Nios® II

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

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com

TU-NIOSII-MCRC/OS-II-1.2

Contents

About this Tutorial

This tutorial introduces you to the Nios® II integrated development environment (IDE) and MicroC/OS-II development flow. It shows you how to use the Nios II IDE to create a new MicroC/OS-II project to configure, build, and run a MicroC/OS-II program on the Nios development board.

Table 1–1 shows the tutorial revision history.

Table 1–1. Tutorial Revision History

Date	Description
January 2007	Updates for Nios II 6.1 release.
September 2004	Updates for Nios II 1.01 release.
May 2004	First publication.

How to Find Information

- The Adobe Acrobat Find feature allows you to search the contents of a PDF file. Click the binoculars toolbar icon to open the Find dialog box.
- Bookmarks serve as an additional table of contents.
- Thumbnail icons, which provide miniature previews of each page, provide a link to the pages.
- Numerous links, shown in green text, allow you to jump to related information.

How to Contact Altera

For the most up-to-date information about Altera® products, go to the Altera world-wide web site at www.altera.com. For technical support on this product, go to www.altera.com/mysupport. For additional information about Altera products, consult the sources shown below.

Information Type	USA & Canada	All Other Locations
Technical support	**www.altera.com**	**www.altera.com**
	(800) 800-EPLD (3753) (7:00 a.m. to 5:00 p.m. Pacific Time)	(408) 544-7000 *(1)* (7:00 a.m. to 5:00 p.m. Pacific Time)
Product literature	**www.altera.com**	**www.altera.com**
Altera literature services	**literature@altera.com** *(1)*	**literature@altera.com** *(1)*

Information Type	USA & Canada	All Other Locations
Non-technical customer service	(800) 767-3753	(408) 544-7000 (7:30 a.m. to 5:30 p.m. Pacific Time)
FTP site	**ftp.altera.com**	**ftp.altera.com**

Note to table:
(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

This document uses the typographic conventions shown below.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Command names, dialog box titles, checkbox options, and dialog box options are shown in bold, initial capital letters. Example: **Save As** dialog box.
bold type	External timing parameters, directory names, project names, disk drive names, filenames, filename extensions, and software utility names are shown in bold type. Examples: $\mathbf{f_{MAX}}$, **\qdesigns** directory, **d:** drive, **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Document titles are shown in italic type with initial capital letters. Example: *AN 75: High-Speed Board Design.*
Italic type	Internal timing parameters and variables are shown in italic type. Examples: t_{PIA}, $n + 1$. Variable names are enclosed in angle brackets (< >) and shown in italic type. Example: *<file name>*, *<project name>***.pof** file.
Initial Capital Letters	Keyboard keys and menu names are shown with initial capital letters. Examples: Delete key, the Options menu.
"Subheading Title"	References to sections within a document and titles of on-line help topics are shown in quotation marks. Example: "Typographic Conventions."
`Courier type`	Signal and port names are shown in lowercase Courier type. Examples: `data1`, `tdi`, `input`. Active-low signals are denoted by suffix `n`, e.g., `resetn`. Anything that must be typed exactly as it appears is shown in Courier type. For example: `c:\qdesigns\tutorial\chiptrip.gdf`. Also, sections of an actual file, such as a Report File, references to parts of files (e.g., the AHDL keyword `SUBDESIGN`), as well as logic function names (e.g., `TRI`) are shown in Courier.
1., 2., 3., and a., b., c., etc.	Numbered steps are used in a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ● •	Bullets are used in a list of items when the sequence of the items is not important.
✓	The checkmark indicates a procedure that consists of one step only.
	The hand points to information that requires special attention.

Visual Cue	Meaning
	A caution calls attention to a condition or possible situation that can damage or destroy the product or the user's work.
	A warning calls attention to a condition or possible situation that can cause injury to the user.
↵	The angled arrow indicates you should press the Enter key.
	The feet direct you to more information on a particular topic.

Tutorial

Introduction

This tutorial familiarizes you with the Nios II IDE and MicroC/OS-II development flow. The Nios II IDE offers designers a rich development platform for Nios II applications. The Nios II IDE contains the MicroC/OS-II real-time operating system, providing designers with 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.

