

Introduction

This chapter familiarizes you with the main features of the Nios® II integrated development environment (IDE). This chapter is only a brief introduction to the look and feel of the Nios II IDE—it is not a user guide. The easiest way to get started using the Nios II IDE is to launch the tool and perform the Nios II software development tutorial, available in the help system.



The figures in this chapter may differ slightly from the actual GUI due to improvements in the software.



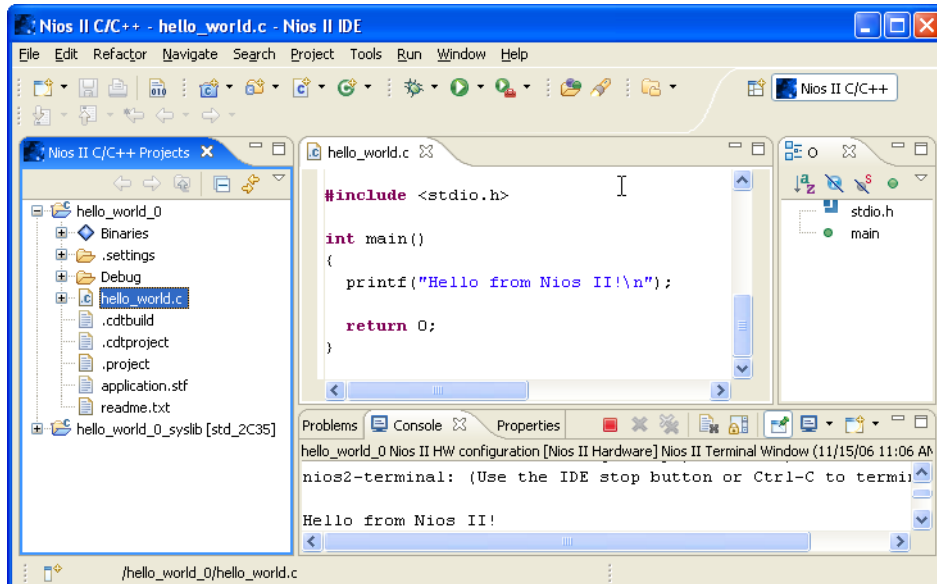
For more information on all IDE-related topics, refer to the Nios II IDE help system.

This chapter contains the following sections:

- “The Nios II IDE Workbench” on page 2–1
- “EDS Design Flows and the IDE” on page 2–3
- “Creating a New IDE-Managed Project” on page 2–4
- “Building and Managing Projects” on page 2–5
- “Running and Debugging Programs” on page 2–7
- “Importing User-Managed Projects” on page 2–10
- “Programming Flash” on page 2–19
- “Help System” on page 2–21

The Nios II IDE Workbench

The term “workbench” refers to the desktop development environment for the Nios II IDE. The workbench is where you edit, compile and debug your programs in the IDE. [Figure 2–1 on page 2–2](#) shows an example of the workbench.

Figure 2–1. The Nios II IDE Workbench

Perspectives, Editors, and Views

Each workbench window contains one or more perspectives. Each perspective provides a set of capabilities for accomplishing a specific type of task. For example, [Figure 2–1](#) shows the Nios II C/C++ development perspective.

Most perspectives in the workbench comprise an editor area and one or more views. An editor allows you to open and edit a project resource (i.e., a file, folder, or project). Views support editors, provide alternative presentations, and ways to navigate the information in your workbench. [Figure 2–1](#) shows a C program open in the editor, and the Nios II C/C++ Projects view in the left-hand pane of the workbench. The Nios II C/C++ Projects view displays information about the contents of open Nios II projects.

Any number of editors can be open at once, but only one can be active at a time. The main menu bar and toolbar for the workbench window contain operations that are applicable to the active editor. Tabs in the editor area indicate the names of resources that are currently open for editing. An asterisk (*) indicates that an editor has unsaved changes. Views can also provide their own menus and toolbars, which, if present, appear along the top edge of the view. To open the menu for a view, click

the drop-down arrow icon at the right of the view's toolbar or right-click in the view. A view might appear on its own, or stacked with other views in a tabbed notebook.

EDS Design Flows and the IDE

The Nios II IDE is an integral part of both Nios II embedded design suite (EDS) design flows. The main distinction between the two design flows is in the management of the project.

IDE-Managed Projects and Makefiles

In the Nios II IDE design flow, the IDE manages Nios II C/C++ application and system library projects and makefiles that you create with the **New Project** wizard in Nios II IDE. In IDE-managed projects, the IDE manages the makefiles for you. The best way to modify and build an IDE-managed project is through the IDE. You manage the system library project settings with the **System Library** page of the **Properties** dialog box.

You can manually convert an IDE-managed project to a user-managed project. For details, see the *Nios II Software Build Tools* chapter of the *Nios II Software Developer's Handbook*.

User-Managed Projects and Makefiles

In the Nios II software build tools design flow, you manage Nios II application, library, and BSP projects and makefiles, giving you total control. Typically, you create user-managed projects outside of the IDE and then import them into the IDE for debugging.

The best way to create a user-managed project, and modify the settings, is with the Nios II software build tools through a command shell or scripting tool. You can create the makefile by hand, or you can use the Nios II software build tools to create it.

When you import a user-managed C/C++ application project into the IDE for debugging, the Nios II IDE does not manage the project or makefile. The IDE does not support management of or changes to a BSP project after import. You must manage the BSP with the software build tools, outside of the IDE.

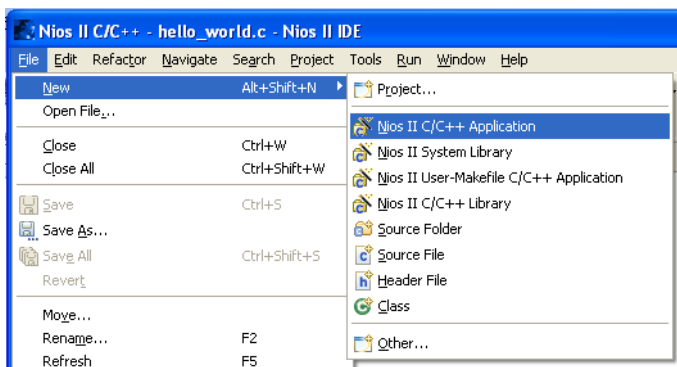


User-managed projects and IDE-managed projects are not interchangeable.

