Peripheral Simulation

in C-SPY®

User Guide

UGPSCS-1

IAR SYSTEMS

COPYRIGHT NOTICE

DISCLAIMER

The information in this document is subject to change without notice and does not represent a commitment on any part of IAR Systems. While the information contained herein is assumed to be accurate, IAR Systems assumes no responsibility for any errors or omissions.

In no event shall IAR Systems, its employees, its contractors, or the authors of this document be liable for special, direct, indirect, or consequential damage, losses, costs, charges, claims, demands, claim for lost profits, fees, or expenses of any nature or kind.

TRADEMARKS

EDITION NOTICE

First edition: December 2009

Part number: UGPSCS-1

Internal reference: ISUD.

Peripheral simulation

This document contains general information about the C-SPY® peripheral simulation system.

Using plug-in modules, the C-SPY Simulator can simulate the on-chip peripheral units of your microcontroller. This makes it possible to debug an application before you have access to actual target hardware. Using the peripheral simulation system, you can also find problems with the software logic which are difficult to debug on actual hardware.

For reference information about the simulated peripheral units and the dialog boxes used for configuring and monitoring them, see the peripheral unit vendor's documentation.

Using the peripheral simulation system in C-SPY

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

- *Requirements for using peripheral simulation*, page 3
- *Reasons for using peripheral simulation*, page 3
- *Briefly about the peripheral system*, page 4
- *Setting up for simulating peripheral units*, page 4
- *Example of recording and replaying*, page 5
- *Related reference information*, page 6.

REQUIREMENTS FOR USING PERIPHERAL SIMULATION

In the simulator there are no specific requirements for using the peripheral system. Peripheral simulation has a very small impact on simulation performance.

REASONS FOR USING PERIPHERAL SIMULATION

Debugging with a simulator instead of real hardware has several advantages. For example, it is easier to monitor the target system and you can, for all practical purposes, stop time. Without peripheral simulation, the scope of simulation covers just the basic CPU, with the addition of simulated interrupts. Only limited device simulation is possible, using immediate breakpoints and C-SPY macros.

Using peripheral simulation, it becomes possible to simulate a wide range of peripheral devices, such as timers, watchdogs, UARTs, buttons, and displays.

BRIEFLY ABOUT THE PERIPHERAL SYSTEM

Peripheral simulation is implemented by *peripheral simulation plug-in* (PSP) *modules*, normally written by third-party vendors using a Software Developer's Kit (SDK) available from IAR Systems. Each peripheral device is implemented as a PSP unit. C-SPY can load several PSP modules, each of which can define several PSP units.

SETTING UP FOR SIMULATING PERIPHERAL UNITS

To simulate peripheral units, you must configure the simulation environment:

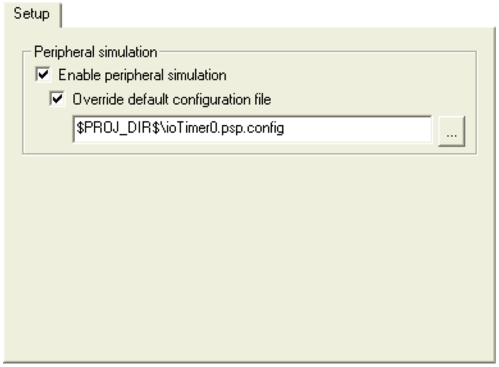
- Select the correct device description file
- Enable the PSP system
- Configure the simulator.

Selecting the correct device description file

1. Choose **Project>Options** to display the **Options** dialog box.
2. In the **Category** list, select **Debugger**. On the **Setup** page, choose the appropriate device description file.

Enabling the PSP system

1. Choose **Project>Options** to display the **Options** dialog box.
2. Select **Debugger>Simulator** in the **Category** list. On the **Setup** page, select the **Enable peripheral simulation** option.
3. If needed, override the default configuration file. The PSP configuration files are located in the `$TOOLKIT_DIR$\config` directory.



Setup

Peripheral simulation

☑ Enable peripheral simulation

☑ Override default configuration file

$PROJ_DIR$\ioTimer0.psp.config

Figure 1: Enabling peripheral simulation

Configuring the simulator

1 After you have built your application and started C-SPY, choose **Simulator Config** on the **Peripherals** menu.

2 The options in the **Simulator Config** dialog box allow you to configure stimuli to/from the simulated system, see *Simulator Config*, page 7.

