Using MicroC/OS-II RTOS with the Nios II Processor

Tutorial

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

TU-NIOSII-MCRC/OS-II-3.0

Document last updated for Altera Complete Design Suite version: 11.0
Document publication date: May 2011

Subscribe

Contents

1. Performing the Tutorial

This tutorial familiarizes you with the Nios® II Software Build Tools (SBT) for Eclipse and the MicroC/OS-II development flow. The Nios II SBT for Eclipse offers designers a rich development platform for Nios II applications. The Nios II SBT for Eclipse enables you to integrate the MicroC/OS-II real-time operating system, giving you the ability to build MicroC/OS-II applications for the Nios II processor quickly. This tutorial provides step-by-step instructions for building a simple program based on the MicroC/OS-II RTOS and an Altera hardware design example.

For complete details about MicroC/OS-II for the Nios II processor, refer to the *MicroC/OS-II Real-Time Operating System* chapter in the *Nios II Software Developer's Handbook*.

Hardware and Software Requirements

This tutorial requires the following hardware and software:

- Quartus® II development software v11.0 or later
- One of the Altera development boards listed on the Nios II Ethernet Standard Design Example page of the Altera website.

  The page contains information about the design example and includes the hardware design files for a number of Altera development boards.

- The Altera® USB-Blaster™ or similar cable
- The hardware design files for the Nios II Ethernet Standard design example
- The software example files for this tutorial

Obtaining the Hardware Design Example Files

To obtain the hardware design files for this tutorial, perform the following steps:

1. Browse to the Nios II Ethernet Standard Design Example page of the Altera website.
2. Download the Nios II Ethernet Standard design example .**zip** file that corresponds to your board.
3. Unzip the file in a project directory of your choosing. The remainder of this tutorial refers to this directory as <*your hardware design directory*>.

Obtaining the Software Example Files

To obtain the software example files for this tutorial, perform the following steps:

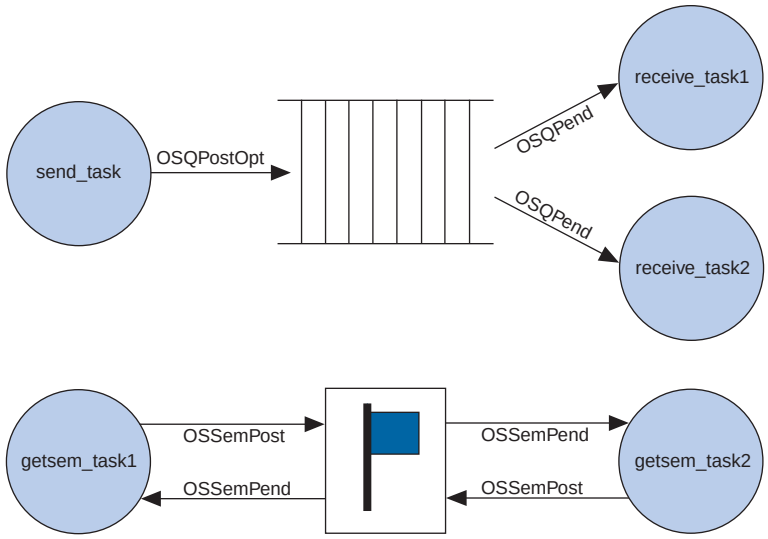
1. Download the file **uCOS_II_Tutorial.zip**. You can find this file on the MicroC/OS-II RTOS with the Nios II Processor web page of the Altera website or alongside this tutorial on the Literature: Nios II Processor page of the Altera website.
2. Unzip the file to a directory on your system. The remainder of this tutorial refers to this directory as <*your software examples directory*>.

When you extract the files from **uCOS_II_Tutorial.zip**, make sure there are no spaces in your destination path.

MicroC/OS-II Tutorial Design

The design example you use for this tutorial is a simple design that exercises some of the basic features of the operating system. Figure 1–1 is a simplified diagram of the application.

