the current function is displayed. |

*func* is selected from the following:

| | |
|---|---|
| rrm | RRM function is selected. |
| trace | Trace function is selected. |
| coverage | Coverage function is selected. |

## Usage example

(IDCON) 1 % dbgopt ?function trace

# download

download - Download of files

## Input format

**download** *?options? filename ?offset?*

## Functions

Downloads files specified with *filename* according to *options*.

If *offset* is specified, the address is shifted by the *offset* (if the data is in binary format, the load start address is specified for *offset*).

| | |
|---|---|
| **-binary** | Binary format data is downloaded. |
| **-coverage** | Coverage data is downloaded. |
| **-append** | Additional download is executed. |
| **-nosymbol** | Download is executed. Symbol information is not read. |
| **-symbolonly** | Symbol information is read. |
| **-erase** | The contents of the internal flash memory are erased all before download (only a product with internal flash memory) (when N-Wire CARD, MINICUBE is connected). |
| **-reset** | CPU is reset after download. |
| **-information** | Download information is displayed. |

## Usage example

(IDCON) 1 % download test.lmf

# erase

When N-Wire CARD, MINICUBE is connected

erase - Deletion of the internal flash memory

## Input format

**erase**

## Functions

Deletes the internal flash memory

## Usage example

(IDCON) 1 % erase

# extwin

extwin - Creation of expansion window

## Input format

**extwin** *scriptfile*

## Functions

Creates expansion window with *scriptfile*.

## Usage example

(IDCON) 1 % extwin d:/foo.tcl

# finish

finish - Returning from function

## Input format

**finish**

## Functions

Executes until it returns to the program that called the current function.

## Usage example

(IDCON) 1 % finish

# flop

When N-Wire CARD, MINICUBE is connected

flop -Manipulation related to internal flash memory

## Input format

**flop** -init

**flop** -user *clock*

## Functions

Selects whether the clock-related settings during internal flash memory write are the device file settings or user settings.

| | |
|---|---|
| **-init** | Processing is performed using the device file setting.<br>If the device file contains no information, an error occurs during internal flash memory write. In this case, use the -user option. |
| **-user *clock*** | IOR of the clock generation function is manipulated.<br>Specify for *clock* the frequency (MHz) of the CPU clock (fCPU) determined by the IOR manipulation.<br>In this case, perform the IOR setting using the hook procedure. |

## Usage example

(IDCON) 1 % flop -user 13.5

# go

go - Continuous execution

## Input format

**go** *?options?*

## Functions

Executes program continuously.If -waitbreak is specified, the command waits until the program stops.

The following are *options*:

| | |
|---|---|
| **-ignorebreak** | Breakpoint is ignored. |
| **-waitbreak** | The command waits for the program to stop. |

## Usage example

(IDCON) 1 % go -w

# help

help - Display of help

## Input format

**help**

## Functions

Displays Dcl help.

## Usage example

(IDCON) 1 % help

# hook

hook - Setting of hook

## Input format

**hook** *scriptfile*

## Functions

Sets the procedure for hook with *scriptfile*.

The hook setting is initialized when the project file is loaded and when the ID850QB is reset.

## Usage example

(IDCON) 1 % hook d:/foo.tcl

# ie

ie - Display/setting of IE register

## Input format

**ie** reg *address ?value?*

**ie** dcu *address ?value?*

## Functions

The ie command depends on the IE.

When reg is specified for the subcommand, referencing and setting of the IE register is executed.

When dcu is specified for the subcommand, referencing and setting of the DCU register is executed.

**Caution:**The value of a register will be reset by 0 if a DCU register is referred to.

## Usage example

(IDCON) 1 % ie reg 0x100 1
(IDCON) 2 % ie dcu 0x100 1

# inspect

inspect - Symbol inspect

## Input format

**inspect** *?options? progname pattern*

## Functions

Searches and displays the load module symbol specified with *progname* using the regular expression of *pattern*.

The following regular expressions can be used.

| | |
|---|---|
| ? | Match 1 character |
| * | Match characters other than 0 |
| [chars] | Match chars character. (Range specification such as [a-z/0-9] also possible.) |
| \x | Match character x. (? * [ ] \ specification also possible.) |

The following are *options*:

| | |
|---|---|
| **-nocase** | The case is distinguished. |
| **-address** | Displays in pair with symbol address. |

## Usage example

(IDCON) 1 % inspect test1.out {[a-z]*}

# jump

jump - Jump to window

## Input format

**jump** -source -line *filename ?line?*

**jump** *?options? address*

## Functions

Displays the window specified by *options*.

| | |
|---|---|
| **-source** | The Source Window is displayed from the address specified by *address*. |
| **-assemble** | The Assemble Window is displayed from the address specified by *address*. |
| **-memory** | The Memory Window is displayed from the address specified by *address*. |
| **-line** | The command is moved to the line specified by *line*. |
| **-focus** | The Focus is moved to the window displayed. |

## Usage example

```
(IDCON) 1 % jump -s main
(IDCON) 2 % jump -s -l mainfile.c 10
(IDCON) 3 % jump -m array
```

# map

map - Setting/deletion of memory mapping

## Input format

**map** *options address1 address2 ?accsize? ?cs?*

## Functions

Sets, deletes, and displays memory mapping.

The access size of 8, 16, or 32 is specified by accsize (unit:byte, the default is 8).

**Caution:** To map an emulation memory (alternate ROM/RAM) with the ID850QB, specify either cs0, cs1, cs2, cs3, cs4, cs5, cs6, or cs7 for *cs* for chip selection.
In the case of the V850ES Series, this specification can be omitted because the allocation by chip selection may be fixed or the chip selection function may not be provided. If *cs* is specified, specification of *accsize* cannot be omitted.

The following are *options*:

| | |
|---|---|
| **-erom** | Alternate ROM is mapped.<br>(when IECUBE is connected and when memory boards are incorporated) |
| **-eram** | Alternate RAM is mapped.<br>(when IECUBE is connected and when memory boards are incorporated) |
| **-target** | Target area is mapped. |
| **-targetrom** | Target ROM area is mapped (when IECUBE is connected). |
| **-protect** | I/O protect area is mapped. |
| **-rrm** | Start address of RRM area is set.<br>If performed during user program execution, CPU is stopped for an instant.<br>RRM area can be divided into 8 partitions.<br>The start address and size are specified in pairs in list format as follows.<br>{{address size} {address size} {address size} ...}<br>size is one of 256, 512, 768, 1024, 1280, 1536, 1792, 2048 bytes, and the total size is up to 2048. |
| **-clear** | All the settings for the mapping are deleted. |
| **-information** | Refer to the setting for the mapping. |

## Usage example

```
(IDCON) 1 % map -i
1: 0 0x7fff 8 {IROM}
2: 0x8000 0x87ff 8 {Target RRM}
3: 0x8800 0x9fff 8 {Target}
4: 0xa000 0xf7ff 8 {NonMap}
5: 0xf800 0xfaff - {NonMap}
6: 0xfb00 0xfedf 8 {Saddr}
7: 0xfee0 0xfeff 8 {Register}
8: 0xff00 0xffff 8 {IOR}
```

# mdi

mdi - Setting of expansion window

## Input format

**mdi** geometry *?x y? width height*

**mdi** title *string*

## Functions

Sets the size and title name of the expansion window.

The command can be used only from the expansion window.

## Usage example

```
(IDCON) 1 % mdi geometry 0 0 100 100
(IDCON) 2 % mdi title foo
```

# memory

memory - Display/setting of memory

## Input format

**memory** *?options? address ?value?*

**memory** *?options?* -fill *address1 address2 value*

**memory** *?options?* -copy *address1 address2 address3*

## Functions

Sets *value* in the memory of the *address* specified by address according to *options*.

If *value* is omitted, display the value of the memory of the address specified by *address*.

If -fill is specified, data from *address1* to *address2* is filled with *value*.

If -copy is specified, data from *address1* to *address2* is copied to *address3*.

The following are *options*:

| | |
|---|---|
| **-byte** | Displayed/set in one-byte units (default). |
| **-halfword** | Displayed/set in halfword units. |
| **-word** | Displayed/set in word units. |
| **-fill** | The data is filled in. |
| **-copy** | The data is copied. |
| **-noverify** | Verification is not executed on writing. |

If memory referencing is performed for other than the real-time RAM monitor area during user program execution, the CPU is stopped for an instant. The CPU is stopped for an instant even if the memory setting is performed.

## Usage example

```
(IDCON) 1 % memory 100
0x10
(IDCON) 2 % memory 100 2
(IDCON) 3 % memory 100
0x02
(IDCON) 4 % memory -fill 0 1ff 0
```

# module

module - Display of the list of files and functions

## Input format

**module** *progname ?filename?*

## Functions

Displays the list of files and functions of the load module specified by *progname*.

If *filename* is not specified, the list of files is displayed.

If *filename* is specified, the list of functions of the specified files is displayed.

## Usage example

```
(IDCON) 1 % module rammon.lmf
1: rammon.c
(IDCON) 2 % module rammon.lmf rammon.c
1: rammon.c sub1
2: rammon.c main
```

# next

next - Procedure step

## Input format

**next** *?options?*

## Functions

Executes the procedure steps. If functions are called, the step stops after executing function.

The following are *options*:

| | |
|---|---|
| **-source** | The command is executed in source line units (default). |
| **-instruction** | The command is executed in command units. |

## Usage example

```
(IDCON) 1 % next -i
(IDCON) 2 % next -s
```

# refresh

refresh - Redrawing of window

## Input format

**refresh**

## Functions

Redraws the window and updates the data.

## Usage example

(IDCON) 1 % batch foo.tcl
(IDCON) 2 % refresh

# register

register - Display/setting of register value and IOR value

## Input format

**register** *?options? regname ?value?*

## Functions

Sets *value* in the register specified with *regname*.

If *value* is omitted, displays the value of the register specified by *regname*.

The following are *options*:

| | |
|---|---|
| **-force** | Compulsory reading or writing is executed. |

If register reference/setting is performed during user program execution, the CPU stops for an instant.

## Usage example

```
(IDCON) 1 % register pc
0x100
(IDCON) 2 % register pc 200
(IDCON) 3 % register pc
0x200
```

# reset

reset - Reset

## Input format

**reset** *?options?*

## Functions

Resets the ID850QB , CPU, symbols or events.

If options are omitted, the CPU is reset.

The following are *options*:

| | |
|---|---|
| **-cpu** | CPU is reset (default). |
| **-debugger** | The ID850QB is reset. |
| **-symbol** | Symbol is reset. |
| **-event** | All events and software breaks are reset. |

## Usage example

(IDCON) 1 % reset

# run

run - Reset and execution of CPU

## Input format

**run** *?options?*

## Functions

Resets the program and executes it.

If -waitbreak is not specified, the command does not wait until the program stops.

The following are *options*:

| | |
|---|---|
| -**waitbreak** | The command waits for the program to stop. |

## Usage example

(IDCON) 1 % run
(IDCON) 2 % run -w

# step

step - Step execution

## Input format

**step** *?options?*

## Functions

Executes step execution.

If functions are called, the command stops at the head of the functions.

The following are *options*:

| | |
|---|---|
| **-source** | The command is executed in source line units (default). |
| **-instruction** | The command is executed in instruction units. |

## Usage example

(IDCON) 1 % step -i
(IDCON) 2 % step -s

# stop

stop - Stop executing

## Input format

**stop**

## Functions

Stops the program forcibly.

## Usage example

(IDCON) 1 % run
(IDCON) 2 % stop

# upload

upload - Upload

## Input format

**upload** *?options? filename address1 address2*

**upload** -coverage *filename*

## Functions

Saves the memory data a within the specified range in a file.

