My First Nios II Software

Tutorial

101 Innovation Drive
San Jose, CA 95134
www.altera.com

TU-01003-2.1

Document last updated for Altera Complete Design Suite version: 12.1
Document publication date: December 2012

Subscribe

Contents

1. My First Nios II Software Design

This tutorial provides comprehensive information to help you understand how to create a software project for a Nios II processor system in an Altera FPGA and run the software project on your development board.

The Nios® II processor core is a soft intellectual property (IP) processor that you download (along with other hardware components that comprise the Nios II system) onto an Altera FPGA. This tutorial introduces you to the basic software development flow for the Nios II processor. In the tutorial, you use a simple, pre-generated Nios II hardware system and create a software program to run on it.

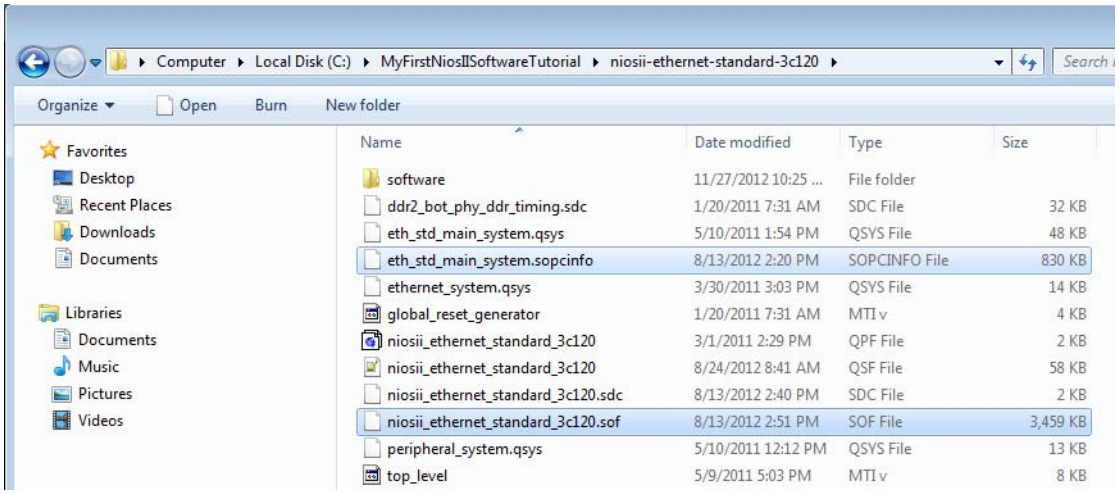
The example Nios II hardware system provides the following necessary components:

- Nios II processor core
- Off-chip DDR memory interface to store and run the software
- USB serial link for communication between the host computer and the target hardware (typically using a USB-Blaster™ cable)
- LED peripheral I/O (PIO)

Software and Hardware Requirements

This section assumes you have already installed the Quartus® II design software and the Nios II Embedded Design Suite. Figure 1–1 shows an example of the project directory structure.

Figure 1–1. Example Project Directory Structure



The tutorial describes how to use the Nios II tools with the hardware design from the Nios II Ethernet Standard Design Example page of the Altera website. The hardware design is available in several different **.zip** files that target different Altera development boards. On the web page, locate the Nios II Ethernet Standard Design Example **.zip** file that corresponds to your board.

Download and copy the design files to the location where you plan to run the tutorial. Throughout the tutorial, *<your project directory>* refers to this directory.

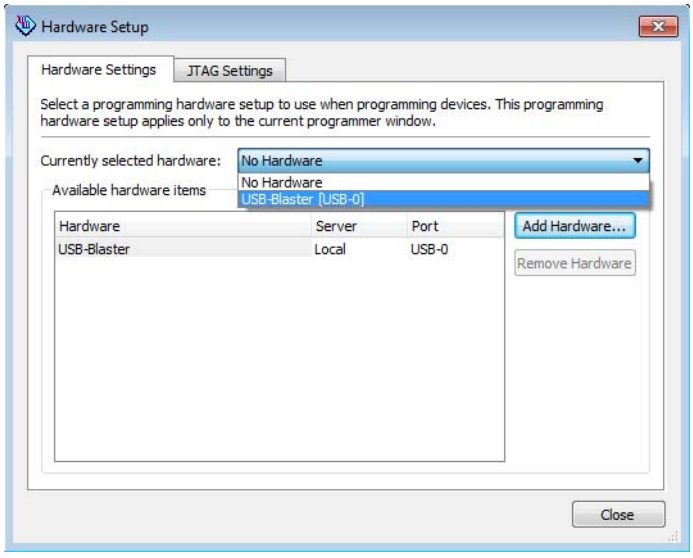
Download Hardware Design to Target FPGA

The software that you build will be executed by a Nios II processor-based system in an FPGA. Therefore, the first step is to configure the FPGA on your development board with the pre-generated Nios II hardware system. Download the FPGA configuration file, that is, the SRAM Object File (**.sof**) that contains the Nios II hardware system, to the board by performing the following steps:

1. Connect the board to the host computer using the USB download cable.
2. Apply power to the board.
3. Start the Nios II Software Build Tools (SBT) for Eclipse. On Windows computers, choose **All Programs > Altera > Nios II EDS** *<version>* **> Nios II** *<version>* **Software Build Tools for Eclipse** in the Windows Start menu.
4. Select **Workbench** and accept the default workspace.
5. On the Nios II Tools menu, click **Quartus II Programmer**.
6. Click **Hardware Setup** in the upper left corner of the Quartus II Programmer window. The **Hardware Setup** dialog box appears.
7. Select **USB-Blaster** from the **Currently selected hardware** list, as shown in Figure 1–2.