Creating a New IDE-Managed Project

The Nios II IDE provides a **New Project** wizard that guides you through the steps to create new IDE-managed projects. To start the **New Project** wizard for Nios II C/C++ application projects, on the File menu in the Nios II C/C++ perspective, point to **New**, and then click **Nios II C/C++ Application**, as shown in [Figure 2–2](#).

Figure 2–2. Starting the Nios II C/C++ Application New Project Wizard

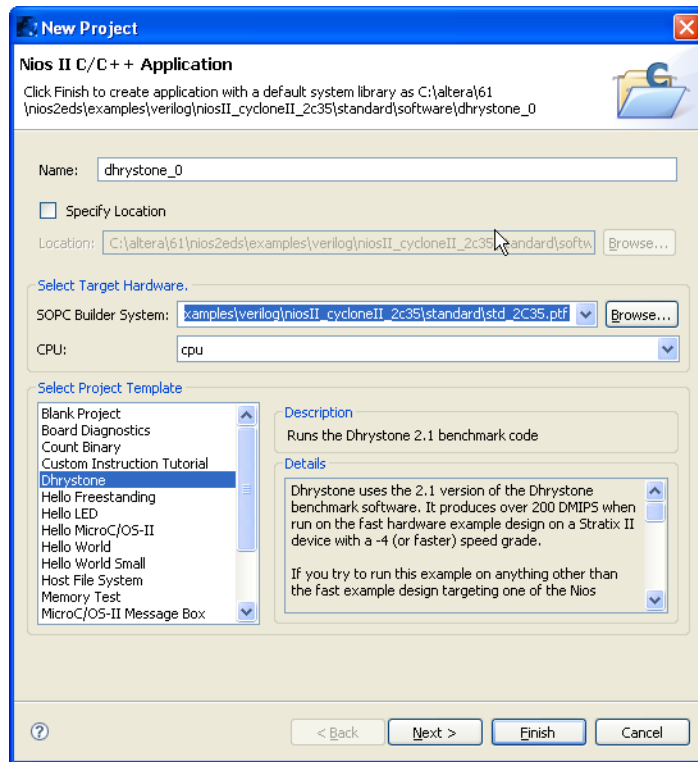


The Nios II C/C++ application **New Project** wizard prompts you to specify:

1. A name for your new Nios II project.
2. The target hardware.
3. A template for the new project.

Project templates are ready-made, working designs that serve as examples to show you how to structure your own Nios II projects. It is often easier to start with a working “Hello World” project, than to start a blank project from scratch.

[Figure 2–3](#) shows the Nios II C/C++ application **New Project** wizard, with the template for a Dhrystone benchmark design selected.

Figure 2–3. The Nios II C/C++ Application New Project Wizard

After you click **Finish**, the Nios II IDE creates the new project. The IDE also creates a system library project, `*_syslib` (for example, `dhystone_0_syslib` for Figure 2–3 on page 2–5). These projects show up in the Nios II C/C++ Projects view of the workbench.

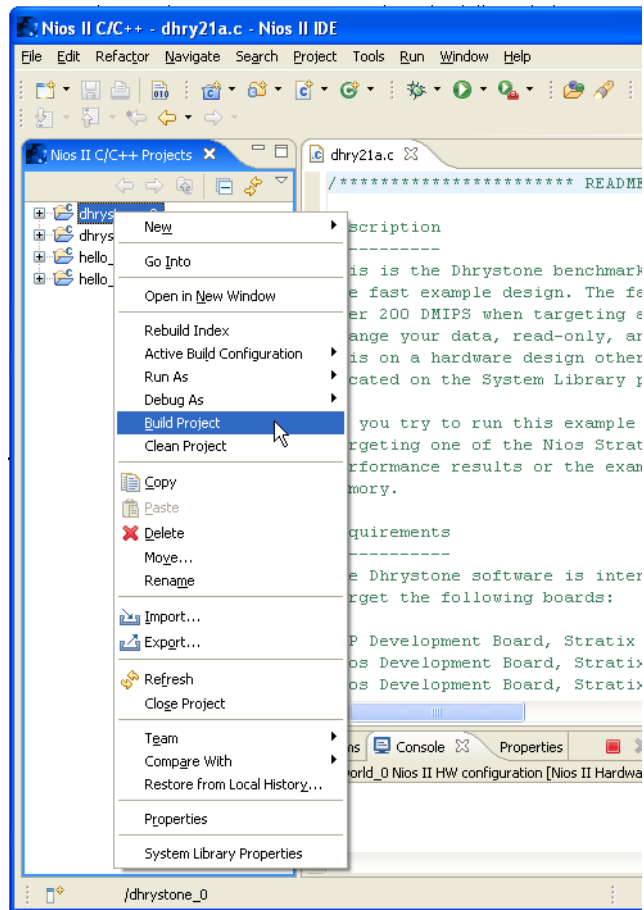
Building and Managing Projects

Right-clicking on any resource (a file, folder, or project) opens a context-sensitive menu containing commands that you can perform on the resource. Right-clicking is usually the quickest way to find the command you need, though commands are also available in menus and toolbars.

To compile a Nios II project, right-click the project in the Nios II C/C++ Projects view, and click **Build Project**. Figure 2–4 on page 2–6 shows the context-sensitive menu for the project `dhystone_0`, with the **Build Project** option chosen. When building, the Nios II IDE first builds the

system library project (and any other project dependencies), and then compiles the main project. Any warnings or errors are displayed in the Tasks view.

