IAR C-SPY® Hardware Debugger Systems

User Guide

for the Renesas
SH Microcomputer Family

CSSHHW-1

IAR SYSTEMS

EDITION NOTICE

First edition: February 2010

Part number: CSSHHW-1

This guide applies to version 2.x of IAR Embedded Workbench® for SH.

Internal reference: M2, 6.0.x, IJOA.

Contents

Tables

Figures

Preface

Welcome to the *IAR C-SPY® Hardware Debugger Systems for SH User Guide*. The purpose of this guide is to provide you with detailed reference information that can help you use the features in the IAR C-SPY® Hardware Debugger Systems for SH.

Who should read this guide

You should read this guide if you want to get the most out of the features in the C-SPY hardware debugger systems. In addition, you should have working knowledge of:

- The C or C++ programming language
- Application development for embedded systems
- The architecture and instruction set of the SH microcontroller (refer to the chip manufacturer's documentation)
- The operating system of your host machine.

This guide also assumes that you already have working knowledge of the target system you are using, as well as some working knowledge of the IAR C-SPY Debugger. For a quick introduction to the IAR C-SPY Debugger, see the tutorials available in the *IAR Embedded Workbench® IDE User Guide*.

How to use this guide

This guide describes the C-SPY interface to the target system you are using; this guide does not describe the general features available in the IAR C-SPY Debugger or the hardware target system. To take full advantage of the whole debugger system, you must read this guide in combination with:

- The *IAR Embedded Workbench® IDE User Guide* which describes the general features available in the C-SPY debugger
- The documentation supplied with the target system you are using.

Note that additional features might have been added to the software after the *IAR C-SPY® Hardware Debugger Systems for SH User Guide* was printed. The IAR Information center contains the latest information.

What this guide contains

Below is a brief outline and summary of the chapters in this guide.

- *Introduction to C-SPY® hardware debugger systems* introduces you to the C-SPY driver for the E10 Emulator. The chapter briefly shows the difference in functionality provided by the different C-SPY drivers.
- *Hardware-specific debugging* describes the additional options, menus, and features provided by the debugger system.

Other documentation

The complete set of IAR development tools for the SH microcontroller are described in a series of guides. These guides can be found in the `sh\doc` directory or reached from the **Help** menu.

All of these guides are delivered in hypertext PDF or HTML format on the installation media.

Recommended web sites:

- The Renesas web site, **www.renesas.com**, contains information and news about the SH microcontrollers.
- The IAR Systems web site, **www.iar.com**, holds application notes and other product information.

Document conventions

When, in this text, we refer to the programming language C, the text also applies to C++, unless otherwise stated.

When referring to a directory in your product installation, for example `sh\doc`, the full path to the location is assumed, for example `c:\Program Files\IAR Systems\Embedded Workbench 6.`*`n`*`\sh\doc`.

TYPOGRAPHIC CONVENTIONS

This guide uses the following typographic conventions:

Style	Used for
`computer`	• Source code examples and file paths. • Text on the command line. • Binary, hexadecimal, and octal numbers.

Table 1: Typographic conventions used in this guide

Style	Used for
parameter	A placeholder for an actual value used as a parameter, for example *filename*.h where *filename* represents the name of the file.
[option]	An optional part of a command.
{option}	A mandatory part of a command.
a|b|c	Alternatives in a command.
bold	Names of menus, menu commands, buttons, and dialog boxes that appear on the screen.
italic	• A cross-reference within this guide or to another guide. • Emphasis.
…	An ellipsis indicates that the previous item can be repeated an arbitrary number of times.
	Identifies instructions specific to the IAR Embedded Workbench® IDE interface.
	Identifies instructions specific to the command line interface.
	Identifies helpful tips and programming hints.
	Identifies warnings.

Table 1: Typographic conventions used in this guide (Continued)

NAMING CONVENTIONS

The following naming conventions are used for the products and tools from IAR Systems® referred to in this guide:

Brand name	Generic term
IAR Embedded Workbench® for SH	IAR Embedded Workbench®
IAR Embedded Workbench® IDE for SH	the IDE
IAR C-SPY® Debugger for SH	C-SPY, the debugger
IAR C/C++ Compiler™ for SH	the compiler
IAR Assembler™ for SH	the assembler
IAR ILINK™ Linker	ILINK, the linker
IAR DLIB Library™	the DLIB library

Table 2: Naming conventions used in this guide

Introduction to C-SPY® hardware debugger systems

This chapter introduces you to the C-SPY hardware debugger system and to how it differs from the C-SPY Simulator.

