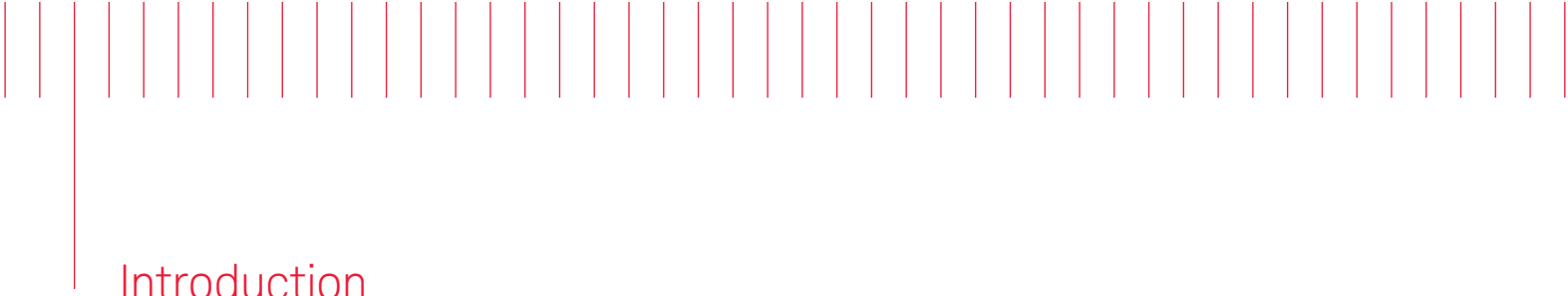


Keysight Technologies

Planning Your Design for Debug: FPGA Dynamic Probe

Design Guide



Introduction

The FPGA dynamic probe is a flexible tool that allows you to view many internal design signals using a few pins connected to the logic analyzer or mixed-signal oscilloscope (MSO). The combination of this tool with advanced logic analysis physical probing and sophisticated triggering produces a powerful combination for finding the toughest FPGA bugs, as well as the easy ones.

Planning for debug during design will allow you to make best use of the FPGA dynamic probe. Planning can make the FPGA debug port a reliable, simple, and effective observation port for in-circuit verification. This document is a guide to help you in this planning process.

You can find more information, about the FPGA dynamic probe, including a list of frequently asked questions, at: www.keysight.com/find/fpga and www.keysight.com/find/MSOfpga

Planning for FPGA Debug

Planning for debug is the best insurance to help quickly get through power up and unexpected in-circuit problems. Planned debug can be the difference between an on time project and one that slips.

On-chip resources

The degree of project risk will influence the resources you will need to allow for debugging. How much of the design is new and untested in-circuit? If much of your design has not been verified in-circuit, your risk will be high. Many problems only manifest themselves in-circuit when the FPGA is interacting with the surrounding system at full speed.

Examples of various risk levels:

- High risk:
- project has an entirely new design with none of the HDL verified in-circuit
- Medium risk:
- project is leveraged from an existing, tested design, with some new untested features added
 - existing design targeted at a different FPGA device family
- Low risk:
- leveraged design from an existing design
 - uses the same FPGA as used in the original
 - few changes to the design

Risk	Reserved debug resource		Differential Pin width for designs > 200 MHz	Single-ended Pin width for designs ≤ 200 MHz	ATC2
	Max flop	Max LUT			
High	25-40%	25-40%	(Widest control path + Widest data path + Widest address path) * 2	(Widest control path + Widest data path + Widest address path)	Equals number of time domains in your design
Medium	10-25%	10-25%	(Widest control path + Widest data path) * 2	(Widest control path + Widest data path)	2-4
Low	5-10%	5-10%	(Widest control path + some data path) * 2	(Widest control path + some data path)	1-2

Table 1. Design risk and associated debug headroom

Planning for FPGA Debug (continued)

Resource headroom

Extra resources help design tools reach timing closure more quickly. For rapid validation and to allow for future product enhancement, some design teams allocate up to 25% to 40% of the FPGA fabric for headroom. Some of this headroom will be taken by the Keysight Technologies, Inc. Trace Core-2 (ATC2) used to facilitate rapid debug of your FPGA and the surrounding system.

Estimating resource consumption

Although a single ATC2 core can use as few as 65 LUTs and 54 flops¹, additional overhead provides headroom if you encounter a situation where you want to route multiple ATC2 cores using full 32-bit banks with wide outputs widths. If you already know the dimensions of most of the units you will be debugging, you can use Keysight's ATC resource calculator [2] to better estimate the fabric use for one or more of these debug cores.

Number of pins

The minimum debug pin width from Table 1 may be difficult to achieve using traditional debug methods. The demands of the mission-critical pins will always take precedence over the debug pins, so you may not have enough spare pins to debug your design. Consider how many pins you would need ideally for debug. This exercise will make you aware ahead of time of the observable and unobservable areas in your design. ATC2 is user configurable from 4 pins to 128 pins for logic analyzers, and 4 pins to 16 pins for MSOs.

Speed

If the design signals run below 200 MHz, you can use single-ended outputs from the ATC2 core to connect to the logic analyzer or MSO. If the design has internal signals that will be probed running above 200 MHz, the core should use differential outputs because signal fidelity is better preserved; thus high-speed data can be easily transferred using differential outputs. Note that if you use differential outputs, you must plan the printed circuit board (PCB) layout accordingly. Some logic analyzers and MSOs do not support differential inputs.

In some cases, 2x pin compression rate is not possible because the edges are too fast for the I/O, as in the LVCMOS 3.30 I/O standard. The only convenience 2x pin compression gives is the ability to view twice the signals with the same number of pins. Note that the MSO does not support 2x pin compression.

Number of cores

The number of ATC2 cores that normally would make sense in the design can be anywhere from 1 to n, where n is the total number of frequency domains in the FPGA design. In general, a single ATC2 core is sufficient to debug most problems. Adding a second core allows you to view two time domains. Two cores also can help you see a combination of two groups of signals. For instance, in one measurement you can view a group of signals on bank 1 on core 1 and a different group of signals on bank 2 on core 2. For another measurement you can keep core 1 on bank 1 but change to bank 3 on core 2. Having more cores enables even more combinations and also simplifies cross-clock domain debug.

Each ATC2 core has its own time base and thus requires a unique logic analysis acquisition module to view all simultaneously. In this paper, we use the following definition for the term "module" for the Keysight 16900:

A module is a group of logic analysis channels that are grouped together on a single time base. The implementation may be a single logic analyzer (for example, an Keysight 1680 or 1690 Series logic analyzer or a single Keysight 16910, 16911, or 16950 logic analyzer module), or a collection of logic analyzer modules that are connected together in a 16900 Series modular logic analysis system. Keysight 1680, 1690, and 16900 Series logic analyzers have a split analyzer mode where a single logic analyzer can be split into two time bases.² Two ATC2 cores can be monitored simultaneously by a single logic analyzer that has been split.