Figure 2–4. Building a Project Using the Context-Sensitive (Right-Click) Menu



Right-clicking a project in the Nios II C/C++ Projects view also allows you to access the following important options for managing the project:

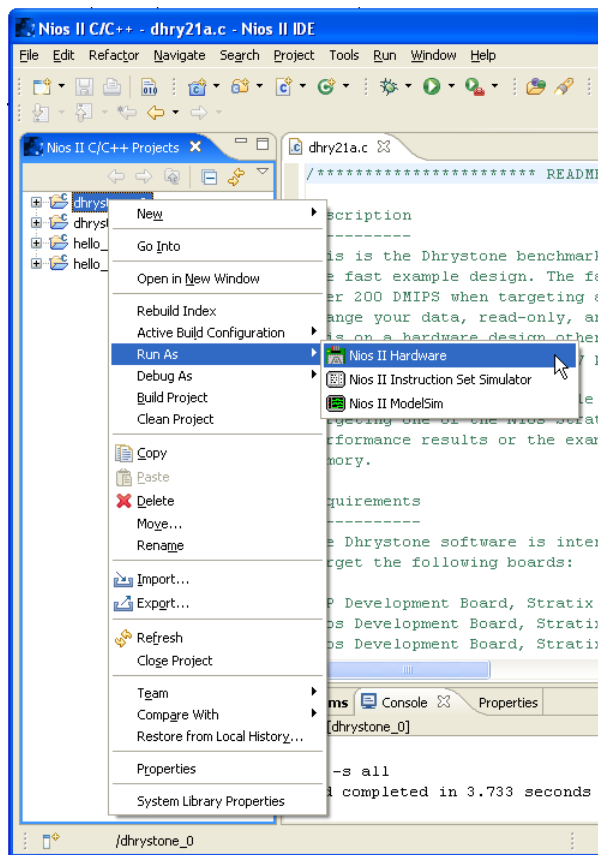
- **Properties**—Manage the dependencies on target hardware and other projects
- **System Library Properties**—Manage hardware-specific settings, such as communication devices and memory partitioning
- **Build Project**—i.e., make

- **Run As**—Run the program on hardware or under simulation
- **Debug As**—Debug the program on hardware or under simulation

Running and Debugging Programs

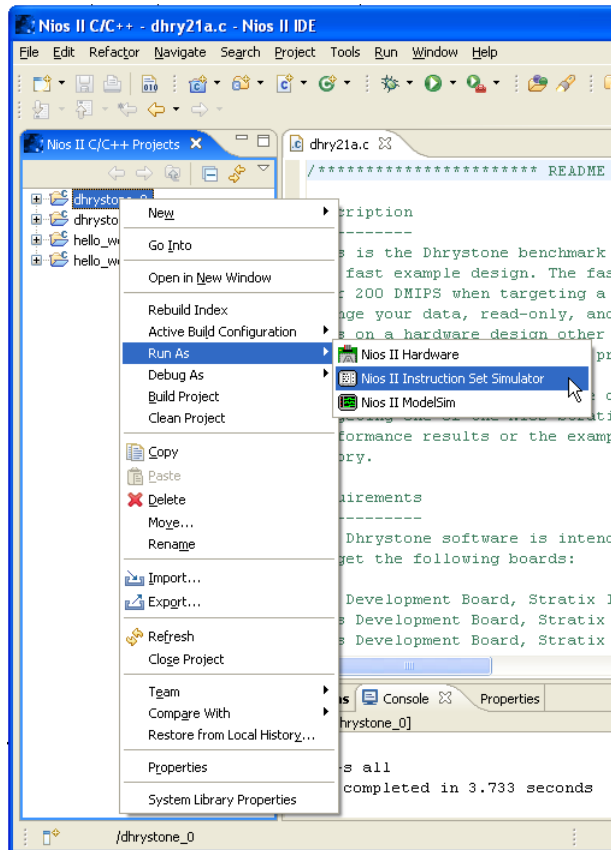
Run and debug operations are available by right-clicking the Nios II project. The Nios II IDE allows you to run or debug the project either on a target board, under the Nios II instruction set simulator (ISS), or under ModelSim®. For example, to run the program on a target board, right-click the project in the Nios II C/C++ Projects view, point to Run As, and then click **Nios II Hardware**. See Figure 2–5. Character I/O to stdout and stderr are displayed in the Console view.

Figure 2–5. Running a Program on Target Hardware



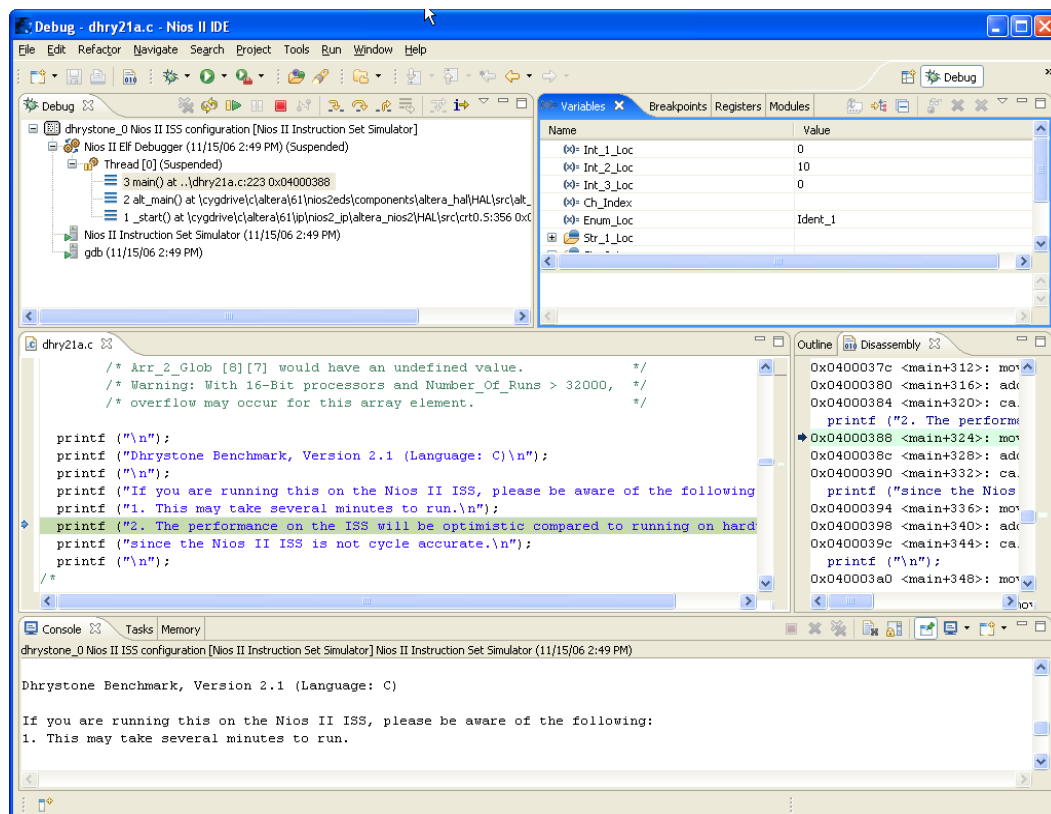
Starting a debug session is similar to starting a run session. For example, to debug the program on the ISS, right-click the project in the Nios II C/C++ Projects view, point to **Debug As**, and then click **Nios II Instruction Set Simulator**. See [Figure 2-6 on page 2-8](#).