If the download cable does not appear in the list, you must first install drivers for the cable. For information about download cables and drivers, refer to the Download Cables page of the Altera website.

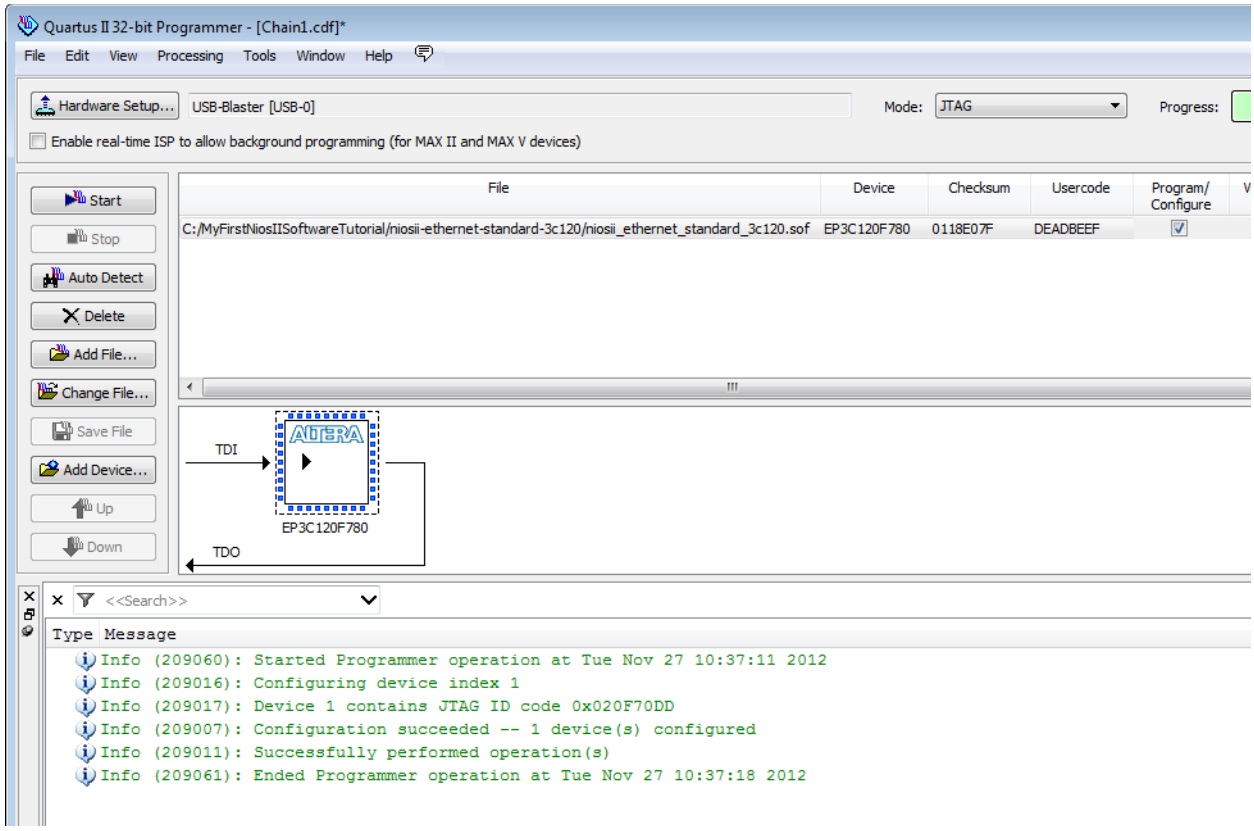
Figure 1–2. Hardware Setup Dialog Box



8. Click **Close**. You return to the Quartus II Programmer window.

9. Click **Auto Detect**. The device on your board is detected automatically.
10. Click the first entry to highlight it. Refer to Figure 1–3 for the location of the first entry.
11. Click **Change File**.
12. Browse to <*your project directory*> and select the SRAM Object File (**.sof**) for your design.
13. Click **OK**.
14. Turn on **Program/Configure** the programming file, as shown in Figure 1–3.

Figure 1–3. Quartus II Programmer



15. Click **Start**.

The **Progress** meter sweeps to 100% as the Quartus II software configures the FPGA. When configuration is complete, the FPGA contains the Nios II system, but does not yet have a program in memory to execute. "Nios II SBT for Eclipse Build Flow" describes how to create, build, download to the FPGA, and run a Nios II program.

Nios II SBT for Eclipse Build Flow

The Nios II SBT for Eclipse is an easy-to-use GUI that automates build and makefile management, and integrates a text editor, debugger, the Nios II flash programmer, and the Quartus II Programmer. Software application templates included in the GUI make it easy for new software programmers to get started quickly.

In this section, you use the Nios II SBT for Eclipse to compile a simple C language example software program to run on the Nios II hardware system configured in the FPGA on your development board. You create a new software project, build it, and run it on the target hardware. You also edit the project, rebuild it, and set up a debugging session.

For a complete tutorial on using the Nios II SBT for Eclipse to develop programs, refer to the software development tutorial in the *Getting Started with the Graphical User Interface* chapter of the *Nios II Software Developer's Handbook*.

Create the Hello World Example Project

In this section, you create a new Nios II application project from an installed example. To begin, perform the following steps in the Nios II SBT for Eclipse:

1. Return to the Nios II SBT for Eclipse.

   You can close the Quartus II Programmer or leave it open in the background if you want to reload the processor system onto your development board quickly.

