


## Introduction

This handbook is the primary reference for the Nios® II family of embedded processors. The handbook describes the Nios II processor from a high-level conceptual description to the low-level details of implementation. The chapters in this handbook define the Nios II processor architecture, the programming model, the instruction set, and more.

 This handbook is part of a larger collection of documents covering the Nios II processor and its usage that you can find on the [Literature: Nios II Processor](#) page on the Altera® website.

This 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. This handbook intentionally minimizes discussion of hardware implementation details of the processor system. That said, the Nios II processors are designed for Altera FPGA devices, and so this handbook does describe some FPGA implementation concepts. Your familiarity with FPGA technology provides a deeper understanding of the engineering trade-offs related to the design and implementation of the Nios II processor.

This *Introduction* chapter introduces the Altera Nios II embedded processor family. The chapter helps hardware and software engineers understand the similarities and differences between the Nios II processor and traditional embedded processors. This chapter contains the following sections:

- “Nios II Processor System Basics”
- “Getting Started with the Nios II Processor” on page 1–2
- “Customizing Nios II Processor Designs” on page 1–3
- “Configurable Soft-Core Processor Concepts” on page 1–4
- “OpenCore Plus Evaluation” on page 1–5

## Nios II Processor System Basics

The Nios II processor is a general-purpose RISC processor core, providing:

- Full 32-bit instruction set, data path, and address space
- 32 general-purpose registers
- Optional shadow register sets
- 32 interrupt sources
- External interrupt controller interface for more interrupt sources
- Single-instruction  $32 \times 32$  multiply and divide producing a 32-bit result
- Dedicated instructions for computing 64-bit and 128-bit products of multiplication

- Floating-point instructions for single-precision floating-point operations
- Single-instruction barrel shifter
- Access to a variety of on-chip peripherals, and interfaces to off-chip memories and peripherals
- Hardware-assisted debug module enabling processor start, stop, step, and trace under control of the Nios II software development tools
- Optional memory management unit (MMU) to support operating systems that require MMUs
- Optional memory protection unit (MPU)
- Software development environment based on the GNU C/C++ tool chain and the Nios II Software Build Tools (SBT) for Eclipse
- Integration with Altera's SignalTap® II Embedded Logic Analyzer, enabling real-time analysis of instructions and data along with other signals in the FPGA design
- Instruction set architecture (ISA) compatible across all Nios II processor systems
- Performance up to 250 DMIPS

A Nios II processor system is equivalent to a microcontroller or “computer on a chip” that includes a processor and a combination of peripherals and memory on a single chip. A Nios II processor system consists of a Nios II processor core, a set of on-chip peripherals, on-chip memory, and interfaces to off-chip memory, all implemented on a single Altera device. Like a microcontroller family, all Nios II processor systems use a consistent instruction set and programming model.

## Getting Started with the Nios II Processor

Getting started with the Nios II processor is similar to any other microcontroller family. The easiest way to start designing effectively is to purchase a development kit from Altera that includes a ready-made evaluation board and the Nios II Embedded Design Suite (EDS) containing all the software development tools necessary to write Nios II software.

The Nios II EDS includes the following two closely-related software development tool flows:

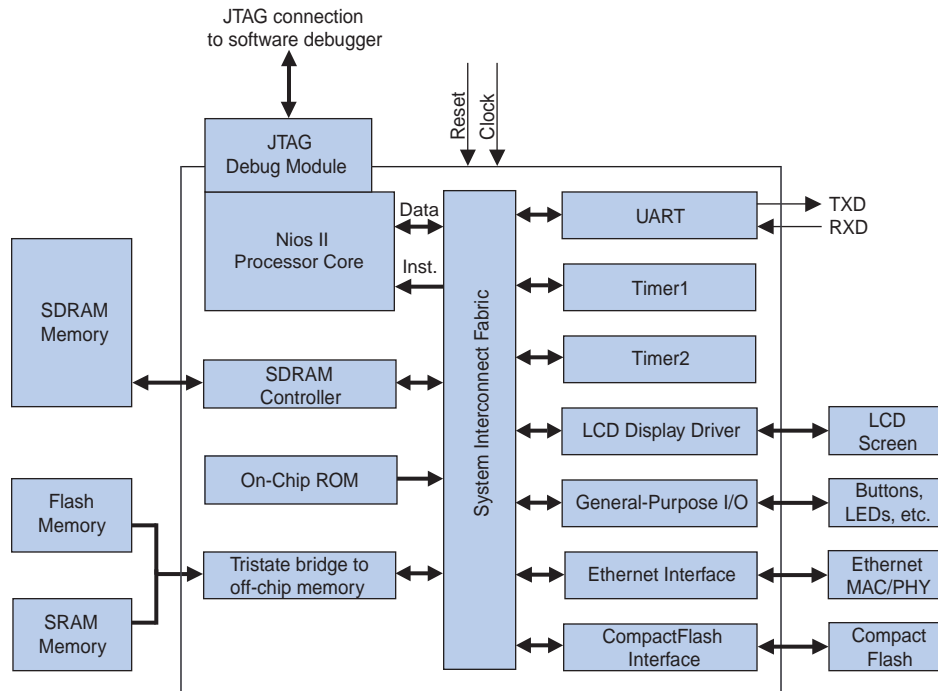
- The Nios II SBT
- The Nios II SBT for Eclipse

