

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 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.

 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 Development Flows and the IDE” on page 2-2
- “Creating a New IDE-Managed Project” on page 2-3
- “Building and Managing Projects” on page 2-3
- “Running and Debugging Programs” on page 2-4
- “Importing Software Build Tools Projects” on page 2-4
- “Programming Flash” on page 2-9
- “Archiving Nios II IDE Software Projects” on page 2-9
- “Help System” on page 2-11

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.

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.

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.

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 Development Flows and the IDE

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

IDE Projects and Makefiles

In the Nios II IDE development flow, the IDE manages Nios II C/C++ application and board support package (BSP) projects and makefiles that you create with the **New Project** wizard in Nios II IDE. The best way to modify and build an IDE project is through the IDE. You manage the BSP project settings with the **System Library** page of the **Properties** dialog box.



In the Nios II IDE, the term “system library” is used for a BSP project.

Software Build Tools Projects and Makefiles

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

The best way to create a software build tools 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 software build tools 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.



Software build tools projects and IDE projects are not interchangeable. However, you can manually convert an IDE project to a software build tools project.



For details, see [Appendix A, Porting Nios II IDE Projects to the Software Build Tools](#), in the *Nios II Software Developer's Handbook*.

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 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**.

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 software projects 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.

When the Nios II IDE creates the new application project, it also creates a BSP project. If the name of the application project is `<name>`, the default name of the BSP project is `<name>*_syslib` (for example, `dhystone_0_syslib`). These projects appear in the Nios II C/C++ Projects view of the workbench.



The first time you create or build a Nios II project, the Nios II IDE automatically creates a project in your workspace called **altera.components**. This project contains links to the source code files for all Altera®-provided device drivers and software packages, enabling you to step through system code in the debugger, set breakpoints, and use other debugger features. The **altera.components** project appears in the Nios II C/C++ Projects view. The Nios II C/C++ view protects the source files in **altera.components** from accidental deletion, because they are shared among all software projects. Do not attempt to circumvent this protection.

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**. When building, the Nios II IDE first builds the BSP project (and any other project dependencies), and then compiles the main project. Any warnings or errors are displayed in the Tasks view.

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

Debug and Release Configurations

You can select a Debug or Release configuration in the **Project Properties** dialog box, under **C/C++ Build**. The project configuration controls the optimization level and debug compiler options.

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 the ModelSim® logic simulator. 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**. Character I/O to `stdout` and `stderr` are displayed in the Console view.

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**.

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, such as the Variables, Expressions, Registers, and Memory views.

Importing Software Build Tools Projects

In the Nios II software build tools development flow, you import software build tools 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 software build tools project. No additional preparation is necessary.

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

- Software build tools C/C++ application project
- Software build tools board support package (BSP) project
- Software build tools 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?
Software build tools C/C++ application project	Yes	Yes	Yes	No
Software build tools BSP project	Yes	No (1)	(2)	No
Software build tools library project	Yes	No (1)	(2)	No
C/C++ source project	Yes	No	(2)	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 software build tools 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 *Introduction 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 software build tools C/C++ application
- Import a supporting project
- Debug a software build tools C/C++ application
- Edit software build tools C/C++ application code


The following sections describe these tasks.

Import a Software Build Tools C/C++ Application

To import a software build tools C/C++ application, perform the following steps:

1. Launch the IDE.
2. On the File menu, click **Import**. The **Import** dialog box appears.
3. Expand the **Altera Nios II** folder, and select **Existing Nios II software build tools project or folder into workspace**.
4. Click **Next**. The **Import** wizard appears.
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. 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 project, or it is a software build tools 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 software build tools makefiles in your imported C/C++ application project. The IDE can display and step through application source code, exactly as if the project were created in the IDE. 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 Software Build Tools C/C++ Application" on page 2-8](#).

 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 2-8](#).


 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 Software Build Tools BSP

To import a software build tools 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 project, or it is a software build tools 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 software build tools BSP looks the same as a software build tools C/C++ application. However, you cannot directly build or run a BSP in the IDE.

Importing a Software Build Tools Library

To import a software build tools 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 project, or it is a software build tools 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 software build tools library looks the same as a software build tools C/C++ application. However, you cannot directly build or run a 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 project, or it is a software build tools 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, software build tools C/C++ source code looks the same as a software build tools C/C++ application. However, you cannot directly build or run a C/C++ source code project in the IDE.