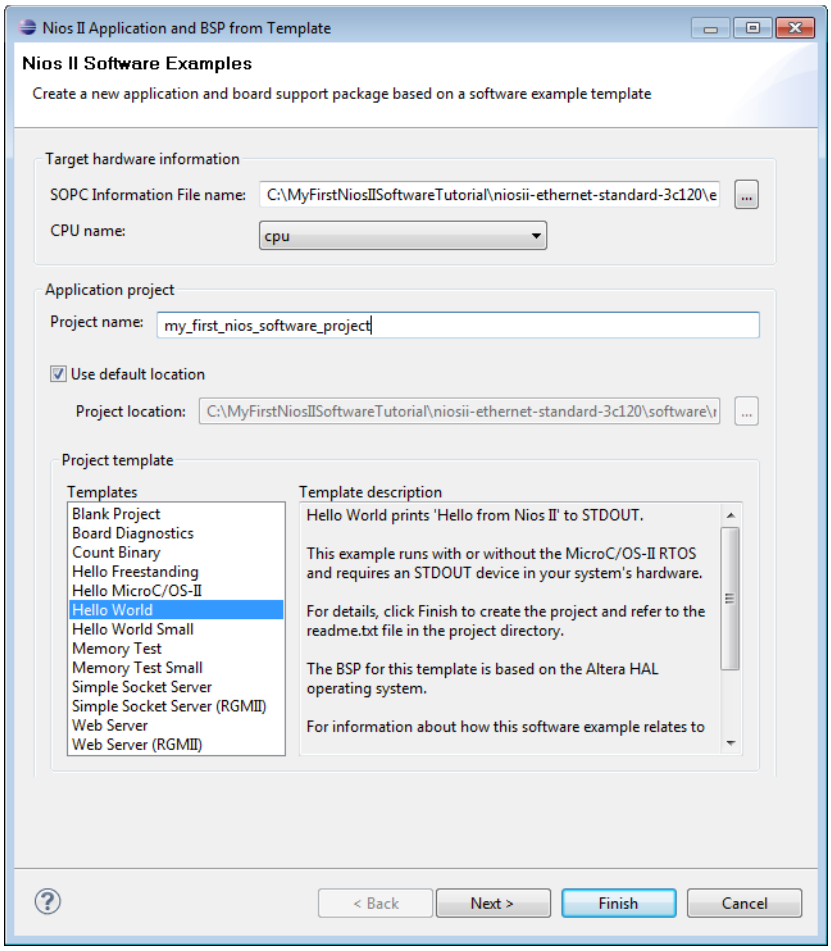
2. On the File menu, point to **New**, and click **Nios II Application and BSP from Template**. The **Nios II Application and BSP from Template** wizard appears.
3. For SOPC Information File name, browse to *<your project directory>* and open the SOPC Information File (**.sopcinfo**) for your design.

   Every Nios II software project needs a system description of the corresponding Nios II hardware system. For the Nios II SBT for Eclipse, this system description is contained in a **.sopcinfo** file.

4. In the **Project name** box, type `my_first_nios_software_project`.

5. In the **Templates** list, select the **Hello World** project template. Figure 1–4 shows the wizard.

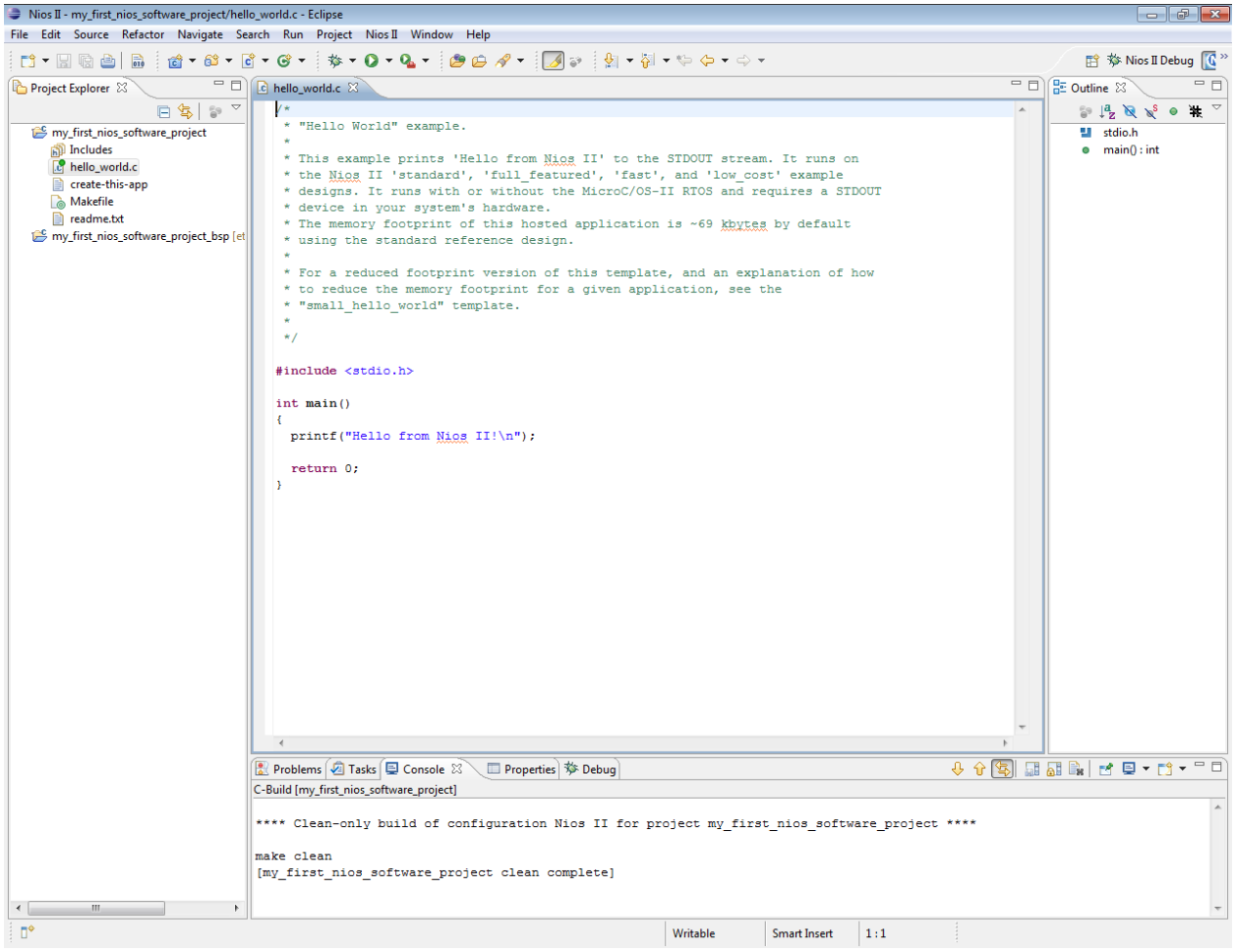
Figure 1–4. Nios II Application and BSP from Template Wizard



