

Introduction

The Quartus II Logic Analyzer Interface (LAI) allows you to examine the behavior of internal signals using an external logic analyzer and using a minimal number of FPGA I/O pins, while your design is running at full speed on your FPGA.



This chapter's use of the term "logic analyzer" includes both logic analyzers and oscilloscopes equipped with digital channels, commonly referred to as mixed signal analyzers or MSOs.

The LAI connects a large set of internal device signals to a small number of output pins. You can connect these output pins to an external logic analyzer for debugging purposes. In the Quartus II LAI, the internal signals are grouped together, distributed to a user-configurable multiplexer, and then output to available I/O pins on your FPGA. Instead of having a one-to-one relationship between internal signals to output pins, the Quartus II LAI enables you map many internal signals to a smaller number of output pins. The exact number of internal signals that you can map to an output pin varies based on the multiplexer settings in the Quartus II LAI.

This chapter details the following topics:

- ["Choosing a Logic Analyzer"](#)
- ["Debugging Your Design Using the Logic Analyzer Interface" on page 15-3](#)
- ["Advanced Features" on page 15-11](#)

Choosing a Logic Analyzer


The Quartus II software offers the following two general purpose on-chip debugging tools for debugging a large set of RTL signals from your design:

- The embedded SignalTap® II logic analyzer
- An external logic analyzer, which connects to internal signals in your FPGA, by using the Quartus II LAI

[Table 15-1](#) describes the advantages to each debugging technologies.

Table 15-1. Comparing the SignalTap II Embedded Logic Analyzer with the Logic Analyzer Interface

Feature and Description	Logic Analyzer Interface	SignalTap II Embedded Logic Analyzer
Sample Depth You have access to a wider sample depth with an external logic analyzer. In the SignalTap II Embedded Logic Analyzer, the maximum sample depth is set to 128 Kb, which is a device constraint. However, with an external logic analyzer, there are no device constraints, providing you a wider sample depth.	✓	—
Debugging Timing Issues Using an external logic analyzer provides you with access to a “timing” mode, which enables you to debug combined streams of data.	✓	—
Performance You frequently have limited routing resources available to place-and-route when you use SignalTap II with your design. An external logic analyzer adds minimal logic, which removes resource limits on place-and-route.	✓	—
Triggering Capability SignalTap II offers triggering capabilities that is comparable with external logic analyzers.	✓	✓
Use of Output Pins Using the SignalTap II Logic Analyzer, no additional output pins are required. Using an external logic analyzer requires the use of additional output pins.	—	✓
Acquisition Speed With the SignalTap II Logic Analyzer, you can acquire data at speeds of over 200 MHz. You can achieve the same acquisition speeds with an external logic analyzer; however, you must consider signal integrity issues.	—	✓

 The Quartus II software offers a portfolio of on-chip debugging tools. For an overview and comparison of all tools available in the Quartus II software on-chip debugging tool suite, refer to *Section V. In-System Debugging* in volume 3 of the *Quartus II Handbook*.

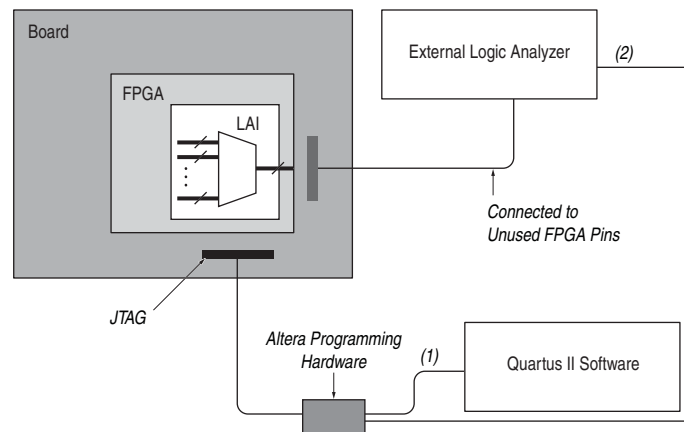
Required Components

You must have the following components to perform analysis using the Quartus II LAI:

- The Quartus II software starting with version 5.1 and later
- The device under test
- An external logic analyzer
- An Altera® communications cable
- A cable to connect the FPGA to the external logic analyzer

Figure 15-1 shows the LAI and the hardware setup.