Figure 1–1. Tutorial Design Example



As shown in Figure 1–1, the design has five active tasks. **send_task** fills up a message queue with incrementing data. **receive_task1** and **receive_task2** periodically pull messages out of the message queue. **getsem_task1** and **getsem_task2** compete over a shared resource that is protected by a semaphore. The design also has two tasks not shown in Figure 1–1: one for initialization and one for printing status information.

The process for creating a MicroC/OS-II software image for the Nios II processor involves the following steps:

1. Create a new Nios II SBT for Eclipse project.
2. Configure the Nios II board support package (BSP).
3. Build and run the Nios II software project.

These steps are described in detail in the following sections.

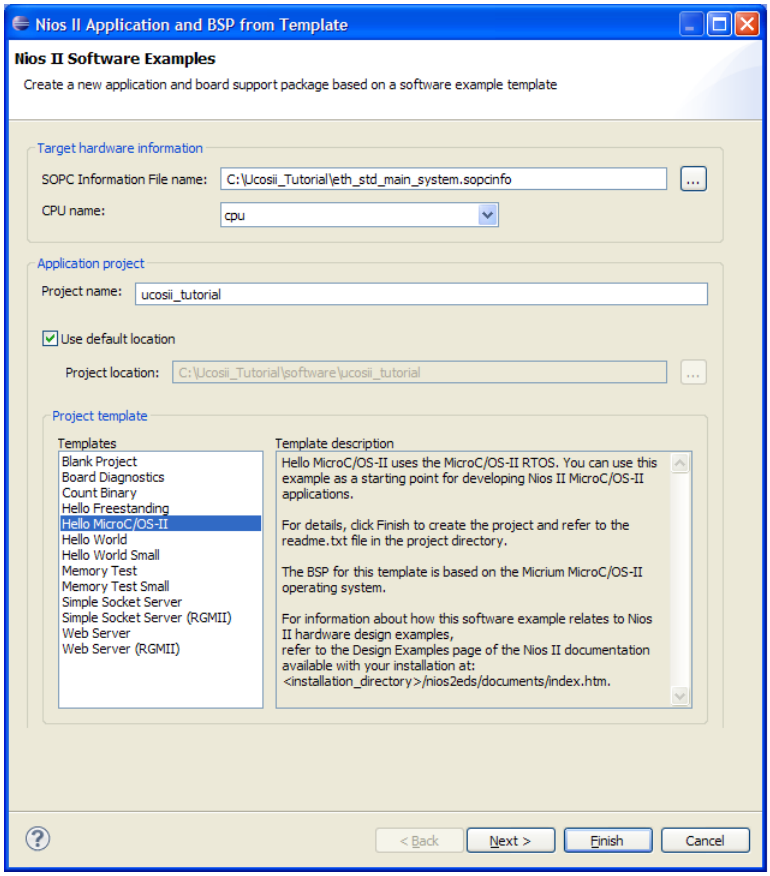
Create the MicroC/OS-II Software Project

In this section, you create new Nios II application and BSP projects using the **Hello MicroC/OS-II** project template available in the Nios II SBT for Eclipse.

1. Start the Nios II SBT for Eclipse by performing one of the following steps, depending on your environment:
   - In the Windows operating system, on the Start menu, point to **Programs** > **Altera** > **Nios II EDS** <*version*>, and click **Nios II** <*version*> **Software Build Tools for Eclipse**.
   - In the Linux operating system, in a command shell, change directories to <*Nios II EDS install path*>, type `./sdk_shell` to start the Nios II Command Shell, and type `eclipse-nios2.`
2. If the **Workspace Launcher** dialog box appears, click **OK** to accept the default workspace location.

3. On the File menu, point to **New**, and click **Nios II Application and BSP from Template**. The first page of the **Nios II Application and BSP from Template** wizard appears, as shown in Figure 1–2.