For complete details on 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 & Software Requirements

This tutorial requires the following hardware and software:

- Quartus® II Development Software version 6.1 or later
- One of the following Nios Development Boards:
  - Stratix® II Edition
  - Cyclone™ II Edition
  - Cyclone™ Edition
  - Stratix Edition
  - Stratix Professional Edition
- Altera USB-Blaster™ or similar cable

To complete this tutorial, you must have the Nios II IDE installed, and your Nios development board must be connected to a host PC. For details on how to install the software and connect the board, see the *Nios II Development Kit Getting Started User Guide*.

Tutorial Files

The Nios II Embedded Design Suite (EDS) includes the tutorial designs. The hardware design is the **standard** reference design and the software design is a Nios II IDE template. After you install the Nios II EDS, you can find the design files in the Nios II EDS directory structure.

- The standard reference hardware design files are located in the *<Nios II EDS install path>***\examples***<verilog or vhdl>**<Nios development board>***\standard** directory.

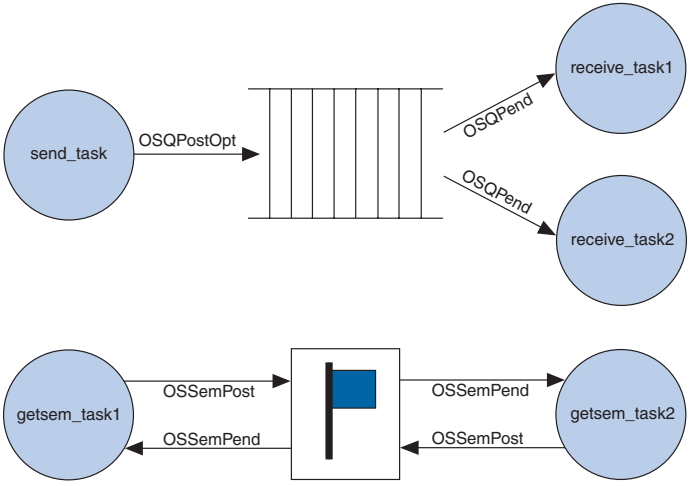
- The MicroC/OS-II tutorial software design files are located in the *<Nios II EDS install path>*\examples\software\ucosii_tutorial directory.

  You do not need to move or copy these software files because the Nios II IDE copies them to your working project directory automatically.

Hello MicroC/OS-II Design

The example design 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 Example Design



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 IDE project.
2. Configure the Nios II system library project.
3. Build and run the Nios II software project.

These steps are described in detail in the following sections.

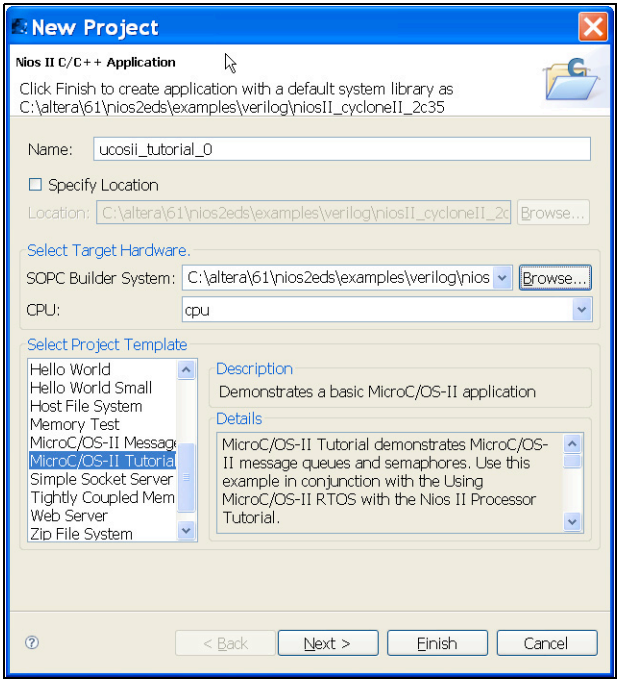
Create a New Nios II IDE Project

In this section you create a new Nios II IDE project using a software template. Perform the following steps:

1. On the Windows Start menu, point to **Programs**, **Altera**, **Nios II EDS <*version number*>**, and click **Nios II <*version number*> IDE**.
2. On the File menu, point to **New**, and click **Nios II C/C++ Application**. The first page of **New Project** wizard opens.
3. Under **Select Project Template**, select the **MicroC/OS-II Tutorial**. The project name and project path are filled in for you automatically. Keep these defaults.
4. Click **Browse** under **Select Target Hardware**.
5. Browse to the standard example directory for the Nios development board that you are targeting, i.e., <*Nios II EDS install path*>**\examples**<*verilog or vhdl*>\<*Nios development board*>**\standard** directory, where <*Nios development board*> is one of the following options:
   - **niosII_cyclone_1c20** for the Cyclone EP1C20 board
   - **niosII_cycloneII_2c35** for the Cyclone EP2C35 board
   - **niosII_stratix_1s10** for the Stratix EP1S10 board
   - **niosII_stratix_1s10_ES** for the Stratix EP1S10ES board
   - **niosII_stratix_1s40** for the Stratix EP1S40 board
   - **niosII_stratixII_2s60** for the Stratix EP2S60 board
   - **niosII_stratixII_2s60_ES** for the Stratix EP2S60ES board
   - **niosII_stratixII_2s60_rohs** for the Stratix EP2S60 ROHS board
6. Click the file **std**_<*device name*>**.ptf**.

7. Click **Open**. You return to the **New Project** wizard. As shown in Figure 1–2, the **SOPC Builder System** box under **Select Target Hardware** contains the path to the **.ptf** file for the standard example design. Additionally, the **CPU** box contains the name of the CPU in the example SOPC Builder system.

Figure 1–2. New Project Wizard Page 1

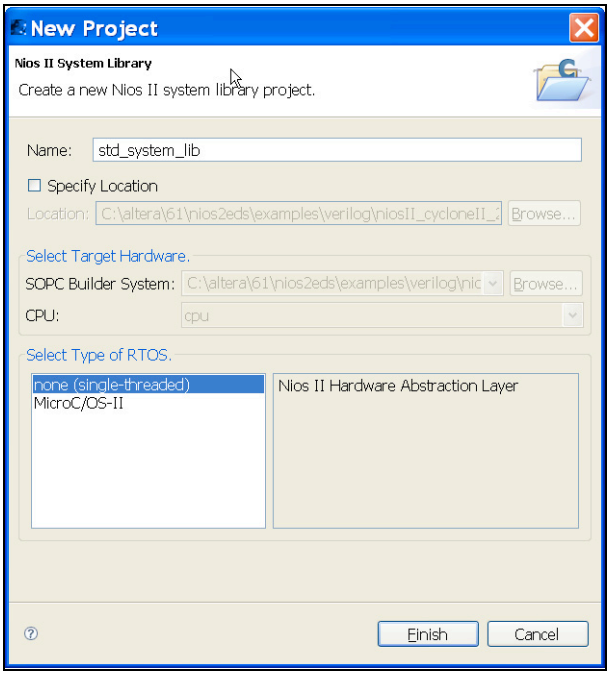


8. Click **Next** to go the second page of the **New Project** wizard.

9. Select the **Select or create a system library** option.

10. Click **New System Library Project** to open the Nios II System Library page. See Figure 1–3.

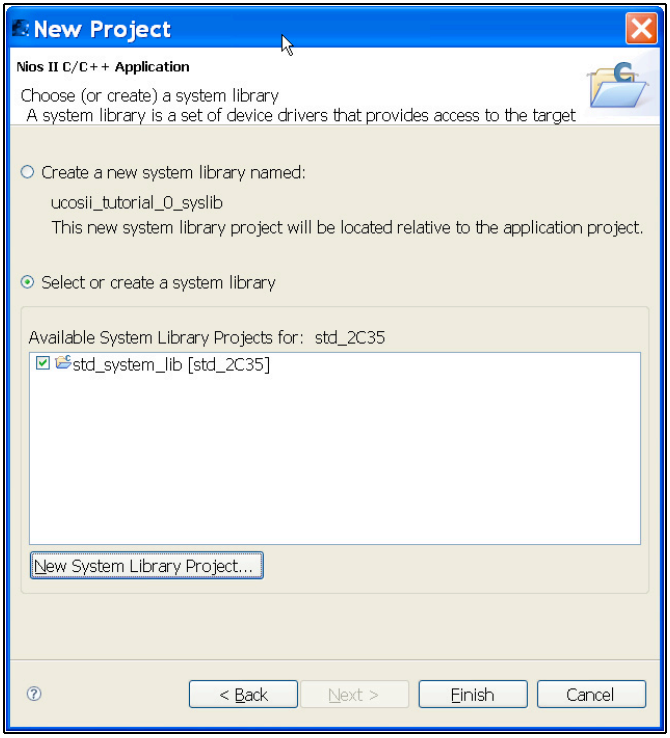
Figure 1–3. New System Library Dialog Box