Figure 2-6. Launching the Instruction Set Simulator



[Figure 2-7 on page 2-9](#) shows a debug session in progress for the `dhrystone_0` project.

Figure 2-7. Debugging dhrystone_0 on the ISS

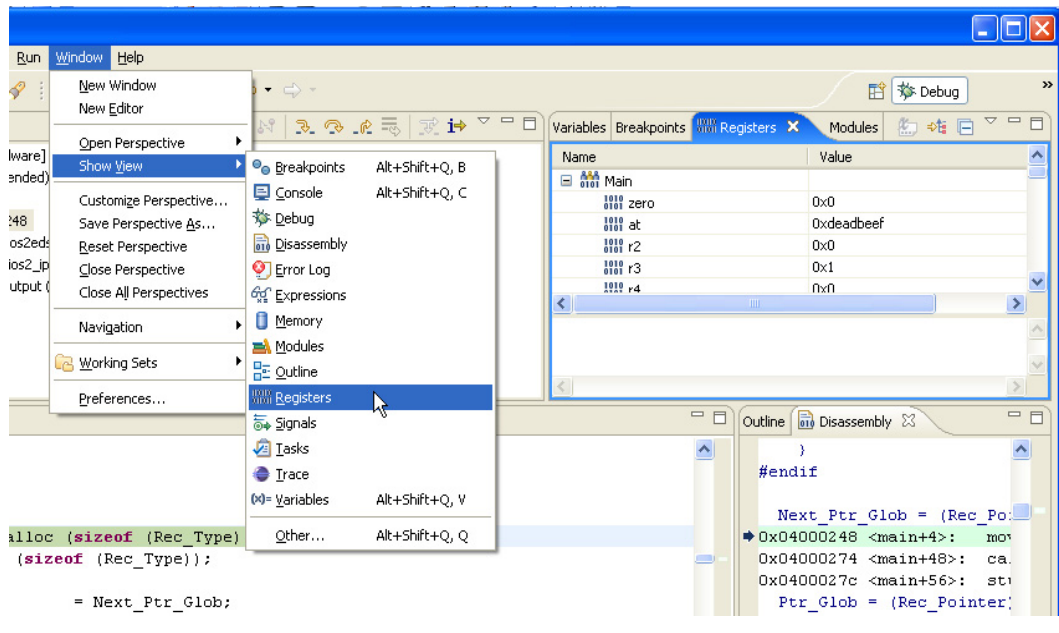


Launching the debugger changes the workbench perspective to the debug perspective. You can easily switch between the debug perspective and the Nios II C/C++ development perspective by clicking on the **Open Perspective** icon at the upper right corner of the workbench window.

After you start a debug session, the debugger loads the program, sets a breakpoint at `main()`, and begins executing the program. You use the usual controls to step through the code: Step Into, Step Over, Resume, Terminate, etc. To set a breakpoint, double click in the left-hand margin of the code view, or right-click in the margin and then click **Add Breakpoint**.

The Nios II IDE offers many views that allow you to examine the status of the processor while debugging: Variables, Expressions, Registers, Memory, etc. [Figure 2-8 on page 2-10](#) shows the Registers view.

Figure 2–8. The Registers View While Debugging



Importing User-Managed Projects

In the Nios II software build tools design flow, you import user-managed projects, created with the Nios II software build tools, into the IDE for debugging. This section discusses that process.

When you create a C/C++ application (and its associated BSP) with the Nios II software build tools, the application is ready to import into the IDE as a user-managed project. No additional preparation is necessary.

The IDE imports four kinds of Nios II software build tools projects:

- User-managed C/C++ application project
- User-managed board support package (BSP) project
- User-managed library project
- C/C++ source project (a directory tree containing supporting source code)

The IDE treats each type of imported project as listed in [Table 2–1](#).

Table 2–1. IDE Capabilities for Imported Projects				
Type of project	Source editable?	Buildable?	Debuggable?	Settings Manageable?
User-managed C/C++ application project	Yes	Yes	Yes	No
User-managed BSP project	Yes	No <i>(1)</i>	<i>(2)</i>	No
User-managed library project	Yes	No <i>(1)</i>	<i>(2)</i>	No
C/C++ source project	Yes	No	<i>(2)</i>	No

Notes to Table 2–1:

- (1) When the IDE builds a C/C++ application project, it also builds the associated BSP, and any associated libraries. It is not necessary to import BSPs and libraries to build them as part of a C/C++ application in the IDE.
- (2) When the IDE debugs a C/C++ application, it can step into any associated BSP, library, or supporting C/C++ source code that you have imported.

The IDE imports each type of project through the **Import** wizard. The **Import** wizard determines the kind of project you are importing, and configures it appropriately.

You can continue to develop project code in your user-managed project after importing the project into the IDE. You can edit source files and rebuild the project, using either the IDE tool chain or the software build tools. However, you must manage BSP settings with the software build tools.