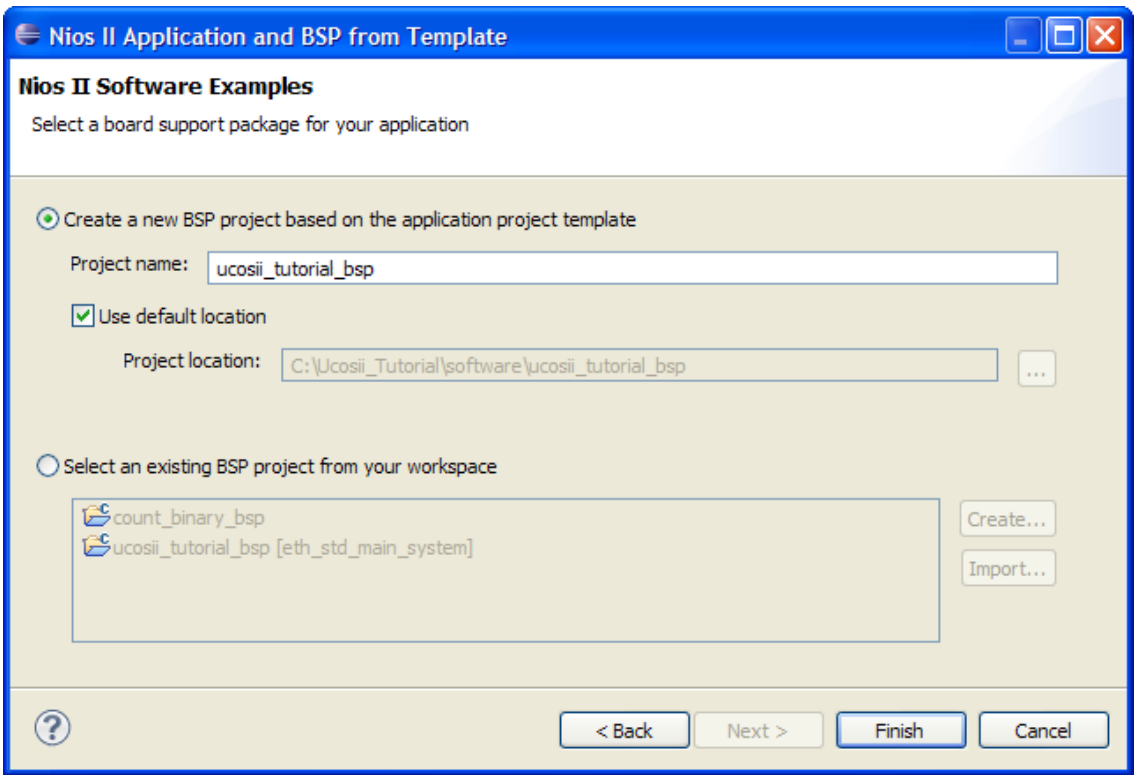
Figure 1–2. Selecting the Application Project Template



4. Under **Target hardware information**, browse to <*your hardware design directory*> and select **eth_std_main_system.sopcinfo**. The Nios II SBT for Eclipse fills in **CPU name** with the processor name found in **eth_std_main_system.sopcinfo**, as shown in Figure 1–2.
5. In the **Project name** box, type `ucosii_tutorial` as the name of your project.

6. Ensure that **Use default location** is on.
7. In the **Templates** list, select **Hello MicroC/OS-II**.
8. Click **Next**. The second page of the **Nios II Application and BSP from Template** wizard appears, as shown in Figure 1–3.