Figure 15-1. Logic Analyzer Interface and Hardware Setup



Notes to Figure 15-1:

- (1) Configuration and control of the LAI using a computer loaded with the Quartus II software via the JTAG port.
- (2) Configuration and control of the LAI using a third-party vendor logic analyzer via the JTAG port. Support varies by vendor.

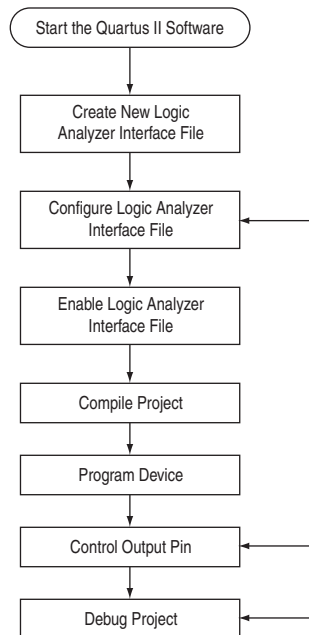
FPGA Device Support

You can use the Quartus II Logic Analyzer Interface (LAI) with the following FPGA device families:

- Arria® GX
- Stratix® series
- Cyclone® series
- MAX® II
- APEX™ 20K
- APEX II

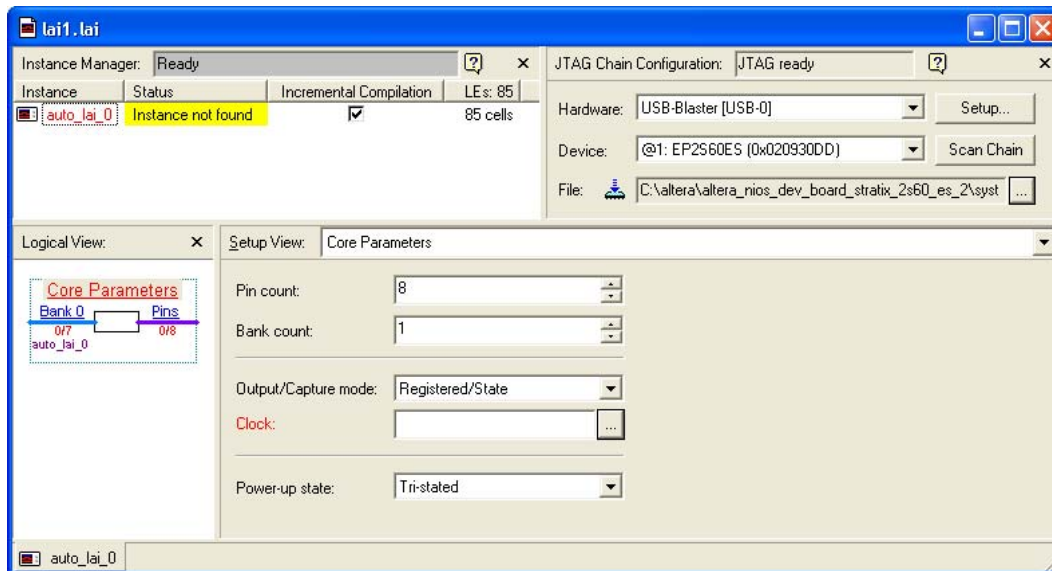
Debugging Your Design Using the Logic Analyzer Interface

Figure 15-2 shows the steps you must follow to debug your design with the Quartus II LAI.

Figure 15-2. LAI Process Flow

Creating an LAI File

The Logic Analyzer Interface (**.lai**) file defines the interface that builds a connection between internal FPGA signals and your external logic analyzer. [Figure 15-3](#) shows an example of an **.lai** editor.

Figure 15-3. The LAI Editor

To define the Quartus II LAI, you can create a new **.lai** file or use an existing **.lai** file.

Creating a New Logic Analyzer Interface File

To create a new .lai file, perform the following steps:

1. In the Quartus II software, on the File menu, click **New**. The **New** dialog box opens.
2. Click the **Other Files** tab.
3. Select **Logic Analyzer Interface File**.
4. Click **OK**. The LAI editor opens. The file name is assigned by the Quartus II software (refer to [Figure 15-3 on page 15-4](#)). When you save the file, you will be prompted for a file name. Refer to [“Saving the External Analyzer Interface File” on page 15-5](#).