Debug a Software Build Tools C/C++ Application

To debug an imported software build tools 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". The debugger needs information about the target system in order to establish communications.

2. Click **Browse** at the right of the **SOPC Builder System PTF File** box.
3. Locate the SOPC Builder system (**.ptf**) file on which the application's BSP is based. For example, if you are using a Nios II software build tools 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. Click **Apply**.

Your software application is ready to run or debug exactly as you would run or debug an IDE project. For details about running and debugging applications in the Nios II IDE, see [“Running and Debugging Programs” on page 2-4](#).

Edit Software Build Tools C/C++ Application Code

You can edit the code in an imported software build tools project with the editor exactly the same way you edit the code in an IDE 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.



To program a software build tools 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 build tools 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 software build tools C/C++ application, as described in [“Debug a Software Build Tools C/C++ Application” on page 2-8](#).

Archiving Nios II IDE Software Projects

This section helps you identify the files you must include when archiving a Nios II IDE software project. With this information, you can archive a Nios II application project and its associated Nios II BSP project.

You may want to archive your projects for one of the following reasons:

- To place them under source control
- To create backups
- To bundle the projects for transfer to another location

This section covers the following information:

- How to find and identify the files that you must include in an archived Nios II IDE software project.
- Which files must have write permission to allow the software projects to be built.

Required Files

This section describes the files required by Nios II IDE software projects. This is the minimum set of files needed to completely rebuild a software project, including the executable and linking format (.elf) file.

Archive your Nios II IDE software projects together with the SOPC Builder system on which they are based. You cannot rebuild a Nios II IDE software project without its associated SOPC Builder system.

Nios II Application Project Files

The files listed in [Table 2-2](#) are located in the Nios II application project directory.

Table 2-2. Files Required for a Nios II Application Project

File Description	File Name	Write Permission Required? (1)
All source files	for example: <code>app.c</code> , <code>header.h</code> , <code>assembly.s</code> , <code>lookuptable.dat</code>	No
Eclipse™ project file	<code>.project</code>	No
C/C++ Development Toolkit project file	<code>.cdtproject</code>	Yes
C/C++ Development Toolkit option file	<code>.cdtbuild</code>	No
Software configuration file	<code>application.stf</code>	No

Note to Table 2-2:

(1) For further information about write permissions, refer to “[File Write Permissions](#)”.

Nios II BSP Project

The files listed in [Table 2-3](#) are located in the Nios II BSP (system library) project directory.

Table 2-3. Files Required for a Nios II BSP Project

File description	File name	Write permission required? (1)
Eclipse project file	<code>.project</code>	Yes
C/C++ Development Toolkit project file	<code>.cdtproject</code>	Yes
C/C++ Development Toolkit option file	<code>.cdtbuild</code>	No
System software configuration file	<code>system.stf</code>	Yes

Note to Table 2-3:

(1) For further information about write permissions, see “[File Write Permissions](#)”.

File Write Permissions

You must have write permission for certain files, shown in [Table 2-2](#) and [Table 2-3](#). The tools write to these files as part of the build process. If the files are not writable, the tool chain fails.

Many source control tools mark local files read-only by default. In this case, you must override this behavior. You do not have to check the files out of source control unless you are modifying the Nios II software project.

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.

Referenced Documents

This chapter references the following documents:

- *Introduction to the Nios II Software Build Tools* chapter of the *Nios II Software Developer's Handbook*
- *Appendix A, Porting Nios II IDE Projects to the Software Build Tools*, in the *Nios II Software Developer's Handbook*

Document Revision History

Table 2-4 shows the revision history for this document.

Table 2-4. Document Revision History (Part 1 of 2)

Date & Document Version	Changes Made	Summary of Changes
March 2009 v9.0.0	<ul style="list-style-type: none"> ■ Reorganized and updated information and terminology to clarify role of Nios II software build tools. ■ Corrected minor typographical errors. 	
May 2008 v8.0.0	No change from previous release.	
October 2007 v7.2.0	altera.components project added	altera.components project added
May 2007 v7.1.0	<ul style="list-style-type: none"> ■ Added instructions for importing software build tools 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.	

Table 2-4. Document Revision History (Part 2 of 2)

Date & Document Version	Changes Made	Summary of Changes
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.	