Figure 1–3. Selecting a Board Support Package



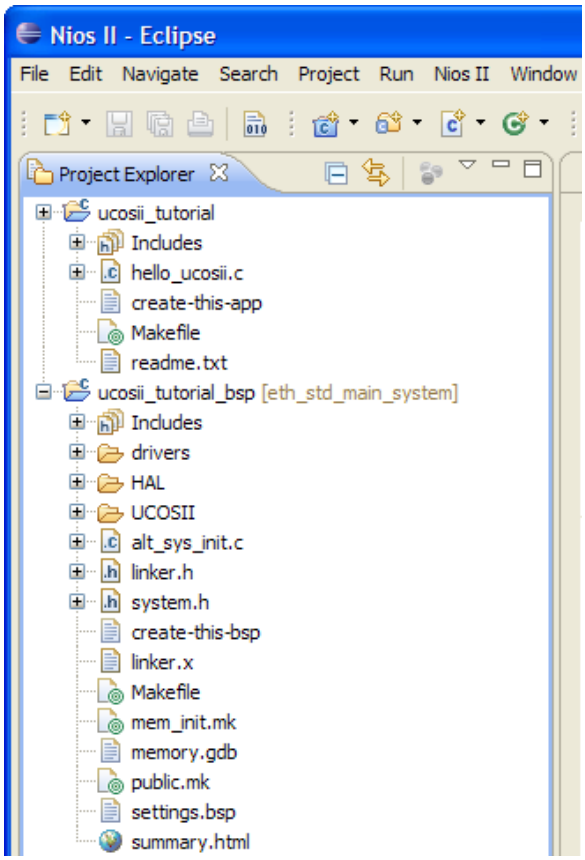
9. Ensure that **Create a new BSP project based on the application project template** is selected and **Use default location** is on.

   In this tutorial, you use the **Hello MicroC/OS-II** project template to easily and quickly create a BSP based on MicroC/OS-II. However, if you wish to use your own BSP, select **Select an existing BSP project from your workspace** and use the bottom portion of the **Nios II Application and BSP from Template** wizard to select, create, or import your BSP. Make sure that your BSP is based on the MicroC/OS-II operating system.

10. Click **Finish** to create the new application and BSP.

11. The Nios II SBT for Eclipse creates two folders, visible in the Project Explorer view: **ucosii_tutorial** and **ucosii_tutorial_bsp**. To view the files created from the **Hello MicroC/OS-II** project template, expand the **ucosii_tutorial** and **ucosii_tutorial_bsp** projects, as shown in Figure 1–4.

Figure 1–4. Files Created from the Project Template



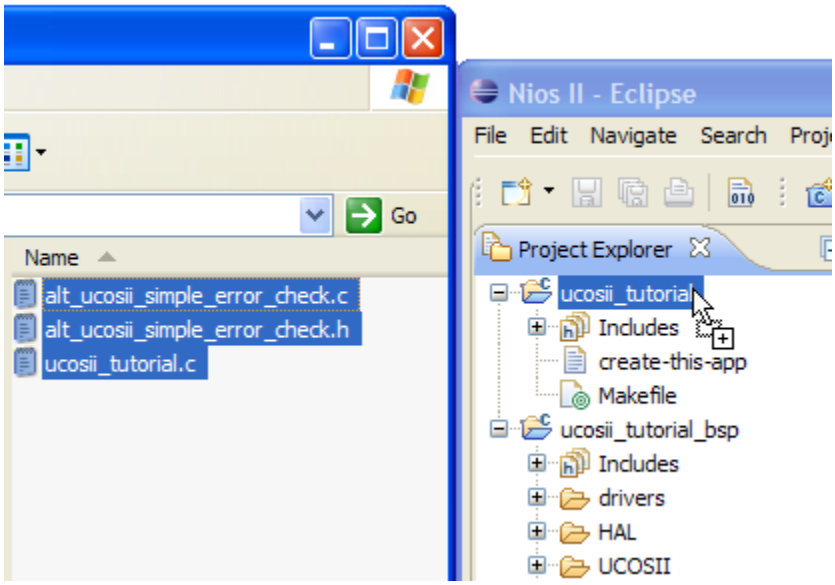
12. In the application project, delete **hello_ucosii.c** (right-click the file and click **Delete**).

