

Introduction

Hardware verification can be a lengthy and expensive process. The SignalProbe™ incremental routing feature can help reduce the hardware verification process and time-to-market for System-On-a-Programmable-Chip (SOPC) designs.

Easy access to internal device signals is important in the debugging of a design. The SignalProbe feature enables efficient design verification by allowing you to quickly route internal signals to I/O pins without affecting the design. Starting with a fully routed design, you can select and route signals for debugging to either previously reserved or currently unused I/O pins.

The SignalProbe™ feature supports the Stratix™, Stratix GX, Cyclone™, APEX™ II, APEX 20KE, APEX 20KC, APEX 20K, and Excalibur™ devices.

Using SignalProbe

You can use the SignalProbe compilation to incrementally route internal signals to reserved output pins. This process completes in a fraction of the time required by a full design recompilation. The incremental routing does not affect source behavior or design operation.

Follow the steps below to use the SignalProbe incremental routing feature:

1. Reserve SignalProbe pins prior to initial compilation.
2. After initial compilation, determine which nodes you want to route to the reserved SignalProbe pins.
3. Assign an I/O standard to the SignalProbe pins.
4. Add registers for pipelining of signals, if necessary.
5. Perform a SignalProbe compilation.
6. Understand the results of the SignalProbe compilation.

Reserving SignalProbe pins

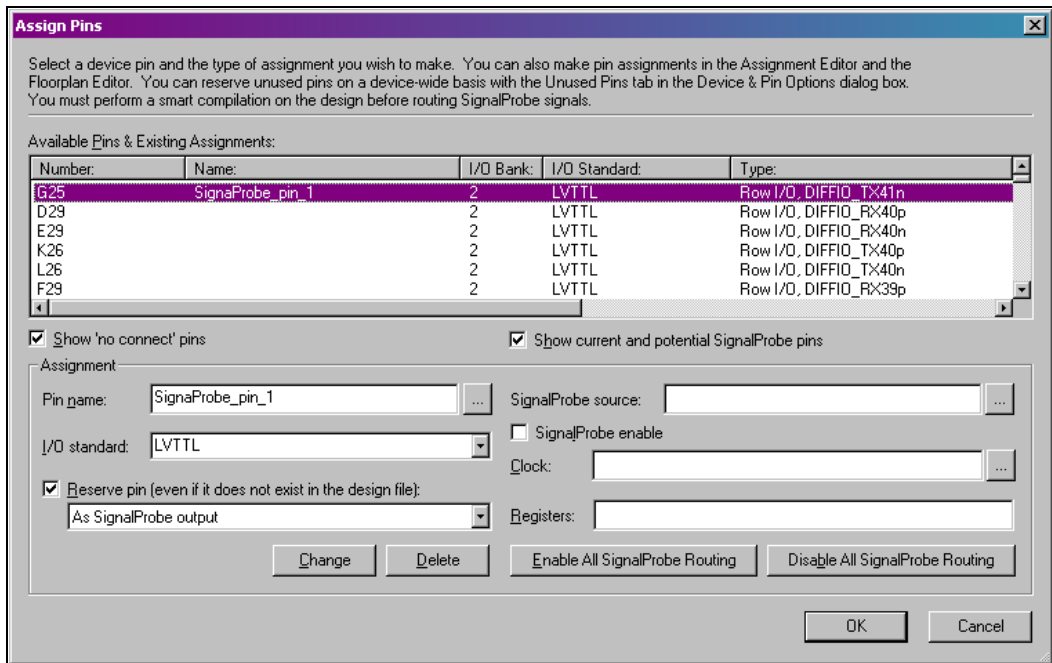
You can reserve an unused pin as a SignalProbe pin before you route an internal signal out of your device. You can reserve your SignalProbe pins before or after a compilation. To ensure that a pin is available for your SignalProbe pin and not to another unassigned user I/O pin, reserve the SignalProbe pin before a compilation.

You may only need a few SignalProbe pins, since you can easily reassign different sources to your SignalProbe pins.

To reserve an unused I/O pin as a SignalProbe pin, perform the following steps:

1. Choose **Assign Pins** (Assignment menu).
2. Turn on **Show current and potential SignalProbe** pins in the **Assign Pins** dialog box.
3. Select a pin **Number** from the **Available Pins & Existing Assignments** list.
4. Type your SignalProbe pin name into the **Pin name** box.
5. Select **As SignalProbe Output** from the **Reserve** pin list.
6. Turn on **Reserve** pin.
7. Click **Add** or **Change** (for a new and an existing SignalProbe pin, respectively).
8. Click **OK**.

