

Introduction

The *Nios II Software Developer's Handbook* provides the basic information needed to develop software for the Altera® Nios® II processor. This document is written for the user of the Nios II integrated development environment (IDE), as well as the user of the Nios II command-line software build tools.

This chapter provides a high-level overview of the software development environment for the Nios II processor. This chapter introduces the Nios II software development environment, the Nios II embedded design suite (EDS) tools available to you, and the process for developing software. This chapter contains the following sections:

- “Getting Started” on page 1-1
- “Nios II Software Development Environment” on page 1-2
- “Nios II Programs” on page 1-2
- “Altera-Provided Development Tools” on page 1-4
- “Development Flows for Creating Nios II Programs” on page 1-9
- “Third-Party Support” on page 1-13
- “Migrating from the First-Generation Nios Processor” on page 1-13
- “Further Nios II Information” on page 1-13

Getting Started

Writing software for the Nios II processor is similar to the software development process for any other microcontroller family. The easiest way to start designing effectively is to purchase a development kit from Altera that includes documentation, a ready-made evaluation board, and all the development tools necessary to write Nios II programs.

Modifying existing code is one of the most common and comfortable ways that you can learn to write programs in a new environment. The Nios II EDS provides many example software designs that you can examine, modify, and use in your own programs. The provided examples range from a simple “Hello world” program, to a working RTOS example, to a full TCP/IP stack running a web server. Each example is documented and ready to compile.

Prerequisites

The *Nios II Software Developer's Handbook* assumes you have a basic familiarity with embedded processor concepts. You do not need to be familiar with any specific Altera technology or with Altera development tools. Familiarity with Altera hardware development tools can give you a deeper understanding of the reasoning behind the Nios II software development environment. However, software developers can develop and debug applications without further knowledge of Altera technology.

Finding Nios II EDS Files

When you install the Nios II EDS, you specify a root directory for the directory structure containing the EDS files. For example, if the Nios II EDS 8.1 is installed on the Windows operating system, the root directory might be `c:\altera\81\nios2eds`.

The root directory can be any accessible location in your file system. For simplicity, this handbook refers to this directory as *<Nios II EDS install path>*.

 The Nios II EDS defines the environment variable `SOPC_KIT_NIOS2` to represent *<Nios II EDS install path>*.

Nios II Software Development Environment

The Nios II EDS provides a consistent software development environment that works for all Nios II processor systems. With a PC, an Altera FPGA, and a JTAG download cable (such as an Altera USB-Blaster™ download cable), you can write programs for, and communicate with, any Nios II processor system. The Nios II processor's JTAG debug module provides a single, consistent method to communicate with the processor using a JTAG download cable. Accessing the processor is the same, regardless of whether a device implements only a Nios II processor system, or whether the Nios II processor is embedded deeply in a complex multiprocessor system. Therefore, you do not need to spend time manually creating interface mechanisms for the embedded processor.

The Nios II EDS provides two distinct development flows and includes many proprietary and open-source tools (such as the GNU C/C++ tool chain) for creating Nios II programs. The Nios II EDS automates board support package (BSP) creation for Nios II processor-based systems, eliminating the need to spend time manually creating BSPs. Altera BSPs contain the Altera hardware abstraction layer (HAL), an optional RTOS, and device drivers. The BSP provides a C/C++ runtime environment, insulating you from the hardware in your embedded system.

Nios II Programs

Each Nios II program you develop in either Nios II EDS development flow consists of an application project, optional library projects, and a BSP project. You build your Nios II program to create an executable and linking format (**.elf**) file which runs on a Nios II processor.

The following sections describe the project types that constitute a Nios II program.

Application Project

A Nios II C/C++ application project consists of a collection of source code compiled to create one **.elf** file. A typical characteristic of an application is that one of the source files contains function `main()`. An application includes code that calls functions in libraries and BSPs.

Library Project

A library project is a collection of source code compiled to create a single library archive (.a) file. Libraries often contain reusable, general purpose functions that multiple application projects can share. A collection of common arithmetical functions is one example. A library does not contain a function `main()`.

BSP Project

A Nios II BSP project is a specialized library containing system-specific support code. A BSP provides a software runtime environment customized for one processor in an SOPC Builder system. The Nios II EDS provides tools to modify settings that control the behavior of the BSP.