You are assumed to already have some working knowledge of the target system you are using, as well as of the IAR C-SPY Debugger. For a quick introduction, see the *IAR Embedded Workbench® IDE User Guide*.

The C-SPY hardware debugger systems

C-SPY consists of both a general part which provides a basic set of C-SPY features, and a driver. The C-SPY driver is the part that provides communication with and control of the target system. The driver also provides a user interface—special menus, windows, and dialog boxes—to the functions provided by the target system, for instance special breakpoints. This driver is automatically installed during the installation of IAR Embedded Workbench.

At the time of writing this guide, the IAR C-SPY Debugger for the SH microcontroller is available with drivers for these target systems:

- Simulator
- E10 Emulator.

For further details about the concepts that are related to C-SPY and general debugger features, see the *IAR Embedded Workbench® IDE User Guide*.

DIFFERENCES BETWEEN THE C-SPY DRIVERS

This table summarizes the key differences between the C-SPY drivers:

Feature	Simulator	Emulator
Code breakpoints	Unlimited	x[1]
Data breakpoints	x	—
Execution in real time	—	x

Table 3: Driver differences

Feature	Simulator	Emulator
Zero memory footprint	x	x
Simulated interrupts	x	—
Real interrupts	—	x
Live watch	—	x
Cycle counter	x	—
Code coverage	x	—
Data coverage	x	—
Function/instruction profiler	x	—
Profiling	x	x
Trace	x	x

Table 3: Driver differences (Continued)

1 For detailed information about available breakpoints, see Table 8, *Available breakpoints*, page 29.

The C-SPY Emulator driver

The C-SPY Emulator driver is automatically installed during the installation of IAR Embedded Workbench. Using the C-SPY Emulator driver, C-SPY can connect to the E10 Emulator.

To use the hardware debugger, you also need a USB driver. The first time you connect an emulator to the USB port on the host computer, a USB setup wizard will help you locate and install the required USB driver. The driver is located in the `sh\drivers\Renesas\e8_e10\` directory of your IAR Systems product installation.

The C-SPY Emulator driver communicates with the emulator via the USB driver. The E10 Emulator communicates with the JTAG interface on the microcontroller.

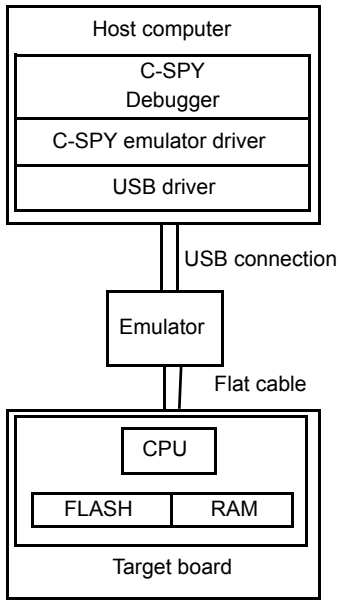

Figure 1: C-SPY Emulator communication overview

Note: For trace, the AUD port is used instead of the JTAG port.

For further information, refer to the documentation supplied with the E10 Emulator.

When a debugging session is started, your application is automatically downloaded and programmed into flash memory. You can disable this feature, if necessary.

Programming the E10A-USB firmware

Before an E10A-USB emulator can be used for the first time, the emulator firmware must be programmed. To program or update the firmware, follow these steps:

1. Make sure the USB cable is not attached to the emulator. Then open the sliding switch cover and make sure that the mode selection switch (number 1) is set to 1.

   Note: The USB cable must never be attached when you change the mode selection switch.

2 Attach the USB cable and connect the E10A-USB emulator to your host computer, and start the `E1setup.exe` setup program found in the `\sh\external\Renesas\E10A-USB\SH\TARGET\`*`core_name`*`\SetupTool\` directory. If you are asked to locate a `sys` file, this can be found on the CD that is supplied with the emulator.

