4. Implementing the Nios II Processor in SOPC Builder

NII51004-6.0.0

Introduction

This chapter describes the Nios® II configuration wizard in SOPC Builder. The Nios II configuration wizard allows you to specify the processor features for a particular Nios II hardware system. This chapter covers only the features of the Nios II processor that can be configured via the Nios II configuration wizard. It is not a user guide for creating complete Nios II processor systems.

To get started using SOPC Builder to design custom Nios II systems, refer to the *Nios II Hardware Development Tutorial*. Nios II development kits also provide a number of ready-made example hardware designs that demonstrate several different configurations of the Nios II processor.

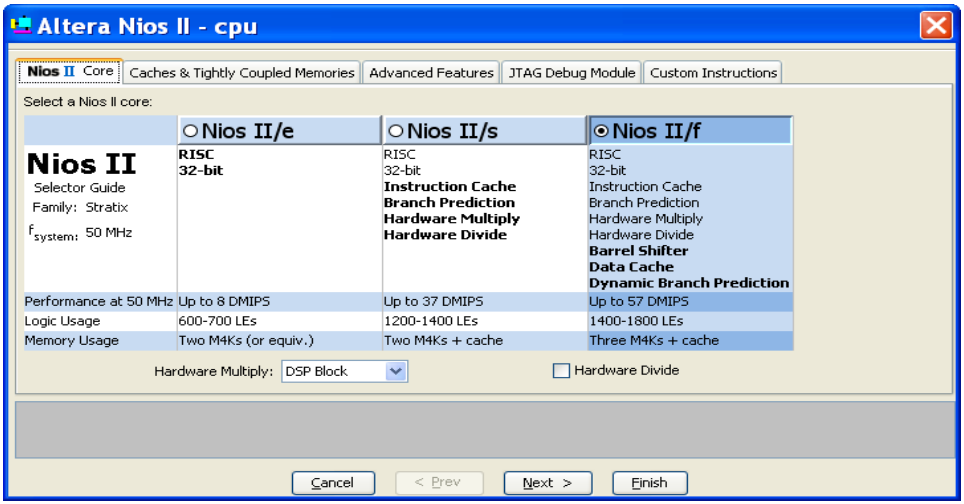
The Nios II processor configuration wizard has several tabs. The following sections describe the settings available on each tab.

Due to evolution and improvement of the Nios II configuration wizard, the figures in this chapter might not match the exact screens that appear in SOPC Builder.

Nios II Core Tab

The **Nios II Core** tab presents the main settings for configuring the Nios II processor core. An example of the **Nios II Core** tab is shown in Figure 4–1.

Figure 4–1. Nios II Core Tab in the Nios II Configuration Wizard



Core Setting

The main purpose of the **Nios II Core** tab is to select the processor core. The core you select on this tab affects other options available on this and other tabs.

Currently, Altera® offers three Nios II cores:

- *Nios II/f*—The Nios II/f "fast" core is designed for fast performance. As a result, this core presents the most configuration options allowing you to fine-tune the processor for performance.
- *Nios II/s*—The Nios II/s "standard" core is designed for small size while maintaining performance.
- *Nios II/e*—The Nios II/e "economy" core is designed to achieve the smallest possible core size. As a result, this core has a limited feature set, and many settings are not available when the Nios II/e core is selected.

As shown in Figure 4–1, the **Nios II Core** tab displays a "selector guide" table that lists the basic properties of each core.

For complete details of each core, see the *Nios II Core Implementation Details* chapter of the *Nios II Processor Reference Handbook*.

Multiply & Divide Settings

The Nios II/s and Nios II/f cores offer different hardware multiply and divide options. You can choose the best option to balance embedded multiplier usage, logic element (LE) usage, and performance.

The **Hardware Multiply** setting provides the following options:

- Include embedded multipliers (e.g., the DSP blocks in Stratix® devices) in the arithmetic logic unit (ALU). This is the default when targeting devices that have embedded multipliers.
- Include LE-based multipliers in the ALU. This option achieves high multiply performance without consuming embedded multiplier resources.
- Omit hardware multiply. This option conserves logic resources by eliminating multiply hardware. Multiply operations are implemented in software.