11. Type `std_system_lib` in the **Name** box.
12. Select **MicroC/OS-II** in the **Select Type of RTOS** box.

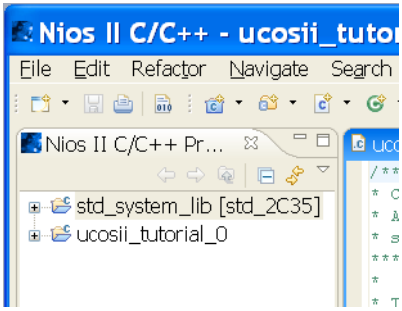
13. Click **Finish** to copy the template files and return to the **New Project** wizard. See Figure 1–4.

Figure 1–4. New Project Wizard Page 2



14. Click **Finish** to complete creation of your new project. The wizard creates two projects in the **Nios II C/C++ Projects** view, as shown in Figure 1–5. **ucosii_tutorial_0** is the application project, and **std_system_lib** is the system library project.

Figure 1–5. Nios II C/C++ Projects



Configure the System Library

In general, after you create a new system library you must configure it, e.g., defining `stdin`, `stdout`, `stderr`, etc. For this tutorial, you must configure MicroC/OS-II. During configuration, the Nios II IDE saves the appropriate values into the **system.h** file. Perform the following steps to configure the MicroC/OS-II kernel.

1. Right-click on the system library, **std_system_lib**, in the **Nios II C/C++ Projects** view.

2. Click **System Library Properties** in the pop-up menu to open the **Properties** dialog box. If you are using uC/OS-II in evaluation mode as distributed by Altera, you might see the license notification in Figure 1–6 on page 1–8. Click **OK**. The system library options dialog box appears as shown in Figure 1–7.