3 A dialog box with information about device and firmware version is displayed. If the firmware information consists of dashes or an older version, this means that no firmware has been programmed. In that case, click **Setup** to display the next dialog box. Otherwise, click **Exit** and skip steps 4 through 8.

4 Follow the instructions in the dialog box; disconnect the USB cable and change the mode selection switch back to 0. Then re-attach the USB cable. Click **OK** to download the firmware update.

   If a hardware wizard is launched and asks you to re-install the USB driver for **HMSE USB Direct**, select a driver for your operating system version from the CD—or from the `drivers\` subdirectory in the Embedded Workbench installation directory—and install it. The original driver is not corrupted or invalid, but the host computer treats the E10 Emulator with the switch in a different position as a different device. When the installation of the driver is completed, click **OK**.

   Note: Do not turn off the host computer or disconnect the USB cable any more until the download has been completed. Doing so can damage the firmware permanently.

5 The firmware download might take some time. When it has been completed successfully, click **OK** in the dialog box that is displayed.

6 Follow the instructions in the new dialog box; disconnect the USB cable and change the mode selection switch back to 1 once more and then re-attach the USB cable again. Click **OK**.

7 The dialog box with information about device and firmware version will now show device group and version information for the firmware.

8 Click **Exit**. The E10A-USB emulator is now ready to be used.

Connecting to the target board

To establish a connection to the target board, follow this procedure:

1 To select the device that matches your target system, choose **Project>Options>General Options>Target** in IAR Embedded Workbench.

2 Select the driver that matches your emulator on the **Project>Options>Debugger>Setup** page.

3 Build the project if it has not been built and choose **Project>Debug** to start C-SPY.

4 The **Select Emulator mode** dialog box is displayed.

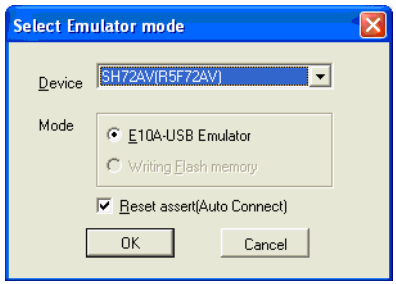

Figure 2: The Select Emulator mode dialog box for the E10A-USB emulator

Select the device name in use from the **Device** drop-down list. The following items can be selected in the **Mode** group box.

Emulator mode	Description
E10A-USB Emulator	The E10A-USB emulator for the specified device is activated. Debugging the program is enabled.
Writing flash memory	This option is not available for this product.

Table 4: The E10A-USB emulator mode options

5 The **Connecting** dialog box is displayed and the emulator connection is started.

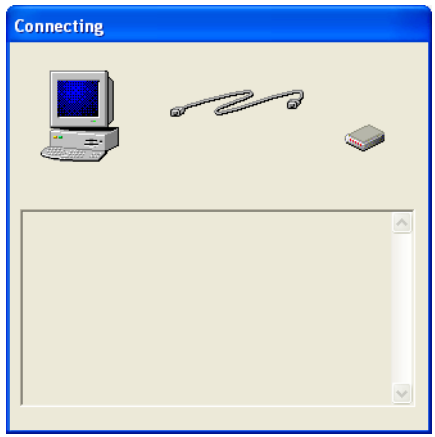

Figure 3: The Connecting dialog box

6 When the **ID Code** dialog box is displayed, enter the hexadecimal ID security code for the flash memory. For some devices, the flash memory contents are erased if the ID code does not match.

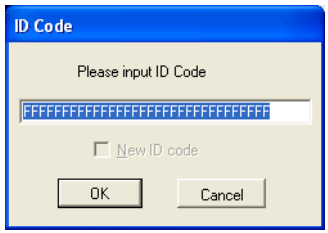

Figure 4: The ID Code dialog box for the E10A-USB emulator

If the **New ID code** option is available and selected, the flash memory contents are erased.

When the emulator is ready to be used, `Connected` will be printed in the IAR Embedded Workbench Debug Log window together with the number of available hardware and software breakpoints.

Getting started

IAR Embedded Workbench comes with example applications. You can use these examples to get started using the development tools from IAR Systems or simply to verify that contact has been established with your target board. You can also use the examples as a starting point for your application project.

You can find the examples in the `sh\examples` directory. The examples are ready to be used as is. They are supplied with ready-made workspace files, together with source code files and all the other related files.