For further information about creating projects with the software build tools, refer to the *Nios II Software Build Tools* chapter of the *Nios II Software Developer's Handbook*.

Road Map

Importing and debugging a project typically involves several of the following tasks. You do not need to perform these tasks in this order, and you can repeat or omit some tasks, depending on your needs.

- Import a user-managed C/C++ application
- Import a supporting project
- Debug a user-managed C/C++ application
- Edit user-managed C/C++ application code

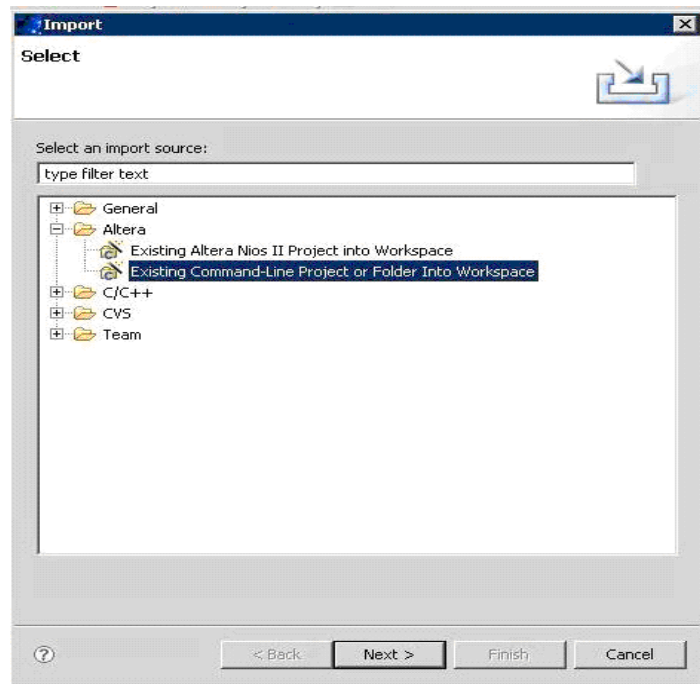
The following sections describe these tasks.

Import a User-Managed C/C++ Application

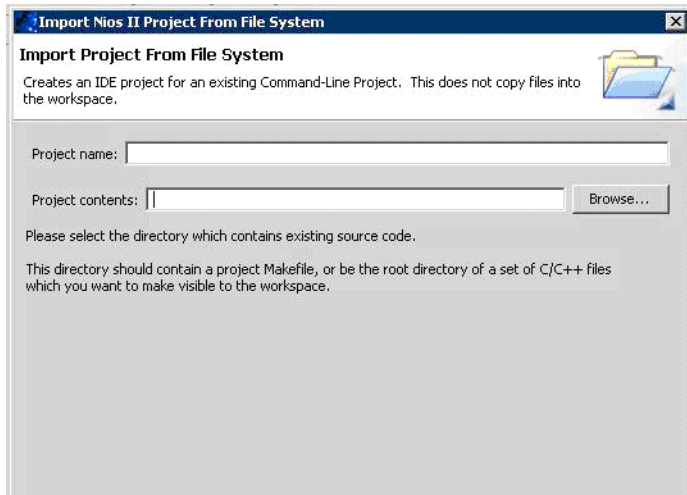
To import a user-managed C/C++ application, perform the following steps:

1. Launch the IDE.
2. On the File menu, click **Import**. The **Import** dialog box appears.

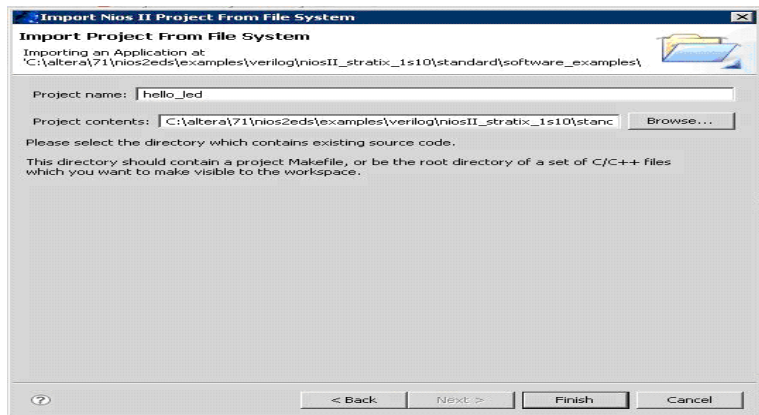
Figure 2–9. Launching the Import Wizard



3. Expand the **Altera Nios II** folder, and select **Existing Nios II software build tools project or folder into workspace**, as shown in [Figure 2–9](#).
4. Click **Next**. The **Import** wizard appears, as shown in [Figure 2–10 on page 2–13](#).

Figure 2–10. User-Managed Project Import Wizard

5. Click **Browse** and locate the directory containing the C/C++ application project to import.
6. Click **OK**. The wizard fills in the project name and path, as shown in [Figure 2–11](#). The project name is the directory name. You can override the project name by typing a new name in the **Project name** box.

Figure 2–11. Importing a User-Managed C/C++ Application



You might see a warning saying "There is already a **.project** file at: *<path>*". This warning indicates that the directory already contains an IDE project. Either it is an IDE-managed project, or it is a user-managed project that is already imported into the IDE.

If the project is not already in your workspace, you can import it, but be aware that the **Import** wizard does not convert it from one type to another.

If the project is already in your workspace, do not re-import it.

7. Click **Finish**. The wizard imports the project, creating a new C/C++ application project in the workspace.

At this point, the IDE can build, debug, and run the complete program, including the BSP and any libraries, by using the user-managed makefiles in your imported C/C++ application project. The IDE can display and step through application source code, exactly as if the project were IDE-managed. However, the IDE does not have direct information about where BSP or library code resides. If you need to view, debug or step through BSP or library source code, you need to import the BSP or library. The process of importing supporting projects, such as BSPs and libraries, is described in the next section.