A BSP contains the following elements:

- [Hardware Abstraction Layer](#)
- [Newlib C Standard Library](#)
- [Device Drivers](#)
- [Optional Software Packages](#)
- [Optional Real-Time Operating System](#)

Hardware Abstraction Layer

The HAL provides a non-threaded, UNIX-like, C/C++ runtime environment. The HAL provides generic I/O devices, allowing you to write programs that access hardware using the newlib C standard library routines, such as `printf()`. The HAL interfaces to HAL device drivers, which access peripheral registers directly, abstracting hardware details from the software application. This abstraction minimizes or eliminates the need to access hardware registers directly to control and communicate with peripherals.



For complete details about the HAL, refer to the [Hardware Abstraction Layer](#) section and the [HAL API Reference](#) chapter of the *Nios II Software Developer's Handbook*.

Newlib C Standard Library

Newlib is an open source implementation of the C standard library intended for use on embedded systems. It is a collection of common routines such as `printf()`, `malloc()`, and `open()`.

Device Drivers

Each device driver manages a hardware component. By default, the HAL instantiates a device driver for each component in your SOPC Builder system that needs a device driver. In the Nios II software development environment, a device driver has the following properties:

- A device driver is associated with a specific SOPC Builder component.
- A device driver might have settings that impact its compilation. These settings become part of the BSP settings.

Optional Software Packages

A software package is source code that you can optionally add to a BSP project to provide additional functionality. The NicheStack® TCP/IP - Nios II Edition is an example of a software package.

In the Nios II software development environment, a software package typically has the following properties:

- A software package is not associated with specific hardware.
- A software package might have settings that impact its compilation. These settings become part of the BSP settings.



In the Nios II software development environment, a software package is distinct from a library project. A software package is part of the BSP project, not a separate library project.

Optional Real-Time Operating System

The Nios II EDS includes an implementation of the third-party MicroC/OS-II RTOS that you can optionally include in your BSP. MicroC/OS-II is built on the HAL, and implements a simple, well-documented RTOS scheduler. You can modify settings that become part of the BSP settings. Other operating systems are available from third-party vendors.

Altera-Provided Development Tools

This section lists all the development tools that Altera provides for the Nios II processor. This section does not describe detailed usage of the tools, but refers you to the most appropriate documentation.

Refer to [“Development Flows for Creating Nios II Programs” on page 1-9](#) an overview of how you use these tools together to develop software for the Nios II processor.

The Nios II Command Shell

The Nios II command shell is a command-line environment with the correct settings to run Nios II command-line tools. To open a Nios II command shell, execute the following steps, depending on your environment:

- In the Windows operating system, on the Start menu, point to **Programs > Altera > Nios II EDS**, and click **Nios II Command Shell**.
- In the Linux operating system, in a command shell, execute the following commands:

```
cd $SOPC_KIT_NIOS2↵  
./sdk_shell↵
```

The Nios II IDE Tools

[Table 1-1](#) describes the tools provided by the Nios II IDE user interface.

Table 1-1. The Nios II IDE and Associated Tools

| Tools | Description |
|------------------------|--|
| The Nios II IDE | The Nios II IDE is a software development user interface for the Nios II processor. All software development tasks can be accomplished in the IDE, including editing, building, and debugging programs. For more information, refer to the Nios II IDE help system. |
| Flash programmer | The Nios II IDE includes a flash programmer utility that allows you to program flash memory chips on a target board. The flash programmer supports programming flash on any board, including Altera development boards and your own custom boards. The flash programmer facilitates programming flash for the following purposes: <ul style="list-style-type: none">■ Executable code and data■ Bootstrap code to copy code from flash to RAM, and then run from RAM■ HAL file subsystems■ FPGA hardware configuration data For more information, refer to the <i>Nios II Flash Programmer User Guide</i> . |
| Quartus® II Programmer | The Quartus II programmer is part of the Quartus II Complete Design Suite, however the Nios II IDE can start the Quartus II programmer directly. The Quartus II programmer allows you to download new FPGA configuration files to the board. For more information, refer to the Nios II IDE help system, or to the Quartus II help system. |

Altera Nios II Build Tools

This section describes the Altera Nios II build tools. You can run these tools from the Nios II command shell. Each tool provides its own documentation in the form of help pages accessible from the command line. To view the help, open the Nios II command shell, and type the following command:

```
<name of tool> --help
```

For details about the Nios II command shell, refer to [“The Nios II Command Shell” on page 1-4](#).