EXAMPLE OF RECORDING AND REPLAYING

For example, to record and replay a debug session, follow these steps:

1 After you have started C-SPY, choose **Simulator Config** from the **Peripherals** menu.

2 In the **Simulator Config** dialog box, select the **Record** option.

3 Execute your application a little while, entering some stimuli, for example by clicking some input buttons. (This assumes that your project contains windows where you can create stimuli interactively.)

4 Stop execution. The text box next to the **Save** button should contain a non-zero number.

5 Click **Save**. The recorded stimuli are saved to the stimuli file `stimuli.rec`.

6 Click **Load**. The stimuli file is loaded into the replay list. The text box next to the **Load** button should contain the same non-zero number.

7 Deselect **Record** and select **Replay**.

8 Choose **Debug>Reset** to reset the debugger.

9. Execute your application once more. Recorded stimuli are replayed and the execution mirrors the first run exactly.

RELATED REFERENCE INFORMATION

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

- *The Peripherals menu*, page 6
- *Simulator Config*, page 7.

THE PERIPHERALS MENU

When you use the simulator driver with peripheral simulation enabled, the **Peripherals** menu is added in the menu bar.

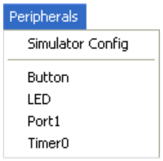


Figure 2: Peripherals menu

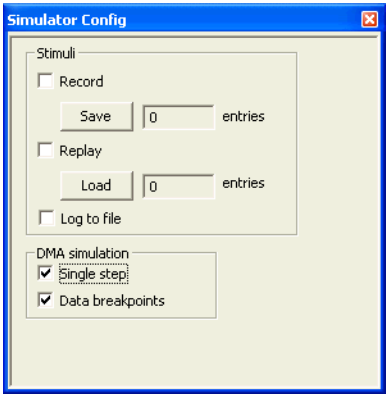
The **Peripherals** menu contains these commands:

Menu command	Description
Simulator Config	Displays a dialog box where you configure stimuli (input and output) to/from the simulated system, see *Simulator Config*, page 7.
Plug-in specific commands	The rest of the menu contains commands that open windows for the PSP units that the simulator has loaded.

Table 1: Description of Peripherals menu commands

SIMULATOR CONFIG

The options in the **Simulator Config** dialog box—available from the **Peripherals** menu—allow you to configure stimuli to/from the simulated system.



Simulator Config

Stimuli

□ Record

Save 0 entries

□ Replay

Load 0 entries

□ Log to file

DMA simulation

☑ Single step

☑ Data breakpoints

Figure 3: Simulator Config dialog box

Stimuli

These options control recording and replaying stimuli during a debug session.

Stimuli means input and output to/from the simulated system. For example, a push button might be part of the simulated system, but actually pushing it requires external stimuli. When running a simulation, stimuli could come from manual interaction, such as clicking a button in a window, or from predefined stimuli stored in a file. Interactive stimuli can also be recorded to a file, so that an exact sequence can be repeated.

Typical uses for stimuli include a UART unit feeding or receiving bits through a port unit, a timer unit feeding bits to a port, and a port feeding bits to external units (LEDs, motors, etc.).

Choose between:

Record The simulator records all stimuli generated in the session. The text box next to the **Save** button shows the number of recorded entries. When the debugger is reset, the recording is set to 0. Click **Save** to save the current stimuli entries to the *stimuli file* `$PROJ_DIR$\stimuli.rec`.

Note: Only "unconnected" stimuli are recorded, not output routed between peripheral units.

Replay Stimuli are fetched from an internal list during the simulation. Click **Load** to populate this list from the stimuli file `stimuli.rec`. The text box next to the **Load** button shows the number of entries in the list. When the debugger is reset, replay will start over from the beginning of the list.

If both the **Record** and **Replay** options are selected at the same time, stimuli replayed from the stimuli file are not re-recorded.

Log to file All signals (unit input and output) are logged to the log file `$PROJ_DIR$\stimuli.log`.

DMA simulation

These options determine how the instruction simulator behaves when a peripheral unit, such as a DMA unit, takes control of the memory bus.

Choose between:

Single step Single-stepping (on machine level, that is, not in source code) will step one clock cycle at a time, as long as the peripheral unit commands the bus. When the option is deselected, single-stepping will always execute exactly one CPU instruction.

Data breakpoints By default, data breakpoints are only triggered when the CPU accesses memory. Select the **Data breakpoints** option if you also want memory accesses by other units (for instance, a DMA unit) to trigger data breakpoints.