Import a Supporting Project

While debugging a C/C++ application, you might need to view, debug or step through source code in a supporting project, such as a BSP or library. To make supporting project source code visible in the IDE debug perspective, you need to import the supporting project.

If you do not need BSP or library source code visible in the debugger, you can skip this task, and go directly to [“Debug a User-Managed C/C++ Application” on page 2-17](#).



To debug or step through the code in the GNU newlib library, import the **nios2-gnutools** folder as a C/C++ source project, as described in [“Importing a C/C++ Source Project” on page 2-16](#).

If you have several C/C++ applications based on one BSP or library, import the BSP or library once, and then import each application that is based on the BSP or library. Each application’s makefile contains the information needed to find and build any associated BSP or libraries.

Importing a User-Managed BSP

To import a user-managed BSP project, perform the following steps:

1. On the File menu, click **Import**. The **Import** dialog box appears.
2. Expand the **Altera Nios II** folder, and select **Existing Nios II software build tools project or folder into workspace**.
3. Click **Next**. The **Import** wizard appears.
4. Click **Browse** and locate the directory containing the BSP project to import.
5. Click **OK**. The wizard fills in the project name and path. The project name is the directory name. You can override the project name by typing a new name in the **Project name** box.



You might see a warning saying "There is already a **.project** file at: *<path>*". This warning indicates that the directory already contains an IDE project. Either it is an IDE-managed project, or it is a user-managed project that is already imported into the IDE.

If the project is not already in your workspace, you can import it, but be aware that the **Import** wizard does not convert it from one type to another.

If the project is already in your workspace, do not re-import it.

6. Click **Finish**. The wizard imports the project, creating a new BSP project in the workspace.



After import, a user-managed BSP looks the same as a user-managed C/C++ application. However, you cannot directly build or run a user-managed BSP in the IDE.

Importing a User-Managed Library

To import a user-managed library, perform the following steps:

1. On the File menu, click **Import**. The **Import** dialog box appears.
2. Expand the **Altera Nios II** folder, and select **Existing Nios II software build tools project or folder into workspace**.
3. Click **Next**. The **Import** wizard appears.

4. Click **Browse** and locate the directory containing the library project to import.
5. Click **OK**. The wizard fills in the project name and path. The project name is the directory name. You can override the project name by typing a new name in the **Project name** box.



You might see a warning saying "There is already a **.project** file at: *<path>*". This warning indicates that the directory already contains an IDE project. Either it is an IDE-managed project, or it is a user-managed project that is already imported into the IDE.

If the project is not already in your workspace, you can import it, but be aware that the **Import** wizard does not convert it from one type to another.

If the project is already in your workspace, do not re-import it.

6. Click **Finish**. The wizard imports the project, creating a new library project in the workspace.



After import, a user-managed library looks the same as a user-managed C/C++ application. However, you cannot directly build or run a user-managed library in the IDE.

Importing a C/C++ Source Project

To import a C/C++ Source Project, such as newlib, perform the following steps:

1. On the File menu, click **Import**. The **Import** dialog box appears.
2. Expand the **Altera Nios II** folder, and select **Existing Nios II software build tools project or folder into workspace**.
3. Click **Next**. The **Import** wizard appears.
4. Click **Browse** and locate the directory containing the C/C++ source project to import.
5. Click **OK**. The wizard fills in the project name and path. The project name is the directory name. You can override the project name by typing a new name in the **Project name** box.



You might see a warning saying "There is already a **.project** file at: *<path>*". This warning indicates that the directory already contains an IDE project. Either it is an IDE-managed project, or it is a user-managed project that is already imported into the IDE.

If the project is not already in your workspace, you can import it, but be aware that the **Import** wizard does not convert it from one type to another.

If the project is already in your workspace, do not re-import it.

6. Click **Finish**. The wizard imports the project, creating a new C/C++ source project in the workspace.



After import, user-managed C/C++ source code looks the same as a user-managed C/C++ application. However, you cannot directly build or run user-managed C/C++ source code in the IDE.

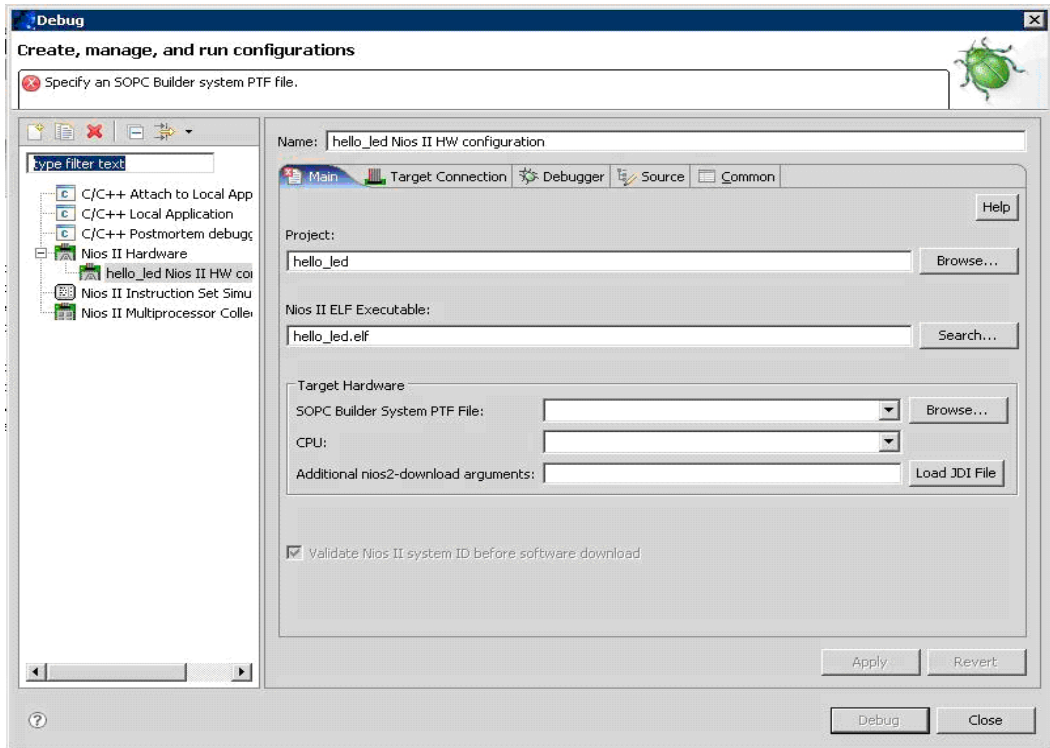
Debug a User-Managed C/C++ Application

To debug an imported user-managed C/C++ application project, perform the following steps:

1. In the Nios II C/C++ Projects view, right click the project name, point to **Debug As**, and click **Nios II Hardware**.