Opening an Existing External Analyzer Interface File

To open an existing .lai file, on the Tools menu, click **Logic Analyzer Interface Editor**. If no .lai file is enabled for the current project, the editor automatically creates a new .lai file. If an .lai file is currently enabled for the project, that file opens when you select the **Logic Analyzer Interface Editor**.

Alternatively, on the File menu, click **Open**, and select the .lai file you want to open.

Saving the External Analyzer Interface File

To save your .lai file, perform the following steps:

1. In the Quartus II software, on the File menu, click **Save As**. The **Save As** dialog box opens.
2. In the **File name** box, enter the desired file name.
3. Click **Save**.

Configuring the Logic Analyzer Interface File Core Parameters

After you have created the .lai file, you must configure the .lai file core parameters.

To configure the .lai file core parameters, from the **Setup View** list, select **Core Parameters**. Refer to [Figure 15-4](#).

Figure 15-4. Logic Analyzer Interface File Core Parameters

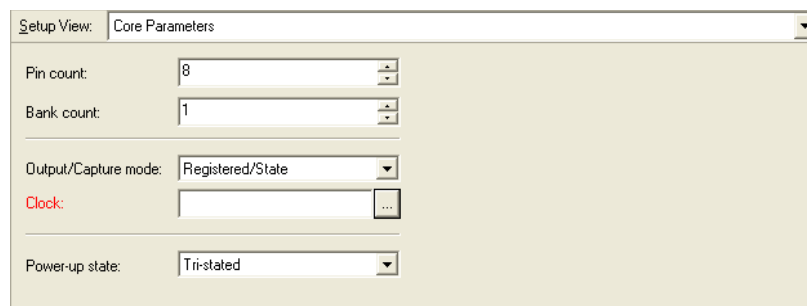


Table 15-2 lists the .lai file core parameters.

Table 15-2. Logic Analyzer Interface File Core Parameters

Parameter	Description
Pin Count	The Pin Count parameter signifies the number of pins you want dedicated to your LAI. The pins must be connected to a debug header on your board. Within the FPGA, each pin is mapped to a user-configurable number of internal signals. The Pin Count parameter can range from 1 to 255 pins.
Bank Count	The Bank Count parameter signifies the number of internal signals that you want to map to each pin. For example, a Bank Count of 8 implies that you will connect eight internal signals to each pin. The Bank Count parameter can range from 1 to 255 banks.
Output/Capture Mode	The Output/Capture Mode parameter signifies the type of acquisition you perform. There are two options that you can select: Combinational/Timing —This acquisition uses your external logic analyzer’s internal clock to determine when to sample data. Because Combinational/Timing acquisition samples data asynchronously to your FPGA, you must determine the sample frequency you should use to debug and verify your system. This mode is effective if you want to measure timing information, such as channel-to-channel skew. For more information about the sampling frequency and the speeds at which it can run, refer to the data sheet for your external logic analyzer. Registered/State —This acquisition uses a signal from your system under test to determine when to sample. Because Registered/State acquisition samples data synchronously with your FPGA, it provides you with a functional view of your FPGA while it is running. This mode is effective when you verify the functionality of your design.
Clock	The clock parameter is available only when Output/Capture Mode is set to Registered State. You must specify the sample clock in the Core Parameters view. The sample clock can be any signal in your design. However, for best results, Altera recommends that you use a clock with an operating frequency fast enough to sample the data you would like to acquire.
Power-Up State	The Power-Up State parameter specifies the power-up state of the pins you have designated for use with the LAI. You have the option of selecting tri-stated for all pins, or selecting a particular bank that you have enabled.


Mapping the Logic Analyzer Interface File Pins to Available I/O Pins

To configure the .lai file I/O pins parameters, select Pins from the Setup View list (Figure 15-5).

Figure 15-5. Logic Analyzer Interface File Pins Parameters

Pin				I/O Standard
Type	Index	Name	Location	
	0	altera_reserved_lai_0_0		
	1	altera_reserved_lai_0_1		
	2	altera_reserved_lai_0_2		
	3	altera_reserved_lai_0_3		
	4	altera_reserved_lai_0_4		
	5	altera_reserved_lai_0_5		
	6	altera_reserved_lai_0_6		
	7	altera_reserved_lai_0_7		









To assign pin locations for the LAI, double-click the **Location** column next to the reserved pins in the **Name** column. This opens the Pin Planner.

