



eSi-RISC Development Suite Getting Started Guide

1 Contents

1	Contents _____	2
2	Overview _____	3
3	Starting the Integrated Development Environment _____	4
4	Hello World Tutorial _____	5
5	Next Steps _____	8
6	Support _____	10

2 Overview

This guide describes how to create and run your first program using the eSi-RISC Development Suite. It assumes that you already have the software installed on your computer. For details of how to do this, please refer to the eSi-RISC Development Suite Installation Guide.

The eSi-RISC Development Suite contains all you need to develop software for an eSi-RISC CPU. It features:

- C/C++ compiler
- Assembler
- Disassembler
- Linker
- Debugger
- Simulator
- IDE
- C and C++ libraries
- FreeRTOS
- lwIP TCP/IP stack
- Verilog PLIs for connecting the debugger to Verilog simulations

Many of these components are based upon ports of tools that most developers will already be familiar with, such as the Eclipse IDE and GNU tools, allowing you to start developing your eSi-RISC applications straight away.

3 Starting the Integrated Development Environment

After the eSi-RISC Development Suite has been installed, the Eclipse Integrated Development Environment can be started, by selecting: *Start / All Programs / eSi-RISC Development Suite / Eclipse*.

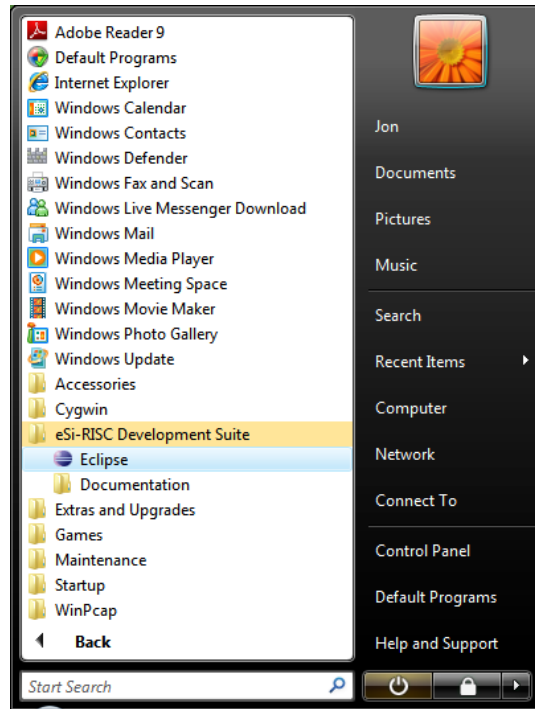


Figure 1: Start the Eclipse IDE

Eclipse will request you to enter a folder to use for its workspace, which is where it will store its configuration files and use as a default location for new projects. It is recommended that you select a new folder for this and do not try to share this workspace folder with other Eclipse based programs you may have. Also, if you intend to use the command line tools from a shell, it is recommended that you chose a folder that does not have spaces in the path.