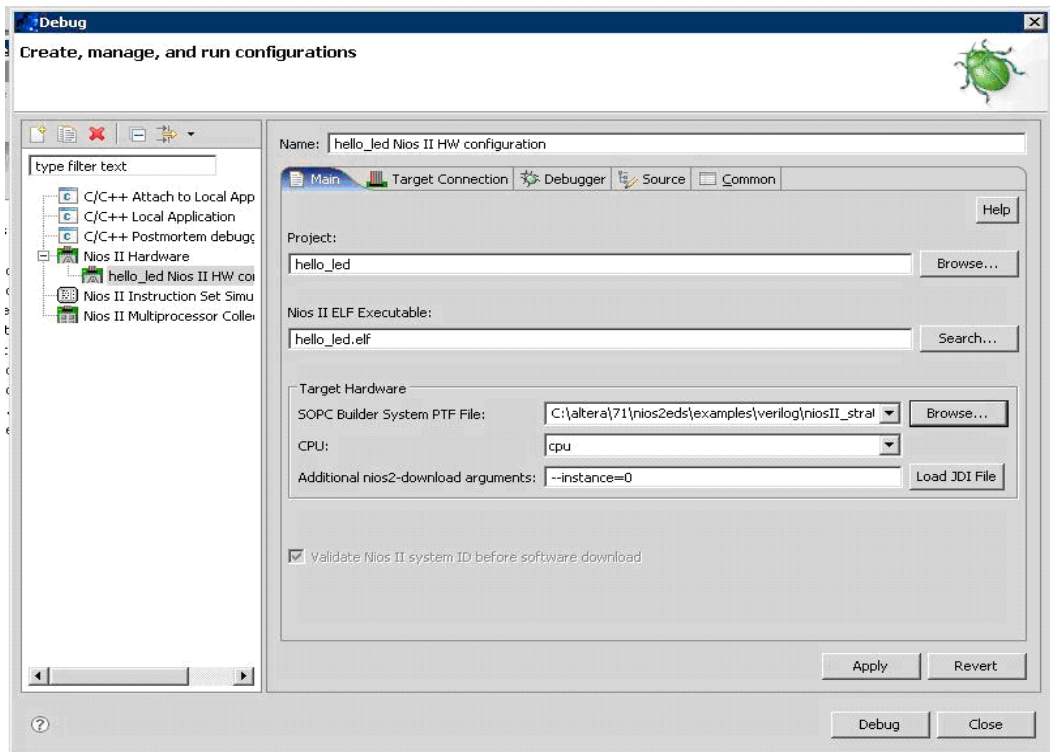
The debug configuration shows the message: "Specify an SOPC Builder system PTF file", as in [Figure 2-12 on page 2-18](#). The debugger needs information about the target system in order to establish communications.

Figure 2–12. Debug Configuration Manager Initial View



2. Click **Browse** at the right of the **SOPC Builder System PTF File** box.
3. Locate the SOPC Builder System File (**.ptf**) on which the application's BSP is based. For example, if you are using a Nios II software example, the SOPC Builder System File is three levels up in the directory tree from the software project.

After you select the file, the message disappears, as shown in [Figure 2–13 on page 2–19](#).

Figure 2–13. Debug Configuration Manager Final View

4. Click **Apply**.

Your software application is ready to run or debug exactly as you would run or debug an IDE-managed project. For details about running and debugging applications in the Nios II IDE, see [“Running and Debugging Programs”](#) on page 2–7.

Edit User-Managed C/C++ Application Code

You can edit the code in an imported user-managed project with the editor exactly the same way you edit the code in an IDE-managed project.

Programming Flash

Many Nios II processor systems use external flash memory to store one or more of the following items:

- Program code

- Program data
- FPGA configuration data
- File systems

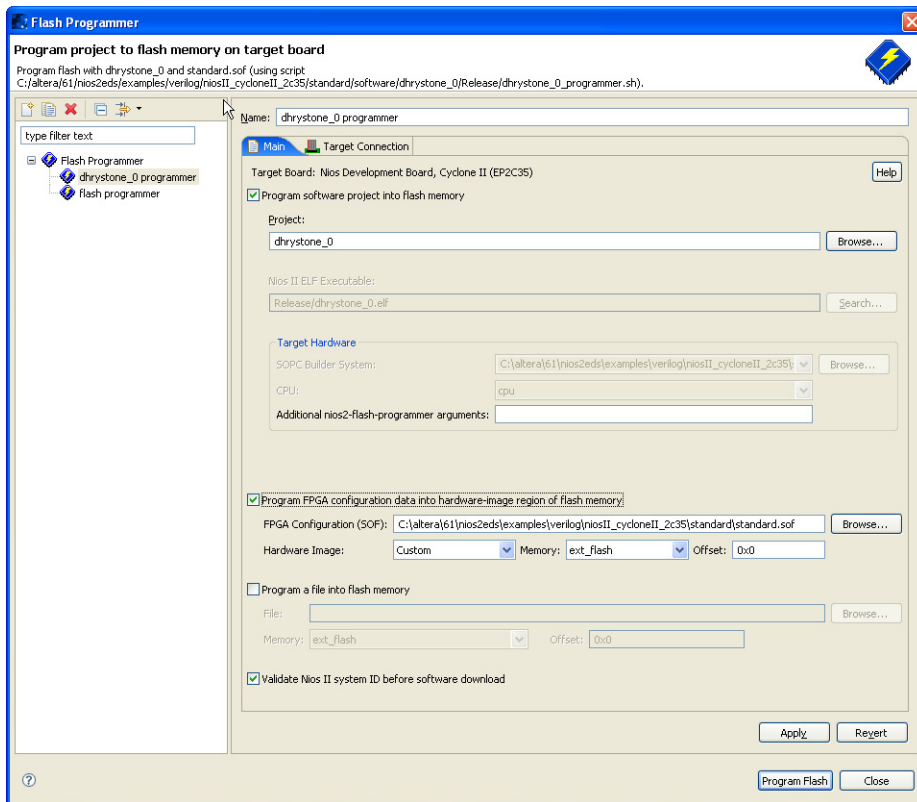
The Nios II IDE provides a Flash Programmer utility to help you manage and program the contents of flash memory. [Figure 2–14 on page 2–21](#) shows the Flash Programmer.