If you are using a 1680 or 1690 logic analyzer, you will have only one main module. This module can be split into two modules, but this is the limit. A 16903A can have up to three cards. If each card is an independent module, you can have up to six ATC2 cores attached to it. The 16900A and 16902A mainframes hold up to six modules. With 6 modules configured as separate modules, you can have up to 12 ATC2 cores connected to it.

If you are using a MSO, only one ATC2 core can be controlled at a time.

Printed Circuit Board Probing Connectors

For the FPGA dynamic probe you will need one connector for the JTAG scan chain and another for the logic analyzer or MSO acquisition channels.

JTAG connector

The JTAG pins on the FPGA must be accessible. The most common approach to making these pins accessible is to connect them to 0.100-inch center header posts that connect to flying leads on the parallel programming cable. An alternative to this header is Xilinx's recommended target interface connector [5], shown in Figure 1. This connector provides a quick way to link to the JTAG scan chain.

Typically, problems on the scan chain are related to glitches on the JTAG signals. Careful layout is critical. Pay close attention to the signal integrity effects on TCK and TMS. It is best not to load the scan chain with more than four devices to help ensure the cable drives all devices with enough power. For longer scan chains you should consider adding buffers on TMS and TCK.

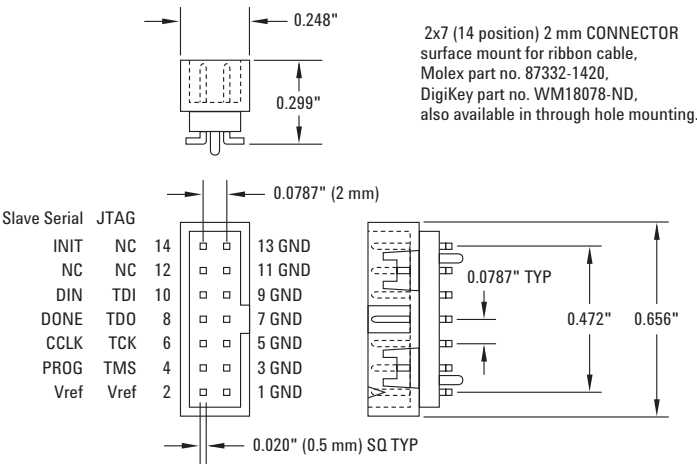


Figure 1. Xilinx target interface connector (reprinted from "Xilinx Parallel Cable IV," with permission from Xilinx Corporation)

1. Timing core with 1 bank and 4 ATD pins + 1 ATCK pin
2. When a module is split it is called split-module.

One way to always get a reliable connection on the scan chain is to allow isolating the FPGA from the scan chain, as shown in Figure 2. Here two zero-ohm resistors, R0 and R1, are added to the chain TMS and TCK signals, called Ch_TMS and Ch_TCK respectively. The resistors are used to break the chain inputs going to this FPGA. The TMS and TCK lines are then routed from the zero-ohm resistors to header posts or connector pins. These header posts or connector pins will be used for connecting the programming cable. The TDI and TDO pins on the FPGA are also routed to the header posts or connector pins. No zero-ohm resistors are used on these pins because these PCB traces will be short compared to the TCK and TMS lines. These signals can still work under noisy conditions, like ringing, because TDI is sampled only on a rising edge of TCK. TDO can have noisy edges because it is sampled long after TCK has gone low.

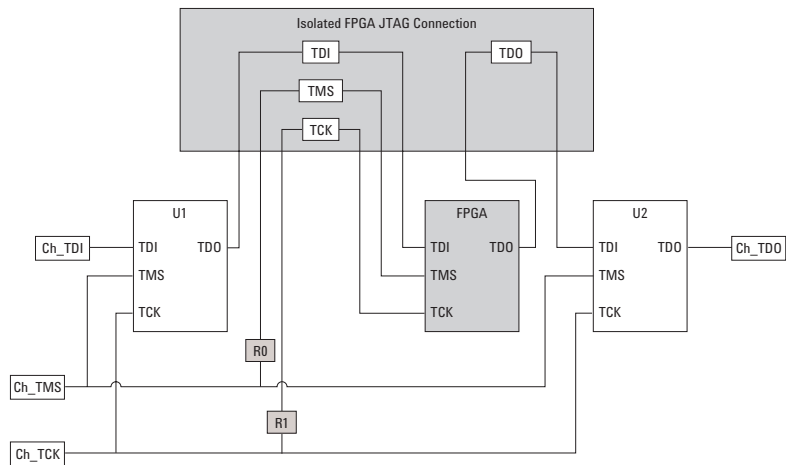


Figure 2. Isolated FPGA

Acquisition connector

A dedicated trace connector is the best way to plug a logic analyzer or MSO into your system. Using a standard connector such as AMP's Mictor connector or Keysight's soft touch connectorless probe, you can have a robust connection to your system in seconds, compared to hours of soldering individual brittle wires to traces on a PCB. Moreover, the improved signal integrity on these connectors provides an added benefit with no additional effort.

Keep in mind these considerations as you plan for connectors with the ATC2 core:

First, each ATC2 core has data and clock pins as shown in Figure 3. There can be from 4 up to 128 data pins for logic analyzers (up to 16 for MSOs), called ATD. Only one clock pin exists for each ATC2 core. The clock line is a state clock line for an ATC2 state core, and it is treated as an individual signal channel for the timing core. We recommend limiting each connector to two cores to simplify trace setup.

For supporting two cores on a connector, route the pins from each ATC2 core to either the even and odd connector pins of the trace connectors. For example, suppose you have 34 debug pins on an FPGA. In this case, you should route 16 pins plus a clock to the even pod pins of the trace connector. Similarly, you should route the next 16 pins and a clock to the odd pod's connector pins. This connection will allow you to use two ATCs from different time domains with just one connector. When you are debugging within one time base, you can reconfigure the ATC2 core to use all 34 channels for a single trace core.

You should split the trace connector into two cores even if all you plan to have is 16 pins for debug. In this case, you can reserve nine FPGA outputs for the odd pod pins of the connector. The remaining seven FPGA outputs are then assigned to the even pod con-

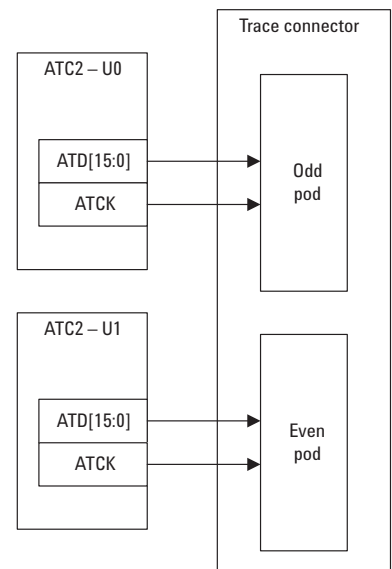


Figure 3. ATC2 core connected to logic analysis acquisition connector

nector pins. When you are debugging wide buses with this arrangement, all 16 pins can be used by a single trace core. This arrangement will also accommodate two cores, one assigned to the even-pod pins and the other to the odd-pod pins. With two cores you can easily inspect two time domains with a single trace connector.