When saving coverage data, all the specified range of the coverage data is saved in the file (specification of start/end addresses not required).

The following are *options*:

| | |
|---|---|
| **-binary** | The data is saved in binary format. |
| **-coverage** | The coverage data is saved. |
| **-intel** | The data is saved in Intel HEX format (default). |
| **-motorola** | The data is saved in Motorola HEX format. |
| **-tektronix** | The data is saved in Tektronix HEX format. |
| **-force** | The file is overwritten. |

## Usage example

(IDCON) 1 % upload -b foo.hex 0 0xffff

# version

version - Display of the version information

## Input format

**version**

## Functions

Displays the version of the ID850QB.

## Usage example

```
(IDCON) 1 % version
GUI          : Vx.xx [XX XXXX 200X]
Devicefile   : V850 Device File [uPD703201] Vx.xx
Debugger     : V850 Debugger Vx.xx [XX XXXX 200X]
Executer     : V850 Executer Vx.xx [XXX XXXX 200X]
Monitor      : V850 Peripheral Vx.xx [XX XXXX 200X]
Assembler    : V850 Asm/Disasm Vx.xx [XX XXXX 200X]
Tcl/Tk       : 8.4.5
```

# watch

watch - Display/setting of variables

## Input format

**watch** *?options? variable ?value?*

## Functions

Displays and sets the variables.

The following are *options*:

| | |
|---|---|
| **-binary** | The value is displayed in binary digits. |
| **-octal** | The value is displayed in octal digits. |
| **-decimal** | The value is displayed in decimal digits. |
| **-hexdecimal** | The value is displayed in hexdecimal digits. |
| **-string** | The value is displayed in character strings. |
| **-sizeof** | The size, instead of the value, of variables is displayed in decimal digits. |
| **-encoding** ***name*** | Encoding during character string display is specified.<br>By default, system encoding is used.<br>*name* (encoding name) is based on the Tcl specification (shiftjis, euc-jp, etc.). |

## Usage example

```
(IDCON) 1 % watch var
0x10
(IDCON) 2 % watch -d var
16
(IDCON) 3 % watch array\[0\] 0xa
```

# where

where - Stack trace

## Input format

**where**

## Functions

Executes the back-trace of the stack.

## Usage example

```
(IDCON) 1 % where
1: test2.c#sub2(int i)#13
2: test.c#num(int i)#71
3: test.c#main()#82
```

# wish

wish - Startup of Tclet

## Input format

**wish** *scriptname*

## Functions

Starts up the script using Tk (Tclet).

The expansion window can be created with Tclet.

## Usage example

(IDCON) 1 % wish test.tcl

# xcoverage

When IECUBE is connected

xcoverage - Operation of coverage

## Input format

**xcoverage** *option*

## Functions

Operates coverage.

The following are *options*:

| | |
|---|---|
| **-clear** | Clears the coverage memory. |

## Usage example

(IDCON) 1 % xcoverage -clear

# xtime

xtime - Operation of timer

## Input format

**xtime** *option*

## Functions

Operates timer.

The following are *options:*

| | |
|---|---|
| **-start** | Timer starts on executing the program. (when IECUBE is connected) |
| **-stop** | Timer stops on executing the program. (when IECUBE is connected) |
| **-gobreak** | Time from Go to Break is displayed in nsec. |

## Usage example

(IDCON) 1 % xtime -start
(IDCON) 2 % xtime -stop

# xtrace

When IECUBE is connected

xtrace - Operation of tracer

## Input format

**xtrace** -dump *?-append? frameno ?filename?*

**xtrace** -start

**xtrace** -stop

**xtrace** -clear

**xtrace** *-addup ?bool*?

**xtrace** -mode *?mode?*

**xtrace** *-complement ?bool*?

## Functions

Operates tracer.

The following are *options:*

| | |
|---|---|
| **-start** | The tracer starts on executing the program. |
| **-stop** | The tracer stops on executing the program. |
| **-clear** | Clears the trace memory. |
| **-dump** | The trace data is dumped (default).<br>The dump result is redirected to the console window.<br>If the file name is specified, the dump result is written in the file. |
| **-append** | The dump result is added to a file. |
| **-addup** *?bool?* | Whether the time tag is totaled or not is selected. (Addition only)<br>When *bool* is omitted, the current mode is displayed. |
| **-mode** *?mode?* | The trace control mode (any one of: all, cond, nonstop, fullstop, fullbreak, delaystop, delaybreak, machine, or event) is selected.<br>When *mode* is omitted, the current mode is displayed. |
| **-complement** *?bool?* | Selects whether to perform trace complementation.<br>When bool is omitted, the current mode is displayed. |

## Usage example

```
(IDCON) 1 % xtrace -start
(IDCON) 2 % xtrace -stop
(IDCON) 3 % xtrace -dump 3
_ 01685 2 000000BC M1 br _sub2+0x2
_ 01686 4 0000009A BRM1 st.w r6, 0x8[sp]
_ 01687 3 0000009E BRM1 st.w r0, 0x0[sp]
(IDCON) 4 % xtrace -clear
(IDCON) 5 % xtrace -addup true
```

# 7.12 Samples (Calculator Script)

The script of the expansion window in which the calculator script is described and its execution screen are shown below.

**Script of expansion window**

```
# Calculator.tcl
mdi geometry 100 100
set top .dcl
entry $top.e -relief sunken -textvariable v
frame $top.f -height 120 -width 120; pack $top.e -fill x; pack $top.f -fill both -expand 1set i 0; set v {}; set r
0.25
foreach n {7 8 9 / 4 5 6 * 1 2 3 - 0 = + C} {
        if {$n == "=" || $n == "C"} {
                button $top.f.b$n -text $n
        } else {
                button $top.f.b$n -text $n -command "$top.e insert end $n"
        }
        place $top.f.b$n -relx [expr ($i%4)*$r] -rely [expr ($i/4)*$r] -relw $r -relh $r    incr I
}
bind $top.f.bC <1> {$top.e delete 0 end}
bind $top.f.b= <1> {catch {expr $v} v}
```

Figure 7-1 Execution Screen

# APPENDIX A  EXPANSION WINDOW



## A.1 Overview

With the ID850QB, the user can create custom windows in addition to the existing windows.

The Tcl (Tool Command Language) interpreter and the commands for controlling the debugger are implemented in the ID850QB.  Users can create windows using this Tcl.

The ID850QB is supplied with samples of the following expansion windows.

## A.2 Sample Window

Table A-1 List of Expansion Window (Sample)

| Window Name | Function |
|---|---|
| List window | Displays a list of the source files and functions. |
| Grep window | Searches a character string. |
| RRM window | This is the memory window for real-time RAM monitoring. |
| Hook window | Sets the hook procedure. |
| Memory Mapped I/O window | Writes to or reads from the specified address. |
| Sym Inspect window | Searches through a list of properly described symbols. |
| Run Break Timer window | Displays two types of time: Time at which the user program starts running (Run) and time at which the user program breaks (Break). |

## A.3 Activation

The expansion window can be activated by selecting List, Grep, RRM, Hook, Memory, SymInspect or RunBreak-Time in [Others] on the [Browse] menu.

**Remark:** Each .tcl file is installed in NEC Electronics Tools\ID850QB*Vx.xx*\bin\idtcl\tools

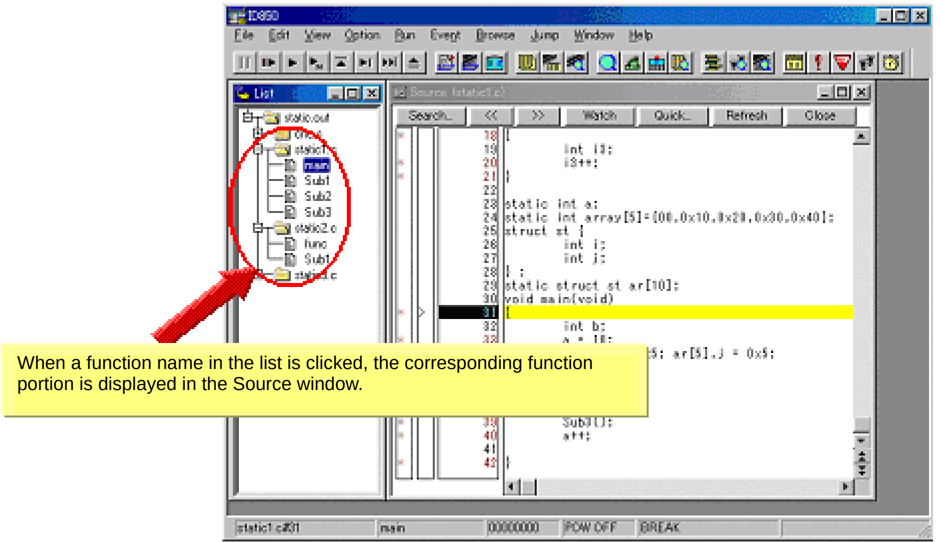
## A.4 Explanation of each sample window

The ID850QB provides the sample window below.

## A.5 List window

The lists of the source files and functions are displayed in a tree format in this window.  When a function name in the list is clicked, the corresponding source is displayed.

Figure A-1 List Window

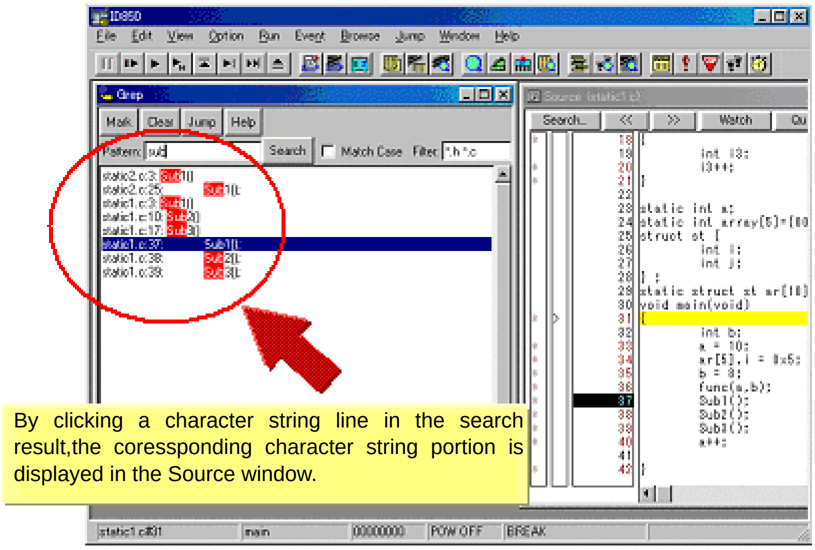


# Grep window

Search for a character string is performed in the files under the source path.

When the search result is clicked, the corresponding source is displayed.

Figure A-2 Grep Window




| Object | Function |
|---|---|
| Pattern | Input the character string to be searched. |
| <Mark> button | Marks the searched character string. |
| <Clear> button | Clears the marking. |
| <Jump> button | Put the cursor on a section in the search result and click this button to open the corresponding file. |
| Match Case | Select whether or not to distinguish uppercase and lowercase. |
| Filter | Specify the type of the file to be searched. |

# RRM window

This is a dedicated window for RAM monitoring.

