SOPC Builder

User Guide

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
www.altera.com

Document Version: 1.0
Document Date: June 2003

About this User Guide

This user guide provides comprehensive information about the Altera® SOPC Builder system development tool, which is available with the Quartus® II software version 3.0 and higher.

Table 1 shows the user guide revision history.

Table 1. User Guide Revision History

Date	Description
June 2003	First publication.

How to Find Information

- The Adobe Acrobat Find feature allows you to search the contents of a PDF file. Click on the binoculars icon in the top toolbar to open the Find dialog box.
- Bookmarks serve as an additional table of contents.
- Thumbnail icons, which provide miniature previews of each page, provide a link to the pages.
- Numerous links, shown in green text, allow you to jump to related information.

How to Contact Altera

For the most up-to-date information about Altera products, go to the Altera world-wide web site at www.altera.com.

For technical support on this product, go to www.altera.com/mysupport. For additional information about Altera products, consult the sources shown in Table 2.

Table 2. How to Contact Altera

Information Type	USA & Canada	All Other Locations
Technical support	www.altera.com/mysupport/	www.altera.com/mysupport/
	(800) 800-EPLD (3753) (7:00 a.m. to 5:00 p.m. Pacific Time)	(408) 544-7000 *(1)* (7:00 a.m. to 5:00 p.m. Pacific Time)
Product literature	www.altera.com	www.altera.com
Altera literature services	lit_req@altera.com *(1)*	lit_req@altera.com *(1)*
Non-technical customer service	(800) 767-3753	(408) 544-7000 (7:30 a.m. to 5:30 p.m. Pacific Time)
FTP site	ftp.altera.com	ftp.altera.com

Note:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

This document uses the typographic conventions shown in Table 3.

Table 3. Conventions

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Command names, dialog box titles, checkbox options, and dialog box options are shown in bold, initial capital letters. Example: **Save As** dialog box.
bold type	External timing parameters, directory names, project names, disk drive names, filenames, filename extensions, and software utility names are shown in bold type. Examples: **f_{MAX}**, **\qdesigns** directory, **d:** drive, **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Document titles are shown in italic type with initial capital letters. Example: *AN 75: High-Speed Board Design.*
Italic type	Internal timing parameters and variables are shown in italic type. Examples: t_{PIA}, $n + 1$. Variable names are enclosed in angle brackets (< >) and shown in italic type. Example: <*file name*>, <*project name*>**.pof** file.
Initial Capital Letters	Keyboard keys and menu names are shown with initial capital letters. Examples: Delete key, the Options menu.
“Subheading Title”	References to sections within a document and titles of online help topics are shown in quotation marks. Example: “Typographic Conventions.”
`Courier type`	Signal and port names are shown in lowercase Courier type. Examples: `data1`, `tdi`, `input`. Active-low signals are denoted by suffix `n`, e.g., `resetn`. Anything that must be typed exactly as it appears is shown in Courier type. For example: `c:\qdesigns\tutorial\chiptrip.gdf`. Also, sections of an actual file, such as a Report File, references to parts of files (e.g., the AHDL keyword `SUBDESIGN`), as well as logic function names (e.g., `TRI`) are shown in Courier.
1., 2., 3., and a., b., c.,...	Numbered steps are used in a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■	Bullets are used in a list of items when the sequence of the items is not important.
✓	The checkmark indicates a procedure that consists of one step only.
☞	The hand points to information that requires special attention.
↵	The angled arrow indicates you should press the Enter key.
👣	The feet direct you to more information on a particular topic.

Notes:

Contents

Notes:

Getting Started

Introduction

The SOPC Builder system development tool dramatically simplifies the task of creating high-performance system-on-a-programmable-chip (SOPC) designs by accelerating system definition and integration. Using SOPC Builder, system designers can define and implement a complete system, from hardware to software, within one tool and in a fraction of the time of traditional system-on-a-chip (SoC) design. SOPC Builder is integrated within the Altera Quartus II software to give FPGA designers immediate access to a revolutionary new development tool.