Nios II Software Build Tools

You can create, modify, and build Nios II programs with commands typed at a command line or embedded in a script.

[Table 1-2](#) summarizes the command-line utilities and scripts included in the software build tools. You can call these utilities and scripts on the command line or from the scripting language of your choice (such as **perl** or **bash**).

Table 1–2. Nios II Software Build Tools Utilities and Scripts

| Command | Summary | BSP Generator | Makefile Generator |
|--|---|---------------|--------------------|
| <code>nios2-app-generate-makefile</code> | Creates an application makefile | | ✓ |
| <code>nios2-lib-generate-makefile</code> | Creates a library makefile | | ✓ |
| <code>nios2-bsp-create-settings</code> | Creates a BSP settings file | ✓ | |
| <code>nios2-bsp-update-settings</code> | Updates the contents of a BSP settings file | ✓ | |
| <code>nios2-bsp-query-settings</code> | Queries the contents of a BSP settings file | ✓ | |
| <code>nios2-bsp-generate-files</code> | Generates all files for a given BSP settings file | ✓ | ✓ |
| <code>nios2-bsp-editor</code> | Nios II BSP editor standalone project GUI | ✓ | ✓ |
| <code>nios2-bsp</code> | Creates or updates a BSP | ✓ | ✓ |
| <code>nios2-c2h-generate-makefile</code> | Creates an application makefile fragment for the Nios II C2H Compiler | | ✓ |


The Nios II BSP editor provides a GUI for the software build tools.

Each command utility or script falls into one or both of the following categories:

- Nios II BSP generator
- Nios II makefile generator

Table 1–2 lists the category or categories of each utility and script. For more information about generators, refer to “Overview of the Nios II Software Build Tools Development Flow” in the *Using the Nios II Software Build Tools* chapter of the *Nios II Software Developer’s Handbook*.

The Nios II software build tools reside in the `<Nios II EDS install path>/sdk2/bin` directory.

 For further information about the Nios II software build tools, refer to the *Introduction to the Nios II Software Build Tools* chapter of the *Nios II Software Developer’s Handbook*.

File Format Conversion Tools

File format conversion is sometimes necessary when passing data from one utility to another. Table 1–3 shows the Altera-provided utilities for converting file formats.

Table 1–3. File Conversion Utilities (Part 1 of 2)

| Utility | Description |
|------------------------|---|
| <code>bin2flash</code> | Converts binary files to a Motorola S-record (.flash) file for programming to flash memory. |
| <code>elf2dat</code> | Converts a .elf file to a .dat file format appropriate for Verilog HDL hardware simulators. |
| <code>elf2flash</code> | Converts a .elf file to a .flash file for programming to flash memory. |
| <code>elf2hex</code> | Converts a .elf file to an Intel hexadecimal (.hex) file. |
| <code>elf2mem</code> | Generates the memory contents for the memory devices in a specific Nios II system. |
| <code>elf2mif</code> | Converts a .elf file to a Quartus II memory initialization (.mif) file. |

Table 1-3. File Conversion Utilities (Part 2 of 2)

| Utility | Description |
|-----------|---|
| flash2dat | Converts a .flash file to the .dat file format appropriate for Verilog HDL hardware simulators. |
| sof2flash | Converts an SRAM object (.sof) file to a .flash file. |

The file format conversion tools are in the `<Nios II EDS install path>/bin/` directory.

Other Command-Line Tools

Table 1-4 shows other Altera-provided command-line tools for developing Nios II programs.

Table 1-4. Altera Command-Line Tools

| Tool | Description |
|------------------------|---|
| nios2-download | Downloads code to a target processor for debugging or running. |
| nios2-flash-programmer | Programs data to flash memory on the target board. |
| nios2-gdb-server | Translates GNU debugger (GDB) remote serial protocol packets over Transmission Control Protocol (TCP) to JTAG transactions with a target Nios II processor. |
| nios2-terminal | Performs terminal I/O with a JTAG UART in a Nios II system |
| validate_zip | Verifies if a specified zip file is compatible with Altera's read-only zip file system. |
| nios2-debug | Downloads a program to a Nios II processor and launches the Insight debugger. |
| nios2-console | Opens the FS2 command-line interface (CLI), connects to the Nios II processor, and (optionally) downloads code. |
| nios2-configure-sof | Configures an Altera configurable part. If no explicit .sof file is specified, it tries to determine the correct file to use. |
| jtagconfig | Allows you configure the JTAG server on the host machine. It can also detect a JTAG chain and set up the download hardware configuration. |

The command-line tools described in this section are in the `<Nios II EDS install path>/bin/` directory.