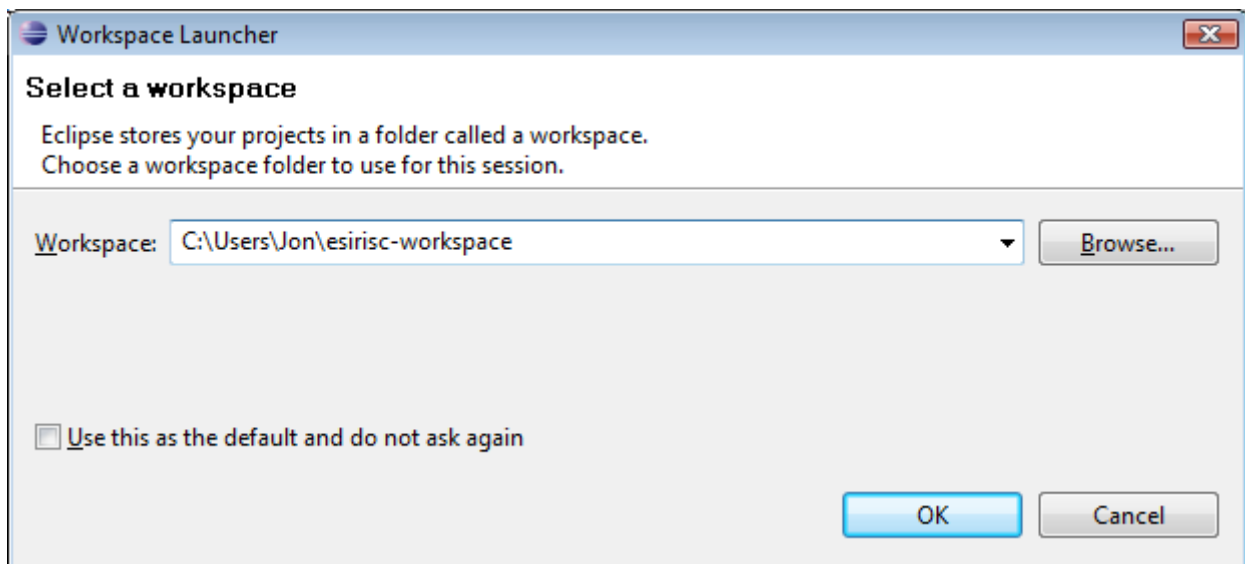


Figure 2: Select a workspace

4 Hello World Tutorial

When Eclipse finishes loading, you will be presented with the Welcome page. From here you can select:

- *Overview* - Get an overview of the features of Eclipse and its installed plugins.
- *What's New* - Find out what is new in this release of Eclipse.
- *Samples* - See a list of sample eSi-RISC applications.
- *Tutorials* - See a list of tutorials for Eclipse features and the eSi-RISC Development Suite.
- *Workbench* - Go straight to the workbench.

For now select *Tutorials*, then on the next page under the list of eSi-RISC tutorials, select *Create a Hello World Application for eSi-RISC*.