You can use SOPC Builder as a powerful platform for composing bus-based systems from common system components placed inside or outside the FPGA. SOPC Builder library components supplied by Altera or other third party developers range from simple blocks of fixed logic, to complex, parameterized, and dynamically generated subsystems. SOPC Builder library components include:

- Processors
- Microcontroller peripherals
- Digital signal processing (DSP) cores
- Intellectual property (IP) cores
- Communications peripherals
- Interfaces
  - Memory (on-chip or off-chip)
  - Buses and bridges
  - ASSPs
  - ASICs
- Software components
  - Header files
  - Generic C drivers
  - Operating system (OS) kernels
  - Middleware libraries

This user guide describes how to use the SOPC Builder user interface and explains how to obtain available library components.

Refer to "Additional Information" on page 18 for additional documentation on SOPC Builder.

SOPC Builder User Interface

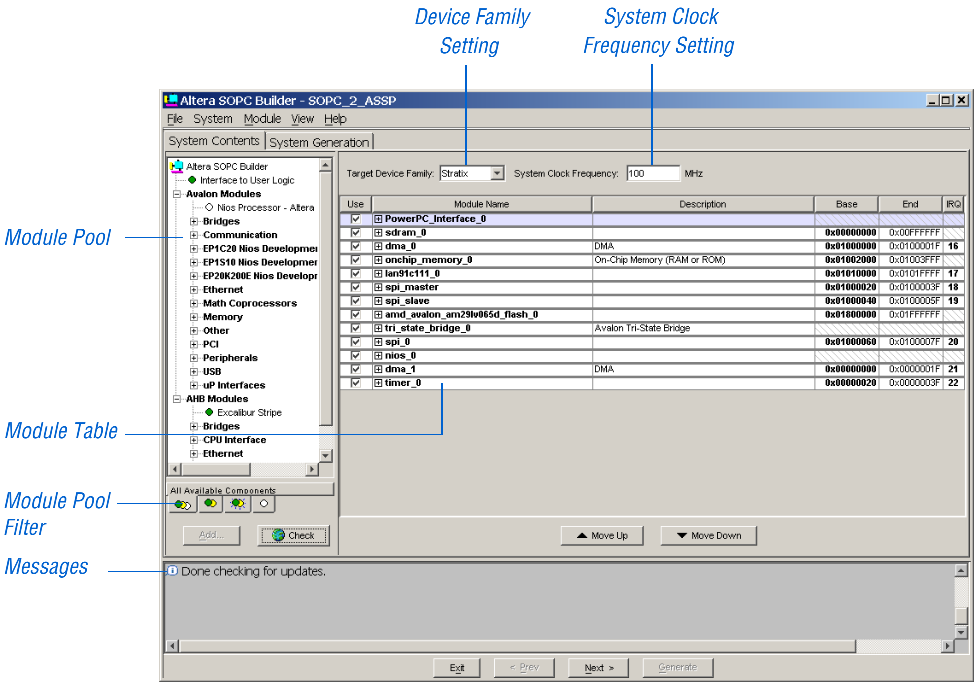
After you open a Quartus II project, launch the SOPC Builder user interface by choosing **SOPC Builder** (Tools menu) in the Quartus II software. The SOPC Builder user interface contains the following pages:

- **System Contents** page
- System dependency page(s)
- **System Generation** page

System Contents Page

This page is where you define your system. It includes a listing of all available library components in the *module pool* and displays all of the components that you have added to your system in the *module table*. When you generate your system with SOPC Builder, it creates a single system module that includes components and interfaces you specified. Additionally, this single system module contains automatically generated bus (interconnection) logic. Figure 1 shows the **System Contents** page.

Figure 1. System Contents Page



Target Device Family: Stratix

System Clock Frequency: 100 MHz

Use	Module Name	Description	Base	End	IRQ
☑	⊞ PowerPC_Interface_0				
☑	⊞ sdram_0		0x00000000	0x00FFFFFF	
☑	⊞ dma_0	DMA	0x01000000	0x0100001F	16
☑	⊞ onchip_memory_0	On-Chip Memory (RAM or ROM)	0x01002000	0x01003FFF	
☑	⊞ lan91c111_0		0x01010000	0x0101FFFF	17
☑	⊞ spi_master		0x01000020	0x0100003F	18
☑	⊞ spi_slave		0x01000040	0x0100005F	19
☑	⊞ amd_avalon_am29lv065d_flash_0		0x01800000	0x01FFFFFF	
☑	⊞ tri_state_bridge_0	Avalon Tri-State Bridge			
☑	⊞ spi_0		0x01000060	0x0100007F	20
☑	⊞ nios_0				
☑	⊞ dma_1	DMA	0x00000000	0x0000001F	21
☑	⊞ timer_0		0x00000020	0x0000003F	22

Done checking for updates.