6. Click **Finish**. The Nios II SBT for Eclipse creates the **my_first_nios_software_project** project and returns to the Nios II perspective.

7. In the Project Explorer view, expand **my_first_nios_software_project**. Double-click **hello_world.c** to view the source code. Figure 1–5 shows the Nios II perspective.

Figure 1–5. Nios II Perspective with Source Code Editor



When you create a new project, the Nios II SBT for Eclipse creates the following new projects in the Project Explorer view:

- **my_first_nios_software_project** is the application project. This project contains the source and header files for your application.

- **my_first_nios_software_project_bsp** is the board support package (BSP) for your Nios II system hardware. The BSP includes the following details:
  - Component device drivers for your Nios II hardware system
  - newlib C library, which is a richly-featured C library for embedded systems
  - Nios II software packages
    - Nios II hardware abstraction layer (HAL)
    - NicheStack TCP/IP Network Stack, Nios II Edition
    - Nios II host file system
    - Nios II read-only zip file system
    - Micrium's MicroC/OS-II real-time operating system (RTOS)
  - **system.h**, which is a header file that encapsulates your hardware system
  - **alt_sys_init.c**, which is an initialization file that initializes the devices in the system
  - **linker.h**, which is a header file that contains information about the linker memory layout.

Build and Run the Program

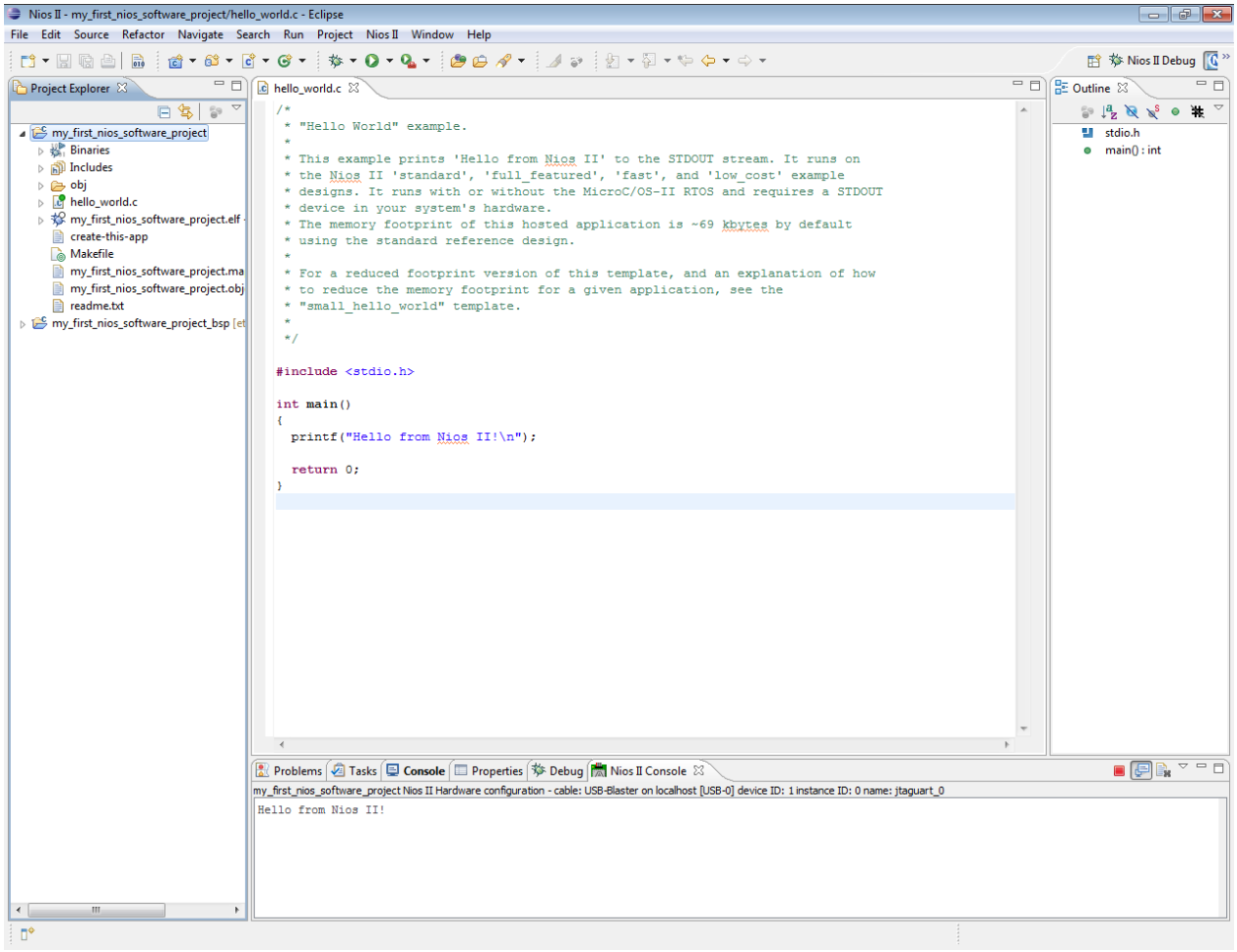
Perform the following steps to build and run the program:

1. To build the program, right-click the **my_first_nios_software_project** project in the Project Explorer view, and click **Build Project**. The **Build Project** dialog box appears and the Nios II SBT for Eclipse begins compiling the project. When compilation completes, the message "Build completed" appears in the Console view. The completion time varies depending on your system.
2. To run the program, right-click **my_first_nios_software_project**, point to **Run As**, and click **Nios II Hardware**. The Nios II SBT for Eclipse downloads the program to the FPGA on the target board and executes the code. The message "Hello from Nios II!" displays in the Nios II Console view.

If the **Run Configurations** dialog box appears, click the **Target Connection** tab and verify the connection to the board. If no connection is shown, click **Refresh Connections**. When a connection appears, click **Run**. For more information about run and debug configurations, refer to the *Getting Started with the Graphical User Interface* chapter of the *Nios II Software Developer's Handbook*.

Figure 1–6 shows the Nios II Console view at the bottom of the Nios II perspective.

Figure 1–6. Nios II Console View



Now that you have created, compiled, and run your first software program, you can perform additional operations, such as configuring the BSP properties, editing and rebuilding the application, and debugging the source code.

Edit and Rerun the Program

You can modify the source code in the Nios II SBT for Eclipse, rebuild the project, run the program, and observe your changes executing on the target board. In this section, you add code that makes an LED on your development board blink.

For more information regarding how the LED blinks, refer to "Debugging the Application" on page 1–9.

Perform the following steps to modify and rerun the program:

1. In the **hello_world.c** file, replace the existing code with the following code:

```
#include <stdio.h>
#include "system.h"
#include "altera_avalon_pio_regs.h"
int main()
{
  int count = 0;
  int delay;
  printf("Hello from Nios II!\n");

while(1)
  {
    IOWR_ALTERA_AVALON_PIO_DATA(PERIPHERAL_SUBSYSTEM_LED_PIO_BASE,
count & 0x01);
    delay = 0;
    while(delay < 2000000)
    {
        delay++;
    }
    count++;
  }
  return 0;
}
```

2. Save the file.
3. Right-click **my_first_nios_software_project** in the Project Explorer view, point to **Run As**, and click **Nios II Hardware**.

   ☞ You do not need to build the project manually; the Nios II SBT for Eclipse automatically rebuilds the program before downloading it to the FPGA.

4. Observe the LED blinking on your development board.

Debugging the Application

To debug your application, perform the following steps:

1. Right-click **my_first_nios_software_project** in the Project Explorer view, point to **Debug As**, and click **Nios II Hardware**.

   ☞ If the **Debug Configurations** dialog box appears, click the **Target Connection** tab and verify the connection to the board. If no connection is shown, click **Refresh Connections**. When a connection appears, click **Debug**. For more information about run and debug configurations, refer to the *Getting Started with the Graphical User Interface* chapter of the *Nios II Software Developer's Handbook*.

2. If the **Confirm Perspective Switch** message box appears, click **Yes**. The context switches to the Nios II Debug perspective.

   After a moment, the `main()` function appears in the editor. A blue arrow next to the first line of code indicates that execution stopped at that line.

3. On the Run menu, click **Resume** to resume execution.

When debugging a project in the Nios II SBT for Eclipse, you can pause, stop, or single step the program, set breakpoints, examine variables, and perform many other common debugging tasks.

To return to the Nios II perspective from the Nios II Debug perspective, click the two arrows (>>) in the upper right corner of the GUI, and click **Nios II**.

For more information about debugging software projects in the Nios II SBT for Eclipse, refer to the *Getting Started with the Graphical User Interface* chapter of the *Nios II Software Developer's Handbook*.

Why the LED Blinks

The Nios II system description header file, **system.h**, contains the software definitions, name, locations, base addresses, and settings for all of the components in the Nios II hardware system. The **system.h** file resides in the **my_first_nios_software_project_bsp** directory. Open the **system.h** file and locate the `PERIPHERAL_SUBSYSTEM_LED_PIO_BASE` macro.

`PERIPHERAL_SUBSYSTEM_LED_PIO_BASE` contains the base address of the PIO peripheral controlling the LEDs. The Nios II processor controls the PIO ports (and therefore the LED) by reading and writing to the register map. For the PIO, there are four registers: `data`, `direction`, `interruptmask`, and `edgecapture`. To turn the LED on and off, the application writes to the PIO `data` register.