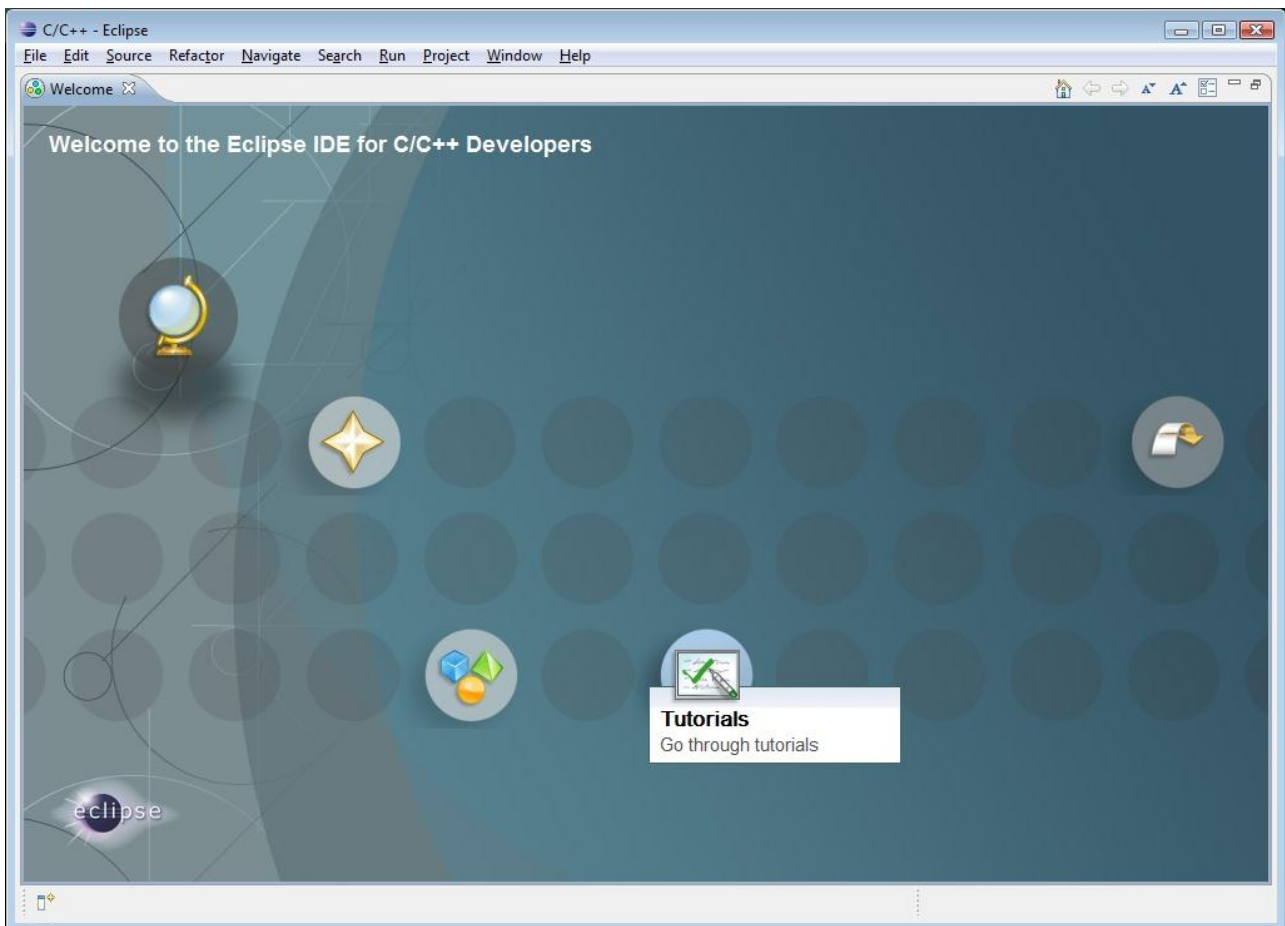


Figure 3: Eclipse Welcome Screen

After selecting the tutorial, Eclipse will switch to the workbench, with step-by-step instructions for walking through the tutorial appearing in a window to the right of the workbench, as illustrated in Figure 4: Eclipse Workbench.

Each of the instructions will have one of the following icons next to it, which should be clicked to move on to the next step of the tutorial:



- Click to indicate you have completed the instruction
- Click to have the instruction performed for you
- Click to skip the instruction