Nios II IDE Command-Line Tools

Table 1-5 shows the command-line utilities that form the basis of the Nios II IDE. These tools can create and build Nios II IDE projects without launching the Nios II IDE GUI. However, Altera recommends that you use the Nios II software build tools for new projects.




For detailed information about the Nios II software build tools, refer to the *Introduction to the Nios II Software Build Tools* chapter of the *Nios II Software Developer's Handbook*.

Each of the Nios II IDE command-line tools launches the Nios II IDE in the background, without displaying the GUI. You cannot use these utilities while the IDE is running, because only one instance of the Nios II IDE can be active at a time.

The Nios II IDE command-line tools are in the `<Nios II EDS install path>/bin/` directory.

Table 1-5. Nios II IDE Command-Line Tools

| Tool | Description |
|---|--|
| <code>nios2-create-system-library</code> | Creates a new system library project. |
| <code>nios2-create-application-project</code> | Creates a new C/C++ application project. |
| <code>nios2-build-project</code> | Builds a project using the Nios II IDE managed-make facilities. Creates or updates the makefiles to build the project, and optionally runs make. <code>nios2-build-project</code> operates only on projects that exist in the current Nios II IDE workspace. |
| <code>nios2-import-project</code> | Imports a previously-created Nios II IDE project into the current workspace. |
| <code>nios2-delete-project</code> | Removes a project from the Nios II IDE workspace, and optionally deletes files from the file system. |

 The Nios II IDE command-line tools must be supplied with a workspace location. This location is supplied by means of the `-data <path to workspace>` command-line argument. The path to the workspace must not contain whitespace. Otherwise, any valid disk location can be used for the workspace. The workspace shown in [Example 1-1](#) is the default workspace which is used by the IDE.

Example 1-1. Specifying a Workspace on the Command Line

```
nios2-create-project \
  -data c:/altera/80/nios2eds/bin/eclipse/nios2-ide-workspace-8.0 \
  <other arguments>
```

GNU Compiler Tool Chain

The Nios II compiler tool chain is based on the standard GNU gcc compiler, assembler, linker, and make facilities.

GNU Tool Chain

Altera provides and supports the standard GNU compiler tool chain for the Nios II processor. Complete HTML documentation for the GNU tools resides in the Nios II EDS directory. The GNU tools are in the `<Nios II EDS install path>/bin/nios2-gnutools` directory.

GNU tools for the Nios II processor are generally named `nios2-elf-<tool name>`. The following list shows some examples:

- `nios2-elf-gcc`
- `nios2-elf-as`
- `nios2-elf-ld`

- `nios2-elf-objdump`
- `nios2-elf-size`

The exception is the make utility, which is simply named make.



For a comprehensive list of Nios II GNU tools, refer to the GNU HTML documentation, installed with the Nios II EDS. You can obtain further information about GNU from the Free Software Foundation, Inc. (www.gnu.org)

Libraries and Embedded Software Packages

Table 1-6 shows the Nios II libraries and software packages.

Table 1-6. Libraries and Software Packages

| Name | Description |
|---|--|
| HAL | Refer to the <i>Overview of the Hardware Abstraction Layer</i> chapter of the <i>Nios II Software Developer's Handbook</i> |
| MicroC/OS-II RTOS | Refer to the <i>MicroC/OS-II Real-Time Operating System</i> chapter of the <i>Nios II Software Developer's Handbook</i> . |
| NicheStack TCP/IP Stack - Nios II Edition | Refer to the <i>Ethernet and the NicheStack TCP/IP Stack - Nios II Edition</i> chapter of the <i>Nios II Software Developer's Handbook</i> . |
| newlib ANSI C standard library | The complete HTML documentation for newlib resides in the Nios II EDS directory. Also refer to the <i>Overview of the Hardware Abstraction Layer</i> chapter of the <i>Nios II Software Developer's Handbook</i> . |
| Read-only zip file system | Refer to the <i>Read-Only Zip File System</i> chapter of the <i>Nios II Software Developer's Handbook</i> . |
| Host file system | Refer to the <i>Developing Programs Using the Hardware Abstraction Layer</i> chapter of the <i>Nios II Software Developer's Handbook</i> . |

Instruction Set Simulator

The Nios II instruction set simulator (ISS) allows you to begin developing programs before the target hardware platform is ready. The Nios II IDE allows you to run programs on the ISS as easily as running on a real hardware target.