If you are using a MSO with its 16 digital timing channels, only the even or odd pod connector can be used at a time.

The footprint of two types of soft touch connectors is shown in Figure 4 and Figure 5. The differential connector in Figure 4 is a one-pod, high-speed footprint for the 16950A module. The right-hand side of this connector has the positive end of the differential signal; the left has the negative signal.

The connector in Figure 5 is a two-pod, single-ended footprint intended for all modules and logic analyzers supported by the FPGA dynamic probe. The top portion above the dashed line is the odd pod. Here the channels have an “A” suffix. The bottom portion connects to the even pod. These channels have a “B” suffix.

If PCB space is limited, you can use a half-size soft touch connector. The footprint of this connector is shown in Figure 6. In this case, only one pod is available on the connector, therefore only one ATC2 core should connect to it. The half-size soft touch connector is ideal for use with the MSO and its single input pod connector.

The Mictor connection footprint for an ATC2 core is shown in Figure 7. The Mictor connector supports two pods so two ATC2 cores can be connected to it. The distinction between the “A” and “B” suffix on the core pins in the diagram is made to point out that two cores can be placed on this connector. The “A” core is placed on the right-hand pins (the odd-pod channels). The “B” core is placed on the left-hand pins (the even-pod channels).

Note that the Mictor connector is lower performance than a soft touch connector. One of the Mictor connector’s limitations is its larger capacitive load. See “Probing Solutions for Logic Analyzers,” [4] for more information.

34-Channel Differential
Soft Touch Footprint

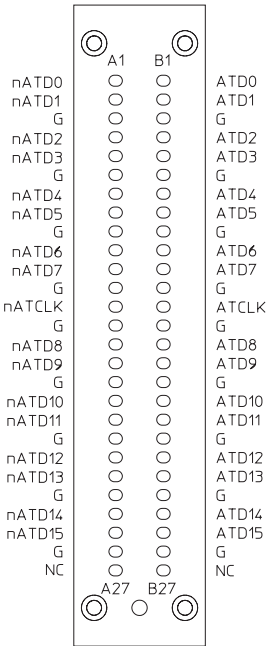


Figure 4. Soft touch differential connector

34-Channel Single-ended
Soft Touch Footprint

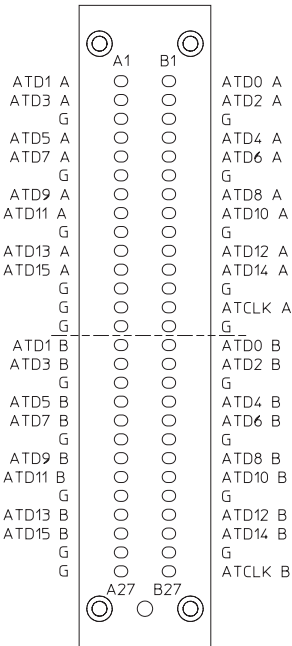


Figure 5. Soft touch single-ended connector

17-Channel Single-ended
Soft Touch Footprint

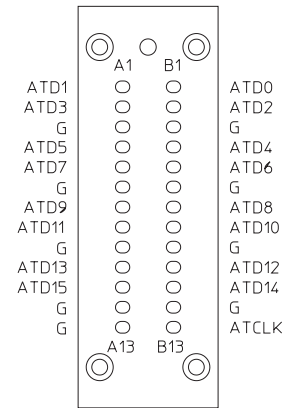


Figure 6. Half soft touch connector

38-Channel Footprint

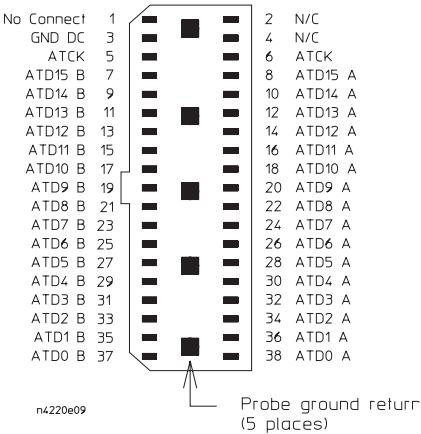


Figure 7. Mictor connector

Adding ATC2 Cores

There are two methods for adding an ATC2 to a design. One is through instantiation and the other is by insertion. Both methods require Xilinx ChipScope version 6.3 or higher [3]. We recommend the core insertion flow.

Core insertion flow

Insertion is done with no modifications to the HDL. The ChipScope core inserter tool takes a synthesized design and adds a customized ATC2 core. It does this using the synthesized EDIF, electronic design interchange format, netlist file.

The basic flow of the inserter is shown in Figure 8. The netlist is the output of the synthesis tool such as Synplify's Synplify Pro. Core inserter reads in this file and then uses a wizard-type GUI to guide you in adding the ATC2 core.

After you complete this process, the inserter saves the insertion information in an insert-er project file with a .cdc extension. This file will later be used on the logic analyzer or MSO for signal name import. When you exit the inserter tool, you will need to rebuild your FPGA design to generate a .bit programming file.

The insertion method is preferred because it requires no modification to the source. When the insertion method is used in the Xilinx ISE tool, the ATC2 core is automatically added every time the FPGA is rebuilt. The .cdc file is part of the ISE project and is associated with this FPGA design. When you need to modify the core, you simply open the inserter and change the parameters.