RUNNING THE DEMO PROGRAM

1 To use an example application, choose **Help>Information Center** and click **EXAMPLE PROJECTS**.

2 Browse to the example that matches the specific evaluation board or starter kit you are using.

Click **Open Project**.

3 In the dialog box that appears, choose a destination folder for your project location. Click **Select** to confirm your choice.

4 The available example projects are displayed in the workspace window. Select one of the projects, and if it is not the active project (highlighted in bold), right-click it and choose **Set As Active** from the context menu.

5 To view the project settings, select the project and choose **Options** from the context menu. Verify the settings of the options **Device** and **Debugger>Driver**. As for other settings, the project is set up to suit the target system you selected.

For further details about the C-SPY options for the hardware target system and how to configure C-SPY to interact with the target board, see *Debugger options for debugging using hardware systems*, page 21.

Click **OK** to close the **Options** dialog box.

6 To compile and link the application, choose **Project>Make** or click the **Make** button.

7 To start C-SPY, choose **Project>Debug** or click the **Debug** button. If C-SPY fails to establish contact with the target system, see *Resolving problems*, page 30.

8 Choose **Execute>Go** or click the **Go** button to start the application.

Click the **Stop** button to stop execution.

Hardware-specific debugging

This chapter describes the additional options, menus, and features provided by the C-SPY® hardware debugger systems. The chapter contains the following sections:

- Debugger options for debugging using hardware systems
- Emulator-specific debugging
- Using the trace system in the emulator
- Using breakpoints
- Resolving problems.

Debugger options for debugging using hardware systems

Before you start any C-SPY hardware debugger you must set some options for the debugger system—both C-SPY generic options and options required for the hardware system (C-SPY driver-specific options). Follow this procedure:

1. To open the **Options** dialog box, choose **Project>Options**.
2. To set C-SPY generic options and select a C-SPY driver:
   - Select **Debugger** from the **Category** list
   - On the **Setup** page, select the appropriate C-SPY driver from the **Driver** drop-down list.

   For information about the options **Setup macros**, **Run to**, and **Device descriptions**, as well as for information about the pages **Extra Options** and **Plugins**, see the *IAR Embedded Workbench® IDE User Guide*.

   Note that a default device description file and linker configuration file is automatically selected depending on your selection of a device on the **General Options>Target** page.
3. To set the driver-specific options, select the appropriate driver from the **Category** list.
4. When you have set all the required options, click **OK** in the **Options** dialog box.

DOWNLOAD

By default, C-SPY downloads the application to RAM or flash when a debug session starts. The **Download** options lets you modify the behavior of the download.

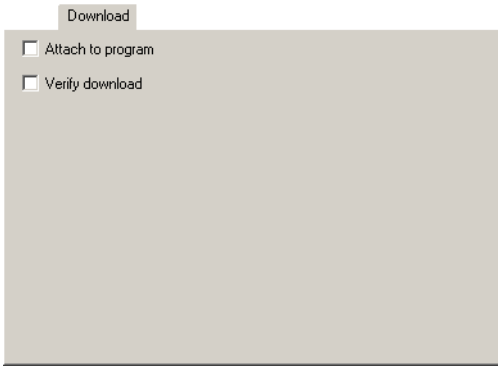

Figure 5: C-SPY Download options

Verify download

Use this option to verify that the downloaded code image can be read back from target memory with the correct contents.

Suppress download

Use this option to debug an application that already resides in target memory. When this option is selected, the code download is disabled, while preserving the present content of the flash.

If this option is combined with the **Verify download** option, the debugger will read back the code image from non-volatile memory and verify that it is identical to the debugged program.

Emulator-specific debugging

In this section you can read about the features specific for the C-SPY Emulator driver.

THE EMULATOR MENU

When you are using the C-SPY Emulator driver, the **Emulator** menu appears in C-SPY.

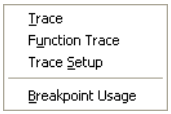

Figure 6: The Emulator menu

These commands are available on the **Emulator** menu:

Menu command	Description
Trace	Opens the Trace window which displays the recorded trace data; see *Trace window*, page 25.
Function Trace	Opens the Function Trace window which displays the trace data for which functions were called or returned from; see *Function Trace window*, page 26.
Trace Setup	Displays the **Trace Settings** dialog box, where you can configure the trace system. See the emulator manufacturer's documentation.
Breakpoint Usage	Displays the **Breakpoint Usage** dialog box which lists all active breakpoints; see *Breakpoint Usage dialog box*, page 29.