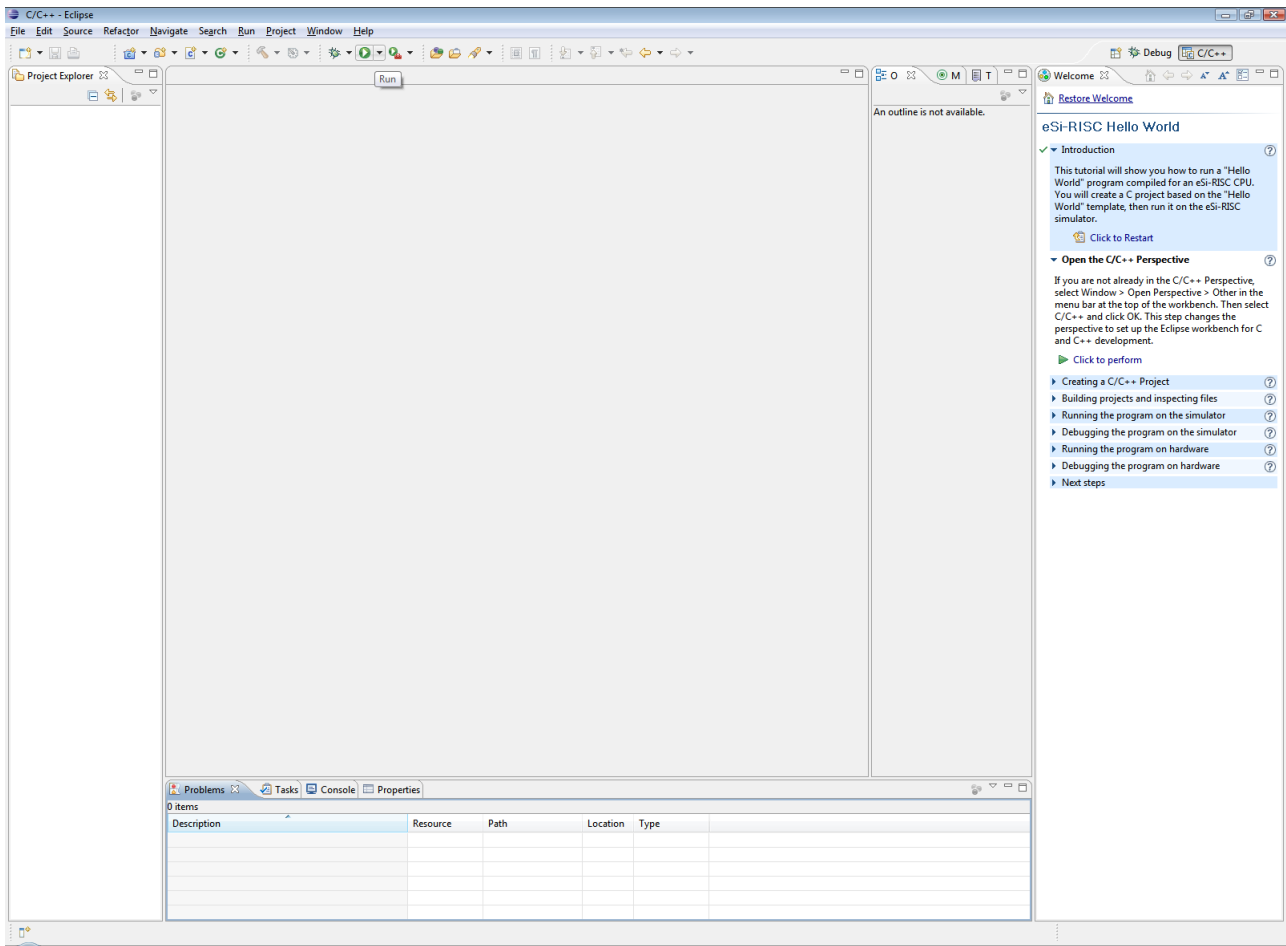


Figure 4: Eclipse Workbench

The tutorial will guide you through:

- Creating a C Project, specifying the target CPU and build configurations
- Compiling the program
- Running the program on the simulator
- Debugging the program on the simulator
- Running the program on hardware
- Debugging the program on hardware

The C project that is created contains the classic Hello World C program, which simply prints the string "Hello World" to the standard output stream. The source code for the program is as follows:

```
#include <stdio.h>

int main() {
    printf("Hello world\n");
    return 0;
}
```

Even a program as simple as this will raise a few questions for the seasoned embedded developer:

- Where does the standard output stream go to on an embedded target?

- What is `iprintf`?
- Why use a return type of `int` for `main` rather than `void` on an embedded target?

Where the standard output stream will be output to will depend upon the configuration of the target SoC (System-on-a-Chip) and the way in which the program is run. If you are targeting the Cyclone III Demo SoC that is assumed for this tutorial, then it can go to one of two places: If the program is run with a debugging connection to a host computer (via a JTAG or USB cable or if the program is being run on the simulator), then you can choose to have the output directed to the console window in Eclipse. If no connection to a host computer is available, then it will be output via a UART.

`iprintf` is identical to the more commonly used `printf` function, except that it does not support floating-point format specifiers (E.g. `%f`, `%e` and `%g`). This has the advantage that for targets without hardware floating-point support (E.g. eSi-1600, eSi-3200 and eSi-3250), the software floating-point emulation libraries are not required to be linked in, which helps to minimize code size. If floating-point format specifiers are needed, then the regular `printf` function can be used.

The return type of `main` is specified as `int` rather than `void`, as is often seen in many embedded programs, in order to be fully compliant with language standards. Also, the eSi-RISC debugger is able to report the value returned by your program (or passed via a call to `exit()`), which can be useful when debugging your programs or for programs that run in a hosted environment.