The PIO core has an associated software file **altera_avalon_pio_regs.h**. This file defines the core's register map, providing symbolic constants to access the low-level hardware. The **altera_avalon_pio_regs.h** file resides in the **my_first_nios_software_project_bsp** directory in the **drivers\inc** subdirectory.

When you include the **altera_avalon_pio_regs.h** file, several useful functions that manipulate the PIO core registers are available to your program. In particular, the function `IOWR_ALTERA_AVALON_PIO_DATA(base, data)` can write to the PIO data register, turning the LED on and off.

The PIO is just one of many the Qsys system integration tool peripherals that you can use in a system. To learn about the PIO core and other embedded peripheral cores, refer to *Volume 5: Embedded Peripherals* of the *Quartus II Handbook*.

When developing your own designs, you can use the software functions and resources that are provided with the Nios II HAL.

Refer to the *Nios II Software Developer's Handbook* for extensive documentation on developing your own Nios II processor-based software applications.

Board Support Package

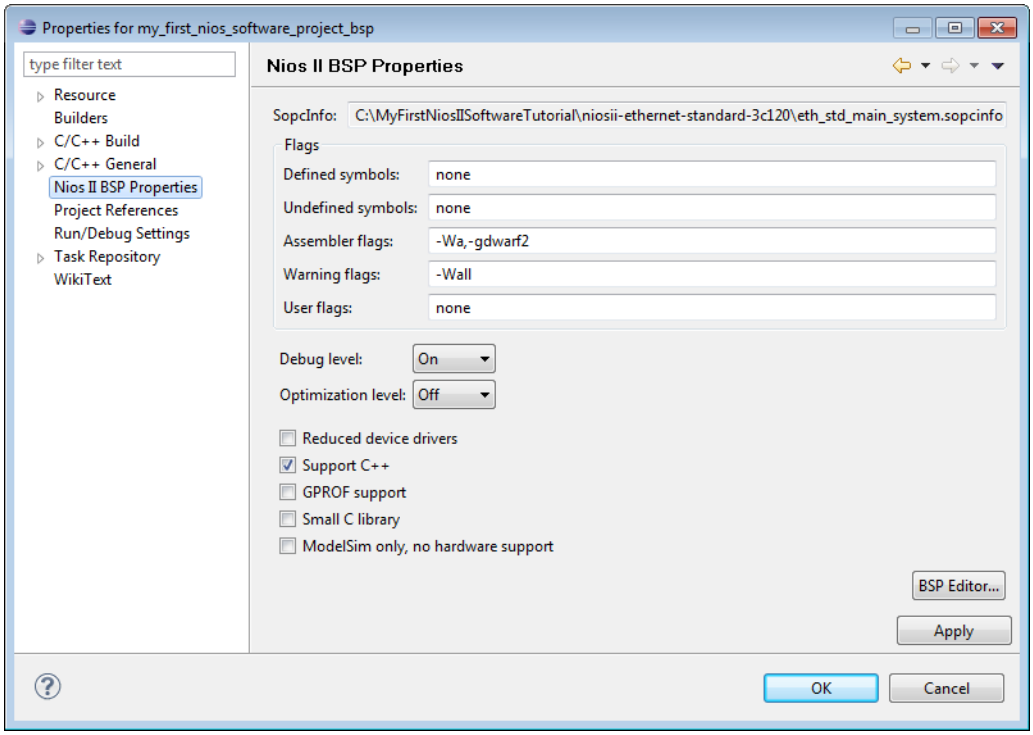
This section explores configuring your BSP.

For BSP properties changes to take affect, you must regenerate your BSP before rerunning your program by either clicking **Generate** in the BSP Editor or by right-clicking the **my_first_nios_software_project_bsp** project in the Project Explorer view, pointing to **Nios II**, and clicking **Generate BSP**.

To access the most common BSP properties, perform the following steps:

1. In the Nios II SBT for Eclipse, right-click **my_first_nios_software_project_bsp** and click **Properties**. The **Properties for my_first_nios_software_project_bsp** dialog box appears.
2. Click **Nios II BSP Properties**. The most-common settings related to how the program interacts with the underlying hardware appear. Figure 1–7 shows the Nios II BSP Properties dialog box.