To program a user-managed C/C++ application to flash memory, you must first specify an SOPC Builder System File, as follows:

1. Click **Browse** at the right of the **SOPC Builder System PTF File** box.
2. Locate the SOPC Builder System File on which the application's BSP is based. For example, if you are using a Nios II software example, the SOPC Builder System File is three levels up in the directory tree from the software project.

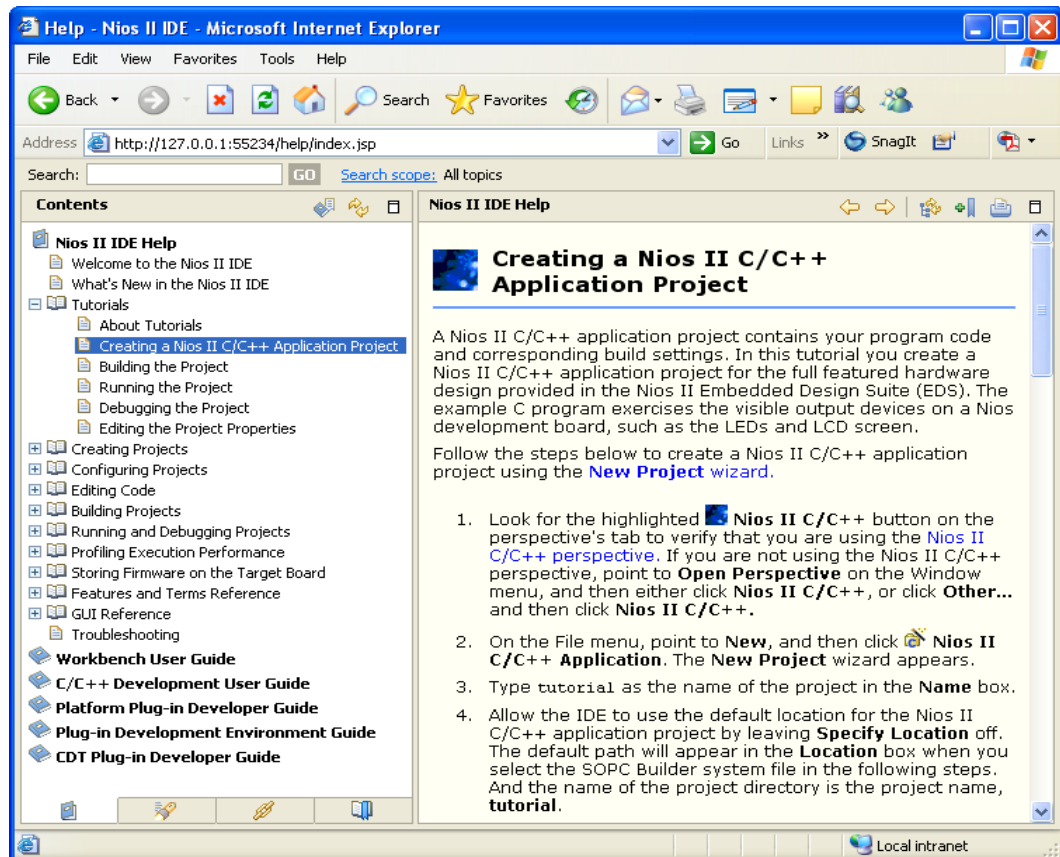
This procedure is identical to specifying the SOPC Builder System File before debugging a user-managed C/C++ application, as described in [“Debug a User-Managed C/C++ Application” on page 2–17](#).

Figure 2–14. The Nios II IDE Flash Programmer

Help System

The Nios II IDE help system provides documentation on all IDE topics. To launch the help system, click **Help Contents** on the Help menu. You can also press F1 on Windows (Shift-F1 on Linux) at any time for context-sensitive help. The Nios II IDE help system contains hands-on tutorials that guide you step-by-step through the process of creating, building, and debugging Nios II projects. [Figure 2–15 on page 2–22](#) shows the Nios II IDE help system displaying a tutorial.

Figure 2–15. Tutorials in the Nios II IDE Help System



Referenced Documents

This chapter references the following document:

Nios II Software Build Tools chapter of the *Nios II Software Developer's Handbook*

Document Revision History

Table 2–2 shows the revision history for this document.

<i>Table 2–2. Document Revision History</i>		
Date & Document Version	Changes Made	Summary of Changes
May 2007 v7.1.0	<ul style="list-style-type: none"> Added instructions for importing user-managed projects Changed chapter title. Added table of contents to Introduction section. Added Referenced Documents section. 	Nios II software build tools
March 2007 v7.0.0	No change from previous release.	
November 2006 v6.1.0	Describes updated look and feel, including Nios II C/C++ perspective and Nios II C/C++ Projects views, renamed project types.	Updated look and feel based on Eclipse 3.2.
May 2006 v6.0.0	No change from previous release.	
October 2005 v5.1.0	Updated for the Nios II IDE version 5.1.	
May 2005 v5.0.0	No change from previous release.	
September 2004 v1.1	Updated screen shots.	
May 2004 v1.0	Initial Release.	