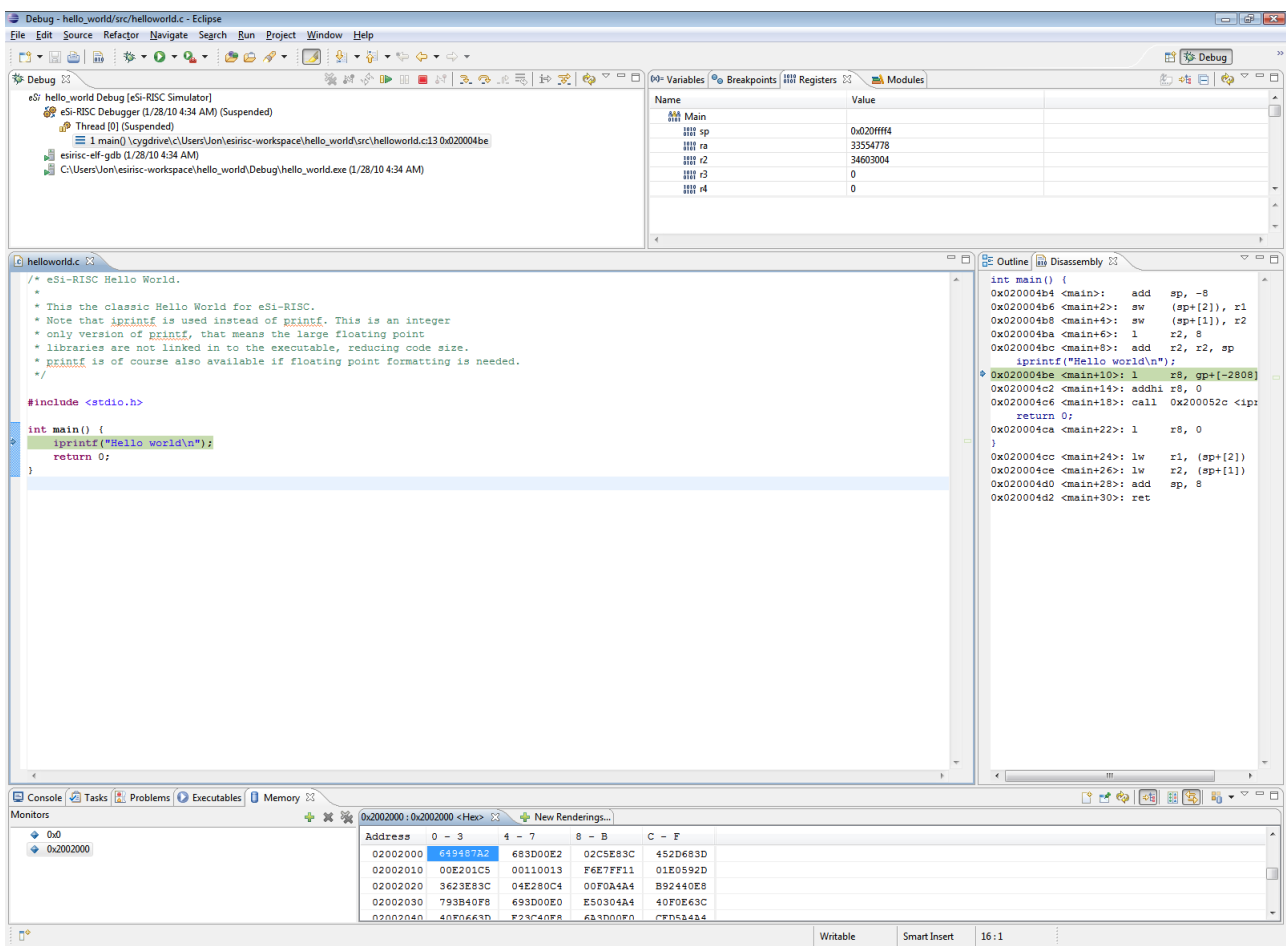


Figure 5: Debugging the Hello World Program

5 Next Steps

After you have finished the Hello World tutorial, there are a number of other tutorials and sample applications you can try out to become more familiar with the eSi-RISC Development Suite. All of these tutorials can be accessed from the Welcome page, which can be accessed at any time by selecting the *Help / Welcome* menu.

Tutorials:

- Profile and optimize an eSi-RISC application.

This tutorial shows how to program an application using the eSi-RISC simulator. It also shows how you can optimize the code and target different eSi-RISC CPUs to see the difference in performance.

Samples:

- c++

Demonstrates several c++ language features, including classes, constructors & destructors (static and dynamic), iostream, operator overloading, exceptions and runtime-type information.

- CFI FLASH Programmer

Demonstrates how to program a CFI FLASH.

- FreeRTOS

Demonstrates the eSi-RISC port of FreeRTOS, a small, open-source RTOS.

- FreeRTOS lwIP

Demonstrates the eSi-RISC port of the lwIP TCP/IP stack with FreeRTOS. The demo consists of a small web-server that dynamically generates a web page showing the tasks being scheduled by FreeRTOS.

- Hosted I/O

Demonstrates how to use hosted I/O, which provides a way to access the file system of a host computer, using the standard C library file functions.

- GPIO

Demonstrates how to use the GPIO peripheral to drive LEDs and read the state of buttons. It also shows how you can write an interrupt handler without writing any assembly code.

- I2C EEPROM

Demonstrates how to use the I2C peripheral to program an EEPROM.

- I2C Master and Slave

Demonstrates how to use the I2C peripheral as both a master and slave.

- LCD

Demonstrates how to use the LCD peripheral by displaying an image on it.

- Makefile

Demonstrates how to build an eSi-RISC application using a handwritten Makefile.

- Smart Card

Demonstrates how to use the UART peripheral in ISO 7816 mode to access a smart card.

- SPI FLASH Programmer

Demonstrates how to use the SPI FLASH peripheral to program a SPI FLASH device.

- SPI SD Card

Demonstrates how to use the SPI peripheral to read the master boot record of a SD Card.

- User-defined Instructions

Demonstrates how to use user-defined instructions in your eSi-RISC programs and how to create a shared library so that they can be used in the eSi-RISC simulator.

- UART

Demonstrates how to use the UART peripheral.

For more information on each of these tutorials and sample applications, please refer to the on-line help, by selecting the *Help / Help Contents* menu.

6 Support

For support issues regarding the eSi-RISC Development Suite, please contact EnSilica support:

e-mail	support@ensilica.com
Telephone	+44 (0)118 3217 310

Table 1: Support Contact Details