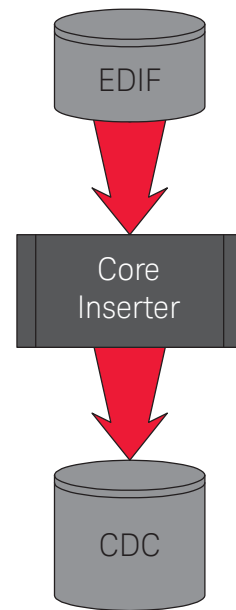


Figure 8. Core inserter flow

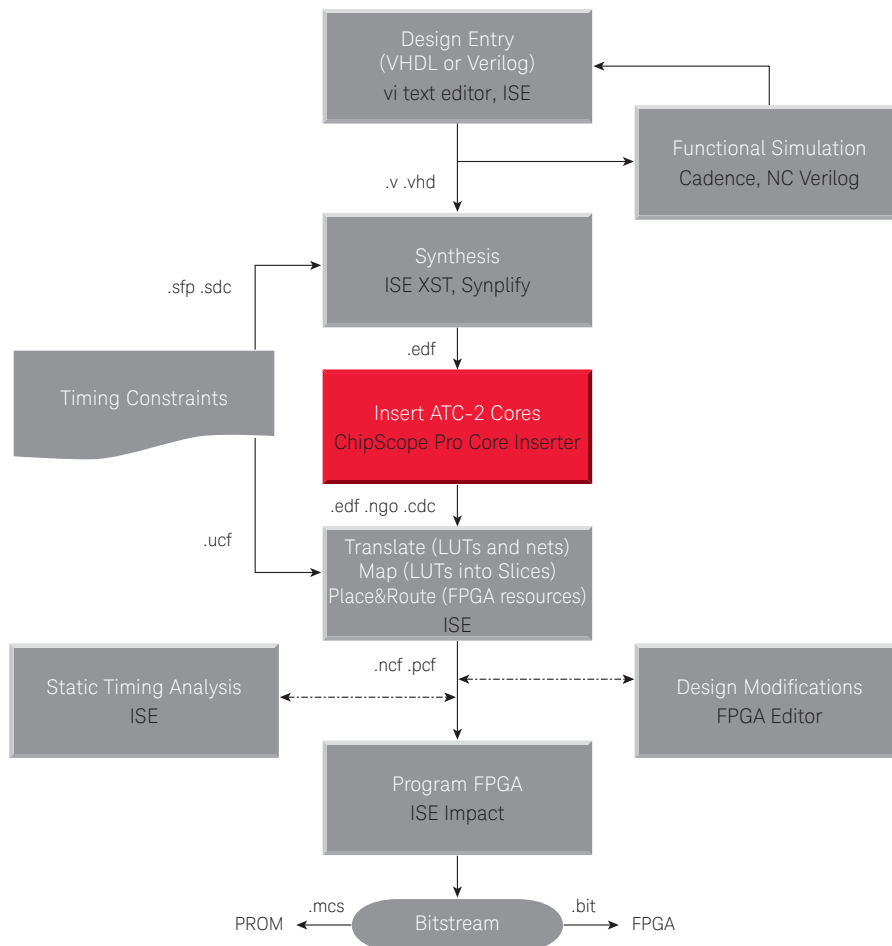


Figure 9. Design flow with core insertion

Adding ATC2 Cores (continued)

If you no longer want the core in the design, you simply remove the .cdc file from the project and rebuild the FPGA bits. Therefore, the insertion method makes it very easy to add and remove ATC2 cores to/from an FPGA.

Core generation flow

Instantiation is done by generating a customized ATC2 core and embedding it in your HDL. The core is customized using the ChipScope Core Generator tool. It guides you through the core parameters and generates the core netlist and an example showing how to instantiate the core in your design.

In your design, you then modify the HDL to instantiate and connect the core to the signals of interest. After these changes you re-synthesize and build the FPGA. When you build the FPGA, the core netlist file must be in a location where the place and route tools can find it.

Core Generation does not produce a .cdc file with signal names for logic analyzer or MSO import. You can manually modify an existing .cdc file to match your modified design.

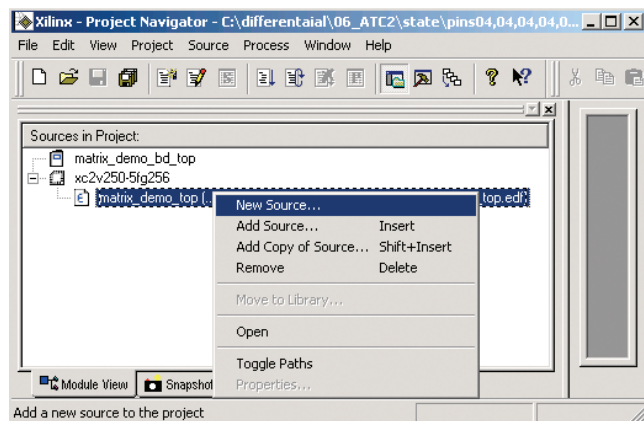


Figure 10. Adding a ChipScope project file

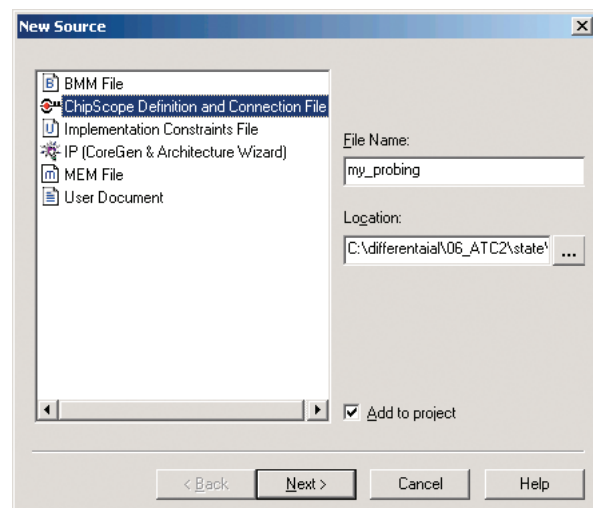


Figure 11. Defining ChipScope new source

Adding ATC2 Cores (continued)

Inserting an ATC2 core

To better understand the ease of use of the core inserter, consider the case of adding an ATC2 core into an existing Xilinx ISE-based project. First you will need to add a ChipScope project file to your ISE project. To add this file, in ISE you right mouse click on the design .edf file in the module view tab of the Sources window, as shown in Figure 10.

Next, the user interface pops-up a dialog box, as shown in Figure 11. Here you select the ChipScope Definition and Connection File and type in a file name in the File Name field. The file will be added to the project with a .cdc extension.

After this dialog box closes, click Next and two more dialogs will follow. The first, shown in Figure 12, has you select the source, which should be the .edf file. The second dialog box, shown in Figure 13, shows what the tool is going to do. In this dialog box, you can back track or click Finish if everything looks correct.

When this process completes, the design EDIF source will have associated with it the newly created .cdc file, as shown in Figure 14. Now, every time this FPGA is built, the settings in the .cdc will be used to add ATC2 probing to the design. If you want to remove the probing from the project, all you do is remove the file from the project and rebuild the FPGA.