Figure 1–6. uC/OS-II License Notification

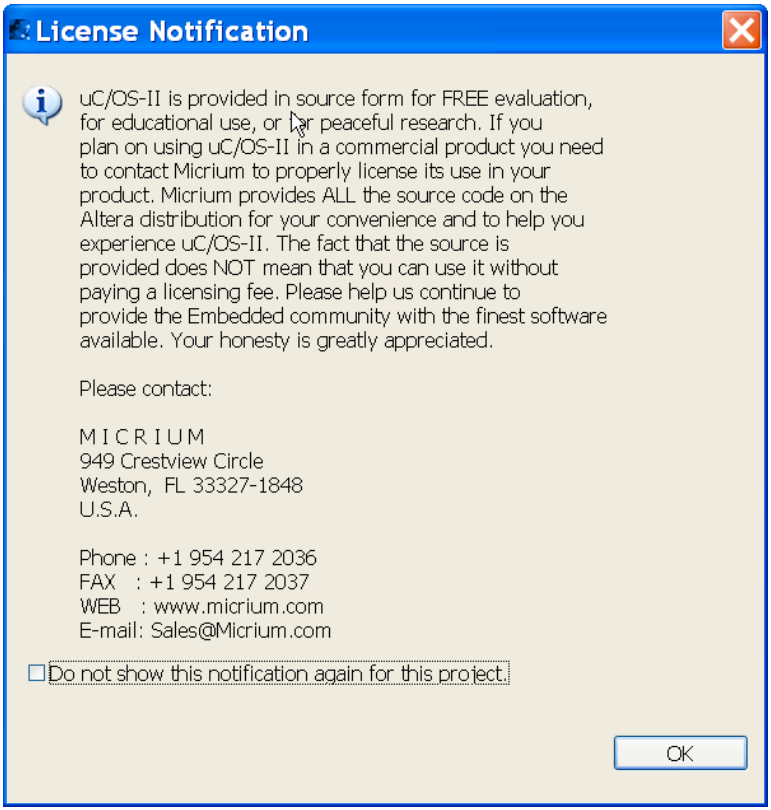
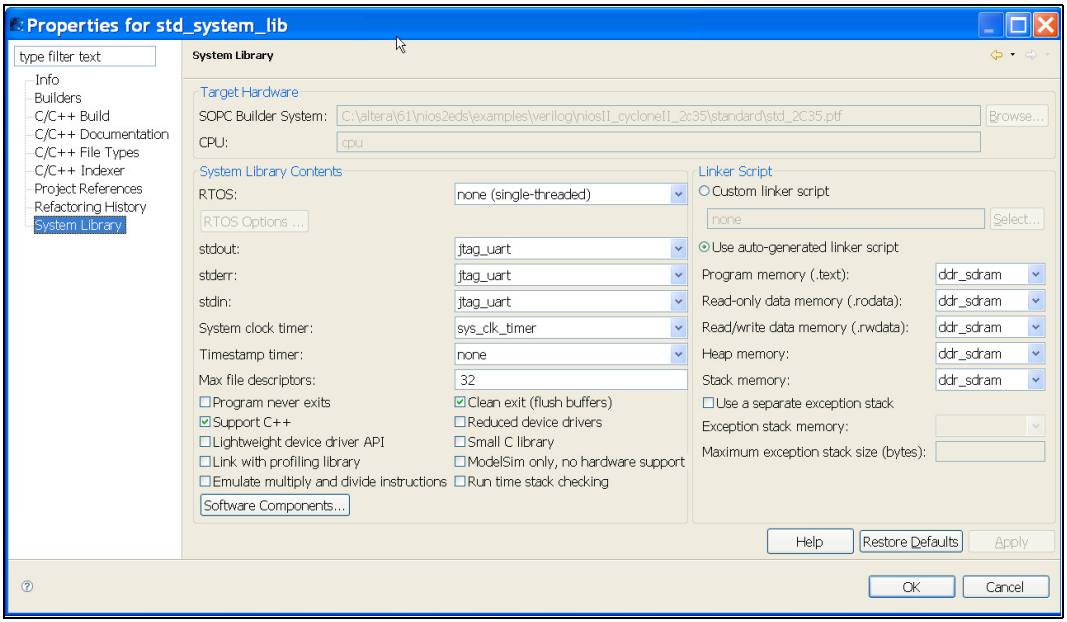
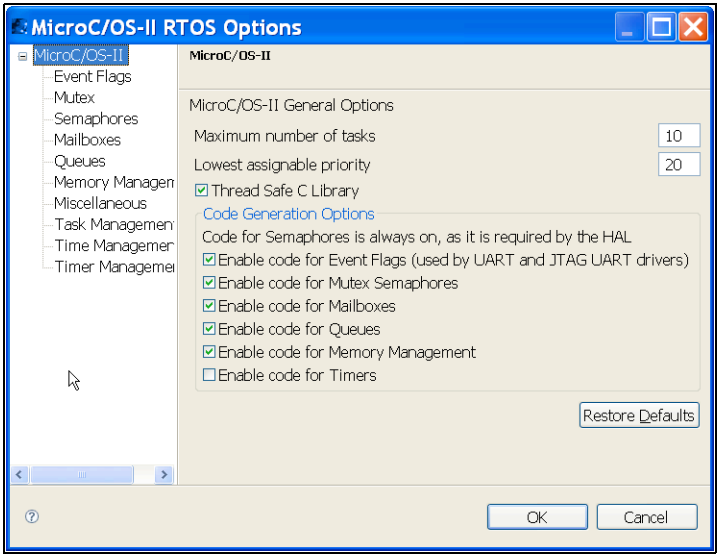


Figure 1–7. System Library Options



3. Click **RTOS Options**. The **MicroC/OS-II RTOS Options** dialog box opens, as shown in Figure 1–8.

Figure 1–8. MicroC/OS-II RTOS Options



4. Click the "+" in the left hand panel to expand the contents under MicroC/OS-II. MicroC/OS-II is highly configurable. The options you select in this dialog box are saved to the **system.h** file and determine which MicroC/OS-II options are included in the binary image. Examine the options you can select by clicking each of the categories listed at the left side of the dialog box, under MicroC/OS-II.

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

5. Click **OK** to use the default settings. You return to the system library properties dialog box.

6. Click **OK** to complete the configuration.

You have finished configuring the system library and are ready to build and run the example as described in the following section.