13. Add the following files to the project:
    - **ucosii_tutorial.c**
    - **alt_ucosii_simple_error_check.c**
    - **alt_ucosii_simple_error_check.h**

    Drag the files from the <*your software examples directory*>**/software_source_files/** folder and drop them into the **ucosii_tutorial** folder, as shown in Figure 1–5.

    If offered the choice to copy or link to the files, copy them.

Figure 1–5. Adding Software Source Files to the Project



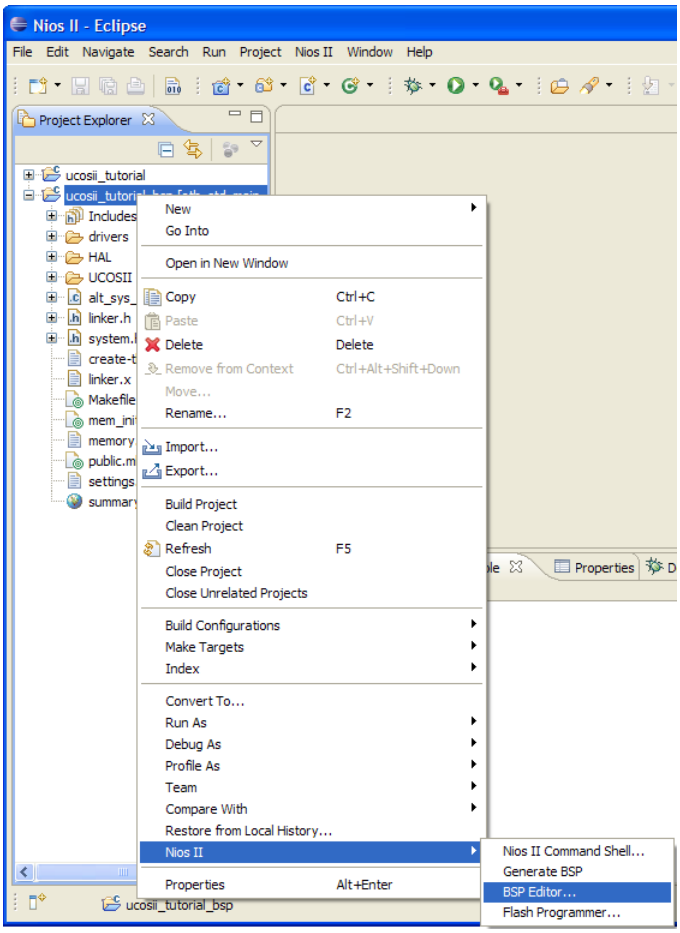
You have created the MicroC/OS-II application. You configure the BSP in the next section.

Configure the BSP

After you create a new BSP, you use the BSP Editor to configure the BSP settings. The BSP uses default settings if you do not configure it.

1. Right-click **ucosii_tutorial_bsp**, point to **Nios II**, and click **BSP Editor**, as shown in Figure 1–6. The Nios II BSP Editor opens.