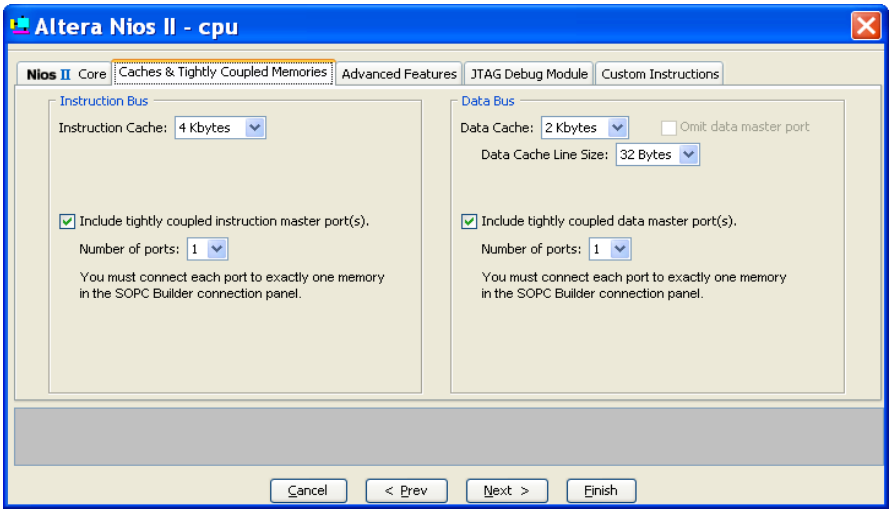
Turning on the **Hardware Divide** setting includes LE-based divide hardware in the ALU. The **Hardware Divide** option achieves much greater performance than software emulation of divide operations.

For details on the effects of the **Hardware Multiply** and **Hardware Divide** options on performance, see the *Nios II Core Implementation Details* chapter of the *Nios II Processor Reference Handbook*.

Caches & Tightly Coupled Memories Tab

The **Caches & Tightly Coupled Memories** tab allows you to configure the cache and tightly coupled memory usage for the instruction and data buses. An example of the **Caches & Tightly Coupled Memories** tab is shown in Figure 4–2.

Figure 4–2. Caches & Tightly Coupled Memories Tab in the Nios II Configuration Wizard



Instruction Settings

The **Instruction** settings provide the following options for the Nios II/f and Nios II/s cores:

- **Instruction Cache** - Specifies the size of the instruction cache. Valid sizes are from 512 bytes to 64 Kbytes, or **None**.

  Choosing **None** disables the instruction cache, which also removes the Avalon instruction master port from the Nios II core. In this case, you must include a tightly coupled instruction memory.

- **Include tightly coupled instruction master port(s)** - When turned on, the Nios II core includes tightly coupled memory ports. You can specify one to four ports with the **Number of ports** setting. Tightly coupled memory ports appear on the connection panel of the Nios II core in the SOPC Builder **System Contents** tab. You must connect each port to exactly one memory component in the system.

Data Settings

The **Data** settings provide the following options for the Nios II/f core:

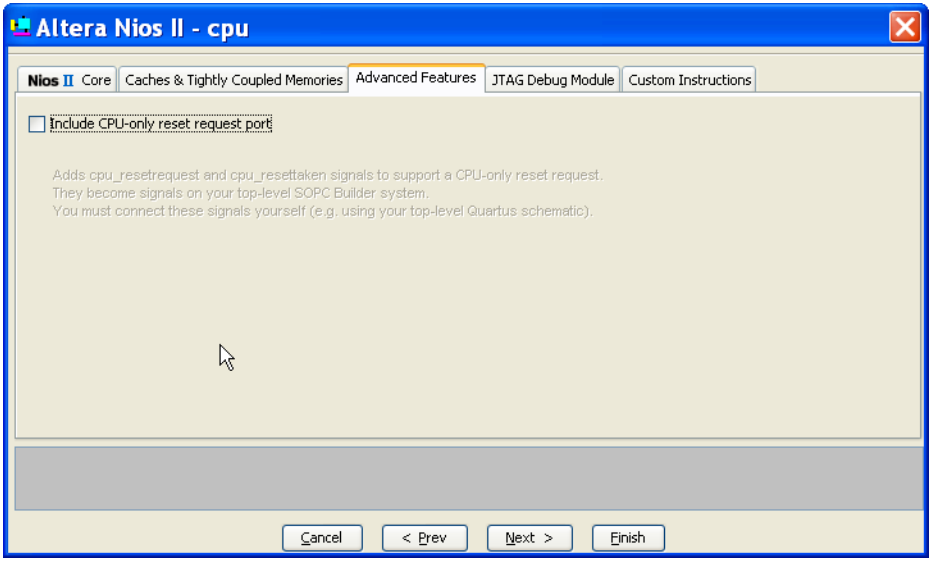
- **Data Cache** - Specifies the size of the data cache. Valid sizes are from 512 bytes to 64 Kbytes, or **None**. Depending on the value specified for **Data Cache**, the following options are available:
  - **Data Cache Line Size** - Valid sizes are 4, 16, or 32 bytes.
  - **Omit data master port** - If you set **Data Cache** to **None**, you can optionally turn on **Omit data master port** to remove the Avalon data master port from the Nios II core. In this case, you must include a tightly coupled data memory.
- **Include tightly coupled data master port(s)** - When turned on, the Nios II core includes tightly coupled memory ports. You can specify one to four ports with the **Number of ports** setting. Tightly coupled memory ports appear on the connection panel of the Nios II core in the SOPC Builder **System Contents** tab. You must connect each port to exactly one memory component in the system.

Advanced Features Tab

The **Advanced Features** tab allows you to enable specialized features of the Nios II processor. It contains one option: **Include cpu_resetrequest and cpu_resettaken signals**. This option adds CPU-only reset request signals to the Nios II processor. These signals let another device individually reset the Nios II processor without resetting the entire SOPC Builder system. The signals are exported to the top level of your SOPC Builder system.

Figure 4–3 on page 4–6 shows the **Advanced Features** tab.

Figure 4–3. Advanced Features Tab in the Nios II Configuration Wizard



For further details on the CPU-only reset request signals, refer to the *Processor Architecture* chapter in the *Nios II Processor Reference Handbook*.

JTAG Debug Module Tab

The **JTAG Debug Module** tab presents settings for configuring the JTAG debug module on the Nios II core. You can select the debug features appropriate for your target application.

Soft-core processors such as the Nios II processor offer unique debug capabilities beyond the features of traditional-fixed processors. The soft-core nature of the Nios II processor allows you to debug a system in development using a full-featured debug core, and later remove the debug features to conserve logic resources. For the release version of a product, you might choose to reduce the JTAG debug module functionality, or remove it altogether.

Table 4–1 describes the debug features available to you for debugging your system.

Table 4–1. Debug Configuration Features

Feature	Description
JTAG Target Connection	The ability to connect to the CPU through the standard JTAG pins on the Altera FPGA. This provides the basic capabilities to start and stop the processor, and examine/edit registers and memory.
Download Software	The ability to download executable code to the processor's memory via the JTAG connection.
Software Breakpoints	The ability to set a breakpoint on instructions residing in RAM
Hardware Breakpoints	The ability to set a breakpoint on instructions residing in nonvolatile memory, such as flash memory.
Data Triggers	The ability to trigger based on address value, data value, or read or write cycle. You can use a trigger to halt the processor on specific events or conditions, or to activate other events, such as starting execution trace, or sending a trigger signal to an external logic analyzer. Two data triggers can be combined to form a trigger that activates on a range of data or addresses.
Instruction Trace	The ability to capture the sequence of instructions executing on the processor in real time.
Data Trace	The ability to capture the addresses and data associated with read and write operations executed by the processor in real time.
On-Chip Trace	The ability to store trace data in on-chip memory.
Off-Chip Trace	The ability to store trace data in an external debug probe. Off-chip trace requires a debug probe from First Silicon Solutions (FS2).

Debug Level Settings

There are five debug levels in the **JTAG Debug Module** tab as shown in Figure 4–4.