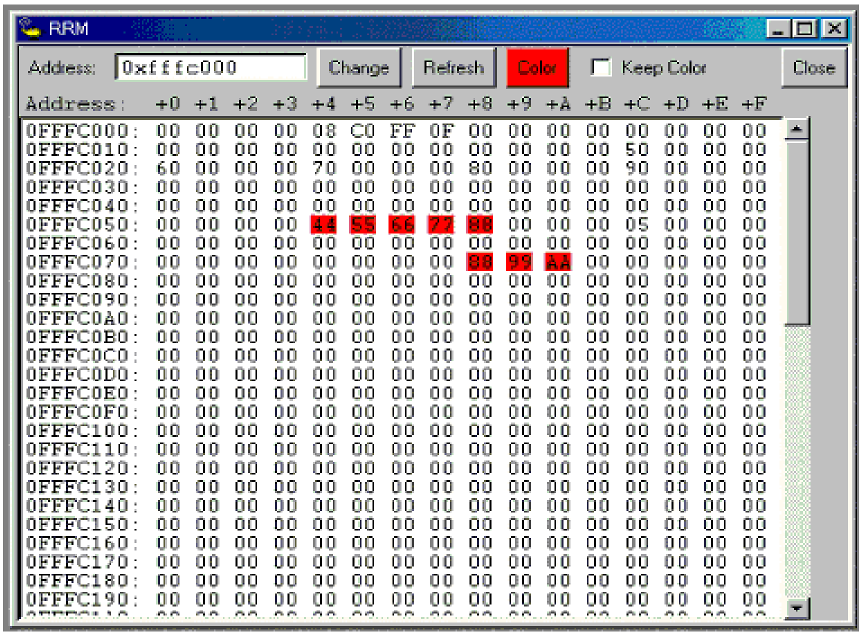
The address area in which a value was changed in the RAM area during program execution is highlighted with a color.  The display range is 1 KB. With reading RAM, execution of the user program momentarily breaks.  On this window, the start address of the RAM area can be changed while the user program is being executed.

**Caution:** All data are not read at the same time (because data of 1 KB is divided and read in word units).

**Remarks1:** This RRM window is opened even when the RAM monitor function is set to OFF in the Extended Option Dialog Box.

**Remarks2:** The sampling interval is about 0.3 - 0.7 seconds (20MHz), but it depend on the frequency of the CPU operation (when the N-Wire CARD, MINICUBE is connected).

Figure A-3 RRM Window




| Object | Function |
|---|---|
| Address | Input the start address to be displayed (automatically aligned to 1 KB.) |
| <Change> button | Switch the start address display. |
| <Refresh> button | Reads data from the memory. |
| <Color> button | The color can be customized. The default color is red. |
| Keep Color | Specify whether or not to hold the color highlighting.<br>Selected: Once a value is changed, the color highlighting is held until a break occurs.<br>Unselected: The color is cleared if there is no change of values(default). |
| <Close> button | Closes this window. |

# Hook window

This window is used to set a hook to the debugger, using a hook procedure.

The hook procedure enables changing the register value before downloading a program, or after a CPU reset.

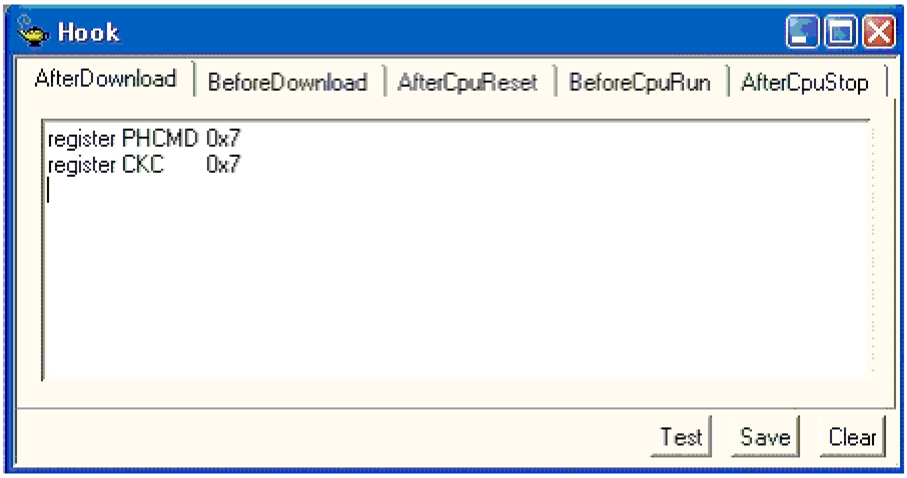
On this window, a hook can be set by using the following five tabs.

- [AfterDownload]  tab: Hook after downloading
- [BeforeDownload] tab: Hook before downloading
- [AfterCpuReset] tab: Hook after CPU reset during break
- [BeforeCpuRun] tab: Hook before start of execution
- [AfterCpuStop]  tab: Hook after break

**Remark:** By setting a IOR by using the [BeforeDownload] tab before downloading the load module, for example, downloading can be executed at high speeds. Access to the external memory is also facilitated by using this tab.

If the setting is saved as "project-file-name.tcl" in the directory where the project is stored, the setting is executed when the project is next opened.

Figure A-4 Hook Window



| Object | Function |
|---|---|
| [AfterDownload] tab | Hook after downloading<br>After downloading is performed, the register values input to the tab are automatically overwritten by the specified value. |
| [BeforeDownload] tab | Hook before downloading<br>Before downloading is performed, the register values input to the tab are automatically overwritten by the specified value. |
| [AfterCpuReset] tab | Hook after CPU reset during break<br>after resetting CPU, the register values input to the tab are automatically overwritten by the specified value. |
| [BeforeCpuRun] tab | Hook before starting execution<br>before starting execution, the register values input to the tab are automatically overwritten by the specified value. |
| [AfterCpuStop] tab | Hook after breaking<br>After breaking, the register values input to the tab are automatically overwritten by the specified value. |
| <Test> button | All the commands described on the tabs are tested. |
| <Save> button | Saves all the tab contents to a file.<br>If the ID850QB was activated from a project file, the file is saved as "project-file-name.tcl". |
| <Clear> button | Clears all the descriptions on the tabs. |

**Remark:** Specify the program register and the peripheral I/O registers for the register name.

# Memory Mapped I/O window

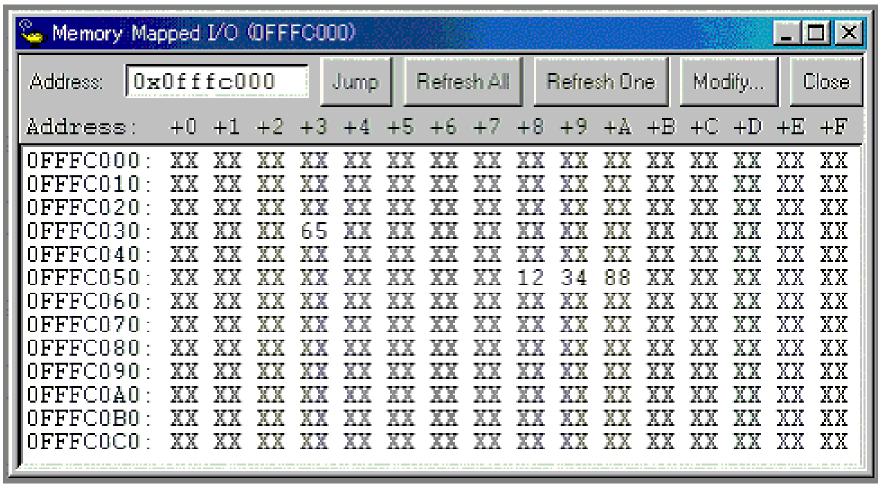
Data is explicitly read or written at a specified address in this window.

When a write is performed in the Memory Window, the data is internally read and verified by the ID850QB. In addition, the memory can also be read simply by scrolling in the Memory Window. On the Memory Mapped I/O window, however, the above operations are not performed.

Therefore, this window is useful for reading or writing a specific address.

While the user program is being executed, it momentarily breaks before data is written in this window.

Figure A-5 Memory Mapped I/O Window




| Object | Function |
|---|---|
| Address | Input the address to display. The display target address changes by pressing the Enter key or clicking the <Jump> button.The data contents are not read at this time, so the address (numerical value) is displayed in the address display section, but "XX" is displayed in the data section. |
| <Jump> button | Jumps to the address input in the Address field. |
| <Refresh All> button | Reads all the areas currently displayed only once.  "ZZ" will be displayed in the data section when an attempt is made to read an unmapped area, or when an error occurs upon a read. |
| <Refresh One> button | Reads data in the memory of the address at which the cursor is placed only once.<br>The read data size depends on the display format.<br>"ZZ" will be displayed in the data section when an attempt is made to read an unmapped area, or when an error occurs upon a read. |
| <Modify...> button | Opens the Memory Mapped I/O dialog box.<br>The address at which the cursor is placed is the input address displayed in the Memory Mapped I/O dialog box.If this button is clicked after the cursor position is changed in the Memory Mapped I/O dialog box, the Address field in the Memory Mapped I/O dialog box is also changed. |
| <Close> button | Closes this window. |
| Context menu | Select the display format from Byte, HalfWord, and Word. |

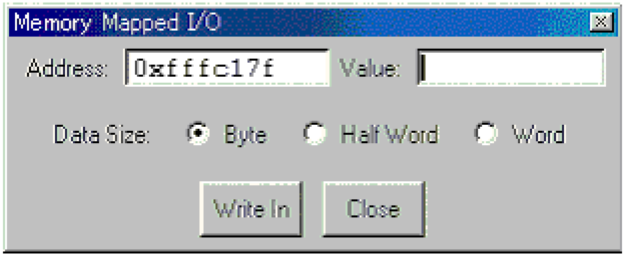
# Memory Mapped I/O dialog box

This dialog box can be opened by clicking the <Modify... > button on the Memory Mapped I/O window.   It is used to write data to any address.

**Cautions1:** When the area to which data is written is displayed in the Memory Window or Watch Window , data is read in these windows after the <Write in> button is clicked.

**Cautions2:** If Data Size is less than Access Size specified in the Configuration Dialog Box, ID850QB reads data in Access Size once, changes the corresponding part of the read data, and writes the changed data in Access Size.

Figure A-6 Memory Mapped I/O Dialog Box

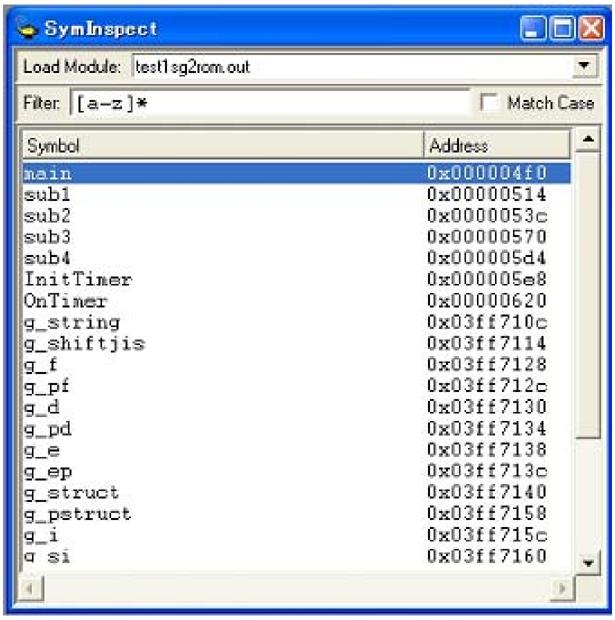



| Object | Function |
|---|---|
| Address | Input the address to be written.<br>The address corresponding to the data for which the cursor is placed in the Memory Mapped I/O window is displayed by default. |
| Value | Input the value to be written. |
| Data Size | Select the size of the data to be written.<br>The size specified in the Memory Mapped I/O window is selected by default. |
| <Write in> button | Data is written to the specified address with the specified size. |
| <Close> button | Closes this dialog box. |

# Sym Inspect window

This window displays the list of the symbols and addresses of loaded module files, and is used for searching the list for the properly described symbol.