 For information about how to use the Pin Planner, refer to the *Pin Planner* section in the *I/O Management* chapter in volume 2 of the *Quartus II Handbook*.

Mapping Internal Signals to the Logic Analyzer Interface Banks

















After you have specified the number of banks to use in the **Core Parameters** settings page, you must assign internal signals for each bank in the LAI. Click the **Setup View** arrow and select **Bank n** or **All Banks** (Figure 15-6).

Figure 15-6. Logic Analyzer Interface Bank Parameters

Setup View: Bank 0				
Pin Index	Node			
	Type	Alias	Name	
0		State Clock	<input type="text" value=""/>	
1			<input type="text" value=""/>	
2			<input type="text" value=""/>	
3			<input type="text" value=""/>	
4			<input type="text" value=""/>	
5			<input type="text" value=""/>	
6			<input type="text" value=""/>	
7			<input type="text" value=""/>	

To view all of your bank connections, click **Setup View** and select **All Banks** (Figure 15-7).

Figure 15-7. Logic Analyzer Interface All Bank Parameters

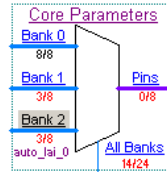
Setup View: All Banks				
Bank Name	Pin Index	Node		
		Type	Alias	Name
Bank 0	0			<input type="text" value="c[7]"/>
	1			<input type="text" value="c[6]"/>
	2			<input type="text" value="c[5]"/>
	3			<input type="text" value="c[4]"/>
	4			<input type="text" value="c[3]"/>
	5			<input type="text" value="c[2]"/>
	6			<input type="text" value="c[1]"/>
	7			<input type="text" value="c[0]"/>
Bank 1	0			<input type="text" value=""/>
	1			<input type="text" value=""/>
	2			<input type="text" value=""/>
	3			<input type="text" value=""/>
	4			<input type="text" value=""/>
	5			<input type="text" value=""/>
	6			<input type="text" value=""/>
	7			<input type="text" value=""/>

Using the Node Finder

Before making bank assignments, on the View menu, point to **Utility Windows** and click **Node Finder**. Find the signals that you want to acquire, then drag and drop the signals from the **Node Finder** dialog box into the bank **Setup View**. When adding signals, use **SignalTap II: pre-synthesis** for non-incrementally routed instances and **SignalTap II: post-fitting** for incrementally routed instances.

As you continue to make assignments in the bank **Setup View**, the schematic of your LAI in the **Logical View** of your **.jai** file begins to reflect your assignments (Figure 15-8).

Figure 15-8. A Logical View of the Logic Analyzer Interface Schematic



Continue making assignments for each bank in the **Setup View** until you have added all of the internal signals for which you wish to acquire data.



You can right-click to switch between the LAI schematic and the LAI **Setup** view.

Enabling the Logic Analyzer Interface Before Compiling Your Quartus II Project

Compile your project after you have completed the following steps:

- Configure your LAI parameters
- Map the LAI pins to available I/O pins
- Map the internal signals to the LAI banks

Compiling Your Quartus II Project

Before compiling your project, you must enable the LAI.

1. On the Assignments menu, click **Settings**. The **Settings** dialog box opens.
2. Under Category, click **Logic Analyzer Interface**. The **Logic Analyzer Interface** displays.
3. Turn on **Enable Logic Analyzer Interface**.
4. Click **Logic Analyzer Interface file name** and specify the full path name to your **.jai** file.

After you have specified the name of your **.jai** file, you must compile your project. To compile your project, on the Processing menu, click **Start Compilation**.

To ensure that the LAI is properly compiled with your project, expand the entity hierarchy in the Project Navigator. (To display the Project Navigator, on the View menu, point to **Utility Windows** and click **Project Navigator**.) If the LAI is compiled with your design, the **sld_hub** and **sld_multitap** entities are shown in the project navigator (Figure 15-9).