Both tools flows are based on the GNU C/C++ compiler. The Nios II SBT for Eclipse provides a familiar and established environment for software development. Using the Nios II SBT for Eclipse, you can immediately begin developing and simulating Nios II software applications.

The Nios II SBT also provides a command line interface.

Using the Nios II hardware reference designs included in an Altera development kit, you can prototype an application running on a board before building a custom hardware platform. [Figure 1–1](#) shows an example of a Nios II processor reference design available in an Altera Nios II development kit.

Figure 1-1. Example of a Nios II Processor System



If the prototype system adequately meets design requirements using an Altera-provided reference design, you can copy the reference design and use it as is in the final hardware platform. Otherwise, you can customize the Nios II processor system until it meets cost or performance requirements.

## Customizing Nios II Processor Designs

In practice, most FPGA designs implement some extra logic in addition to the processor system. Altera FPGAs provide flexibility to add features and enhance performance of the Nios II processor system. Conversely, you can eliminate unnecessary processor features and peripherals to fit the design in a smaller, lower-cost device.

Because the pins and logic resources in Altera devices are programmable, many customizations are possible:

- You can rearrange the pins on the chip to simplify the board design. For example, you can move address and data pins for external SDRAM memory to any side of the chip to shorten board traces.
- You can use extra pins and logic resources on the chip for functions unrelated to the processor. Extra resources can provide a few extra gates and registers as glue logic for the board design; or extra resources can implement entire systems. For example, a Nios II processor system consumes only 5% of a large Altera FPGA, leaving the rest of the chip's resources available to implement other functions.
- You can use extra pins and logic on the chip to implement additional peripherals for the Nios II processor system. Altera offers a library of peripherals that easily connect to Nios II processor systems.

## Configurable Soft-Core Processor Concepts

This section introduces Nios II concepts that are unique or different from other discrete microcontrollers. The concepts described in this section provide a foundation for understanding the rest of the features discussed in this document.

### Configurable Soft-Core Processor

The Nios II processor is a configurable soft-core processor, as opposed to a fixed, off-the-shelf microcontroller. In this context, configurable means that you can add or remove features on a system-by-system basis to meet performance or price goals. Soft-core means the processor core is not fixed in silicon and can be targeted to any Altera FPGA family.

Configurability does not require you to create a new Nios II processor configuration for every new design. Altera provides ready-made Nios II system designs that you can use as is. If these designs meet your system requirements, there is no need to configure the design further. In addition, software designers can use the Nios II instruction set simulator to begin writing and debugging Nios II applications before the final hardware configuration is determined.

### Flexible Peripheral Set and Address Map

A flexible peripheral set is one of the most notable differences between Nios II processor systems and fixed microcontrollers. Because the Nios II processor is implemented in programmable logic, you can easily build made-to-order Nios II processor systems with the exact peripheral set required for the target applications.

A corollary of flexible peripherals is a flexible address map. Altera provides software constructs to access memory and peripherals generically, independently of address location. Therefore, the flexible peripheral set and address map does not affect application developers.

There are two broad classes of peripherals: standard peripherals and custom peripherals.

#### Standard Peripherals


Altera provides a set of peripherals commonly used in microcontrollers, such as timers, serial communication interfaces, general-purpose I/O, SDRAM controllers, and other memory interfaces. The list of available peripherals continues to grow as Altera and third-party vendors release new peripherals.