Figure A-7 Sym Inspect Window



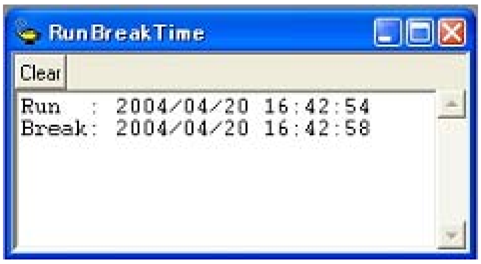

| Object | Function |
|---|---|
| Load Module: | Selects a load module file. |
| Filter: | Specifies a properly described symbol so that the symbol is retrieved. |
| Match Case | In Filter:, specify to differentiate or not differentiate case sensitivity. Check this box to differentiate case sensitivity. |
| Symbol | Displays the symbols.  Clicking this icon has the symbols sorted in alphabetical order. |
| Address | Displays the addresses.  Clicking this icon has the addresses sorted in ascending numerical order. |

| Context Menu | Function |
|---|---|
| Copy | Copies the selected address to the clipboard. |
| Jump to Source | Jumps from the address in the selected line to the identical address displayed in the Source Window. |
| Jump to Assemble | Jumps from the address in the selected line to the identical address displayed in the Assemble Window. |
| Jump to Memory | Jumps from the address in the selected line to the identical address displayed in the Memory Window. |

## Run Break Timer window

This window displays two types of time: Time at which the user program starts running (Run) and time at which the user program breaks (Break).  The window is helpful when measuring takes a long time.  The Windows timer function is utilized for this window; the time is displayed in hours, minutes, and seconds.

Figure A-8 RunBreakTimer Window




| Object | Function |
|---|---|
| <Clear> button | Clears the time display |

# APPENDIX B  INPUT CONVENTIONS

## B.1 Usable Character Set

Table B-1 List of Character Set

| Classification | Character |
|---|---|
| Alphabetic characters | Uppercase: 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<br>Lowercase: 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 |
| Numerals | 0123456789 |
| Character equivalent to alphabetic character | @ _ |

Table B-2 List of Special Characters

<table>
<tr><th>Character</th><th>Name</th><th>Usage</th></tr>
<tr><td>(</td><td>Left parenthesis</td><td>Changes operation order.</td></tr>
<tr><td>)</td><td>Right parenthesis</td><td>Changes operation order.</td></tr>
<tr><td>+</td><td>Plus</td><td>Addition operator or positive sign</td></tr>
<tr><td>-</td><td>Minus</td><td>Subtraction operator or negative sign</td></tr>
<tr><td>*</td><td>Asterisk</td><td>Multiplication operator or indirect reference operator</td></tr>
<tr><td>/</td><td>Slash</td><td>Division operator</td></tr>
<tr><td>%</td><td>Percent</td><td>Remainder operator</td></tr>
<tr><td>~</td><td>Tilde</td><td>Complement operator</td></tr>
<tr><td>|</td><td>Vertical line</td><td>Bit sum operator</td></tr>
<tr><td>^</td><td>Circumflex</td><td>Bit difference operator</td></tr>
<tr><td>&</td><td>Ampersand</td><td>Bit product operator or address operator</td></tr>
<tr><td>[</td><td>Left bracket</td><td rowspan="2">Array subscript operator or base register specification symbol</td></tr>
<tr><td>]</td><td>Right bracket</td></tr>
<tr><td>.</td><td>Period</td><td>Direct member operator or bit position specifier</td></tr>
<tr><td>,</td><td>Comma</td><td>Delimiter between operands</td></tr>
</table>

## B.2 Symbols

(1) A symbol consists of characters A to Z, a to z, @, _ (underbar), . (period) and 0 to 9.

(2) A symbol must start with a character other than numerals 0 to 9.

(3) Uppercase characters (A to Z) and lowercase characters (a to z) are distinguished.

(4) A symbol must be no more than 2048 characters long (if a symbol of more than 2048 characters is defined, only the first 2048 characters are valid).

(5) A symbol is defined by loading a load module file.

(6) Symbols are classified into the following types by the valid range:

- Global symbol (assembly language,C language)
- Static symbol
  In-file static symbol (C language)
  In-function static symbol (C language)
- Local symbol
  In-file local symbol (C language)
  In-function local symbol (C language)
  In-block local symbol (C language)

(7) The following symbols are available for each language used:

- Assembly language
  label name, bit symbol name
- C language
  Variable name (including pointer variable name, enumeration type variable name, array name, structure name, and union name)
  Function name, label name
  Array element, structure element, union element, bit field (if the symbol is an array, structure, or union)

(8) A symbol can be described instead of an address or numeric value.

(9) The valid range of a symbol is determined based on the source debug information when the source file is assembled or compiled.

(10) Describe only the symbol name of a global symbol.

(11) A local symbol is expressed in pairs with a file name.

## B.3 Numeric Values

The following four types of numeric values can be used. The input format of each type is as shown below.

The suffix (bold) and the alphabetic characters of hexadecimal numbers may be uppercase or lowercase characters. If the first character is A to F, 0 must be prefixed to it.

In the input field of ID850QB, decimal numbers or hexadecimal numbers are alternately selected, depending on the default radix.

Table B-3 Input Format of Numeric Values

| Numeric Value | Input Format |
|---|---|
| Binary number | n**Y**<br>n...n**Y** (n=0,1) |
| Octal number | n**O**<br>n...n**O** (n=0,1,2,3,4,5,6,7)<br>n**Q**<br>n...n**Q** (n=0,1,2,3,4,5,6,7) |
| Decimal number | n<br>n...n<br>n**T**<br>n...n**T** (n=0,1,2,3,4,5,6,7,8,9) |
| Hexadecimal numbers | n<br>n...n<br>n**H**<br>n...n**H**<br>**0x**n<br>**0x**n...n (n=0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F) |

## B.4 Expressions and Operators

**(1) Expressions**

An expression consists of constants, register names, peripheral I/O registers name and symbols coupled by operators.

If peripheral I/O registers name, label name, function name, or variable name is described as a symbol, an address is calculated as the value of the symbol.

The elements making up an expression, except operators, are called terms (constants and labels). Terms are called the first term, the second term, and so on, starting from the left.

**(2) Operators**

The following operators of the C language can be used:

Table B-4 List of Operators

| Symbol | Meaning | Explanation |
|---|---|---|
| **Arithmetic operator** | | |
| + | Addition | Returns the sum of the first and second terms. |
| - | Subtraction | Returns the difference between the first and second terms. |
| * | Multiplication | Returns the product of the first and second terms. |
| / | Division | Divides the value of the first term by the value of the second term, and returns the integer of the results. |
| MOD<br>% | Remainder | Divides the value of the first term by the value of the second term, and returns the remainder of the results. |
| - sign | Unary operator (negative) | Returns 2's complement of the value of the term. |
| + sign | Unary operator (positive) | Returns the value of the term. |
| **Logical operator** | | |
| NOT<br>~ | Negation | Logically negates each bit of the term, and returns the results. |
| AND<br>& | Logical product | Obtains the logical product of the values of the first and second terms on each bit, and returns the results. |
| OR<br>\| | Logical sum | Obtains the logical sum of the values of the first and second terms on each bit, and returns the results. |
| XOR<br>^ | Exclusive logical sum | Obtains the exclusive logical sum of the values of the first and second terms on each bit, and returns the results. |
| **Shift operator** | | |
| SHR<br>>> | Right shift | Shifts the value of the first term by the value (number of bits) of the second term to the right, and returns the results. As many 0s as the number of shifted bits are inserted in the higher bits. |

| Symbol | Meaning | Explanation |
|---|---|---|
| SHL<br><< | Left shift | Shifts the value of the first term by the value (number of bits) of the second term to the left, and returns the results. As many 0s as the number of shifted bits are inserted in the lower bits. |
| **Byte separation operator** | | |
| HIGH | Higher byte | Of the lowest 16 bits of the term, returns the higher 8 bits. |
| LOW | Lower byte | Of the lowest 16 bits of the term, returns the lower 8 bits. |
| **Word separation operator** | | |
| HIGHW | Higher word | Of the 32 bits of the term, returns the higher 16 bits. |
| LOWW | Lower word | Of the 32 bits of the term, returns the lower 16 bits. |
| **Other** | | |
| ( | Left parenthesis | Performs the operation in ( ) before the operation outside ( ).<br>'(' and ')' are always used in pairs. |
| ) | Right parenthesis | |

**(3) Rules of operation**

Operations are performed according to the priority of the operators.

Table B-5 Operator Priority

| Priority | Operators |
|---|---|
| 1 Higher | ( , ) |
| 2 | + sign, - sign, NOT, ~, HIGHT, LOW, HIGHW, LOWW |
| 3 | *, /, MOD, %, SHR, >>, SHL, << |
| 4 | +, - |
| 5 | AND, & |
| 6 Lower | OR, \|, XOR, ^ |

- If the priorities of the operators are the same, the operation is performed from the left toward the right.
- Performs the operation in ( ) before the operation outside ( ).
- Each term in an operation is treated as unsigned 32-bit data.
- All operation results are treated as unsigned 32-bit data.
- If an overflow occurs during operation, the lower 32 bits are valid, and the overflow is not detected.

**(4) Terms**

To describe a constant for a term, the following numeric values can be described.

Table B-6 Range of Radixes

| Radix | Range |
| --- | --- |
| Binary number | 0**Y** <= value <= 11111111111111111111111111111111**Y** (32 digits) |
| Octal number | 0**O** <= value <= 37777777777**O** |
| Decimal number | -2147483648 <= value <= 4294967295<br>(A negative decimal number is internally converted into a 2's complement.) |
| Hexadecimal numbers | 0**H** <= value <= 0FFFFFFFF**H** |

## B.5 File Names

The following regulations apply to the source file names and load module file names.

**(1) Source file names and load module file names**

File names are composed of a to z, A to Z, 0 to 9, ., _, +, and -.

File names must start with a character other than ".".

File names cannot be prefixed or suffixed by a period (.) or space.

File names are not case-sensitive.

A file name consists of up to 259 characters including the path.

**(2) Other file names**

Other file names comply with Windows file name regulations.

The following characters cannot be used in file names.

\ / : * ? " < > | ;

File names cannot be prefixed or suffixed by a period (.) or space.

File names are not case-sensitive.

A file name consists of up to 259 characters including the path.