Figure 15-9. Project Navigator

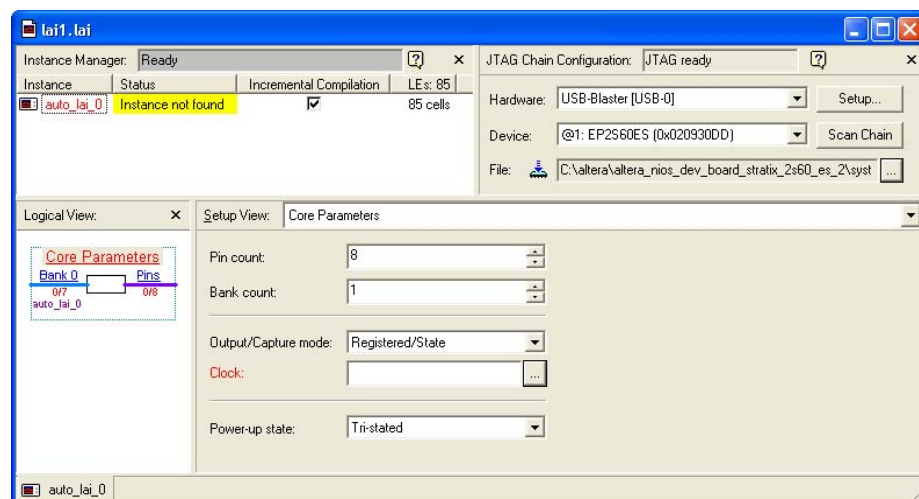
Entity	Logic Cells	LC Registers
Stratix: EP1S10B672C7		
test	136 (1)	81
sld_multitap:auto_lai_0	35 (11)	15
sld_hub:sld_hub_inst	100 (25)	65

Programming Your FPGA Using the Logic Analyzer Interface

After compilation completes, you must configure your FPGA before using the LAI. To configure a device for use with the LAI, perform the following steps:

1. Open the .lai file Editor (Figure 15-10).
2. Under **JTAG Chain Configuration**, click **Hardware** and select your hardware communications device. You may have to click **Settings** to configure your hardware.
3. Click **Device** and select the FPGA device to which you want to download the design (it may be automatically detected). You may have to click **Scan Chain** to configure your device.
4. Click **File** and select the SRAM Object File (.sof) that includes the .lai file (it may be automatically detected).
5. If desired, turn on **Incremental Compilation**.
6. Save the .lai file.
7. Click the **Program Device** icon to program the device.


Figure 15-10. The JTAG Section of the Logic Analyzer Interface File



Using the Logic Analyzer Interface with Multiple Devices

You can use the LAI with multiple devices in your JTAG chain. Your JTAG chain can also consist of devices that do not support the LAI or non-Altera, JTAG-compliant devices. To use the LAI in more than one FPGA, create an LAI and configure an `.lai` file for each FPGA that you want to analyze. To perform multi-FPGA analysis, perform the following steps:

1. Open the Quartus II software.
2. Create, configure, and compile an `.lai` file for each design.
3. Open one `.lai` file at a time.

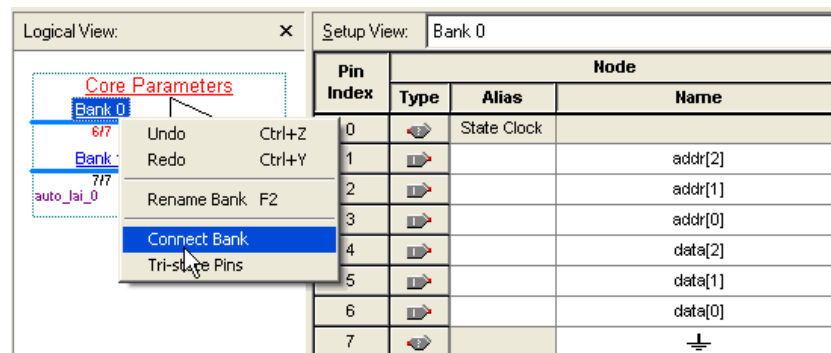
 You do not have to open a Quartus II project to open an `.lai` file.

4. Follow Steps 2 through 6 under “Programming Your FPGA Using the Logic Analyzer Interface” on page 15-9.
5. Click the **Program Device** icon to program the device.
6. Control each `.lai` file independently.

Configuring Banks in the Logic Analyzer Interface File


When you have programmed your FPGA, you can control which bank is mapped to the reserved `.lai` file output pins. To control which bank is mapped, in the schematic in the logical view, right-click the bank and click **Connect Bank** (Figure 15-11).