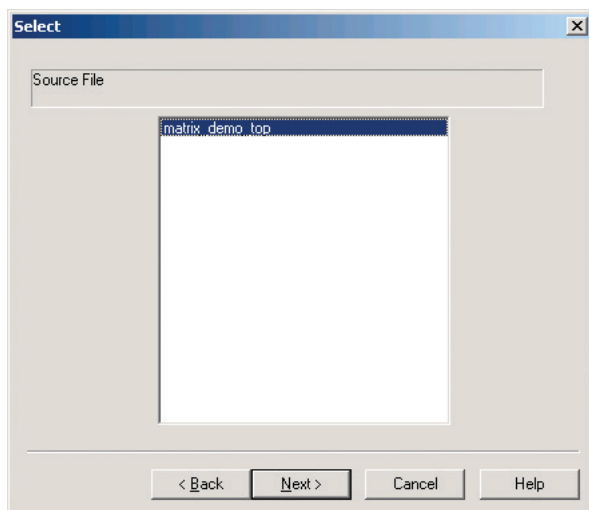


Figure 12. Select source

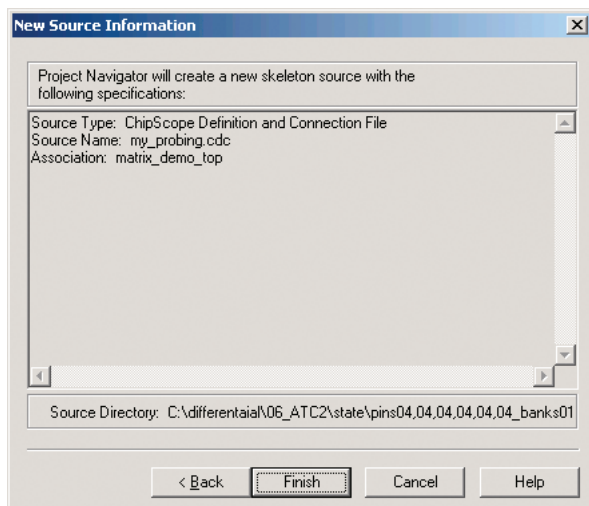


Figure 13. Finish process

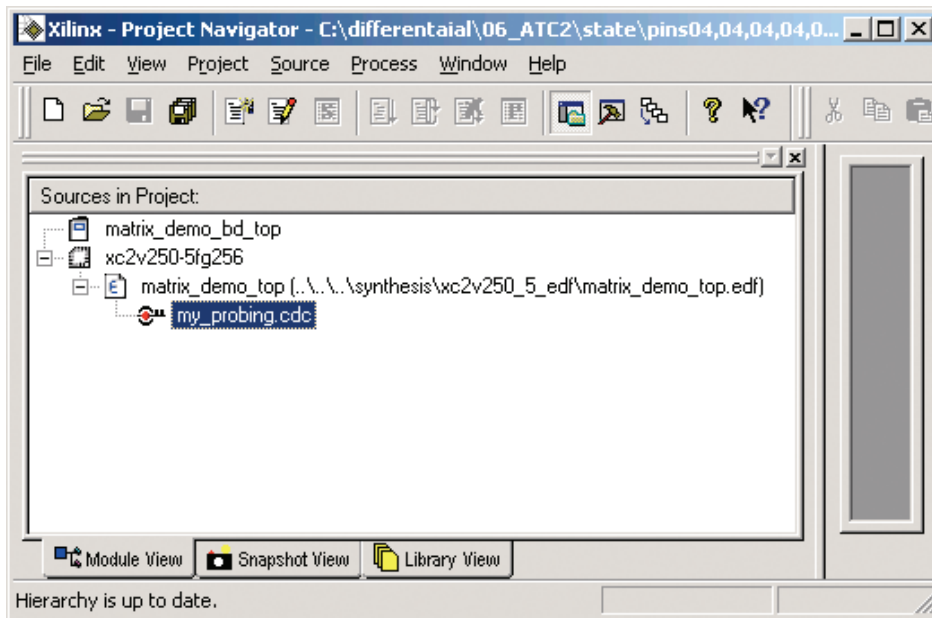


Figure 14. CDC file added

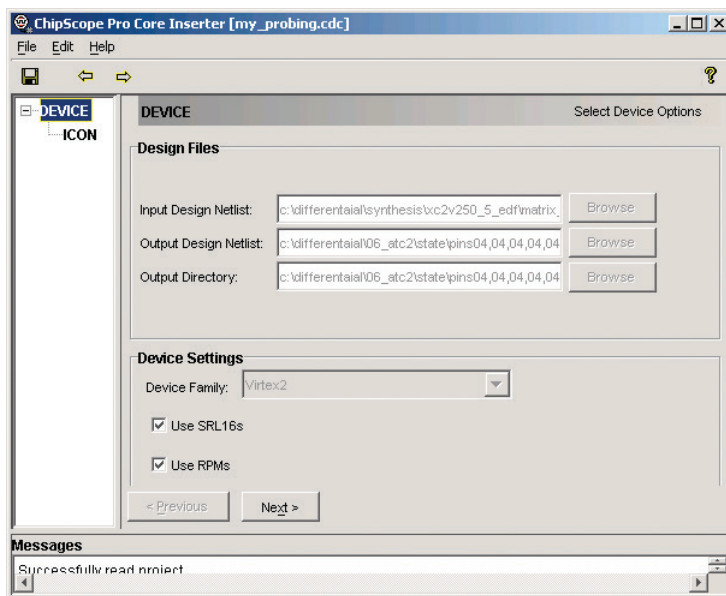


Figure 15. Core inserter

To add ATC2 cores to the design, double click on the .cdc file. For instance, in Figure 15 you double click on my_probing.cdc to add ATC2 cores to it. When you double click, the ChipScope core inserter interface is launched.

Since the .cdc file was created using ISE, the device information for core inserter is automatically filled in. Here, the only optional fields are Use SRL16s and the Use RPMs fields. In general, it is best to leave these enabled so that optimal size and performance can be achieved. After this first screen, click Next.

Adding ATC2 Cores (continued)

The next screen in the core inserter defines the ICON. ICON provides the JTAG interface used by the ATC2 core for communication between it and the logic analyzer or MSO. Through ICON the logic analyzer or MSO is able to get core status information and change the bank MUX select lines. In this dialog, you have the choice of adding a BUFG for use in JTAG communication. In general, you can disable the BUFG as long as the JTAG communication is run at a low TCK rate, like 200 KHz.

After configuring ICON, click New ATC2 Unit. When this button is clicked, the window updates and shows U0: ATC2 on the device tree, as shown in Figure 17. This unit is an unconfigured core. It must be selected and then customized.