Figure 8–1. Reserving a Pin for SignalProbe in the Assign Pin Dialog Box



Adding SignalProbe Sources

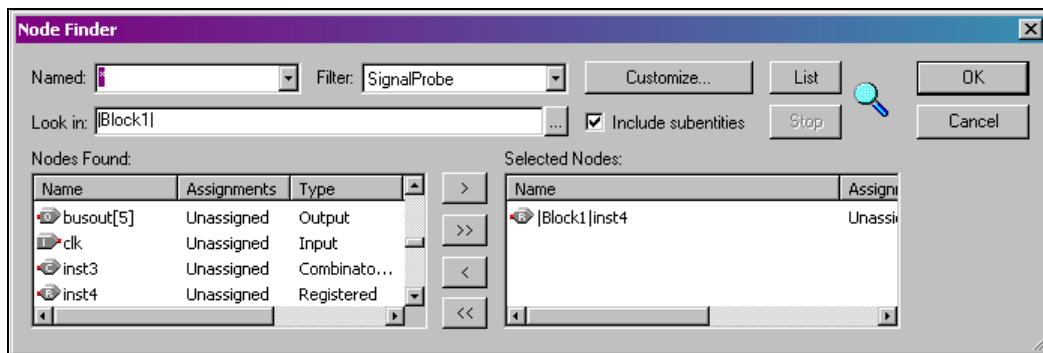
A SignalProbe source is a signal in the post-compilation design database with a possible route to an output pin. You can assign a SignalProbe source to a SignalProbe pin, an unused output pin, or a reserved output pin by performing the following steps:

1. Choose **Assign Pins** (Assignments menu).
2. In the **Available Pins & Existing Assignments** list, select the pin number for the pin to which you want to add a SignalProbe source. The pin must be a reserved SignalProbe pin, an unused output pin, or a reserved output pin.
3. Browse to a **SignalProbe source**.

The **Node Finder** dialog box appears when you click **Browse** and automatically selects **SignalProbe** in the **Filter** list (see Figure 8–2). Click **List** to view all the available SignalProbe sources. If you cannot find a

specific node with the SignalProbe filter, then the node has been either removed by the Quartus® II software during optimization or placed somewhere in the device where there are no possible routes to a pin.

Figure 8–2. Available SignalProbe Sources in the Node Finder



4. Click **Add** or **Change** (for a new and an existing SignalProbe pin respectively).
5. Click **OK**.

Assigning I/O Standards

The I/O standard of each SignalProbe pin must be compatible with the I/O bank the pin is in.

You can use the following two methods to assign I/O standards for your SignalProbe pins.

1. Choose **Assign Pins** (Assignments menu), select your SignalProbe output and select an I/O standard from the I/O standard list in the **Assignment** box in the **Assign Pins** dialog box.
2. Choose **Assignment Editor** (Assignments menu), select **I/O Standard** in the **Category** list, type the SignalProbe pin name in the **To** column and select the I/O standard in the **I/O Standard** column of the spreadsheet.

Adding Registers for Pipelining

You can specify the number of registers to be placed between a SignalProbe source and a SignalProbe pin to synchronize the data with respect to a clock and control the latency. The SignalProbe incremental routing feature automatically inserts the number of registers specified in the SignalProbe path.

For example, you can add a single register between the SignalProbe source and the SignalProbe output pin to reduce the propagation time (t_{CO}). You can add multiple registers to your SignalProbe output pins to synchronize the data with other output pins in your design.



When you add one register to a SignalProbe pin, the SignalProbe compilation always attempts to place the register into the I/O element. If it is unable to place the register into the I/O element, it places the register as close to the SignalProbe pin as possible to reduce clock to output delays (t_{CO}).

You can add registers to your SignalProbe pin by performing the following steps:

1. Choose **Assign Pins** (Assignments menu).
2. In the **Available Pins & Existing Assignments** list, select the pin number for the SignalProbe output pin you want to register.
3. Under **Assignment**, type a new **Clock name** in the **Clock box**.
4. Under **Assignment**, type the number of registers necessary to pipeline your SignalProbe source in the **Register box**.



Altera strongly recommends using global clock signals to clock the added registers.

The Stratix, Stratix II, Stratix GX, and Cyclone devices support adding registers to a SignalProbe pin.