Table 5: Commands on the Emulator menu

Using the trace system in the emulator

This section gives you information about using the trace system. More specifically, these topics are covered:

- *Requirements for using the trace system*, page 24
- *Reasons for using the trace system*, page 24
- *Briefly about the trace system*, page 24
- *How to use the trace system*, page 24
- *Related reference information*, page 24.

REQUIREMENTS FOR USING THE TRACE SYSTEM

The SH2 processor on the target board must support tracing and the AUD port must be used instead of the JTAG port.

REASONS FOR USING THE TRACE SYSTEM

By using the trace system, you can trace the program flow up to a specific state, for instance an application crash, and use the trace information to locate the origin of the problem. Trace information can be useful for locating programming errors that have irregular symptoms and occur sporadically. Trace information can also be useful as test documentation.

BRIEFLY ABOUT THE TRACE SYSTEM

In C-SPY, a *trace* is a recorded sequence of executed machine instructions. The Function Trace window only shows trace data corresponding to calls to and returns from functions, whereas the Trace window displays all instructions.

Level of support

The level of trace support varies from emulator to emulator. The level of trace support offered by your particular emulator model is described in the manufacturer's emulator documentation.

HOW TO USE THE TRACE SYSTEM

For information about using the trace system, see the *IAR Embedded Workbench® IDE User Guide.*

RELATED REFERENCE INFORMATION

To use the trace system, you might need reference information about these windows and dialog boxes:

- *Trace window*, page 25
- *Function Trace window*, page 26
- *Find In Trace window*, page 27
- *Find in Trace dialog box*, page 27.

TRACE WINDOW

The Trace window—available from the **Emulator** menu—displays a recorded sequence of executed machine instructions.

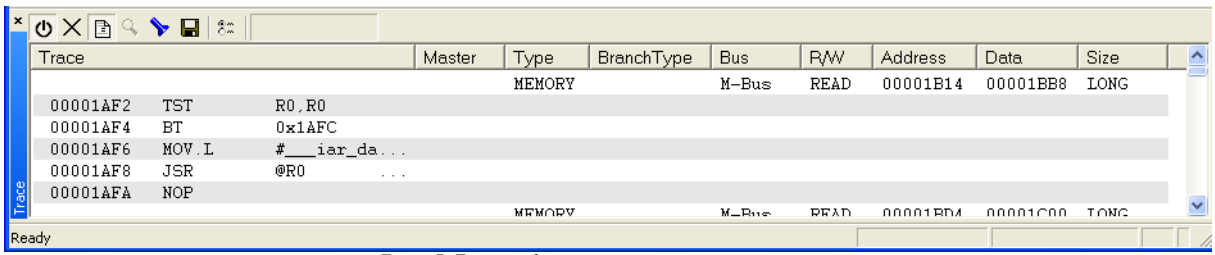

Figure 7: Trace window

C-SPY generates trace information based on the location of the program counter.

Trace toolbar

The Trace toolbar at the top of the Trace window and in the Function Trace window provides these toolbar buttons:

Toolbar button	Description
Enable/Disable	Enables and disables tracing. This button is not available in the Function trace window.
Clear trace data	Clears the trace buffer. Both the Trace window and the Function trace window are cleared.
Toggle Source	Toggles the Trace column between showing only disassembly or disassembly together with corresponding source code.
Browse	Toggles browse mode on and off for a selected item in the Trace window. For more information about browse mode, see the *IAR Embedded Workbench® IDE User Guide*.
Find	Opens the **Find In Trace** dialog box where you can perform a search; see *Find in Trace dialog box*, page 27.
Save	Opens a standard **Save As** dialog box where you can save the recorded trace information to a text file, with tab-separated columns.
Edit Settings	Opens the **Trace Settings** dialog box, where you can configure the trace system. See the emulator manufacturer's documentation.

Table 6: Trace toolbar buttons

Trace display area

The display area displays trace information in these columns:

Trace window column	Description
Trace	The recorded sequence of executed machine instructions. Optionally, the corresponding source code can also be displayed.