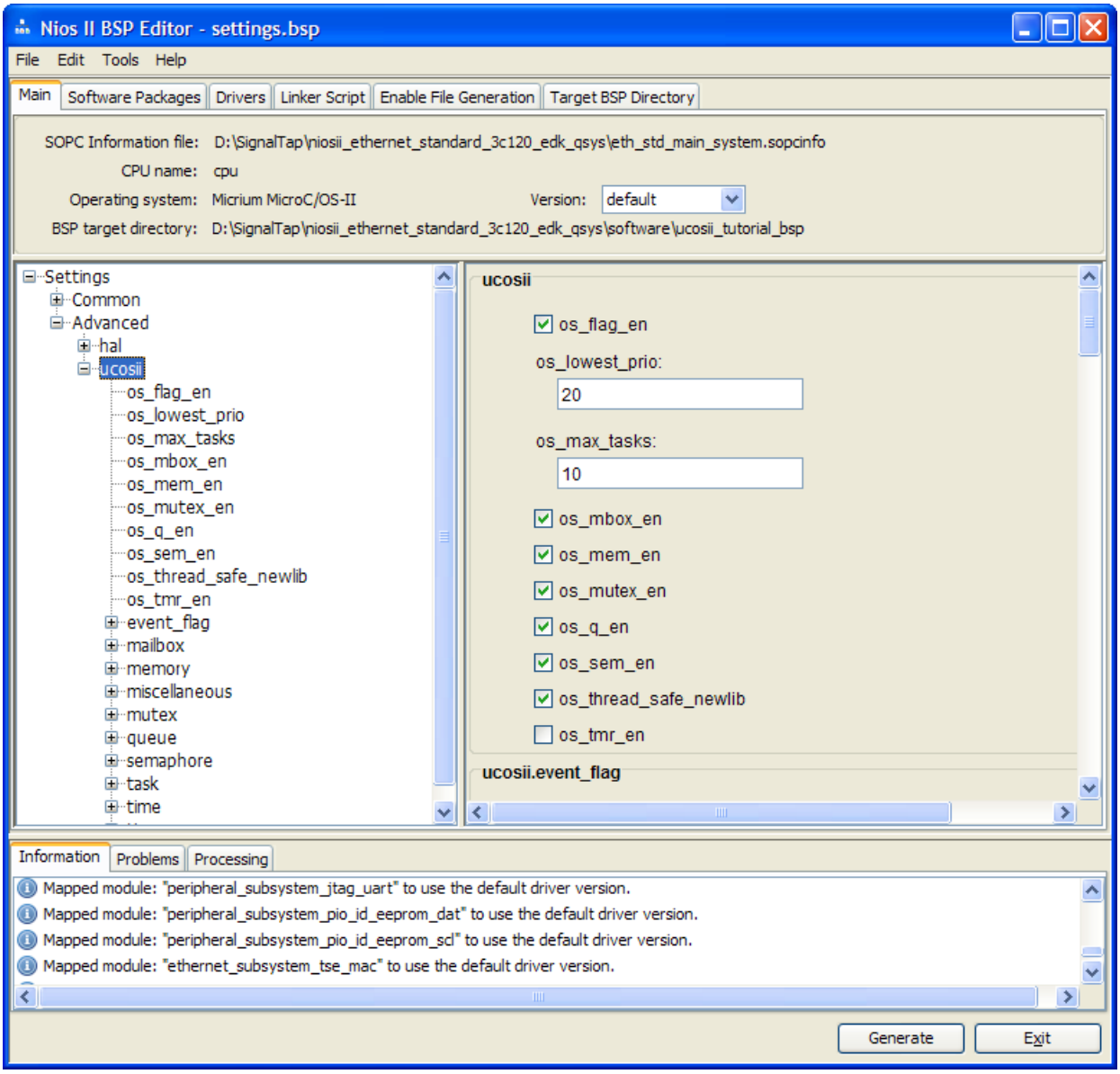
Figure 1–6. Launching the Nios II BSP Editor



2. In the BSP Editor **Main** tab, locate the expandable tree of settings, expand **Advanced**, and then expand and select **ucosii**.

   You can see that the BSP settings for MicroC/OS-II are highly configurable, as shown in Figure 1–7. The settings determine which MicroC/OS-II options are included in the binary image. The Nios II SBT also saves the setting values to the **system.h** file.

Figure 1–7. BSP Settings for MicroC/OS-II



3. To see descriptions of the settings, expand and select each setting.

For details about MicroC/OS-II settings, refer to the *MicroC/OS-II Real-Time Operating System* chapter in the *Nios II Software Developer's Handbook*.

4. Expand the **Common** settings, expand and select **hal**, and ensure that **stderr**, **stdin** and **stdout** are all set to **peripheral_subsystem_jtag_uart**.
5. If you make any changes, save the BSP and click **Generate** to recreate the BSP files.
6. Click **Exit** to close the BSP Editor.