☞ Right-click messages to go to the source of the error or message.

Module Pool

The module pool shows all available library components organized according to bus type and category. Each component appears with a colored dot next to its name:

- *Green dot*—Indicates fully licensed components that you can add to your system.
- *Yellow dot*—Indicates that a component may be evaluated in a system design in some limited form, typically subject to a hard timeout or reduced functionality.
- *White dot*—Indicates SOPC Builder Ready components from Altera and other vendors that are not currently installed on your system. You can download these components from the web.

You can use the module pool filter to display available components, installed components, web-available components, and components with web-available updates. Additionally, if you have an Internet connection, you can update the view to show new components from Altera and Altera Megafunction Partners Program (AMPP^{SM}) partners as they become available.

✓ To view new components, click the **Check** button. Newly available components are shown on the updated page. You can then click the component name and click **Add** to be directed to the location for downloading the component.

Right-click components to view a menu with component details, links to documentation, and links to updates for that component. If you right-click an installed component for which an update is available, the pop-up menu contains a submenu (with the component's version number as the title) that provides links specific to that update.

Module Table

The module table is where you add components to your system, including bridges, bus interfaces, CPUs, memory interfaces, peripherals, etc. Additionally, you use the module table to describe the following elements:

- Masters and slave connectivity
- System address map
- System IRQ assignments
- Arbitration priorities for shared slaves

To add a component to the module table, perform the following steps:

1. Click the component name in the module pool.

2. Click **Add**. One of two things happens:

   – For available, installed components with additional settings, a wizard appears for you to specify options. Make the desired settings and click **Finish** to add the component to the module table. If the component has no wizard, it is added to the module table.

   – For available, uninstalled components, an information dialog box appears with a link to download the component from the web or to request it from the vendor. After you install the component, you can add it to your system.

If **Show Master Connections** (View menu) is turned on, the left side of the module table displays interconnections between master and slave components. Any component can have one or more master or slave ports. Any master can be connected to any slave if both master and slave use the same bus protocol. If the master and the slave use different bus protocols, you can still connect them by using a bridge component, such as the AMBA-AHB-to-Avalon™ Bridge.

Whenever two, or more, masters share (i.e., have access to) the same slave, SOPC Builder automatically inserts an arbiter to control access to the slave. The arbiter determines which master is granted access to the slave when simultaneous requests occur.

✓ To view arbitration priorities, choose **Show Arbitration Priorities** (View menu).

For more information on master/slave connections and arbitration priorities, refer to *AN 184: Simultaneous Multi-Mastering with the Avalon Bus*.

Additional Settings

The **System Contents** page includes the following additional options:

- *Device Family*—You should choose your target device family from the **Target Device Family** list. This setting is important because when SOPC Builder generates logic for the system, it uses this setting to take advantage of the device family's architectural features.

> ☞ The Quartus II software does not use this device family setting. You must also specify it in the Quartus II software.

- *System Clock Frequency*—Peripherals use the system clock frequency to generate clock dividers/baud rate generators, etc. SOPC Builder's built-in testbench generator also uses the settings to make a clock of the requested frequency.

> ☞ You must set the clock frequency to match the frequency you plan to use on your board.

> ☞ The Quartus II Timing Analyzer does not use this system clock frequency setting. You must also specify it in the Quartus II software.

System Dependency Page(s)

When you add certain components to your system, such as a CPU like the Nios embedded processor, an additional page appears in SOPC Builder. These pages allow you to set additional parameters or associations of the component with respect to the other components in the system. For example, you can specify the relationship between a CPU and the memory components to indicate which is used as the program memory and which is used as data memory.. For components that use system dependency pages, a separate system dependency page is created for each instance of the component that you add to your system.