Examples

The Nios II EDS includes documented hardware example designs and software examples to demonstrate all prominent features of the Nios II processor and the development environment.

Development Flows for Creating Nios II Programs

The Nios II EDS provides the following development flows for creating Nios II programs:

- The Nios II IDE
- The Nios II software build tools at the command line

You can work entirely within the Nios II integrated development environment (IDE), or you can use the Nios II software build tools in command line and scripted environments and then import your work into the IDE for debugging.

The two design flows are not interchangeable. Source code for your applications, libraries, and drivers works in either flow, but the makefiles in the two flows are different and not compatible. Once you have committed to using one design flow, you cannot switch to using the other design flow for that project without starting over.



If your hardware design was created with SOPC Builder 7.0 or earlier, you must either use the Nios II IDE development flow, or update your hardware design.

The Nios II IDE Development Flow

The Nios II IDE development flow is an integrated environment in which you can create, modify, build, run, and debug Nios II programs with the Nios II IDE GUI. The IDE not only creates your project makefiles for you, but performs management of source files.

The Nios II IDE flow is best if you only need limited control over the build process and the project settings, and do not require customized scripting.

The Nios II IDE is based on the popular Eclipse™ framework and the Eclipse C/C++ development toolkit (CDT) plug-ins. The Nios II IDE runs other tools behind the scenes, shields you from the details of low-level tools, and presents a unified development environment.

With wizards to assist in creating and configuring projects, the Nios II IDE is easy to use, and is especially helpful for Nios II beginners. The Nios II IDE is available on both Windows and Linux operating systems.

The Nios II Command-Line Software Build Tools Development Flow

In the Nios II command-line software build tools development flow, you create, modify, build, and run Nios II programs with Nios II software build tools commands typed at a command line or embedded in a script. This development flow is best if you need fine control over the build process and the project settings, or if you require customized scripting.

The Nios II command-line software build tools development flow allows you to integrate your Nios II software development with other parts of your development flow. Using scripting, your software development flow is fully repeatable and archivable.


At debug time, you import your software build tools projects to the IDE for debugging. You can further edit, rebuild, run, and debug your imported project in the IDE.

Like the Nios II IDE, the software build tools are available on both Windows and Linux operating systems.

Running the Nios II Command-Line Software Build Tools

You run the Nios II command-line software build tools from the Nios II command shell.

For details about the Nios II command shell, refer to “[The Nios II Command Shell](#)” on page 1-4.

 For further information about the Nios II software build tools, refer to the [Introduction to the Nios II Software Build Tools](#) chapter of the *Nios II Software Developer’s Handbook*.

Terminology Differences

The Nios II software build tools and the Nios II IDE are described with somewhat different project terminology. Where the meaning is unambiguous, this handbook uses the software build tools terminology for both development flows. The IDE terminology is used where needed to distinguish the IDE development flow from the software build tools development flow.

The terminology differences are listed in [Table 1-7](#).

Table 1-7. Nios II IDE Terminology


| IDE Terminology | Software Build Tools Terminology |
|---------------------------|----------------------------------|
| Nios II C/C++ application | Nios II application |
| Nios II C/C++ library | Nios II user library |
| System library | Board support package (BSP) |
| System library option | BSP setting |
| Software component | Software package |

Makefiles

A major difference between the Nios II IDE software development flow and the Nios II software build tools flow is the difference in makefile implementation. The Nios II software build tools include makefile generators, which generate user-managed makefiles that you can edit. In the Nios II IDE development flow, the IDE creates and manages your project makefiles for you.

The key differences between user-managed makefiles and IDE makefiles are as follows:

- The Nios II IDE has control over the contents of a makefile in an IDE project.
- In an IDE makefile, the structure and syntax are optimized for automation rather than for human readability.
- It is not normally necessary or recommended for you to read or modify an IDE makefile.