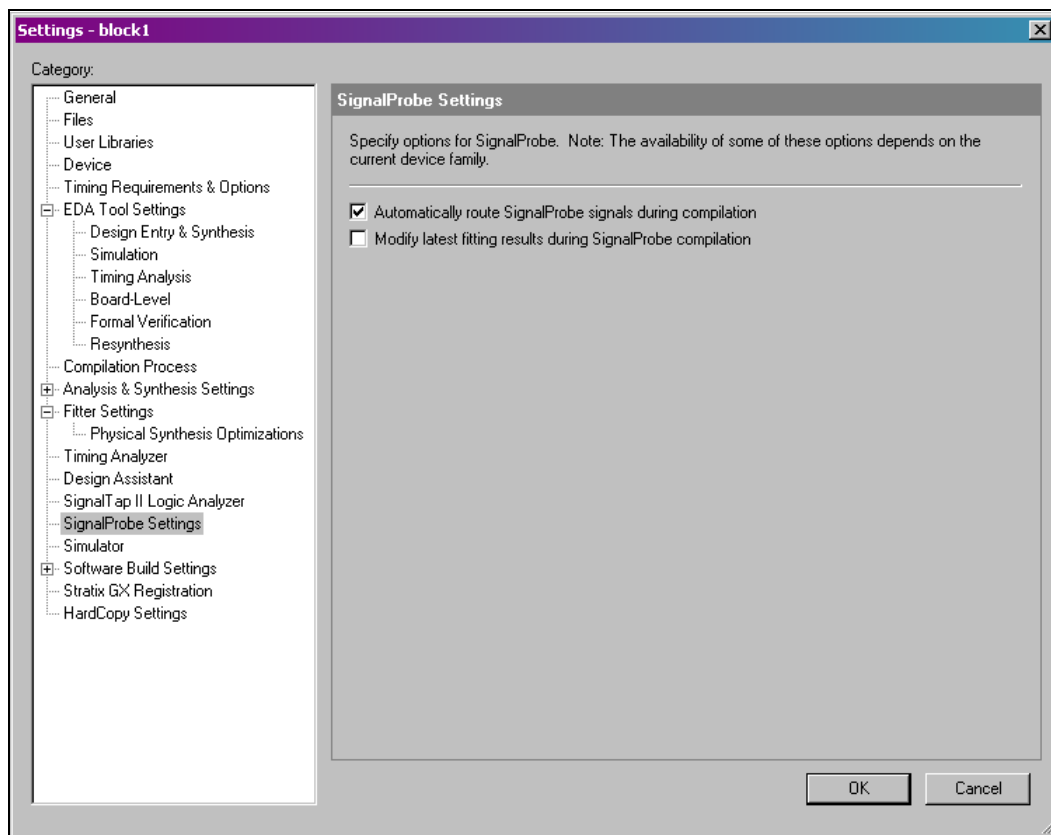
Performing a SignalProbe Compilation

You can start a SignalProbe compilation manually or automatically after a full compilation. A SignalProbe compilation performs the following steps:

1. Validate SignalProbe pins.
2. Validate your specified SignalProbe sources.
3. If applicable, add registers into SignalProbe paths.
4. Attempt to route from SignalProbe sources, through registers, to SignalProbe pins.

To make the SignalProbe compilation run automatically after a full compile, turn on **Automatically route SignalProbe** sources during compilation in the **SignalProbe Settings** page in the **Settings** dialog box (Assignments menu), as shown in [Figure 8–3](#).

Figure 8–3. SignalProbe Settings Page in the Settings Dialog Box



To run a SignalProbe compilation manually after a full compilation, choose **Start SignalProbe Compilation** (Processing menu).



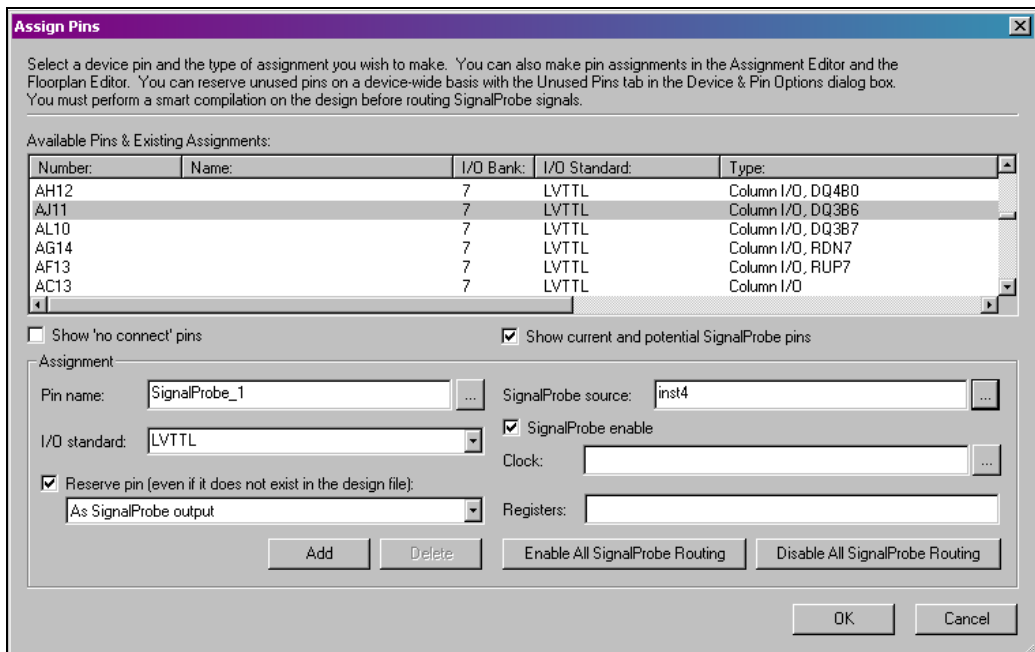
You must run the Fitter before a SignalProbe compilation. The Fitter generates a list of all internal nodes that can be used as SignalProbe sources.