# APPENDIX C  KEY FUNCTION LIST

Table C-1 Key Function List

| Key | Function |
|---|---|
| BackSpace | Deletes one character before the cursor and moves the cursor to the position of the deleted character.<br>At this time, the character string following the cursor moves forward. |
| Delete | - Deletes one character after the cursor and move the character string following the cursor forward.<br>- Deletes a various event condition selected in the Event Manager or each event dialog box.<br>- Deletes the data selected in the Watch Window. |
| Insert | Alternately selects the insert mode and overwrite mode in the Source Window and Assemble Window. However, this key is invalid in the Memory, Register, and IOR Windows, and only the overwrite mode can be used as an input mode. |
| PrintScreen | Loads the entire display screen to the clipboard as a bitmap image (function of Windows). |
| Esc | - Closes the pull-down menu.<br>- Closes the modal dialog box.<br>- Restores the input data. |
| Alt | Moves the cursor to the menu bar. |
| End | Moves the cursor to the end of the line. |
| Home | Moves the cursor to the beginning of the line. |
| PageUp | Scrolls the screen one screen up. The cursor also moves up to the top of the screen. |
| PageDown | Scrolls the screen one screen down.  The cursor also moves up to the top of the screen. |
| Space | Inserts one blank character. |
| Tab | Moves the cursor to the next item. |
| Up arrow key | Moves the cursor up.<br>If the cursor is at the bottom of the screen, scrolls the screen up one line at a time. |
| Down arrow key | Moves the cursor down.<br>If the cursor is at the top of the screen, scrolls the screen down one line at a time. |
| Right arrow key | Moves the cursor to the left.<br>If the cursor is at the left most position on the screen, scrolls the screen one column to the right. |
| Left arrow key | Moves the cursor to the right.<br>If the cursor is at the right most position on the screen, scrolls the screen one column to the left. |
| Enter | - Sets the input data.<br>- Presses the default push button. |
| F1 | Opens the Help window. |

| Key | Function |
|---|---|
| F2 | Forcibly stops program execution.  Same function as [Run] menu -> [Stop]. |
| F3 | Resets the CPU.  Same function as [Run] menu -> [CPU Reset]. |
| F4 | Resets the CPU and executes the program.  Same function as [Run] menu -> [Restart]. |
| F5 | Executes the program.   Same function as [Run] menu -> [Go]. |
| F6 | Executes the program to the cursor position in the Source or Assemble Window.<br>Same function as [Run] menu -> [Come Here]. |
| F7 | The user program is real-time executed until execution returns.<br>Same function as [Run] menu -> [Return Out]. |
| F8 | Step execution.  Same function as [Run] menu -> [Step In]. |
| F9 | Sets a breakpoint at cursor position in Source or Assemble Window.<br>Same function as [Run] menu -> [Break Point]. |
| F10 | Next step execution.  Same function as [Run] menu -> [Next Over]. |
| F11 | Sets or deletes a software breakpoint.<br>Same function as [Run] menu -> [Software Break Point]. |
| Shift+End | Expands the selection range to the end of the line. |
| Shift+Home | Expands the selection range to the beginning of the line. |
| Shift+Left arrow key | Expands the selection range one character to the left. |
| Shift+Right arrow key | Expands the selection range one character to the right. |
| Shift+F6 | Executes the program from the cursor position in the Source or Assemble Window.<br>Same function as [Run] menu -> [Start From Here]. |
| Shift+F9 | Resets the CPU.<br>Same function as [Run] menu -> [CPU Reset]. |
| Ctrl+End | Displays the last line.  The cursor will also move to the last line. |
| Ctrl+Home | Displays the first line.  The cursor will also move to the first line. |
| Ctrl+Left arrow key | Moves the cursor one word to the left.<br>If the cursor at the left most position on the screen, scrolls the screen one column to the right. |
| Ctrl+Right arrow key | Moves the cursor one word to the right.<br>If the cursor at the right most position on the screen, scrolls the screen one column to the left. |
| Ctrl+F5 | Ignores break points being set, and executes the program.<br>Same function as [Run] menu -> [Ignore break points and Go]. |
| Ctrl+F9 | Sets the address at the cursor position in the Source Window or Assemble Window to the PC.<br>Same function as [Run] menu -> [Change PC]. |
| Ctrl+A | Selects all the events registered to the Event Manager.<br>Same function as [View] menu -> [Select All Event] in the Event Manager. |
| Ctrl+C | Copies a selected character string and saves it to the clipboard buffer. |

| Key | Function |
|---|---|
| Ctrl+D | Disassembles and displays the results from the jump destination address specified by the data value selected in the current window.Opens the Assemble Window.<br>Same function as [Jump] menu -> [Assemble]. |
| Ctrl+E | Opens the source file displayed in the active Source Window with the editor specified by the PM plus when the PM plus is running.  Same function as [Edit] menu -> [Edit Source]. |
| Ctrl+G | Performs a search.Opens the search dialog box corresponding to the current window.<br>Same function as [View] menu -> [Search...]. |
| Ctrl+J | Moves the display position. Opens the each dialog box, depending on the current window.<br>Same function as [View] menu -> [Move...]. |
| Ctrl+M | Displays the memory contents from the jump destination address specified by the data value selected in the current window. Opens the Memory Window.<br>Same function as [Jump] menu -> [Memory...]. |
| Ctrl+O | Loads a view file, source file, or text file.<br>Opens the View File Load Dialog Box. The operation will differ depending on the extension of the file.<br>view file: Displays the file in the corresponding window.<br>Others: Displays the file in the Source Window.<br>Same function as [File] menu -> [Open...]. |
| Ctrl+S | Saves the data displayed in the current window to the view file.<br>Same function as [View] menu -> [Save...]. |
| Ctrl+U | Displays the corresponding source text and source line, using the data value selected in the current window as the jump destination address. Opens the Source Window.<br>Same function as [Jump] menu -> [Source Text]. |
| Ctrl+V | Pastes the contents of the clipboard buffer to the text cursor position. |
| Ctrl+W | Temporarily displays the contents of the specified data.<br>Opens the Quick Watch Dialog Box.<br>Same function as [View] menu -> [Quick Watch...]. |
| Ctrl+X | Cuts a selected character string and saves it to the clipboard buffer.<br>Same function as [Edit] menu -> [Cut]. |
| Ctrl+Shift+Left arrow key | Expands the selection range one word to the left. |
| Ctrl+Shift+Right arrow key | Expands the selection range one word to the right. |

# APPENDIX D  MESSAGES

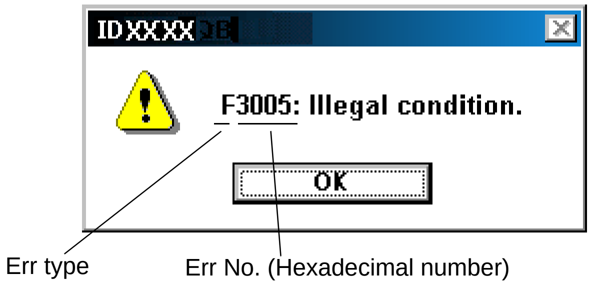
- Display Format
- Types of Messages
- Message Lists

## D.1 Display Format

Messages are output to the error/warning dialog box.

By pressing the F1 key while the error/warning dialog box is open, the related online help files are displayed.

Figure D-1 Error/Warning Messages




## D.2 Types of Messages

The ID850QB outputs the following types of messages.

Table D-1 Types of Messages

| Types | Meaning |
|---|---|
| A*xxxx* | A: Abort Error<br>Stops processing,  and terminates the debugger.<br>If this error occurs, debugging cannot be continued. |
| F*xxxx* | F: Fail<br>Stops processing, and opened windows and dialog boxes are closed. |
| W*xxxx* | W: Warning<br>Stops processing, but opened windows and dialog boxes are not closed. |

## D.3 Message Lists

< X0000~ > < X1000~ > < X2000~ > < X3000~ > < X4000~ > < X5000~ > < X6000~ > < X7000~ > < X8000~ > < X9000~ > < Xa000~ > < Xb000~ > < Xc000~ > <Xd000~ > < Xe000~ > < Xf000~ >

**(1) *X*0000~**

| | |
|---|---|
| F0002: | This feature is not supported. |
| F0100: | Can not communicate with ICE. Please confirm the installation of the device driver for the PC interface board.<br>1) The driver may not be correctly installed. Reinstall the driver. |
| A0101: | Can not find initialization file (expc.ini). |
| A0102: | Host name not found. |
| F0103: | Data transfer to ICE is timed out. Please confirm the power of ICE, connection of the interface cable, or I/O address of the PC interface board. |
| F0104: | Data receive from ICE is timed out. Please confirm the power of ICE, connection of the interface cable, or I/O address of the PC interface board. |
| A0105: | Failed in reading device file (d3xxx.800).<br>1) Necessary files may be damaged. Reinstall the device file. |
| A0106: | Illegal data received.<br>1) Check the power of the in-circuit emulator, cable connections, and setting of the interface board and restart the debugger. |
| A0107: | Can not communicate with ICE. |
| A0108: | Failed in reading initialization file (expc.ini). |
| A0109: | Can not communicate with ICE. Please terminate the debugger and check the power of ICE or the connection of cable then restart the debugger.<br>1) An error may have occurred during USB communication (such as disconnection of power or cable) or IECUBE is faulty (IECUBE). |
| F010a: | Can not communicate. Please confirm the availability of the communication port. |
| A01a0: | No response from the evachip. Please confirm the signal of the CLOCK or RESET WAIT, HLDRQ and so on.(IECUBE)<br>No response from the CPU. Please confirm the signal of the CLOCK or RESET WAIT, HLDRQ and so on.(N-Wire CARD, MINICUBE)<br>1) Check the HOLD signal, WAIT signal, clock signal, etc. The IOR value may not be correct. |
| A01a1: | Failed in reading ie703000.ie. |
| A01a2: | Break board is not connected. |
| A01a3: | Emulation board is not connected. |
| A01a4: | Board configuration of ICE is not consistent. |
| A01a5: | POD/EM1 board is not connected. |

| | |
|---|---|
| A01a6: | Executor is running. |
| A01a8: | Failed to find configuration file (lv8hw.ini). |
| A01ad: | Please update the device driver for the PC interface board.<br>1) The device driver may be old. Install the latest device driver. |
| A01ae: | Failed in reading configuration file (lv8hw.ini). |
| A01af: | Failed in executing monitor command. |
| A01b0: | Can not communicate with monitor program. Please check the availability of communication port, the setting of CPU board or the type of cable. |
| A01b1: | Can not communicate with monitor program. Please terminate the debugger and check the power of CPU board or the connection of cable then restart the debugger. |
| F0200: | Verification error occurred. Failed in writing memory.<br>1) External memory could not be accessed, as it is not set. Change the register values necessary for accessing the external memory using the IOR Window or Hook Procedure before download. |
| F02a0: | Bus hold error.<br>1) CPU is in the bus-hold status. Reset the debugger. |
| F02a2: | Can not compulsory break. |
| F02a3: | Reset under continuation. |
| F02d2: | Not enough memory for trace-buffer. |
| F0300: | User program is running. |
| F0301: | User program is being breaked. |
| F0302: | User program is being traced. |
| F0303: | Not traced. |
| F0304: | Trace memory is not set. |
| F0306: | No trace block exists. |
| F0307: | No event condition exists. |
| F0308: | No timer measurement is done. |
| F0309: | No trigger frame exists. |
| F030a: | Tracer is being stopped. |
| F030b: | Specified snap-event has not been registered. |
| F030c: | Specified stub-event has not been registered. |
| F030d: | Timer is running. |
| F030e: | Memory copy area is overlapped. |
| F030f: | Trace has been already set. |
| F0310: | Event condition is not set. |
| F0311: | Too many valid timer event conditions. |
| F0312: | Specified timer event is not set. |