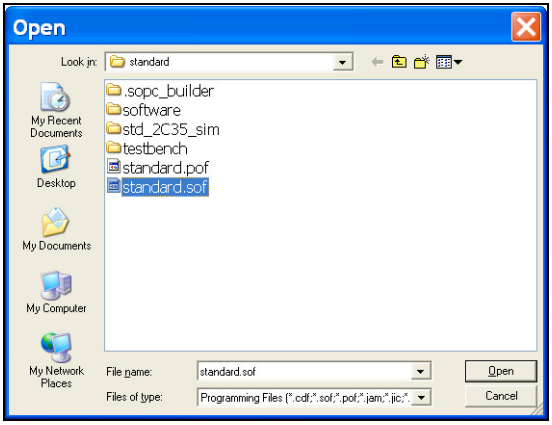
For additional details on how to build and run programs with the Nios II IDE, see the *Nios II Software Development Tutorial* in the Nios II IDE help system.

Build & Run the Nios II Software Project

In this section, you run the example design on a development board. Using Nios II IDE, you build the application, configure the development board with a valid configuration (**.sof**) file, and download the Executable and Linkable Format File (**.elf**).

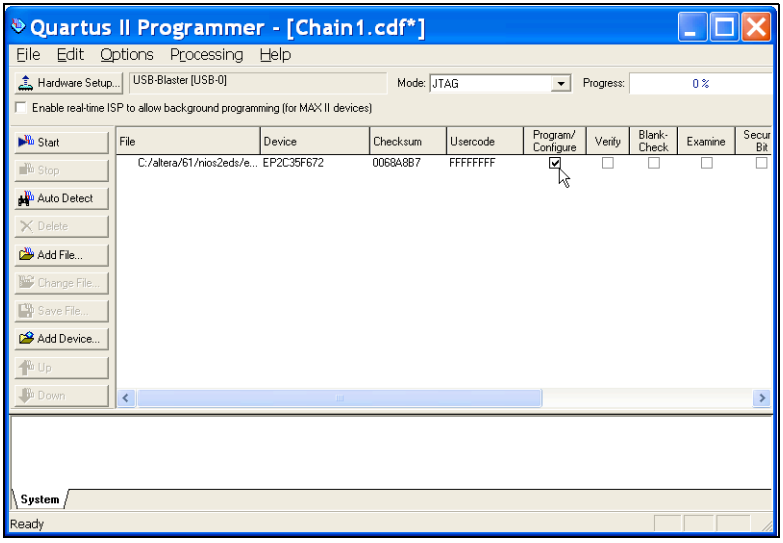
1. In the Nios II **C/C++ Projects** view, select the **ucosii_tutorial_0** project.
2. On the Tools menu, click **Quartus II Programmer ...** to open the Quartus II Programmer.
3. On the Quartus II Programmer's File menu, click **Open**.
4. Select **standard.sof** as shown in Figure 1–9.

Figure 1–9. Open Dialog Box



5. Click **Open**. You return to the Quartus II Programmer.

Figure 1–10. Quartus II Programmer



6. Turn on the **Program/Configure** check box to the right of the **.sof** filename, as shown in Figure 1–10.

7. Click **Start** to configure the FPGA on the development board with the SOF. Upon successful completion, the Quartus II Programmer displays messages similar to this:

```
Info: Configuration succeeded -- 1 device(s) configured
Info: Successfully performed operation(s)
Info: Ended Programmer operation at Wed Jan 03 17:50:03 2007
```

8. On the File menu, click **Exit** to close the Quartus II Programmer. You return to the Nios II IDE.

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

9. On the Run menu, click **Run As**, and click **Nios II Hardware** to build the program, download it to the board, and run it.

For additional information on the Nios II IDE build process and run configuration setup, see the *Nios II Software Development Tutorial* within the Nios II IDE online help.

After download is complete, the Nios II IDE **Console** view is updated periodically by `print_status_task()` as shown in Figure 1–11. The numbers displayed vary depending on the type of Nios development board the example runs on.

Figure 1–11. 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

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

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

For further details on this tutorial, you can examine the source file **ucosii_tutorial.c**. For complete reference on MicroC/OS-II, refer to the "MicroC/OS-II Real Time Operating System" chapter in the *Nios II Software Developer's Handbook*.