 For further information about user-managed makefiles, refer to “[Makefiles](#)” in the [Using the Nios II Software Build Tools](#) chapter of the *Nios II Software Developer’s Handbook*.

Development Flow Tools

This section introduces the tools you use to create Nios II programs for each development flow. The tables are organized to help you determine the level of control you need. You can use the tables as a quick-reference guide to remind you of the differences between the development flows.

Table 1-8 shows the tools that offer a highly automated level of control for tasks in each Nios II development flow. At this level of control, you create an entire example Nios II program (consisting of an application project and BSP project) or just an example BSP project from Altera-provided software examples using default settings. Use the highly automated tools as a starting point for your development or when you do not need customization.

Table 1-8. Highly Automated Development Flow Tools

| Task | Nios II IDE Development Flow | Nios II Command-Line Software Build Tools Development Flow |
|-------------------------------------|--|--|
| Creating an example Nios II program | File > New > Nios II C/C++ Application | create-this-app script |
| Creating an example BSP | File > New > Nios II System Library | create-this-bsp script |
| Debugging | Run > Debug As > Nios II Hardware | Requires import into the Nios II IDE. For details, refer to “Importing User-Managed Projects” in the <i>Nios II Integrated Development Environment</i> chapter of the <i>Nios II Software Developer’s Handbook</i> . |

Table 1-9 shows the tools that offer an intermediate level of control for tasks in each Nios II development flow. At this level of control, you create a Nios II program from your custom code or from Altera-provided software examples. Use the intermediate tools as a starting point for your development when you need more control than the default settings provide.

Table 1-9. Intermediate Development Flow Tools

| Task | Nios II IDE Development Flow | Nios II Command-Line Software Build Tools Development Flow |
|-------------------------|--|--|
| Creating an application | File > New > Nios II C/C++ Application | nios2-app-generate-makefile utility |
| Creating a library | File > New > Nios II C/C++ Library | nios2-lib-generate-makefile utility |
| Creating a BSP | <ul style="list-style-type: none"> ■ File > New > Nios II System Library ■ Project > Properties > System Library | nios2-bsp script |
| Debugging | Run > Debug As > Nios II Hardware | Requires import into the Nios II IDE. For details, refer to “Importing User-Managed Projects” in the <i>Nios II Integrated Development Environment</i> chapter of the <i>Nios II Software Developer’s Handbook</i> . |

Table 1-10 shows the tools that offer an advanced level of control for tasks related to BSPs in each Nios II development flow. At this level of control, you create a Nios II BSP with sophisticated, scriptable control. Use the advanced tools when you need total control over the BSP creation and build process and the BSP project settings.

Table 1-10. Advanced BSP Development Flow Tools

| Task | Nios II IDE Development Flow | Nios II Command-Line Software Build Tools Development Flow |
|--------------------|---|---|
| Creating a BSP | Scriptable control of a BSP project is not supported. | <ul style="list-style-type: none"> ■ <code>nios2-bsp-create-settings</code> utility ■ <code>nios2-bsp-generate-files</code> utility ■ Tcl scriptable |
| Updating a BSP | Scriptable control of a BSP project is not supported. | <ul style="list-style-type: none"> ■ <code>nios2-bsp-update-settings</code> utility ■ <code>nios2-bsp-generate-files</code> utility ■ Tcl scriptable |
| Querying a BSP | Not supported. | <ul style="list-style-type: none"> ■ <code>nios2-bsp-query-settings</code> utility ■ Tcl scriptable |
| Customizing newlib | Not supported. | <code>nios2-bsp</code> script using <code>CUSTOM_NEWLIB_FLAGS</code> setting |

Third-Party Support

Several third-party vendors support the Nios II processor, providing products such as design services, operating systems, stacks, other software libraries, and development tools.



For the most up-to-date information about third-party support for the Nios II processor, visit the [Nios II Processor](#) page of the Altera website.

Migrating from the First-Generation Nios Processor

If you are a user of the first-generation Nios processor, Altera recommends that you migrate to the Nios II processor for future designs. The straightforward migration process is discussed in *AN 350: Upgrading Nios Processor Systems to the Nios II Processor*.