Figure 4–4. JTAG Debug Module Tab in the Nios II Configuration Wizard

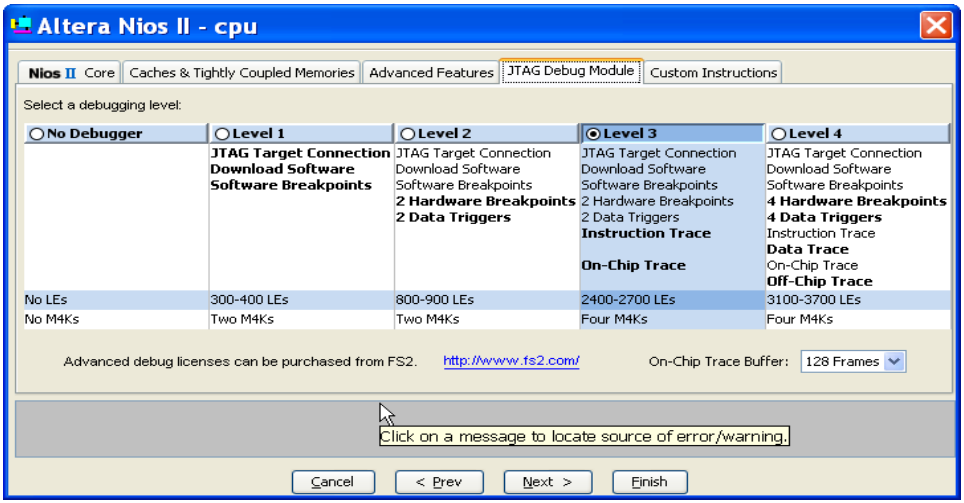


Table 4–2 on page 4–9 is a detailed list of the characteristics of each debug level. Different levels consume different amounts of on-chip resources. Certain Nios II cores have restricted debug options, and certain options require debug tools provided by First Silicon Solutions (FS2).

For details on the Nios II debug features available from FS2, visit **www.fs2.com**.

Table 4–2. JTAG Debug Module Levels

Debug Feature	No Debug	Level 1	Level 2	Level 3	Level 4 *(1)*
Logic Usage	0	300 - 400 LEs	800 - 900 LEs	2,400 - 2,700 LEs	3,100 - 3,700 LEs
On-Chip Memory Usage	0	Two M4Ks	Two M4Ks	Four M4Ks	Four M4Ks
External I/O Pins Required *(2)*	0	0	0	0	20
JTAG Target Connection	No	Yes	Yes	Yes	Yes
Download Software	No	Yes	Yes	Yes	Yes
Software Breakpoints	None	Unlimited	Unlimited	Unlimited	Unlimited
Hardware Execution Breakpoints	0	None	2	2	4
Data Triggers	0	None	2	2	4
On-Chip Trace	0	None	None	Up to 64K Frames *(3)*	Up to 64K Frames
Off-Chip Trace *(4)*	0	None	None	None	128K Frames

Notes to Table 4–2:
(1) Level 4 requires the purchase of a software upgrade from FS2.
(2) Not including the dedicated JTAG pins on the Altera FPGA.
(3) An additional license from FS2 is required to use more than 16 frames.
(4) Off-chip trace requires the purchase of additional hardware from FS2.

On-Chip Trace Buffer Settings

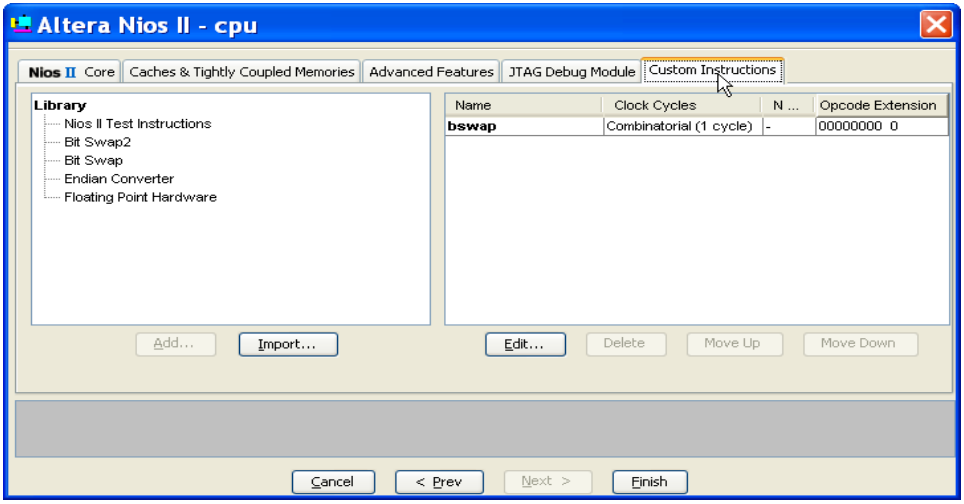
Debug levels 3 and 4 support trace data collection into an on-chip memory buffer. The on-chip trace buffer size can be set to sizes from 128 to 64K trace frames.