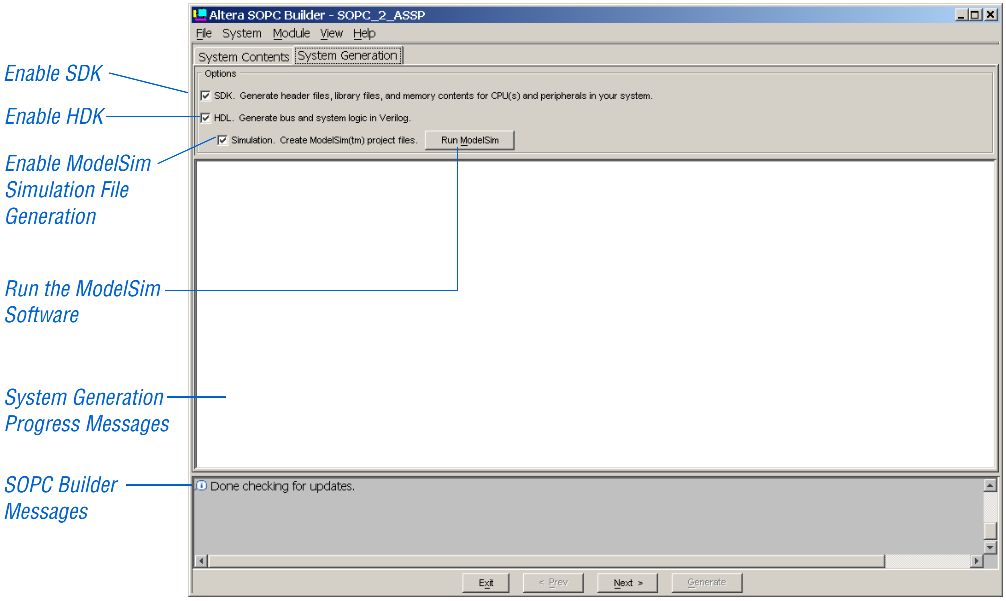
Additionally, processor components may have associated software components that are shown on the system dependency pages. Examples of software components range from utility libraries to real-time-operating systems (RTOSs). Altera provides several software components in development kits, such as the Plugs Library (a lightweight, full-featured TCP/IP protocol stack) that comes with the Nios development kits.

Refer to the Nios hardware development tutorials for information on how to use the system dependency page(s). See "Additional Information" on page 18 for a listing of available tutorials.

System Generation Page

This page is where you generate your system. It includes options you can set to control the generation process such as device family support and simulation. During system generation, this page reports the system generation progress message(s). Figure 2 shows the **System Generation** page.

Figure 2. System Generation Page



☑ SDK. Generate header files, library files, and memory contents for CPU(s) and peripherals in your system.

☑ HDL. Generate bus and system logic in Verilog.

☑ Simulation. Create ModelSim(tm) project files.

Done checking for updates.

Enable SDK

Enable HDK

Enable ModelSim Simulation File Generation

Run the ModelSim Software

System Generation Progress Messages

SOPC Builder Messages

SDK

When you turn on the **SDK** option, SOPC Builder creates a custom SDK for each CPU in your system every time you generate the system. This SDK contains software files (drivers, libraries, and utilities) for any system components that provide software support in their library definition. Processor components, such as the Nios embedded processor and Excalibur^{TM} devices, can build software applications as part of the generation process.

The software files are arranged into the following directories:

- **inc**—This directory contains header files with the definitions of memory maps, register declarations for the peripherals, and macros that you can use to create embedded software applications.
- **lib**—This directory contains library files. If the component supports GNU tools, SOPC Builder compiles the libraries during system generation.
- **src**—This directory contains the source code. You can modify the source code and write software for the system using any text editor. You can also use the Quartus II Text Editor, which supports syntax

coloring for C and C++ source code. Altera recommends that you save your user source code in one of the following locations:

- **src** directory
- A subdirectory of the **src** directory
- A directory on the same level as the **src** directory in the SDK

If you work with software engineers who do not have access to the Quartus II software, you should provide them with the SDK files associated with the SOPC Builder-created system.

☞ By default, the **SDK** option is turned on.