You can enable and disable each SignalProbe pin by turning **on** and **off** the **SignalProbe enable** option in the **Assignment** box in the **Assign Pins** dialog box (Assignments menu). You can also enable or disable all

SignalProbe pins by clicking **Enable All SignalProbe Routing** and **Disable All SignalProbe Routing** respectively in the **Assignment** box in the **Assign Pins** dialog box (see [Figure 8-4](#)).

The **Enable All SignalProbe Routing** and **Disable All SignalProbe Routing** options appear grayed out until you turn on **Show current and potential SignalProbe pins** in the **Assign Pins** dialog box (Assignments menu).

Figure 8-4. Enabling and Disabling SignalProbe Pins in the Assign Pins Dialog Box



Running SignalProbe with Smart Compilation

Smart compilation reduces compilation times by running only necessary modules in a compilation. However, a full compilation is required if any design files, Analysis & Synthesis settings, or Fitter settings have changed.

To turn on **Smart compilation**, turn on **Use Smart compilation** in the **Compilation Process** page in the **Settings** dialog box (Assignments menu).

You get the following message after running a SignalProbe compilation if you turned on smart compilation, made any change in a design file, or changed settings related to the Analysis & Synthesis or Fitter modules.

Error: Can't perform SignalProbe compilation because design requires a full compilation.



Altera recommends turning on smart compilation so that you are always working with the latest settings and design files.

Understanding SignalProbe Routing Failures

If the SignalProbe compilation starts and fails, it could be because of one of the following reasons:

1. The SignalProbe compilation failed to find a route from the SignalProbe source to the SignalProbe pin because of routing congestion.
2. You entered a SignalProbe source that does not exist or is an invalid SignalProbe source.
3. The output pin selected is found to be unusable.

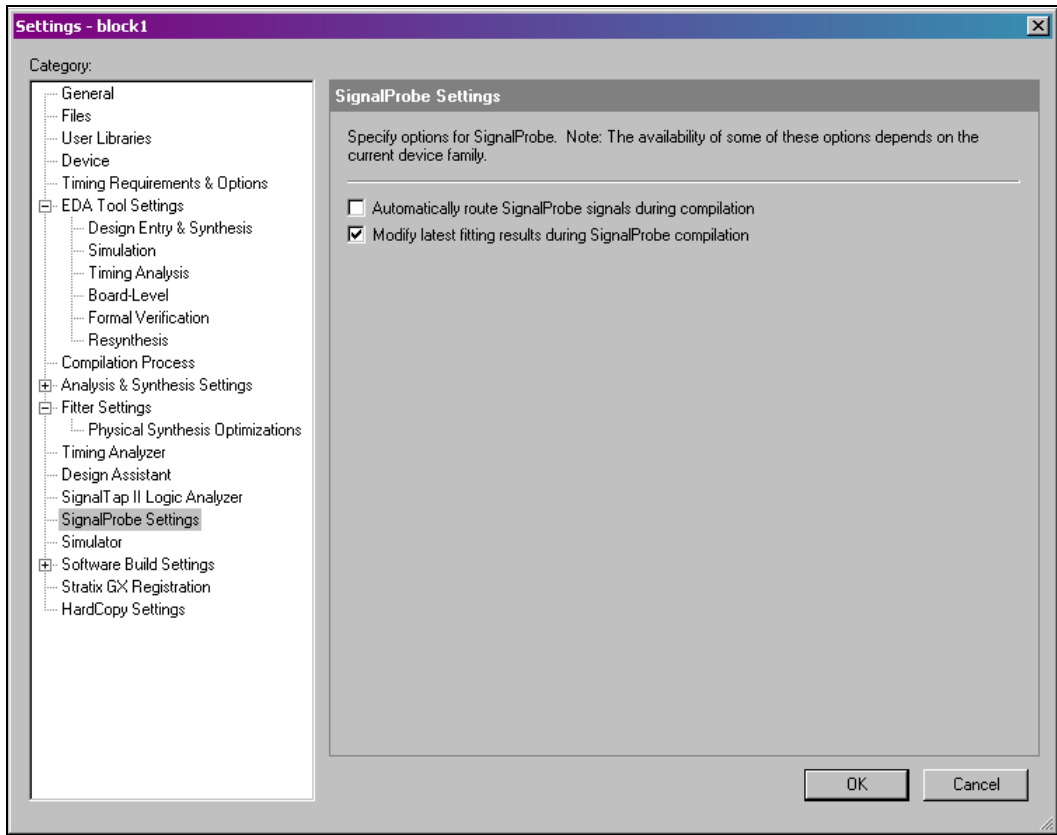
This can occur if the SignalProbe pin's I/O standard conflicts with other I/O standards in the same I/O Bank.