| | |
|---|---|
| F0313: | Illegal map range.<br>1) Check the map range in the Configuration Dialog Box. When mapping to external memory has been performed, change the register values necessary for accessing the external memory using the IOR Window or Hook Procedure before download). |
| F0314: | Only trace delay mode can set with delay trigger. |
| F0315: | Delay trigger cannot set without trace delay mode. |
| F0316: | Overflowed the number of mapping. |
| F03a0: | Target is not turned on.<br>1) Check the target power supply. Check the cable connecting the in-circuit emulator and target board. Check that the VDD signal is input to the connector of the target board. |
| F03a1: | Step execution is being done. |
| F03a2: | Timer and Tracer are running. |
| F03a3: | Event link and BRS events are mixed. |
| F03d0: | Back-trace is being executed. |
| F03d1: | Back-trace is being stopped. |
| F03d2: | Back-trace execution point overrun oldest frame. |
| F03d3: | Register status or Memory status cannot be set up other than Phase1 of event link. |
| F03d4: | No back-trace information exists. |
| F03d5: | Last command can not be backstepped. |
| F0400: | Illegal condition.<br>1) Settings of the used in-circuit emulator and those of the Configuration Dialog Box may not match. Check the Chip selection. |
| F0401: | Result of timer measurement overflowed. |
| F0402: | Too many event conditions with path count. |
| F0403: | Too many address range conditions. |
| F0404: | Too many simultaneously-usable-event conditions. |
| F0405: | Too many snap-events. |
| F0406: | Too many stub-events. |
| F0407: | Too many initialization data. |
| F0408: | Too large search data (> 16 byte). |
| F0409: | Too large search data (> search range). |
| F040a: | Too many Linking-event conditions. |
| F04a0: | Software break conditions number overflow. |
| F04a1: | Not enough memory for emulation. |
| F04a2: | Too many partition of bus size. |
| F04a3: | Too many execution-event conditions. |
| F04a4: | Too many bus-event conditions. |

| | |
|---|---|
| A0600: | Not enough memory for buffer.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| A0601: | Not enough resource of operating system. |
| F0b20: | This event number can not be used. |
| F0b61: | Section Trace event conditions overflow. |
| F0b66: | Cannot use the break before execution event and the software break at the same time.<br>1) This is because a break before execution is used for implementing a software break (when MINICUBE is connected). |
| F0b80: | Reset by hardware error. |
| F0c00: | Monitor file read error.<br>1) Necessary files may be damaged. Reinstall the debugger. |
| A0c01: | During access of register, CPU did time out.<br>1) Check the clock signal, etc. The register value may not be correct. |
| A0c02: | During access of memory, CPU did time out.<br>1) Check the HOLD signal, WAIT signal, clock signal, etc. The memory value may not be correct. |
| A0c03: | During access of I/O register, CPU did time out.<br>1) Check the HOLD signal, WAIT signal, clock signal, etc. The I/O register value may not be correct. |
| F0c20: | Guarded area can not be accessed. |
| F0c21: | Memory was unready status. |
| F0c22: | Memory unready status was canceled. |
| F0c23: | Bus hold under continuation.<br>1) Check the setting of the target board, or mask the HOLD pin. |
| F0c24: | It cannot shift to debug mode.<br>1) Check the clock signal. This may be caused by a stopped clock or a slow clock. |
| F0c25: | Flash macro service ROM was accessed or stepped in.<br>1) Please perform [Go] execution or CPU reset. |
| F0c26: | FLMD terminal is in a write-protected state.<br>1) FLMD is not in the write-enabled status. Check the status of the FLMD0 and FLMD1 pins. |
| F0c27: | Security flag is in a write-protected state.<br>1) The security flag of the flash memory has disabled writing, block erasure, or chip erasure. Nothing can be written to the flash memory. |
| F0c28: | Internal RAM is not enough, the writing to flash memory is not made.<br>1) The internal RAM size is less than 4 KB and flash self-programming cannot be executed. |
| F0c29: | The blank check of flash memory failed. |
| F0c2a: | The erasing of flash memory failed. |
| F0c2b: | The writing of flash memory failed. |
| F0c2c: | The internal verification of flash memory failed. |

| | |
|---|---|
| F0c2d: | Failed in writing flash memory. |
| F0c2e: | There is no response from flash macro service. |
| F0c2f : | Response from flash macro service is not right. |
| F0c30: | Flash I/O register operation prohibition setup needs to be canceled. |
| F0c31: | STOP mode under continuation. Can not compulsory break. Please release STOP mode or reset the CPU. |
| F0c32: | Please write in flash memory in the single chip mode 0. |
| F0c33: | Disabling the on-chip debug function is prohibited. |
| F0c34: | Writing to the on-chip debug reserved area is prohibited. |
| F0c36: | Abnormal Internal ROM size. The size is different from the default of the device. |
| F0c37: | The voltage is too low to operate flash programming. |
| F0c40: | Status of effective event conditions cannot be changed. |
| F0c41: | Coverage test is being executed. |
| F0c42: | Monitor has failed in shift in the debugging mode.Please reset the CPU. |
| F0c43: | Can not communicate with ICE. Please confirm the power of ICE, connection of the interface cable.<br>1) The switch setting may be wrong if a desktop computer is used and two or more PC cards are inserted. Check the setting. Or it may have malfunctioned (when N-Wire CARD, MINICUBE connected).<br>2) Check the power to the in-circuit emulator and cable connections The switch setting may be wrong if a desktop computer is used and two or more PC cards are inserted. Check the setting. Or it may have malfunctioned (when IECUBE connected). |
| F0c44: | Coverage test is being executed. |
| F0c45: | Inside of Power off reset emulation cannot carry out program execution. |
| F0c46: | This function is not valid during Flash Self Emulation. |
| F0c60: | Event before execution cannot be set up other than break conditions. |
| F0c61: | Can not register event numbers which can not be used for hardware break. |
| F0c62: | Event numbers reserved for hardware breaks can not be used. |
| F0c63: | Event link conditions cannot set. |
| F0c64: | Too many ROM-emulation-RAM areas. |
| F0c67: | Writing of flash memory during block is not made. |
| F0c70: | DCU cannot be accessed.<br>1) The device file selection may be incorrect. Select a device file that supports the target chip in Chip Selection in the Configuration Dialog Box. Check the power to the chip. Check the connection of the signal lines (DCK, DMS, DDI, DDO, and DRSTZ). Check the noise level by a DCK waveform test on the N-Wire Checker (when the N-Wire CARD, MINICUBE is connected).<br>2) IE may be malfunctioning (when the IECUBE is connected). |

| | |
|---|---|
| F0c71: | Reset cannot be performed.<br>1) Check the clock signal. This may be caused by a stopped clock or a slow clock. |
| F0c72: | Monitor memory cannot be accessed.<br>1) Revise the Main OSC value in the Configuration Dialog Box. Check the setting of main clock in the Configuration Dialog Box , the noise level by a DCK waveform test on the N-Wire Checker. The problem may also be caused by an internal chip problem (when the is N-Wire CARD, MINICUBE, IECUBE connected).<br>2) Revise the Main OSC value in the Configuration Dialog Box. If this does not solve the problem, IE may be malfunctioning (when the IECUBE is connected). |
| F0c73: | Monitor execution cannot be performed.<br>1) Check the noise level by a DCK waveform test on the N-Wire Checker. The problem may also be caused by an internal chip problem (when the N-Wire CARD, MINICUBE is connected).<br>2) IE may be malfunctioning (when the IECUBE is connected). |
| F0c74: | CPU register cannot be accessed.<br>1) Check the noise level by a DCK waveform test on the N-Wire Checker. The problem may also be caused by an internal chip problem (when theN-Wire CARD, MINICUBE is connected).<br>2) The device file selection may be incorrect. Select a device file that supports the target chip in Chip Selection in the Configuration Dialog Box. If this does not solve the problem, the IE may be malfunctioning (when the IECUBE is connected). |
| F0c75: | Monitor has failed in shift in the debugging mode. Please reset the CPU. |
| F0c76: | Initial state at the time of DCU access start is unusual.<br>1) Initial state at the time of DCU access start is unusual (does not start and remains in the reset state). Check the connection of the signal lines (DCK, DMS, DDI, DDO, and DRSTZ). Check the noise level by a DCK waveform test on the N-Wire Checker (when the N-Wire CARD, MINICUBE is connected).<br>2) The device file selection may be incorrect. Select a device file that supports the target chip in Chip Selection in the Configuration Dialog Box. If this does not solve the problem, the IE may be malfunctioning (when the IECUBE is connected). |
| F0c77: | DCU access is unusuall.<br>1) DCU access is unusuall (verify error). Check the connection of the signal lines (DCK, DMS, DDI, DDO, and DRSTZ). Check the noise level by a DCK waveform test on the N-Wire Checker (when the N-Wire CARD, MINICUBE is connected).<br>2) IE may be malfunctioning (when the IECUBE is connected). |
| F0c78: | Failed in reading of trace data. |
| F0ca0: | Can not communicate with ICE. Please confirm the power of ICE, connection of the interface cable, or I/O address of the PC interface board.<br>Error occurred inside debugger. (IECUBE)<br>1) Can not communicate with in-circuit emulator. Check the power of the in-circuit emulator, cable connections, and setting of the interface board etc. |

| | |
|---|---|
| F0ca1: | Monitor file not found.<br>1) Necessary files may be damaged. Reinstall the debugger. |
| F0ca2: | This device file does not include the on-chip debug information.<br>1) An attempt was made to start with a device file not supporting on-chip debugging. The device file may be old. Install the latest device file (when the N-Wire CARD, MINICUBE is connected).<br>2) IE may be malfunctioning (when the IECUBE is connected). |
| F0ca3: | Unsupported information is included in the on-chip debug information in the device file.<br>1) An unknown flag is included in the on-chip debug information of the device file.The exec module may be old. Install the latest exec module. |
| F0ca4: | This device file does not include the IECUBE information.<br>1) An attempt was made to start with a device file not supporting IECUBE. The device file may be old. Install the latest device file. |
| F0caf: | Trace block can not be stepped over. |

**(2) *X*1000~**

| | |
|---|---|
| A1000: | Failed in initializing ICE. |
| A1001: | No entry exists for specified number. |
| A1002: | Can not relocate internal RAM. |
| F1003: | Illegal relocation address. |
| F1004: | Illegal condition. |
| A1005: | Invalid attribute. |
| F1006: | Illegal address. |
| A1007: | Not enough memory on ICE. |
| A1008: | Not enough memory for tables.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| A1009: | Already initialized. |
| A100a: | Not initialized. |
| F100b: | User program is running. |
| F100c: | Different bus size has been already specified. |
| F100d: | Too large bus size. |
| F100e: | Too large bus partition size. |
| W100f: | Target is not turned on. |
| F1010: | Illegal map range. |
| F1011: | Failed in setting internal ROM and RAM. |
| F1012: | This feature is not supported. |
| F1013: | No terminal name. |
| W1014: | Data is not exist. |

| | |
|---|---|
| A1015: | Programmable-IOR does not exist. |
| F1016: | Programmable-IOR does not movable.<br>1) Necessary files may be damaged. Reinstall the latest device file. |
| F1017: | I/O Protect mapping is possible a target attribute only. |
| F1018: | Illegal Internal ROM size. |
| A10ff: | Can not communicate with ICE. |
| A1dbe: | Error occurred inside debugger. |

**(3) *X*2000~**

| | |
|---|---|
| F2000: | Illegal IOR name. |
| A2001: | Illegal address. |
| F2002: | User program is running. |
| F2003: | Illegal IOR number. |
| F2004: | Illegal bit number. |
| W2005: | IOR of Read Protect attribute was specified. |
| F2006: | Hidden IOR was specified. |
| F2007: | IOR of ban read or write was specified. |
| F2008: | IOR not existing was specified. |
| A2009: | Device file is damaged or error is in file. |
| F200a: | Illegal value specified for IOR. |
| A200b: | Can not copy. |
| A200c: | Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| W200d: | No initialize data for IOR. |
| F200e: | IOR area can not be accessed. |
| A20ff: | Can not communicate with ICE. |
| A2222: | Illegal condition. |