Figure 15-11. Configuring Banks



Acquiring Data on Your Logic Analyzer

To acquire data on your logic analyzer, you must establish a connection between your device and the external logic analyzer.

 For more information about this process and for guidelines on how to establish connections between debugging headers and logic analyzers, refer to the documentation for your logic analyzer.

Advanced Features

This section describes the following advanced features:

- [Using the Logic Analyzer Interface with Incremental Compilation](#)
- [Creating Multiple Logic Analyzer Interface Instances in One FPGA](#)

Using the Logic Analyzer Interface with Incremental Compilation

Using the LAI with Incremental Compilation enables you to preserve the synthesis and fitting of your original design, and add the LAI to your design without recompiling your original source code.

To use the LAI with Incremental Compilation, perform the following steps:

1. Start the Quartus II software.
2. Enable Design Partitions. To enable Partitions, perform the following steps:
 - a. On the Assignments menu, click **Design Partitions**.
 - b. In the **Incremental Compilation** list, select **Full Incremental Compilation**.
 - c. Create Design Partitions for the entities in your design, and set the Netlist Type to **Post-fit**.
 - d. On the Processing menu, click **Start Compilation**.
3. Enable LAI Incremental Compilation by performing these steps:
 - a. In your **.lai** file, under **Instance Manager**, click **Incremental Compilation**.



When you enable **Incremental Compilation**, all existing presynthesis signals are converted into post-fitting signals. Only post-fitting signals can be used with the LAI with Incremental Compilation.

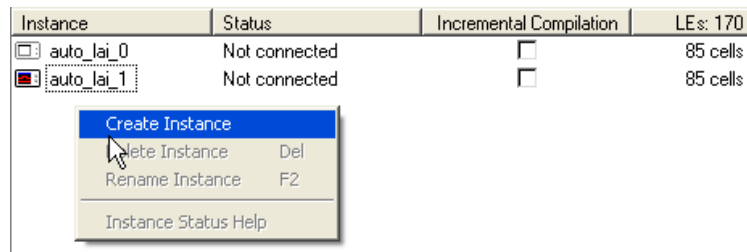
- b. Add Post-Fitting nodes to your **.lai** file.
- c. On the Processing menu, click **Start Compilation**.

Creating Multiple Logic Analyzer Interface Instances in One FPGA

The LAI includes support for multiple interfaces in one FPGA. This feature is particularly useful when you want to build LAI configurations that contain different settings. For example, you can build one LAI instance to perform Registered/State analysis and build another instance that performs Combinational/Timing analysis on the same set of signals.

Another example would be performing Registered/State analysis on portions of your design that are in different clock domains.

To create multiple LAIs, on the Edit menu, click **Create Instance**. Alternatively, you can right-click the **Instance Manager** window, and click **Create Instance** (Figure 15-12).

Figure 15-12. Creating Multiple Logic Analyzer Interface Instances in One FPGA

Conclusion

As the FPGA industry continues to make technological advancements, outdated debugging methodologies must be replaced with new technologies that maximize productivity. The LAI feature enables you to connect many internal signals within your FPGA to an external logic analyzer with the use of a small number of I/O pins. This new technology in the Quartus II software enables you to use feature-rich external logic analyzers to debug your FPGA design, ultimately enabling you to deliver your product in the shortest amount of time.

Referenced Documents

This chapter references the following documents:


- [Section V. In-System Debugging](#) in volume 3 of the *Quartus II Handbook*
- [I/O Management](#) chapter in volume 2 of the *Quartus II Handbook*

Document Revision History

Table 15-3 shows the revision history for this chapter.

Table 15-3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
March 2009 v9.0.0	Minor editorial updates. Removed Figures 15-4, 15-5, and 15-11 from 8.1 version.	Updated for the Quartus II software version 9.0 release.
November 2008 v8.1.0	Changed to 8-1/2 x 11 page size. No change to content.	Updated for the Quartus II software version 8.1 release.
May 2008 v8.0.0	<ul style="list-style-type: none"> ■ Updated device support list on page 15-3 ■ Added links to referenced documents throughout the chapter ■ Added “Referenced Documents” ■ Added reference to Section V. In-System Debugging ■ Minor editorial updates 	Updated for the Quartus II software version 8.0 release.

 For previous versions of the *Quartus II Handbook*, refer to the [Quartus II Handbook Archive](#).