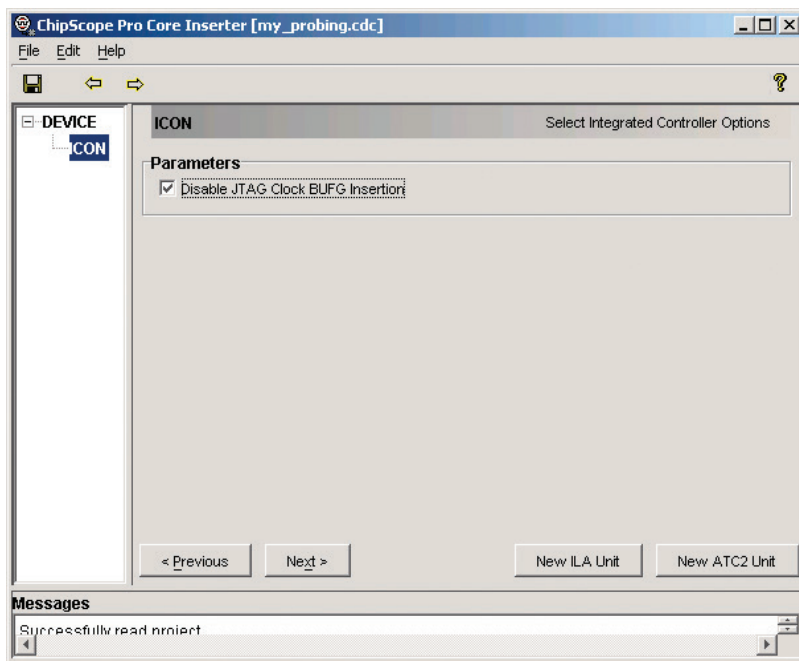


Figure 16. ICON settings

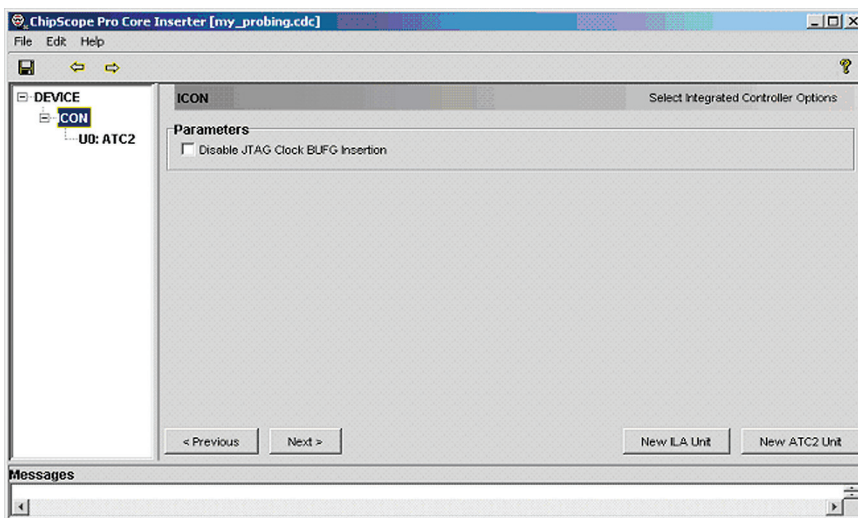


Figure 17. New ATC2 unit

When you select U0, the window displays all the core parameters, as shown in Figure 18. Here, you select the type of core, number of pins and banks, and the TDM rate. You also set the I/O standard for the pins and their location. Once this is set, you select the Net Connections tab to connect the design signals to the core.

The Net Connections field shows the signals connected to the core. If one or more connections are missing, this window shows the net connection in red. As expected, when the core is created all connections are red, as shown in Figure 19. Here, click Modify Connections to make the signal connections between the design and the core.

When you click Modify Connections, a new window appears. This window shows the structure of the design, read in from the EDIF, in the top left-hand side. On the bottom left side is the individual signals contained in the selected level of hierarchy. You select a signal name and click Make Connections on the lower right-hand side to connect that signal with the selected channel.

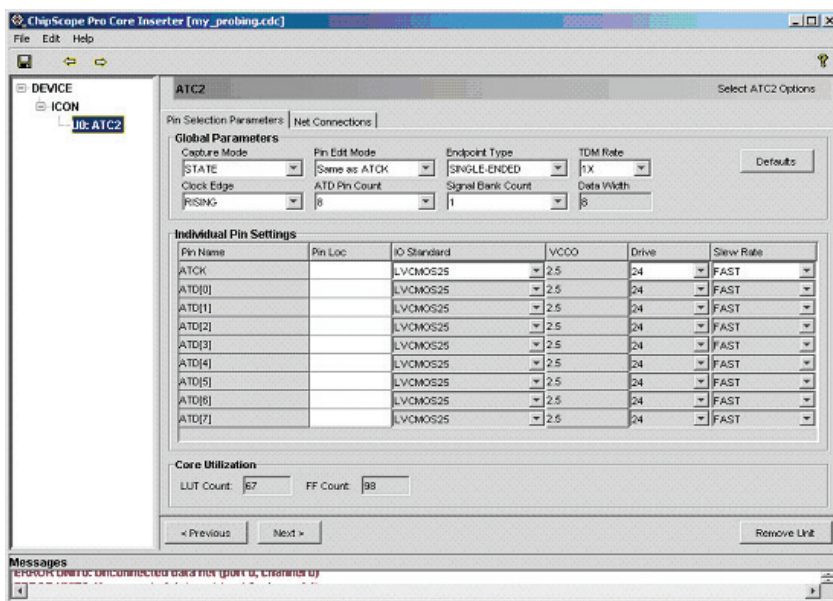


Figure 18. Unconfigured ATC2 core

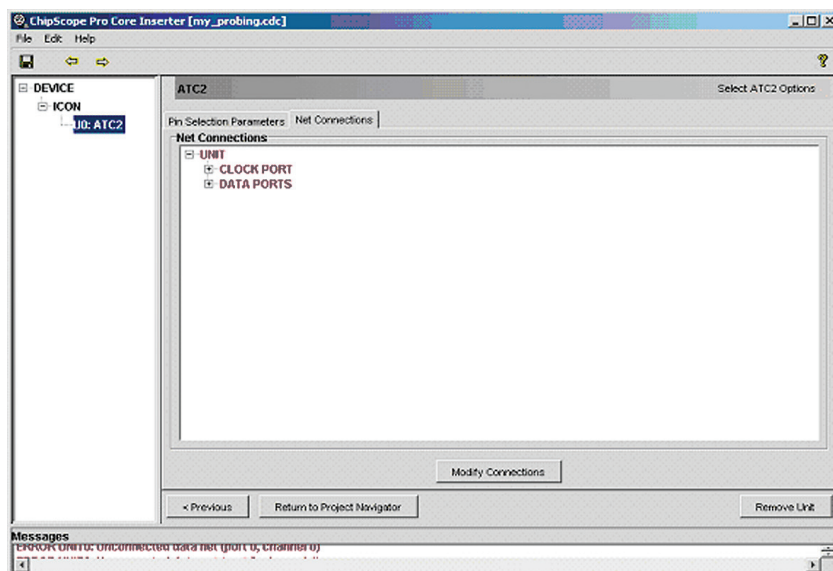


Figure 19. Net Connections tab

Adding ATC2 Cores (continued)

The channel input is split up into two categories. One is for the clock, and the other is for data. The clock channel is used for state cores, which need a sample clock. For this channel, you connect your system clock that is associated with the time domain probed by this core.