HDL

When you turn on the **HDL** option, SOPC Builder generates a system-level hardware description language (HDL) file in Verilog HDL or VHDL, depending on which language you specified when you first set up the system in SOPC Builder. The HDL file contains:

- An instance of every component in the system
- Bus logic to interconnect the components, including the following items:
  - Address decoders
  - Data bus multiplexers
  - Arbiters for shared resources
  - Reset-generation and conditioning logic
  - Interrupt prioritization logic
  - Dynamic bus sizing (for adapting masters to slaves with wider or narrower data buses)
  - Passive interconnections between master and slave ports
- A simulation testbench that:
  - Instantiates the system module
  - Drives clock and reset inputs with default behaviors
  - Instantiates and connects any simulation models for system external components if provided (e.g., memory models)

☞ By default, the **HDL** option is turned on.

Simulation

If you turn on the **Simulation** option, during system generation SOPC Builder creates a ModelSim® project directory that includes the following files:

- Simulation data files for all memory components that have specific contents
- A **setup_sim.do** file that contains setup information and aliases customized for simulating the generated system
- A **wave_presets.do** file that has an initial set of bus interface waveforms
- A ModelSim Project File (**.mpf**) for the current system

You can run the ModelSim software from within SOPC Builder by clicking the **Run ModelSim** button. If the ModelSim software is not in your path, you must specify its location using **SOPC Builder Setup** (File menu).

By default, the **Simulation** option is turned on.

To use simulators other than the ModelSim simulator, you can use the SOPC Builder-generated HDL files. However, you must set up the project and perform other simulation tasks instead of using the SOPC Builder-generated Tcl script. You may want to review the ModelSim files for ideas on setting up your simulation project.

Generating a System

After you have built your system and specified generation options, you generate the system by clicking the **Generate** button. This button is available from any page in the SOPC Builder user interface. When you click **Generate**, SOPC Builder creates the following items:

- The SDK
- HDL files for each component in the system
- A Block Symbol File (**.bsf**) for the top-level system module
- ModelSim files
- A Tcl script that sets up all of the files needed for Quartus II compilation

As the generation process proceeds, SOPC Builder displays messages and information in the system generation progress messages box. SOPC Builder displays the message “Generation Complete” when it finishes generating the system and places a log file in the root directory of the project you generated.

SOPC Builder Menu Commands

Table 1 describes the menu commands available in SOPC Builder.

Table 1. SOPC Builder Menu Commands (Part 1 of 2)

Menu	Command	Description
File	**New**	Create a new SOPC Builder system file in the current project directory.
	Open	Open an SOPC Builder project, <*filename*>**.ptf**.
	Browse Project Directory	Open a browse window for the project directory.
	SOPC Builder Setup	Open the **SOPC Builder Setup** dialog box. You can set paths and other preferences for SOPC Builder in this dialog box, for example, alternate paths for component libraries and Internet connectivity setup.
	Add Library Component	Choose this option to import your user-defined logic as a component.
	Exit	Close SOPC Builder.
System	**Add Interface to User Logic**	Opens the **Interface to User Logic -** <*component name*> dialog box. In this dialog box, you specify the settings for the logic you want to add to your system, including the ports, instantiation, and timing.
	Auto-Assign Base Addresses	Choose this option to have SOPC Builder assign base addresses to the components in the system.
	Auto-Assign IRQs	Choose this option to have SOPC Builder assign IRQs to the components in the system.
Module	**Edit**	Choose this option after selecting a component in the module table to open the component's wizard.
	Rename	Choose this option after selecting a component in the module table to rename the component.
	Delete	Choose this option after selecting a component in the module table to delete the component.
	Lock Base Address	Choose this option to lock a component to a specific base address.
View	**Show Master Connections**	Turn on this option to view the master/slave connection patch panel of the components in the module table.
	Show Arbitration Priorities	Turn on this option to view the master/slave arbitration priorities of the components in the module table.
	Show Bus Type	Turn on this option to view the bus type of the components in the module table.