**(4) *X*3000~**

| | |
|---|---|
| F3000: | No mapped address was accessed.<br>1) The allocation addresses of the program and the addresses of the debugger may not match. Set the mapping to the external memory in the Configuration Dialog Box according to the allocation addresses specified in the link directive file on compilation. When mapping to external memory has been executed, change the register values necessary for accessing the external memory using the IOR Window or Hook Procedure before download. |
| F3001: | Memory has different value. |

| | |
|---|---|
| F3002: | Illegal start address. |
| F3003: | Illegal end address |
| F3004: | Illegal start address and end address. |
| F3005: | Illegal condition. |
| F3006: | User program is running. |
| F3007: | Verification error. |
| F3008: | No condition specified. |
| F3009: | Parameter size does not align with access size alignment. |
| F300a: | Specified address does not align with access size alignment. |
| F300b: | Source address does not align with access size alignment. |
| F300c: | Destination address does not align with access size alignment. |
| F300d: | Illegal end address. |
| F300e: | Different access size in specified area. |
| F300f: | Different access size both in source and destination areas. |
| F3010: | Different access size in destination area. |
| F3011: | Different access size, source & destination. |
| A3012: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| F3013: | Failed in writing DMM. |
| F3014: | Oveflowed mapping area. |
| F3015: | Processing was interrupted. |
| F3016: | This feature is not supported. |
| A30ff: | Can not communicate with ICE. |

**(5)  *X*4000~**

| | |
|---|---|
| F4000: | Can not delete specified event.<br>1) The specified event cannot be deleted as it is being used under another condition.  Invalidate it for other usages before deleting. |
| F4001: | Illegal table number. |
| F4002: | Illegal start address. |
| F4003: | Illegal end address. |
| F4004: | Illegal status. |
| F4005: | Illegal data. |
| F4006: | Specified event number has been already used. |
| F4007: | Too many same events are registered. |
| F4008: | Specified event has not been registered. |

| | |
|---|---|
| F4009: | Illegal data size. |
| F400a: | Illegal mode. |
| F400b: | Setting value is inaccurate. |
| F400c: | Event link conditions cannot be used for section trace conditions. |
| F400d: | Too many identical events are registered (>= 32767). |
| F400e: | Specified event condition does not exist. |
| F400f: | Illegal event link condition. |
| F4010: | Function not found. |
| A4011: | Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| F4012: | Timer is being disabled. |
| W4013: | Access size is different from its mapped bus size. |
| F4014: | Can not use software break. |
| F4015: | Can not use event condition specifying address range. |
| F4016: | Can not change event condition. |
| F4017: | Can not access word at odd address. |
| A4018: | Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| F4019: | This feature is not supported. |
| F401a: | No Event. |
| F401b: | Can not use tag-event. |
| W401c: | Software break can not be set on this area. |
| F401d: | Start event and end event of timer are not made to the same setup. |
| F401e: | Too many trace-events. |
| F401f: | Path count cannot be set up. |
| F4020: | Address range cannot be set up in event before execution. |
| F4021: | Event conditions number overflow. |
| F4022: | Software DMM conditions number overflow. |
| F4023: | Real-time call conditions number overflow. |
| F4024: | Software break call conditions number overflow. |
| F4025: | Illegal snap condition. |
| F4026: | Too many event conditions cannot be set as Phase1 and Phase2 of event link conditions. |
| F4027: | Software break conditions number which can be set as internal ROM was overflow. |
| F4318: | Illegal memory bank setting. |

**(6) *X*5000~**

| | |
|---|---|
| A5000: | Illegal device file type. |
| A5001: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| A5002: | Can not open device file. |
| A5003: | Reading of device file went wrong. |
| A5004: | Can not close device file. |
| A5005: | Illegal device file format.<br>1) Necessary files may be damaged.  Reinstall the device file. |
| A5006: | Failed in initializing ICE. |
| A5007: | Device file has broken or error is in a file. |
| F5008: | Can not open device file.<br>1) Necessary files may be damaged.  Reinstall the device file. |
| F5009: | Can not open ie703000.ie. |
| F500a: | Specified device file is illegal version.<br>1) Necessary files may be damaged.  Reinstall the device file. |
| W500b: | Specified device file does not relocate IRAM. |
| A500c: | Failed in reading expc.ini. |
| A500d: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| W500e: | No tag data which it was going to refer to device file. |
| A5300: | Illegal device file type. |
| A5301: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| A5302: | Can not open database file.<br>1) Necessary files may be damaged.  Reinstall the debugger and device file. |
| A5303: | Reading of database file went wrong. |
| A5304: | Can not close database file. |
| A5305: | Illegal database file format.<br>1) Necessary files may be damaged.  Reinstall the debugger, and device file. |
| A5306: | Database information has been already initialized. |
| A5307: | Database information does not exist. |
| F5308: | Can not open specified database file.<br>1) Necessary files may be damaged.  Reinstall the debugger. |
| F5309: | Specified database file is illegal version.<br>1) Necessary files may be damaged.  Reinstall the debugger, and the device file. |

**(7) *X*6000~**

| |
|---|
| F6000: Current function does not exist. |
| F6001: Illegal symbol name. |
| F6002: Illegal condition. |
| F6003: Illegal function name. |
| F6004: Overflowed output buffer size. |
| F6005: Illegal expression. |

**(8) *X*7000~**

| |
|---|
| F7000: Illegal mode. |
| F7001: User program is running. |
| F7002: User program has been stopped. |
| F7003: Trace enabled. |
| F7004: Trace memory is not set. |
| F7005: Function return address does not exist, can not do step execution. |
| W7010: No source information exists. |
| W7011: Unknown result of step execution. |
| A7012: Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| A70fe: Bus hold error.<br>1) CPU is in the bus-hold status. Reset the debugger. |
| A70ff: Can not communicate with ICE. |
| F7801: End waiting state of step execution was canceled. |
| F7802: End waiting state of step execution was canceled. |
| F7f00: Aborted step execution. |
| F7f02: Suspended step execution. |
| A7f03: Failed in canceling RUN/STEP. |
| F7f04: Can not execute non-mapped area. |
| F7f05: This feature is not supported. |

**(9) *X*8000~**

| |
|---|
| F8000: Specified file was not found. |
| F8001: Illegal line number. |
| F8002: Current information is not set. |
| F8003: Illegal address. |
| F8004: This feature is not supported. |

**(10) *X*9000~**

| | |
|---|---|
| A9000: | Specified register symbol does not exist. |
| A9001: | Specified register symbol ID does not exist. |
| F9002: | Illegal value. |
| A9003: | Illegal condition. |
| A9004: | Too large register size. |
| F9005: | This feature is not supported. |

**(11) *X*a000~**

| | |
|---|---|
| Fa001: | Illegal expression. |
| Fa002: | Start address is bigger than the end address. |
| Fa003: | Illegal source path. |
| Fa004: | Too long expression. |
| Aa005: | Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| Fa006: | Illegal argument. |
| Fa007: | Illegal program number. |
| Fa008: | Source path is not set. |
| Fa009: | File not found. |
| Fa00a: | Can not open file.<br>1) The file is damaged or does not exist. Recreate the file. |
| Aa00b: | Can not close file. |
| Aa00c: | Failed in reading file.<br>1) The file is damaged or does not exist. Recreate the file. |
| Fa00d: | Not source file of load module. |
| Fa00e: | Illegal line number. |
| Fa00f: | Variable does not exist. |
| Aa010: | Can not communicate with ICE. |
| Fa011: | Can not access register. |
| Fa012: | Can not access memory. |
| Aa013: | Reading of file went wrong. |
| Fa014: | It was going to open the binary file. |
| Fa015: | Can not get temporary path.<br>1) The disk is full. Delete or move unnecessary files and increase the available memory in the disk. |
| Fa016: | Can not create temporary file.<br>1) The disk is full. Delete or move unnecessary files and increase the available memory in the disk. |

| | |
|---|---|
| Fa017: | Can not remove temporary file. |
| Fa020: | This feature is not supported. |
| Fa021: | Symbol assigned to register cannot be specified. |
| Fa022: | The character whitch cannot be used for the folder is contained or the folder does not exist. |

**(12)  *X*b000~**

| | |
|---|---|
| Fb000: | Illegal command line. |
| Fb001: | Program information does not exist in specified load module file. |
| Fb002: | File not found. |
| Fb003: | Function not found. |
| Fb004: | Selected load module different from kind (Chip) was loaded. |
| Fb005: | Symbol not found.<br>1) The address could not be found.  Specify a location holding address information. |
| Fb008: | Illegal expression. |
| Ab009: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| Fb00a: | Illegal symbol in load module file. |
| Fb00b: | Current program does not exist. |
| Fb00c: | Current file does not exist. |
| Ab00d: | Current function does not exist. |
| Ab00e: | Current line does not exist. |
| Ab00f: | Tag not found. |
| Ab010: | Failed in loading symbol table. |
| Ab011: | Illegal line number. |
| Fb012: | Too large line number. |
| Ab015: | Reading of file went wrong.<br>1) The file is damaged or does not exist.  Recreate the file. |
| Ab016: | Can not open file.<br>1) The file is damaged or does not exist.  Recreate the file. |
| Ab017: | Failed in writing file.<br>1) The file is damaged or does not exist.  Recreate the file. |
| Ab019: | Reading of file went wrong. |
| Ab01a: | Can not close file. |
| Fb01b: | Too long load module file name. |
| Ab01c: | Too many entries of the task kind. |
| Fb01d: | Address not found. |

| | |
|---|---|
| Wb01e: | No debug information (not compiled in Debug Build mode). |
| Fb01f: | Can not find structure member. |
| Fb020: | Can not find value. |
| Fb021: | No debug information exists in load module file.<br>1) To create a load module with appended debug information, execute build in build mode of Debug Build. |
| Fb022: | Illegal line number. |
| Ab023: | Current stack frame is not active. |
| Ab024: | Different section. |
| Fb026: | Too many array dimensions (> 4). |
| Fb027: | Found end of file.<br>1) The specified file may be damaged. Recreate the file. |
| Fb028: | This feature is not supported. |
| Fb029: | Illegal address. |
| Ab02a: | Can not communicate with ICE. |
| Fb02b: | Can not stack trace with current PC value. |
| Fb02c: | Too many blocks for one function. |
| Fb02d: | Illegal argument. |
| Fb02e: | The file does not exist in the SOURCE PATH.<br>1) On stopping the program, the source that the debugger tried to display could not be found. Check if the path connects to the source in the Debugger Option Dialog Box, or check if the source is in the same directory as the out file. Refer to the Assemble Window on which the error message is displayed, and check if the corresponding path connects. |
| Fb02f: | Information has been deleted because of optimization. |
| Ab030: | Monitor timed out.<br>1) Check the power of the in-circuit emulator, cable connections, and setting of the interface board and restart the debugger. |
| Ab031: | Already set in memory. |
| Ab032: | Out of scope. |
| Ab033: | LP is not stored. |
| Fb034: | Return execution from present PC position cannot be performed. |
| Fb037: | Too Many Line-Numbers Information. |
| Fb038: | Compiler version mismatch.<br>1) Recreate the load module with the latest compiler. |
| Ab039: | Failed in loading debug information. |
| Ab03a: | No more section information. |

| | |
|---|---|
| Fb040: | Specified file is not load module.<br>1) This is not a linker output file. Source debug cannot be executed with the load module before output from the linker. Specify the load module output from the linker. |
| Ab041: | Too many files in load module to download. |
| Wb042: | Symbol module is not initialized. |
| Fb32e: | Illegal port number. |
| Fb32f: | Illegal port name. |
| Fb330: | Illegal port position. |
| Fb331: | Illegal increment number. |
| Fb332: | Port for memory bank is not set. |
| Fb333: | Illegal bank number. |
| Fb334: | Area for memory bank is not set. |
| Wb335: | Too long symbol name. |