Table 7: Trace window columns

FUNCTION TRACE WINDOW

The Function Trace window—available from the **Emulator** menu—displays a subset of the trace data displayed in the Trace window. Instead of displaying all rows, the Function Trace window only shows trace data corresponding to calls to and returns from functions.

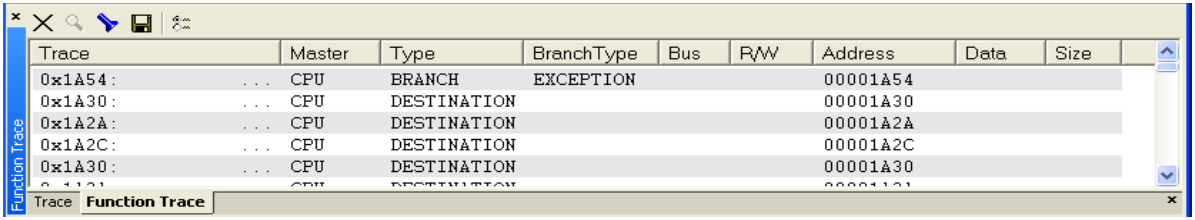

Figure 8: Function Trace window

Toolbar

For information about the toolbar, see *Trace toolbar*, page 25.

The display area

For information about the columns in the display area, see *Trace display area*, page 26.

FIND IN TRACE WINDOW

The Find In Trace window—available from the **View>Messages** menu—displays the result of searches in the trace data.

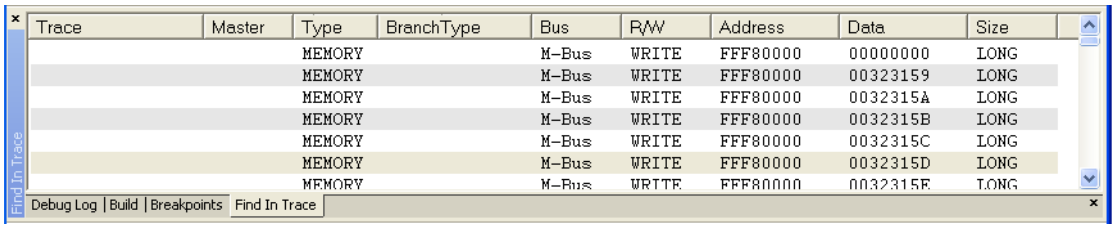

Figure 9: Find In Trace window

The Find In Trace window looks like the Trace window and shows the same columns and data, but *only* those rows that match the specified search criteria. Double-click an item in the Find in Trace window to bring up the same item in the Trace window.

You specify the search criteria in the **Find in Trace** dialog box. For information about how to open this dialog box, see *Find in Trace dialog box*, page 27.

FIND IN TRACE DIALOG BOX

Use the **Find in Trace** dialog box—available by choosing **Edit>Find and Replace>Find** or from the Trace window toolbar—to specify the search criteria for advanced searches in the trace data. Note that the **Edit>Find and Replace>Find** command is context-dependent. It displays the **Find in Trace** dialog box if the Trace window is the current window or the **Find** dialog box if the editor window is the current window.

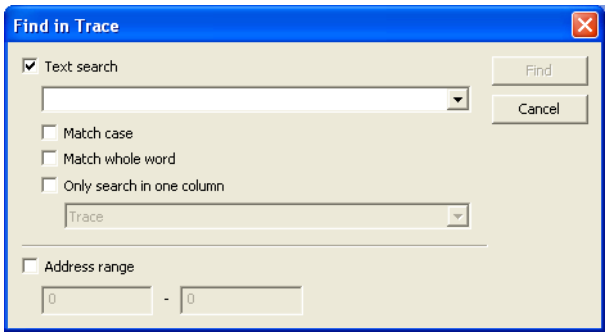

Figure 10: Find in Trace dialog box

The search results are displayed in the Find In Trace window—available by choosing the **View>Messages** command, see *Find In Trace window*, page 27.

In the **Find in Trace** dialog box, you specify the search criteria with the following settings.