Table 1. SOPC Builder Menu Commands (Part 2 of 2)

Menu	Command	Description
Help	**Accessing Help & Documentation**	Describes how to get pop-up help for components in SOPC Builder.
	Documentation Library Index	Opens the *Documentation Index* in Adobe PDF.
	Read Me (Release Notes)	Opens the SOPC Builder readme text file.
	FAQ (Frequently Asked Questions)	Opens the SOPC Builder FAQ text file.
	Installed Documentation	Opens a browse window in the documentation directory.
	Online Documentation	Opens a web browser to the SOPC Builder literature page on the Altera web site.
	Installed Tutorials	Opens a browse window to the SOPC Builder tutorials directory.
	Online Tutorials	Opens a web browser to the SOPC Builder literature page on the Altera web site, which also contains links to tutorial files.
	Altera Home Page	Opens a web browser to the Altera home page.
	SOPC Builder Home Page	Opens a web browser to the SOPC Builder page on the Altera web site.
	About Altera SOPC Builder	Provides release information and legal notices for SOPC Builder.

Next Steps

After you generate the system, you can compile it in the Quartus II software to target an FPGA or you can include the system in a larger design. In addition, you can develop application software for the system. Refer to the following sections for more information.

Additional Information

For more information on using SOPC Builder, refer to the following documents:

- *Avalon Bus Specification Reference Manual*—This reference manual is for developers creating custom peripherals for the Avalon bus. It defines terms and concepts of SOPC designs based on the Avalon bus architecture used for connecting on-chip processors and peripherals into a system on a programmable chip (SOPC). Avalon bus signal functions and timing are defined.

- *Using SOPC Builder with Excalibur Devices Tutorial*—This tutorial introduces you to the Excalibur devices used with the SOPC Builder. It shows you how to use SOPC Builder and the Quartus II software to create and process your own Excalibur device system module design that interfaces with components provided on the Excalibur development board.

- *Nios Hardware Development Tutorial*—This tutorial shows you how to use SOPC Builder to build a Nios processor-based system targeted for the Nios development board, Cyclone edition; Nios development board, Stratix edition; and the Nios development board, Stratix professional edition.

- *Nios Tutorial (APEX Device)*—This tutorial shows you how to use SOPC Builder to build a Nios processor-based system targeted for the Nios development board (APEX device).

- *Nios Software Development Tutorial*—This tutorial shows you how to compile code, download code and debug a Nios processor system on the Nios development board.

- *Simultaneous Multi-Mastering with the Nios Processor Tutorial*—This tutorial describes how to optimize an embedded system's performance using the simultaneous multi-master bus architecture. It describes the new features in the SOPC Builder software that allow you to customize a system bus architecture easily and shows you how to use the SOPC Builder software to define a custom bus architecture to improve the example design's performance.

- *Nios Custom Instructions Tutorial*—This tutorial introduces you to custom instruction using the Nios embedded processor. The tutorial guides you through the steps for implementing two example custom instructions in a Nios system module and then describes how to access these custom instructions through software.

- Refer to Quartus II help for additional information on how to use the software.

- *SOPC Builder PTF File Reference Manual*—This reference manual is for IP developers who wish to create new library components for SOPC Builder. This manual contains reference material on the internal workings of the PTF file structure and the development phases of SOPC Builder. This manual is recommended for advanced system designers with basic familiarity of the SOPC Builder tool.

You can also find more information on the Literature pages on the Altera web site.

SOPC Builder Ready Cores

Altera awards the SOPC Builder Ready certification to IP cores available from Altera and AMPP partners that have plug-and-play integration with the Excalibur devices or the Nios embedded processors via the SOPC Builder. These cores support interfaces for the Avalon bus for the Nios processor or AMBA AHB on-chip bus, and may include software drivers, low-level routines, or other software design files.

Altera offers a “free test drive” of IP cores through the Quartus II software’s OpenCore® evaluation feature. The OpenCore Plus feature set supplements the OpenCore evaluation flow by incorporating free RTL simulation and hardware evaluation. These evaluations feature synthesizable functions that can be parameterized, compiled, and simulated using Altera’s development tools. You can quickly and easily verify the functionality of a cores, as well as evaluate its size and speed before making a purchase decision. You only need to purchase a license when they are completely satisfied with a core’s functionality and performance and would like to write a device programming file.

You can download OpenCore evaluation cores directly from Altera’s web site. Many AMPP partners support the OpenCore evaluation flow. Contact AMPP partners directly to obtain an AMPP OpenCore evaluation. SOPC Builder lists SOPC Builder Ready cores in the module pool, and provides links to where you can download them from the web or request an evaluation.