Note that the MSO does not have a sample clock input and no state acquisition mode. The MSO does have a pseudo-state bus display mode using post-acquisition processing. When using this mode the system clock should be connected to one of the data input channels.

You set the data input channels by selecting the Data Signals tab. In this field, shown in the upper right of Figure 21, you connect data signals associated to this clock source. If the ATC2 core is configured with a bank size greater than 1, use the tabs extending from the bottom edge of the Data Signals tab to set data input channels for each signal bank input. Each signal bank must be connected with design signals before you build the design.

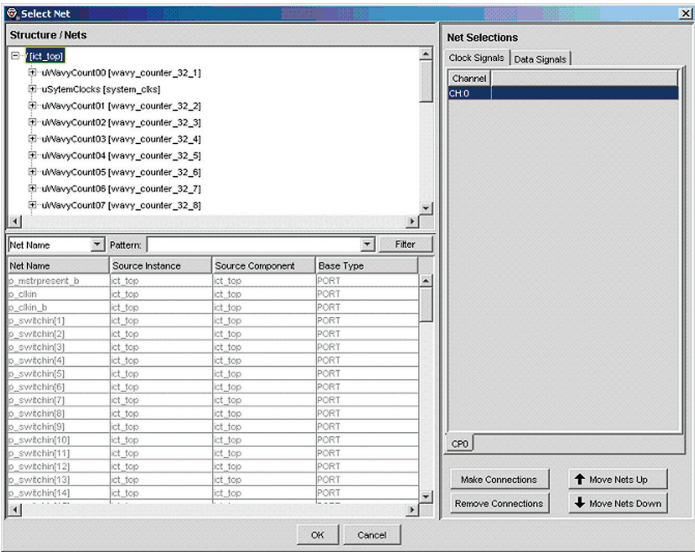


Figure 20. Net Connections window

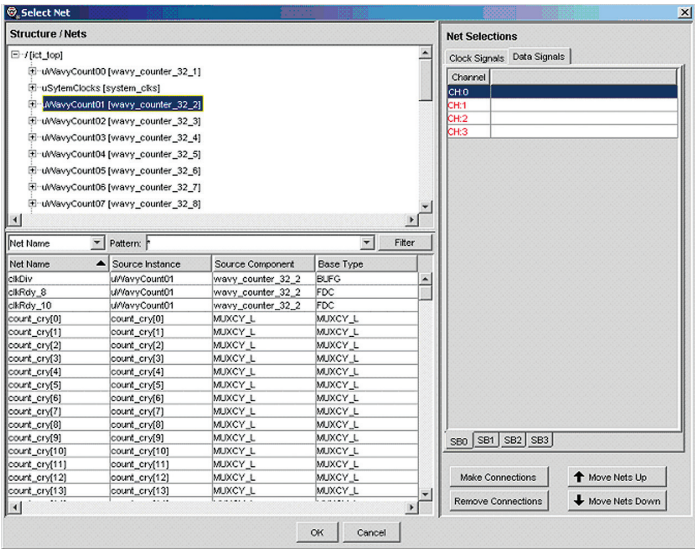


Figure 21. Data Signals tab

If a bank has all the signals you need but still has unconnected pins, then you can fill them with a generic signal, such as GND. To do this, search for GND in your design and then connect it to as many channels as you have available.

When you use more than one signal bank, you should consider duplicating signals across signal banks. Duplicating signals allows you to see certain signals in combination with other groups of signals. For instance, you may consider grouping a signal called “Enable” with a counter on bank 0 and also connect it with FIFO data on bank 1. Doing so allows you to view the “Enable” signal with respect to the counter and also with the FIFO data.

Debugging with ATC2

The core inserter makes it easy to add the ATC2 core to a design. However, there are some things to consider since the inserter is run post-synthesis. Keep in mind that the signal names may have changed. For example, a wire in the design that traverses several layers of hierarchy may not show up in the EDIF file. Logic that is optimized may lose some of the intermediate signals.

Synthesis has options that will generate different EDIF outputs based on input constraints.

For instance, a state machine can be encoded as a one-hot state machine by synthesis tools in order to improve timing. Synthesis will also replicate logic in order to reduce fanout and achieve desired timing results.

One way to make design signals visible is to use keep rules so that net names and intermediate signals are preserved. In Synplicity, the directive `syn_keep` will preserve the signal or register that has this attribute. This causes Synplicity to add a stand-in buffer, BUF, to the circuit and not optimize this signal. When synthesis builds the EDIF, the BUF shows up and the signal name is preserved. Therefore, with `syn_keep` you can probe your circuit using the signal names from your HDL. The one downside to using the `syn_keep` directive is that Synplify Pro will not optimize the circuit as well as it could have. For this reason, use `syn_keep` sparingly.

At top-level I/O ports, `syn_keep` does not affect the design. When `syn_keep` is placed on input, output, or input and output ports, the timing is the same as it would have been without the property. This is because no optimizable logic exists at the ports – only pins and flops are at the IO ring.

The benefit of using `syn_keep` at the ports is that they become probeable by the ATC2, because the inserter will be able to connect the output of the keep BUF to the ATC2 signal bank. If the port does not have a `syn_keep` directive, the inserter will not be able to probe it. If you attempt to probe a port directly, you will see errors when the FPGA is built.

To probe multiple clock domains, you can use two state cores or one timing core and a state core. Note that using multiple cores to debug multiple time domains is not available using a MSO. If you use two state cores, you will set each core to a particular time base. Each core will have signal bank data from its respective time base, as shown in Figure 22. On the logic analyzer, a split- or two-module system can trigger on one or two time bases. In this case, the measurement can be separate or combined such that one time base trigger arms the second.

Debugging with ATC2 (continued)

When you use a timing core and a state core, the logic analyzer correlates both measurements together. The timing core will have detail on glitches but will not necessarily have de-skewed data.

The timing core and state core can be quite useful in cases where glitches are suspicious. When multiple time domains are being debugged, placing a timing core on the control logic, as Figure 23 shows, can unveil glitches that cause the cross clock domain handshakes to fail. Or, two state cores can help debug situations where handshakes appear to work but the data is corrupted going from one time domain to the other.