Further Nios II Information

This handbook is one part of the complete Nios II processor documentation suite. Consult the following references for further Nios II information:

- The *Nios II Processor Reference Handbook* defines the processor hardware architecture and features, including the instruction set architecture.
- *Volume 5: Embedded Peripherals* of the *Quartus II Handbook* provides a reference for the peripherals distributed with the Nios II processor. This handbook describes the hardware structure and Nios II software drivers for each peripheral.
- The *Embedded Design Handbook* describes how to use Altera software development tools effectively, and recommends design styles and practices for developing, debugging, and optimizing embedded systems.

- The Nios II IDE provides tutorials and complete reference for using the features of the GUI. The Nios II IDE help system is available in the Nios II IDE.
- The Altera Knowledge Database is an Internet resource that offers solutions to frequently asked questions with an easy-to-use search engine. Visit the [Knowledge Database](#) page of the Altera website.
- Altera application notes and tutorials offer step-by-step instructions on using the Nios II processor for a specific application or purpose. These documents are available on the [Literature: Nios II Processor](#) page of the Altera website.
- The Nios II EDS documentation launchpad. The launchpad is an HTML page installed with the Nios II EDS, which provides links to Nios II documentation, examples, and other resources. The way you open the launchpad depends on your software platform.
 - In the Windows operating system, on the Start menu, point to **Programs > Altera > Nios II EDS**, and click **Literature**.
 - In the Linux operating system, open `<Nios II EDS install path>/documents/index.html` in a web browser.

Referenced Documents

This chapter references the following documents:

- *Nios II Integrated Development Environment* chapter of the *Nios II Software Developer's Handbook*
- *Introduction to the Nios II Software Build Tools* chapter of the *Nios II Software Developer's Handbook*
- *Using the Nios II Software Build Tools* chapter of the *Nios II Software Developer's Handbook*
- *Hardware Abstraction Layer* section of the *Nios II Software Developer's Handbook*
- *Overview of the Hardware Abstraction Layer* chapter of the *Nios II Software Developer's Handbook*
- *Developing Programs Using the Hardware Abstraction Layer* chapter of the *Nios II Software Developer's Handbook*
- *MicroC/OS-II Real-Time Operating System* chapter of the *Nios II Software Developer's Handbook*
- *Ethernet and the NicheStack TCP/IP Stack - Nios II Edition* chapter of the *Nios II Software Developer's Handbook*
- *Read-Only Zip File System* chapter of the *Nios II Software Developer's Handbook*
- *HAL API Reference* chapter of the *Nios II Software Developer's Handbook*
- *AN 350: Upgrading Nios Processor Systems to the Nios II Processor*
- *Nios II Processor Reference Handbook*
- *Volume 5: Embedded Peripherals of the Quartus II Handbook*
- *Embedded Design Handbook*
- *GNU documentation* installed with the Nios II EDS

- The [Nios II Processor](#) page of the Altera website.
- The [Literature: Nios II Processor](#) page of the Altera website.

Document Revision History

Table 1-11 shows the revision history for this document.

Table 1-11. Document Revision History

| Date & Document Version | Changes Made | Summary of Changes |
|-------------------------|---|---|
| March 2009 v9.0.0 | <ul style="list-style-type: none"> ■ Incorporated information formerly in <i>Altera-Provided Development Tools</i> chapter. ■ Described BSP editor. ■ Reorganized and updated information and terminology to clarify role of Nios II software build tools. ■ Described <code>-data</code> argument for IDE command-line tools. ■ Corrected minor typographical errors. | <ul style="list-style-type: none"> ■ List of development tools ■ BSP editor |
| May 2008 v8.0.0 | Added “What’s New” section. | <ul style="list-style-type: none"> ■ SOPC Builder system (<code>.sopcinfo</code>) file ■ Example designs removed from EDS ■ memory management unit (MMU) added to Nios II core |
| October 2007 v7.2.0 | No change from previous release. | |
| May 2007 v7.1.0 | <ul style="list-style-type: none"> ■ Revised entire chapter to introduce Nios II EDS design flows, Nios II programs, Nios II software build tools, and Nios II BSPs. ■ 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 | No change from previous release. | |
| May 2006 v6.0.0 | No change from previous release. | |
| October 2005 v5.1.0 | No change from previous release. | |
| May 2005 v5.0.0 | No change from previous release. | |
| May 2004 v1.0 | Initial Release. | |