You have finished configuring the BSP. You are ready to build and run the software example as described in the next section.

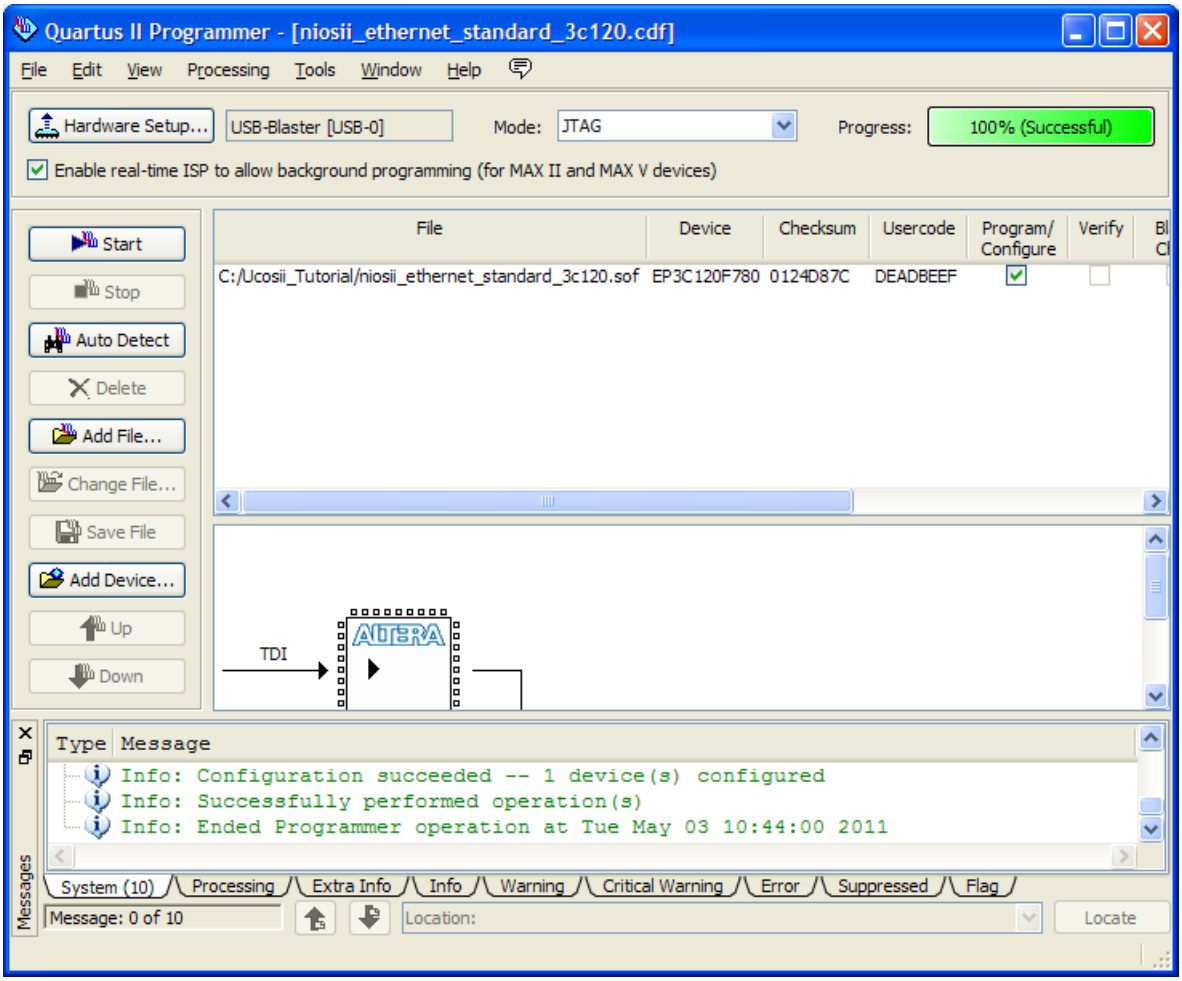
Run the MicroC/OS-II Software Project

In this section, you run the design example on a development board. Using the Quartus II Programmer, you configure the board with an SRAM Object File (**.sof**). Using the Nios II SBT for Eclipse, you build the application, download the Executable and Linking Format File (**.elf**), start it, and observe its output.