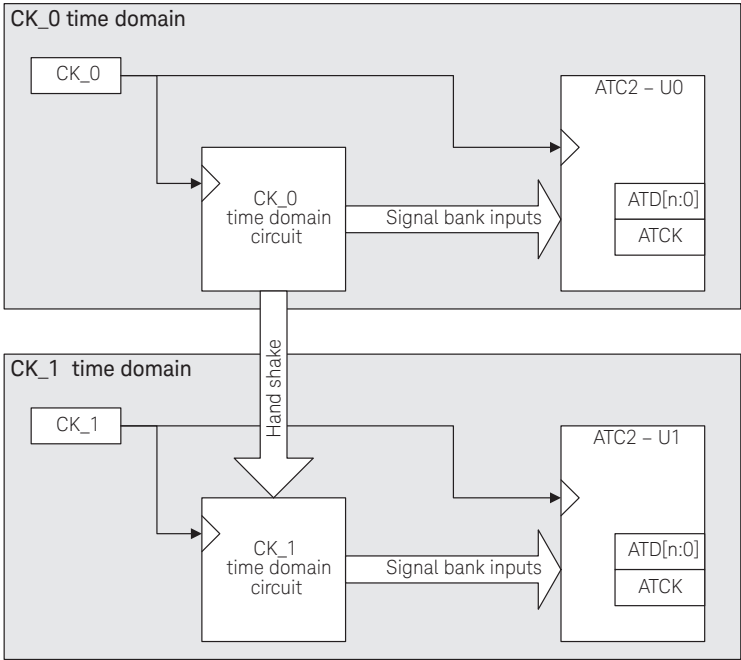


Figure 22. Two state cores for two time bases

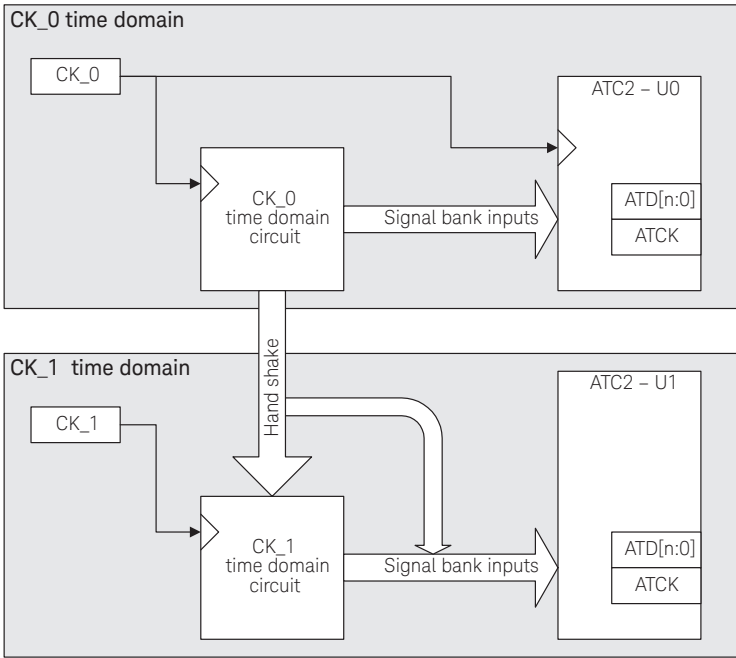


Figure 23. One state core and one timing core

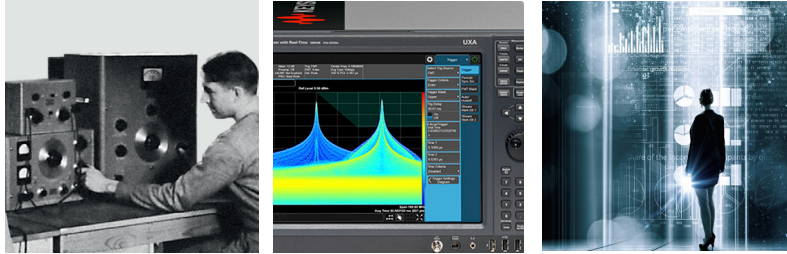
References

- [1] M. Keating and P. Bricaud, Reuse Methodology Manual. Boston: Kluwer Academic Publishers, p 51, 2002
- [2] "Keysight Trace Core Resource Calculator," Keysight Technologies, http://we.home.keysight.com/upload/cmc_upload/All/ATC2_resource_calculation_0408.xls, 2004
- [3] "ChipScope Pro," Xilinx Corporation, http://www.xilinx.com/ise/verification/chip-scope_pro.htm, 2004
- [4] "Probing Solutions for Logic Analyzers," Keysight Technologies, <http://cp.literature.keysight.com/litweb/pdf/5968-4632E.pdf>, 2004
- [5] "Xilinx Parallel Cable IV," Xilinx Corporation, <http://www.xilinx.com/isp/cableQSG.pdf>, 2004

Evolving Since 1939

Our unique combination of hardware, software, services, and people can help you reach your next breakthrough. We are unlocking the future of technology.

From Hewlett-Packard to Agilent to Keysight.



myKeysight

myKeysight

www.keysight.com/find/mykeysight

A personalized view into the information most relevant to you.

http://www.keysight.com/find/emt_product_registration

Register your products to get up-to-date product information and find warranty information.

KEYSIGHT SERVICES

Accelerate Technology Adoption.
Lower costs.

Keysight Services

www.keysight.com/find/service

Keysight Services can help from acquisition to renewal across your instrument's lifecycle. Our comprehensive service offerings—one-stop calibration, repair, asset management, technology refresh, consulting, training and more—helps you improve product quality and lower costs.



Keysight Assurance Plans

www.keysight.com/find/AssurancePlans

Up to ten years of protection and no budgetary surprises to ensure your instruments are operating to specification, so you can rely on accurate measurements.

Keysight Channel Partners

www.keysight.com/find/channelpartners

Get the best of both worlds: Keysight's measurement expertise and product breadth, combined with channel partner convenience.

For more information on Keysight Technologies' products, applications or services, please contact your local Keysight office. The complete list is available at:

www.keysight.com/find/contactus

Americas

Canada	(877) 894 4414
Brazil	55 11 3351 7010
Mexico	001 800 254 2440
United States	(800) 829 4444

Asia Pacific

Australia	1 800 629 485
China	800 810 0189
Hong Kong	800 938 693
India	1 800 11 2626
Japan	0120 (421) 345
Korea	080 769 0800
Malaysia	1 800 888 848
Singapore	1 800 375 8100
Taiwan	0800 047 866
Other AP Countries	(65) 6375 8100

Europe & Middle East

Austria	0800 001122
Belgium	0800 58580
Finland	0800 523252
France	0805 980333
Germany	0800 6270999
Ireland	1800 832700
Israel	1 809 343051
Italy	800 599100
Luxembourg	+32 800 58580
Netherlands	0800 0233200
Russia	8800 5009286
Spain	800 000154
Sweden	0200 882255
Switzerland	0800 805353
	Opt. 1 (DE)
	Opt. 2 (FR)
	Opt. 3 (IT)
United Kingdom	0800 0260637

For other unlisted countries:

www.keysight.com/find/contactus
(BP-9-7-17)

DEKRA Certified
ISO 9001 Quality Management System

www.keysight.com/go/quality
Keysight Technologies, Inc.
DEKRA Certified ISO 9001:2015
Quality Management System



This information is subject to change without notice.
© Keysight Technologies, 2017
Published in USA, December 1, 2017
5989-1593EN
www.keysight.com