Figure 1–7. Nios II BSP Properties

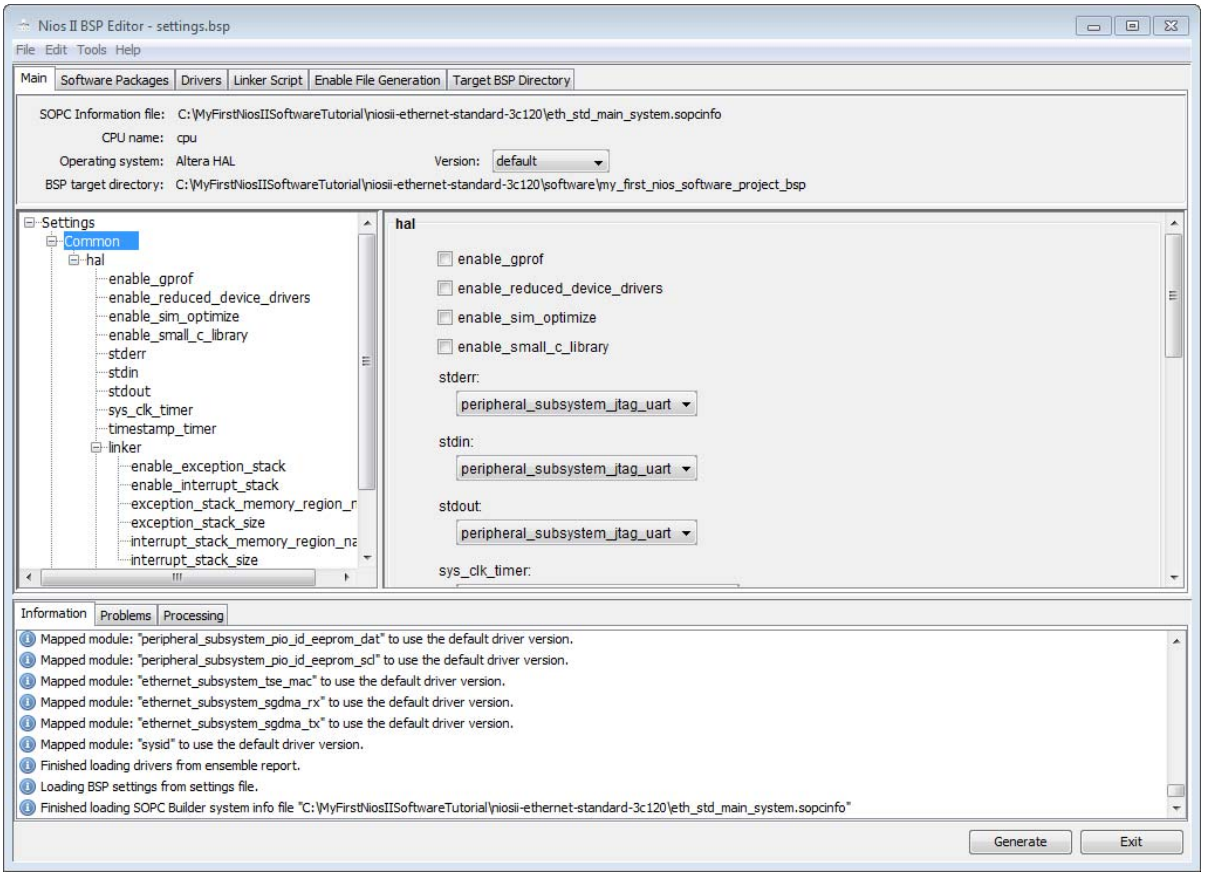


An extensive set of build properties and options are available using the BSP Editor. To access all the BSP properties, perform the following steps:

1. In the lower right corner of the **Properties for my_first_nios_software_project_bsp** dialog box, click **BSP Editor**. The Nios II BSP Editor appears. Figure 1–8 shows the Nios II BSP Editor.

   You can also access the Nios II BSP Editor from the Nios II menu.