Text search

A text field where you type the string you want to search for. Use these options to fine-tune the search:

Match Case	Searches only for occurrences that exactly match the case of the specified text. Otherwise specifying `int` will also find `INT` and `Int`.
Match whole word	Searches only for the string when it occurs as a separate word. Otherwise `int` will also find `print`, `sprintf` and so on.
Only search in one column	Searches only in the column you selected from the drop-down list.

Address Range

Use the text fields to specify an address range. The trace data within the address range is displayed. If you also have specified a text string in the **Text search** field, the text string is searched for within the address range.

Using breakpoints

This section provides an overview of the available breakpoints for the C-SPY hardware debugger systems. This is described:

- *Available breakpoints*, page 28
- *Breakpoint Usage dialog box*, page 29.

For information about the various methods for setting breakpoints, the facilities for monitoring breakpoints, and the various breakpoint consumers, see the *IAR Embedded Workbench® IDE User Guide*.

AVAILABLE BREAKPOINTS

Using the C-SPY driver for the hardware debugger system, you can set code breakpoints. The amount of breakpoints you can set depends on the number of breakpoints available on the target system.

This table summarizes the characteristics of breakpoints for the target system:

Target system	Code breakpoints	Hardware breakpoints	Software breakpoints
E10 Emulator	Yes	11	255

Table 8: Available breakpoints

The debugger will first use any available hardware breakpoints before using software breakpoints.

Exceeding the number of available breakpoints will cause the debugger to single step, which will significantly reduce the execution speed. For this reason you must be aware of the different breakpoint consumers; see the *IAR Embedded Workbench® IDE User Guide*.

BREAKPOINT USAGE DIALOG BOX

The **Breakpoint Usage** dialog box—available from the driver-specific menu—lists all active breakpoints.

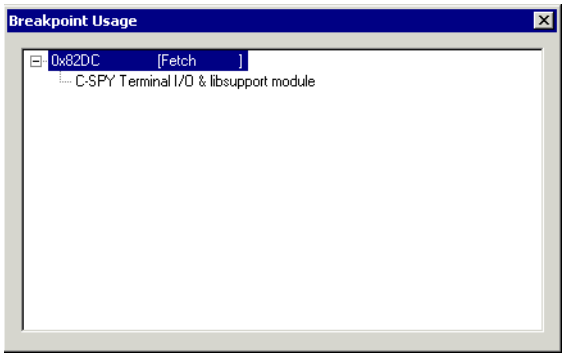

Figure 11: Breakpoint Usage dialog box

In addition to listing all breakpoints that you have defined, this dialog box also lists the internal breakpoints that the debugger is using.

For each breakpoint in the list the address and access type are shown. Each breakpoint in the list can also be expanded to show its originator.

For more information, see the *IAR Embedded Workbench® IDE User Guide*.

Resolving problems

Debugging using the C-SPY hardware debugger systems requires interaction between many systems, independent from each other. For this reason, setting up this debug system can be a complex task. If something goes wrong, it might at first be difficult to locate the cause of the problem.

This section includes suggestions for resolving the most common problems that can occur when debugging with the C-SPY hardware debugger systems.

For problems concerning the operation of the evaluation board, refer to the documentation supplied with it, or contact your hardware distributor.

WRITE FAILURE DURING LOAD

There are several possible reasons for write failure during load. The most common is that your application has been incorrectly linked:

- Check the contents of your linker configuration file and make sure that your application has not been linked to the wrong address (examine the linker memory map file)
- Check that you are using correct linker configuration file.

If you are using the IAR Embedded Workbench, the linker configuration file is automatically selected based on your choice of device:

- Choose **Project>Options**
- Select the **General Options** category
- Click the **Target** tab
- Choose the appropriate device from the **Device** drop-down list.

To override the default linker configuration file:

- Choose **Project>Options**
- Select the **Linker** category
- Click the **Config** tab
- Choose the appropriate linker configuration file in the **Linker configuration file** area.

NO CONTACT WITH THE TARGET HARDWARE

There are several possible reasons for C-SPY to fail to establish contact with the target hardware.

- Check the communication devices on your host computer
- Verify that the cable is properly plugged in and not damaged or of the wrong type

- Verify that the target chip is properly mounted on the evaluation board
- Make sure that the evaluation board is supplied with sufficient power.

Examine the linker configuration file to make sure that your application has not been linked to the wrong address.