If routing congestion is preventing a successful SignalProbe compilation, you can turn on **Modify latest fitting** results during SignalProbe **compilation** in the **SignalProbe Settings** page in the **Settings** dialog box (Assignments menu) to allow the compiler to modify the routing to the specified SignalProbe source (see [Figure 8-5](#)). This setting allows the Fitter to modify the existing routing channels used by your design.



Turning on **Modify latest fitting** results during SignalProbe compilation may change the performance of your design.

Figure 8–5. SignalProbe Settings Page in the Settings Dialog Box



Understanding the Results of a SignalProbe Compilation

Use the **Messages** window to view the results of the SignalProbe compilation. This window lists successfully routed SignalProbe pins. In addition, it displays slack information for each successfully routed SignalProbe pin.

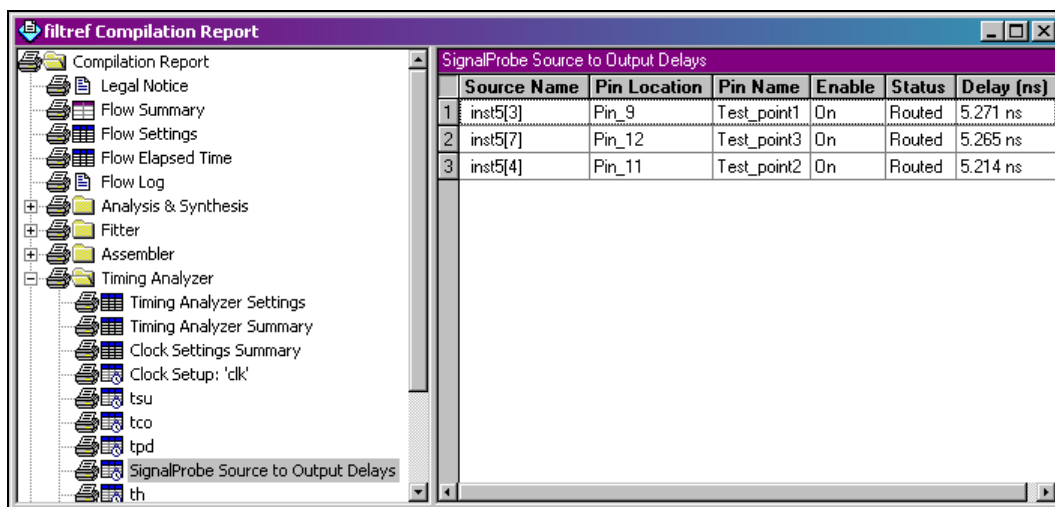
You can view the status and delays of each SignalProbe pin by viewing the **Status** column in the **Assign Pins** dialog box (Assignments menu). [Table 8-1](#) describes the possible values for the **Status** column.

Table 8-1. Status Values

Status	Description
Routed	Connected and routed successfully
Not Routed	Not enabled
Failed to Route	Failed routing during last SignalProbe compilation
Need to Compile	Assignment changed since last SignalProbe compilation

You can find source to output delays for each routed SignalProbe pin in the **SignalProbe Source to Output Delays** page under **Timing Analyzer** in the **Compilation Report** window (see [Figure 8-6](#)).

Figure 8-6. SignalProbe Source to Output Delays Page in the Compilation Report Window



Conclusion

Using the SignalProbe incremental routing feature, the SignalProbe source routing process can complete in a fraction of the time required for a full recompilation. You can use the SignalProbe incremental routing feature to get quick access to internal design signals to perform system-level debugging.