**(13) *X*c000~**

| | |
|---|---|
| Fc001: | Can not open file.<br>1) The file is damaged or does not exist. Recreate the file. |
| Ac002: | Can not close file. |
| Ac003: | Reading of file went wrong.<br>1) The file is damaged or does not exist. Recreate the file. |
| Ac004: | Reading of file went wrong. |
| Fc005: | Illegal file type. |
| Fc006: | Kind (Chip) of load module is illegal. |
| Fc007: | Specified file is not load module.<br>1) This is not a linker output file. Source debug cannot be executed with the load module before output from the linker. Specify the load module output from the linker. |
| Fc008: | Specified load module file (ELF) is old version. |
| Ac009: | Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| Fc00a: | No mapped address was accessed. |
| Fc00b: | Load module is not loaded. |
| Fc00c: | Illegal argument. |
| Fc00d: | User program is running. |
| Fc00e: | User program is being traced. |
| Fc00f: | Interrupted. |
| Ac010: | Can not communicate with ICE. |
| Fc011: | Illegal load module file format. |

| |
|---|
| Fc012: Check sum error. |
| Fc013: Too wide address range to upload (> 1M byte). |
| Fc014: Failed in writing file.<br>1) The file is damaged or does not exist. Recreate the file. |
| Fc015: Illegal program number. |
| Fc016: Load information is full. |
| Wc017: Symbol information is duplicated, please reset symbols. |
| Fc018:Specified file is not load module.<br>1) This is not a linker output file. Source debug cannot be executed with the load module before output from the linker. Specify the load module output from the linker. |
| Fc019: Failed in writing memory. |
| Wc01a: BSS area is assigned to non-mapped area.<br>1) When the program is executed, a non-map break may occur. Either allocate the BSS area to the internal RAM by using a link directive, or map the emulation memory or target memory to the BSS area using the Configuration Dialog Box of the debugger. |
| Fc01b: Programmable-IOR address not specified.<br>1) Necessary files may be damaged. Reinstall the debugger. |
| Wc01c: Programmable IOR address mismatch.<br>1) Necessary files may be damaged. Reinstall the debugger. |
| Wc01d: Selected load module different from kind (Chip) was loaded. |
| Fc01e: .Flash erase is not supported. |
| Fc100: This feature is not supported. |

**(14) *X*d000~**

| |
|---|
| Ad000: Error occurred inside debugger. |
| Ad001: Not enough memory.<br>1) There is not enough system memory. Close the applications being executed and the open files. |
| Ad002: Failed in reading initialization file (expc.ini). |
| Ad003: ICE is not connected. |
| Fd004: Can not find Dynamic Link Library. |

**(15) *X*e000~**

| |
|---|
| Fe000: Illegal argument. |
| Fe001: Illegal start address. |
| Fe002: Illegal end address. |
| Fe003: Too large size. |

| | |
|---|---|
| Fe004: | Can not open file.<br>1) The file is damaged or does not exist.  Recreate the file. |
| Fe005: | Failed in reading file.<br>1) The file is damaged or does not exist.  Recreate the file. |
| Fe006: | Reading of file went wrong. |
| Fe007: | Failed in writing file.<br>1) The file is damaged or does not exist.  Recreate the file. |
| Ae008: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| Fe009: | Illegal file format. |
| Fe00a: | Verification error. |
| Fe010: | This feature is not supported. |

**(16)  *X*f000~**

| | |
|---|---|
| Af000: | Not enough memory.<br>1) There is not enough system memory.  Close the applications being executed and the open files. |
| Ff000: | Not enough memory. |
| Ff001: | [XXX] not found. |
| Wf002: | Not found [XXX]. Search from the beginning? |
| Wf003: | Already exceed search region. |
| Ff004: | Missing parameter. |
| Ff005: | Illegal function name. |
| Ff006: | Illegal number. |
| Ff007: | Start address is bigger than end address. |
| Ff008: | Illegal symbol or expression. |
| Ff009: | [XXX] This file is illegal type. |
| Ff100: | Disk cannot write or full. |
| Ff101: | File not found. |
| Ff102: | File not Create. |
| Ff103: | Old project file version. |
| Ff104: | Illegal project file format. |
| Ff105: | This file is a project file for [XXX].Please select a correct file. |
| Wf106: | CPU in the Project File was Changed.  You must exit the debugger for the new CPU.  Do you exit the Debugger? |
| Wf107: | CPU in the Project File was Changed.  Do you start the Debugger with this CPU? |
| Wf108: | Selected project file different [YYY] from chip [XXX] was opened.  Does it open, although the chip cannot be changed? |

| | |
|---|---|
| Wf109: | Project Manager cannot be used with the debugger of this version. Please use PMplus. |
| Wf200: | No difference encountered. |
| Ff201: | Memory mapping error. |
| Ff202: | Verify error.<br>1) External memory could not be accessed, as it is not set. Change the register values necessary for accessing the external memory using the IOR Window or Hook Procedure before download . |
| Wf203: | When a program is running, while rewriting a memory, program execution stops for a moment. Do you wish to rewrite a memory? |
| Wf300: | Would you like to save the changes made in [XXX]? |
| Ff301: | The symbol being used on the event condition can't be evaluated. |
| Wf302: | Delete: [XXX] |
| Wf303: | [XXX] is edited. Delete: [YYY]? |
| Wf304: | [XXX] is edited. Save: [YYY]? |
| Wf305: | [XXX] is already exist. Do you replace it? |
| Ff306: | This name is too long. |
| Ff307: | There is the same name in other kinds. |
| Ff308: | An address can't be omitted. |
| Ff309: | Illegal address mask. |
| Ff30a: | Illegal data mask. |
| Ff30b: | Illegal ext probe mask. |
| Ff30c: | Illegal ext probe data. |
| Ff30d: | Illegal pass count. |
| Ff30e: | Illegal register name. |
| Ff310: | Illegal delay count. |
| Wf311: | Only one [XXX] can be enabled. Do you make this [YYY] to enable? |
| Ff312: | [XXX] is already there. |
| Ff313: | Event number already exist. |
| Ff314: | Event name is not set. |
| Ff315: | [XXX] is already there. |
| Ff316: | Max number of enabled [XXX] event is over. Please disable other enabled [YYY] event. |
| Ff317: | Max number of set event is over. |
| Ff31e: | Illegal start address. |
| Ff31f: | Illegal end address. |
| Ff322: | Illegal count rate. |
| Ff323: | Illegal time out break count. |
| Ff324: | Section and Qualify can be specified at the same time. |
| Wf325: | User program is running. Do you want to stop user program for a moment and set it? |

| | |
|---|---|
| Wf326: | User program is running. Do you want to stop user program for a moment and delete it? |
| Ff350: | There is a phase which event are not in the middle. |
| Ff351: | The same event is contained in Link and Disable. |
| Ff352: | An event isn't specified. |
| Ff357: | AND event is in Phase. |
| Ff400: | Coverage mapping error. |
| Wf401: | Clear coverage? |
| Ff500: | Illegal symbol. |
| Ff501: | Illegal value. |
| Ff502: | Illegal parameter. |
| Ff503: | Max number of symbol is over. |
| Ff504: | This variable cannot be set as a break.<br>1) Break cannot be set for the following variables.<br>- Local variables, static variables<br>- Array variables, member variables of structures/unions<br>- Register/peripheral I/O registers<br>- Variable expressions |
| Wf600: | Save project file? |
| Wf601: | When connecting the target system, please turn on the target system.<br>1) When a target is not connected, simply click the <OK> button. |
| Wf602: | Please change a MODE mask condition or connect the target system. |
| Ff603: | Incorrect ID Code.<br>1) This may be caused by the following (when the N-Wire CARD, MINICUBE is connected).<br>- The ID code is incorrect.<br>-> Input the correct ID code.<br>- The internal flash memory is in the write mode because the FLMD0 pin is high.<br>-> Make the FLMD0 pin low.<br>- The emulator connection prohibition mode is set because the ID code (bit 7 of address 0x79) is 0.<br>->Erase the internal flash memory once. |
| Af604: | Incorrect ID Code. Abort the debugger.<br>1) This may be caused by the following (when the N-Wire CARD, MINICUBE is connected)<br>- The ID code is incorrect.<br>-> Input the correct ID code.<br>- The internal flash memory is in the write mode because the FLMD0 pin is high.<br>-> Make the FLMD0 pin low.<br>- The emulator connection prohibition mode is set because the ID code (bit 7 of address 0x79) is 0.<br>->Erase the internal flash memory once. |

| | |
|---|---|
| Ff605: | Please check connection with the target board.<br>1) Check the connection of the target connector (TC). If a target is not connected, review the Target setting in the Configuration Dialog Box. |
| Ff606: | Please check connection with the target board, and power on it.<br>1) Check the target power supply. If a target is not connected, review the Target setting in the Configuration Dialog Box. |
| Wf607: | Please check connection of the exchange adapter.<br>1) Check the connection of the exchange adapter (EA).<br>Recommend wearing of the exchange adapter, if the target is not connected. |
| Ff608: | Please disconnect the target board.<br>1) A current may flow from the internal power supply to the target. Disconnect the target connector (TC) from the conversion adapter (EA). Review the setting in the Configuration Dialog Box if the target is not connected. |
| Ff609: | Please power off the target board, and disconnect it. |
| Af60c: | During break Target was not turned on. |
| Wf700: | Do you want to download Load Module File? |
| Wf701: | Do you load symbol information only? |
| Ff802: | All events are deleted. because the use of external probe was changed. |
| Ff803: | This event address is invalid on current configuration. |
| Ff804: | Invalid PC value. |
| Ff805: | Cannot set temporary break on this address. |
| Ff806: | External data is being used by Debugger. |
| Ff900: | Illegal I/O port name. |
| Ff901: | Memory mapping error.<br>1) The specification of the address is illegal. Check the addresses that can be specified in the Add I/O Port Dialog Box. |
| Ff902: | Illegal access size. |
| Ff903: | Illegal access type. |
| Ff904: | There is the same name. |
| Wf905: | [XXX] is already exist. Do you replace it? |
| Wf906: | Would you like to register the change made in [XXX]? |
| Ffa00: | The [XXX] function of current program on PC position not found.<br>1) The symbol specified in main() label: in the Debugger Option Dialog Box could be found. Set a symbol of the main routine of the program. Default is _main. |

| | |
|---|---|
| Ffa01: | The line information on PC position not found.<br>1) The source file corresponding to program counter (PC) value when the program was stopped could not be found. The following reasons are possible.<br>-The source file exists in a location that the source path does not connect to.<br>-The program stopped where the source files, such as library or RX, do not exist.<br>-The program looped, jumped to an address that is not used by the program, and stopped there. |
| Wfb00: | User program is running. Do you want to stop user program?<br>1) <Yes> button is selected, execution of the user program is stopped and then the Exit Debugger Dialog Box is displayed. If it is specified in the Debugger Option Dialog Box that the Exit Debugger dialog box is not to be displayed, however, the ID850QB is terminated.<br><No> button is selected, execution of the user program is not stopped and the Exit Debugger Dialog Box is not displayed. The ID850QB is not terminated. |
| Wfb01: | Since bit 7 of address 0x79 in the ID code are 0, The N-Wire emulator becomes prohibition of use henceforth. Do you exit the debugger as it is? |
| Ffc00: | Online help window cannot be started. Please install HTML Help environment with reference to a users manual. |
| Ffd00: | Failed to specify [XXX]. |
| Ffe00: | The maximum size of RRM was exceeded. |
| Wfe01: | There is a duplicate RRM address. |
| Wfe0b: | It shift to the flash mode. Is it completely cleared but is the present event. Doesn't it care? |
| Fffff: | Interrupted. |

# APPENDIX E INDEX