For details on the Altera-provided cores, refer to the *Embedded Peripherals IP User Guide*.

#### Custom Peripherals

You can also create custom peripherals and integrate them in Nios II processor systems. For performance-critical systems that spend most CPU cycles executing a specific section of code, it is a common technique to create a custom peripheral that implements the same function in hardware. This approach offers a double performance benefit: the hardware implementation is faster than software; and the processor is free to perform other functions in parallel while the custom peripheral operates on data.

 For details on creating custom peripherals, refer to the *SOPC Builder Components* chapter in volume 4 of the *Quartus II Handbook*.

### Custom Instructions

Like custom peripherals, custom instructions allow you to increase system performance by augmenting the processor with custom hardware. The custom logic is integrated into the Nios II processor's arithmetic logic unit (ALU). Similar to native Nios II instructions, custom instruction logic can take values from up to two source registers and optionally write back a result to a destination register.

Because the processor is implemented on reprogrammable Altera FPGAs, software and hardware engineers can work together to iteratively optimize the hardware and test the results of software running on hardware.

From the software perspective, custom instructions appear as machine-generated assembly macros or C functions, so programmers do not need to understand assembly language to use custom instructions.

## Automated System Generation

Altera's SOPC Builder design tool fully automates the process of configuring processor features and generating a hardware design that you program in an FPGA. The SOPC Builder graphical user interface (GUI) enables you to configure Nios II processor systems with any number of peripherals and memory interfaces. You can create entire processor systems without performing any schematic or HDL design entry. SOPC Builder can also import HDL design files, providing an easy mechanism to integrate custom logic in a Nios II processor system.

After system generation, you can download the design onto a board, and debug software executing on the board. To the software developer, the processor architecture of the design is set. Software development proceeds in the same manner as for traditional, nonconfigurable processors.

## OpenCore Plus Evaluation

You can evaluate the Nios II processor without a license. With Altera's free OpenCore Plus evaluation feature, you can perform the following actions:

- Simulate the behavior of a Nios II processor within your system
- Verify the functionality of your design, as well as evaluate its size and speed quickly and easily
- Generate time-limited device programming files for designs that include Nios II processors
- Program a device and verify your design in hardware

You only need to purchase a license for the Nios II processor when you are completely satisfied with its functionality and performance, and want to take your design to production.

 For more information about OpenCore Plus, refer to *AN 320: OpenCore Plus Evaluation of Megafunctions*.

## Referenced Documents

This chapter references the following documents:

- *Embedded Peripherals IP User Guide*
- *SOPC Builder Components* chapter in volume 4 of the *Quartus II Handbook*
- *AN 320: OpenCore Plus Evaluation of Megafunctions*
- **Literature: Nios II Processor** page on the Altera website

## Document Revision History

Table 1–1 shows the revision history for this document.

**Table 1–1.** Document Revision History (Part 1 of 2)

<b>Date &amp; Document Version</b>	<b>Changes Made</b>	<b>Summary of Changes</b>
July 2010 v10.0.0	Maintenance release.	—
November 2009 v9.1.0	<ul style="list-style-type: none"> <li>■ Added external interrupt controller interface information.</li> <li>■ Added shadow register set information.</li> </ul>	Added shadow register sets and external interrupt controller support
March 2009 v9.0.0	Maintenance release.	—
November 2008 v8.1.0	Maintenance release.	—
May 2008 v8.0.0	Added MMU and MPU to bullet list of features.	Added MMU and MPU.
October 2007 v7.2.0	Added OpenCore Plus section.	—
May 2007 v7.1.0	<ul style="list-style-type: none"> <li>■ Added table of contents to Introduction section.</li> <li>■ Added Referenced Documents section.</li> </ul>	—
March 2007 v7.0.0	Maintenance release.	—
November 2006 v6.1.0	Maintenance release.	—
May 2006 v6.0.0	<ul style="list-style-type: none"> <li>■ Added single precision floating-point and integration with SignalTap® II logic analyzer to features list.</li> <li>■ Updated performance to 250 DMIPS.</li> </ul>	—
October 2005 v5.1.0	Maintenance release.	—
May 2005 v5.0.0	Maintenance release.	—

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

<b>Date &amp; Document Version</b>	<b>Changes Made</b>	<b>Summary of Changes</b>
September 2004 v1.1	Updates for Nios II 1.01 release.	—
May 2004 v1.0	Initial release.	—