To run the tutorial software project, perform the following steps:

1. On the Nios II menu in the Nios II SBT for Eclipse, click **Quartus II Programmer** to start the Quartus II Programmer, shown in Figure 1–8.

Figure 1–8. Quartus II Programmer



2. Make sure power on the development board is on, and the USB cable is properly attached and detected by the Quartus II Programmer.
3. Click **Hardware Setup** in the upper left corner of the Quartus II Programmer to verify your download cable settings. The **Hardware Setup** dialog box appears.
4. On the **Hardware Settings** tab, select the appropriate download cable in the **Currently selected hardware** list. If the appropriate 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**.**

5. Click **Close**.
6. On the File menu in Quartus II Programmer, click **Open**.
7. Navigate to <*your hardware design directory*> and select **niosii_ethernet_standard_**<*board*>**.sof**.
8. Click **Open** to select the **.sof** file and return to the Quartus II Programmer.
9. In the table row for the opened **.sof**, ensure that **Program/Configure** is on.
10. Click **Start** to configure the FPGA. Upon successful completion, the Quartus II Programmer displays messages similar to the following example:

```
Info: Configuration succeeded -- 1 device(s) configured
Info: Successfully performed operation(s)
Info: Ended Programmer operation at Tue May 03 10:44:00 2011
```

11. Exit the Quartus II Programmer. You return to the Nios II SBT for Eclipse.

    If the Quartus II Programmer asks if you want to save changes to the **chain1.cdf** file, click **No**.

12. Right-click **ucosii_tutorial** and select **Build Project** to build the program. Building the software might take several minutes.
13. Right-click **ucosii_tutorial**, point to **Run As**, and click **Nios II Hardware** to download the program to the board and run it.

    If the **Run Configurations** dialog box appears, click the **Target Connection** tab. Then click **Refresh Connections** and **Apply** until a board connection establishes. Once the board connection is established, click **Run**.

After download is complete, the Nios II **Console** view is updated periodically by tutorial software's `print_status_task()` function as shown in Example 1–1.

The numbers displayed might vary.

Example 1–1. print_status_task() Output

```
**************************************************************
Hello From MicroC/OS-II Running on NIOS II.  Here is the status:

The number of messages sent by the send_task:         123

The number of messages received by the receive_task1: 73

The number of messages received by the receive_task2: 24

The shared resource is owned by: getsem_task2

The Number of times getsem_task1 acquired the semaphore 240

The Number of times getsem_task2 acquired the semaphore 185

**************************************************************
```

If the correct output does not appear on the Nios II Console view, you might need to turn on **Reset the selected target system** in the **Target Connection** tab of the **Run Configurations** dialog box. You must relaunch your application after modifying the run configuration.

Congratulations! You have successfully configured, built, and run a MicroC/OS-II program.

For more information about this example, examine the source file **ucosii_tutorial.c**. For more information about the MicroC/OS-II operating system, refer to the *MicroC/OS-II Real-Time Operating System* chapter in the *Nios II Software Developer's Handbook*.

This chapter provides additional information about the document and Altera.

Document Revision History

The following table shows the revision history for this document.

Date	Version	Changes
May 2011	3.0	Updates for Qsys
June 2010	2.0	Updates for Altera Complete Design Suite 9.1 SP1 release
January 2007	1.2	Updates for Nios II 6.1 release
September 2004	1.1	Updates for Nios II 1.01 release
May 2004	1.0	First publication

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*.

Visual Cue	Meaning
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.
	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.