Larger buffer sizes consume more on-chip M4K RAM blocks. Every M4K RAM block can store up to 128 trace frames.

Custom Instructions Tab

The **Custom Instructions** tab allows you to connect custom instruction logic to the Nios II arithmetic logic unit (ALU). You can achieve significant performance improvements—often on the order of 10x to 100x—by implementing performance-critical operations in hardware using custom-instruction logic. Figure 4–5 shows an example of the **Custom Instructions** tab.

Figure 4–5. Custom Instructions Tab in the Nios II Configuration Wizard



To add a custom instruction to the Nios II processor, select the custom instruction from the **Library** list at the left side of the dialog box, and click **Add**.

A complete discussion of the hardware and software design process for custom instructions is beyond the scope of this chapter. For full details on the topic of custom instructions, including working example designs, see the *Nios II Custom Instruction User Guide*.

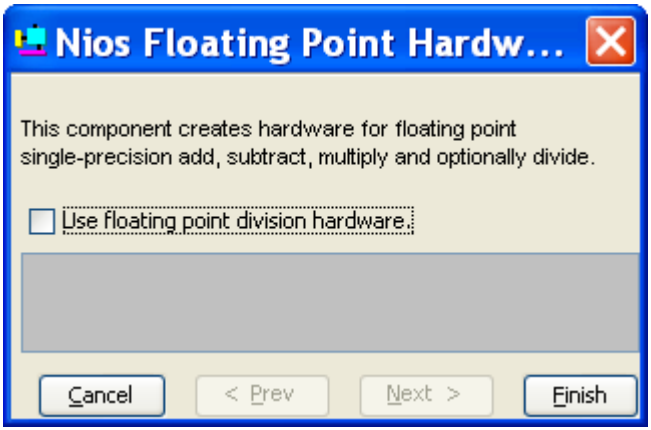
Floating-Point Custom Instructions

The Nios II core offers a set of optional predefined custom instructions that implement floating-point arithmetic operations. You can choose to include these custom instructions to support computation-intensive floating-point applications.

The basic set of floating-point custom instructions includes single precision (32-bit) floating-point addition, subtraction, and multiplication. Floating-point division is available as an extension to the basic instruction set. The best choice for your hardware design depends on a balance among floating-point usage, hardware resource usage, and performance.

To add the floating-point custom instructions to the Nios II processor, select **Floating Point Hardware** from the **Library** list, and click **Add**.

Figure 4–6. Nios II Floating Point Hardware Dialog Box



The **Nios II Floating Point Hardware** dialog box, shown in Figure 4–6, provides one option: **Use floating point division hardware.** If you leave this check box off, SOPC Builder omits floating-point division from the Nios II processor, while including addition, subtraction, and multiplication. The floating-point division hardware requires more resources than the other instructions, so you might wish to omit it if your application does not make heavy use of floating-point division.

Click **Finish** to add the floating point custom instructions to the Nios II processor.

If the target device includes on-chip multiplier blocks, the floating-point custom instructions incorporates them as needed. If there are no on-chip multiplier blocks, the floating-point custom instructions are entirely based on general-purpose logic elements.

The opcode extensions for the floating-point custom instructions are 252 through 255 (0xFC through 0xFF). These opcode extensions cannot be modified.

For details on floating-point instruction usage in the Nios II Embedded Design Suite (EDS), see the tutorial *Using Nios II Floating-Point Custom Instructions.*