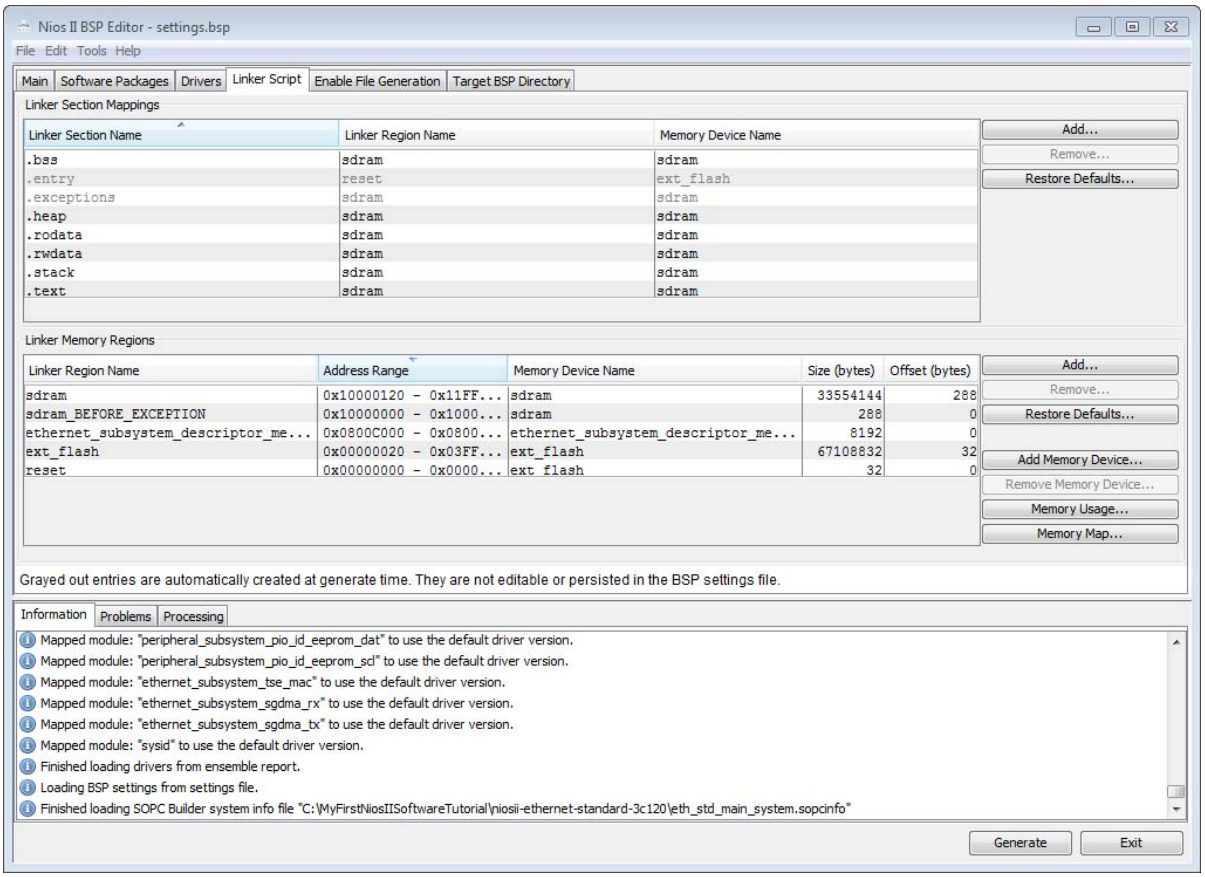
Figure 1–8. Nios II BSP Editor



2. Click the **Settings** tab to see the available settings. For example, you can set the interfaces to use for `stdio`, `stdin`, and `stderr`.
3. Click the **Software Packages** tab to see the software packages, such as files systems, graphics libraries, network stacks, available in your BSP.
4. Click the **Drivers** tab to see the available drivers for the Altera-provided IP in your system.

5. Click the **Linker Script** tab to see the linker section memory assignments. The linker section assignments determine what memory is used to store the compiled executable program when the **my_first_nios_software_project** program runs. Figure 1–9 shows the **Linker Script** tab of the Nios II BSP Editor.

Figure 1–9. Linker Script Tab of the Nios II BSP Editor



6. Click the **Target Generation File** tab to see the files that get added to your BSP at build time.
7. If you have made any changes to your BSP, click **Generate** to update your BSP.
8. On the File menu, click **Exit**.
9. Click **OK** to close the **Properties for my_first_nios_software_project_bsp** dialog box.

Next Steps

The following documents provide next steps to further your understanding of the Nios II processor:

- *Developing Software for Nios II Processors*—These short, online software tutorials walk you through the basics of developing software for the Nios II processor. Access these tutorials on the Embedded Training Resources page of the Altera website.

- *Nios II Software Developer's Handbook* —This handbook provides a complete reference on developing software for the Nios II processor.
- The "Getting Started" section of the *Getting Started with the Graphical User Interface* chapter of the *Nios II Software Developer's Handbook*—This tutorial teaches in detail how to use the Nios II SBT for Eclipse to develop, run, and debug new Nios II application projects.
- *Nios II Processor Reference Handbook*—This handbook provides a complete reference for the Nios II processor hardware.
- The *System Design with Qsys* section of *Volume 1: Design and Synthesis* of the *Quartus II Handbook*—This volume provides a complete reference on using Qsys, including building memory subsystems and creating custom components.
- *Volume 5: Embedded Peripherals* of the *Quartus II Handbook*—This volume contains details on the peripherals provided with the Nios II Embedded Design Suite.

For a complete list of documents available for the Nios II processor, refer to the Literature: Nios II Processor page of the Altera website.

This chapter provides additional information about the document and Altera.

Document Revision History

The following table lists the revision history for this document.

Date	Version	Changes
December 2012	2.1	Updated to match Altera Complete Design Suite version 12.1 flow.
January 2010	2.0	Revised for Nios II Software Build Tools for Eclipse.
July 2008	1.4	Removed command line flow to simplify tutorial.
May 2007	1.3	Minor Fixes.
May 2007	1.2	Minor Fixes.
May 2007	1.1	Minor Fixes.
May 2007	1.0	Initial release.

How to Contact Altera

To locate the most up-to-date information about Altera products, refer to the following table.

Contact (1)	Contact Method	Address
Technical support	Website	www.altera.com/support
Technical training	Website	www.altera.com/training
	Email	custrain@altera.com
Product literature	Website	www.altera.com/literature
Nontechnical support (general)	Email	nacomp@altera.com
(software licensing)	Email	authorization@altera.com

Note to Table:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

The following table shows the typographic conventions this document uses.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Indicate command names, dialog box titles, dialog box options, and other GUI labels. For example, **Save As** dialog box. For GUI elements, capitalization matches the GUI.
bold type	Indicates directory names, project names, disk drive names, file names, file name extensions, software utility names, and GUI labels. For example, **\qdesigns** directory, **D:** drive, and **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Indicate document titles. For example, *Stratix IV Design Guidelines*.
italic type	Indicates variables. For example, *n* + 1. Variable names are enclosed in angle brackets (< >). For example, *<file name>* and *<project name>***.pof** file.
Initial Capital Letters	Indicate keyboard keys and menu names. For example, the Delete key and the Options menu.
"Subheading Title"	Quotation marks indicate references to sections in a document and titles of Quartus II Help topics. For example, "Typographic Conventions."
`Courier type`	Indicates signal, port, register, bit, block, and primitive names. For example, `data1`, `tdi`, and `input`. The suffix n denotes an active-low signal. For example, `resetn`. Indicates command line commands and anything that must be typed exactly as it appears. For example, `c:\qdesigns\tutorial\chiptrip.gdf`. Also indicates sections of an actual file, such as a Report File, references to parts of files (for example, the AHDL keyword `SUBDESIGN`), and logic function names (for example, `TRI`).
↵	An angled arrow instructs you to press the Enter key.
1., 2., 3., and a., b., c., and so on	Numbered steps indicate a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ■ ■	Bullets indicate a list of items when the sequence of the items is not important.
	The hand points to information that requires special attention.
	The question mark directs you to a software help system with related information.
	The feet direct you to another document or website with related information.
	The multimedia icon directs you to a related multimedia presentation.
	A caution calls attention to a condition or possible situation that can damage or destroy the product or your work.
	A warning calls attention to a condition or possible situation that can cause you injury.
	The envelope links to the Email Subscription Management Center page of the Altera website, where you can sign up to receive update notifications for Altera documents.