LogiCORE IP ChipScope Pro Integrated Controller (ICON) (v1.05a)

DS646 June 22, 2011

Product Specification

Introduction

The LogiCORE™ IP ChipScope™ Pro Integrated CONtroller core (ICON) provides an interface between the JTAG Boundary Scan (BSCAN) component of the FPGA and the ChipScope Pro cores, including the following types of cores:

- Integrated Logic Analyzer (ILA)
- Virtual Input/Output (VIO)
- ChipScope AXI Monitor
- Agilent Trace Core 2 (ATC2)

These debug cores are directly attached to the ICON core control ports. Once generated, the ICON core is easily instantiated and connected to these cores using standard Verilog and VHDL syntax. The ICON core can be added to an embedded processor system design using the Xilinx Embedded Development Kit (EDK) tools. Finally, through the core inserter interface the ICON core is automatically added without requiring any change to the Verilog or VHDL sources.

Features

- Provides a communication path, using the JTAG port, between the ChipScope Pro Analyzer software and the ILA, VIO, ATC2, and the ChipScope AXI monitor cores
- Connects to the JTAG chain through the USER scan chain feature of the BSCAN component
- Supports up to 15 ChipScope debug cores attached to the ICON control ports
- Supports internal or external BSCAN connection

LogiCORE IP Facts Table					
Core Specifics					
Supported Device Family (1)	Artix-7(3), Virtex-7(3), Kintex-7(3), Virtex-6, Virtex-5, Virtex-4, Spartan-6/6L, Spartan-3E, Spartan-3, Spartan-3A/Spartan-3A DSP, XA Spartan-3/3A DSP, XA Spartan-3E, XA Spartan-6, XA Virtex-4, Spartan-6Q/6QL, Virtex-4Q, Virtex-5Q, Virtex-6Q/6QL				
Supported User Interfaces	Not applicable.				
Provided with Core					
	Resources				**Frequency**
Configuration (4)	**LUTs**	**FFs**	**DSP Slices**	**Block RAMs**	**Max Freq**
Config1	90	108	0	0	398.867 MHz
Config2	317	335	0	0	384.919 MHz
Config3	541	559	0	0	380.055 MHz
Documentation	Product Specification User Guide				
Design Files	Netlist				
Example Design	Verilog/VHDL				
Test Bench	Provided				
Constraints File	Xilinx Constraints				
Simulation Model	Not Provided				
Tested Design Tools (2)					
Design Entry Tools	Core Generator tool, System Generator, XPS				
Simulation	Not provided				
Support					
Provided by Xilinx, Inc.					

Notes:

1. For a complete listing of supported derivative devices, see the IDS Embedded Edition Derivative Device Support.
2. For a listing of the supported tool versions, see the ISE Design Suite 13: Release Note Guide.
3. For more information, see DS180 7 Series FPGAs Overview.
4. For configuration details, see Table 4, page 10.

© Copyright 2008-2009, 2011 Xilinx, Inc. XILINX, the Xilinx logo, Kintex, Virtex, Spartan, ISE and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

Functional Description

The ICON core provides an interface between the ChipScope Pro Analyzer tool and up to 15 ChipScope Pro target cores (such as ILA, IBA, VIO, and ATC2) via the JTAG Boundary Scan port of the target FPGA. The ICON core bridges the gap between the JTAG Boundary Scan TAP controller of the FPGA and the target cores using the USER scan chains provided by the BSCAN primitive component of the FPGA. The ICON core is responsible for routing various commands sent from the Analyzer tool to the intended target cores.

The ICON core can be configured to automatically include the BSCAN primitive component (see Figure 1) or to use a BSCAN elsewhere in the design (see Figure 3, page 3). The ICON core can also route unused BSCAN USER scan chains to port signals if the BSCAN is included in the ICON core and the FPGA device family supports multiple USER scan chains per BSCAN component (see Figure 3, page 3).

The ICON core connection to the target cores is implemented as a dedicated bi-directional control port. This control port includes JTAG clock, input and output data, and control signals necessary to configure and communicate with the target core.

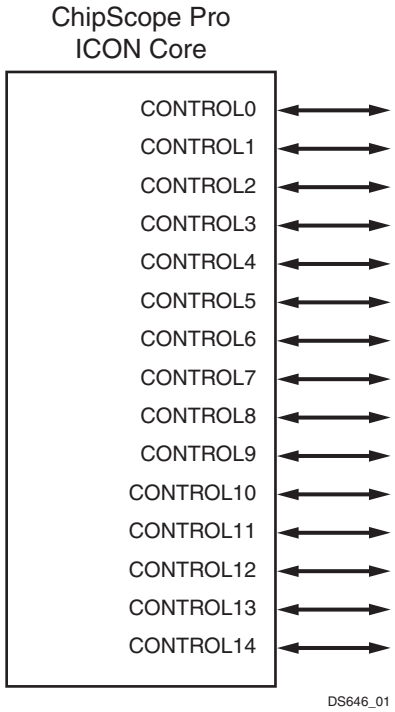

Figure 1: **ICON Core with Internal BSCAN Component**

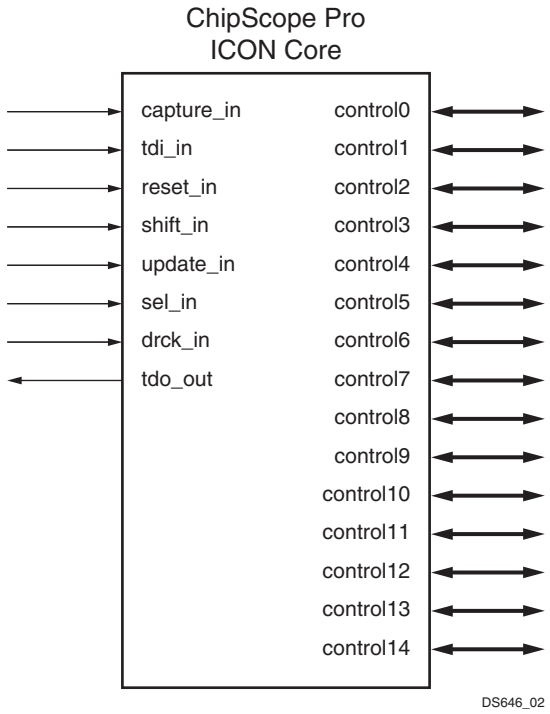

Figure 2: **ICON Core with External BSCAN Component**

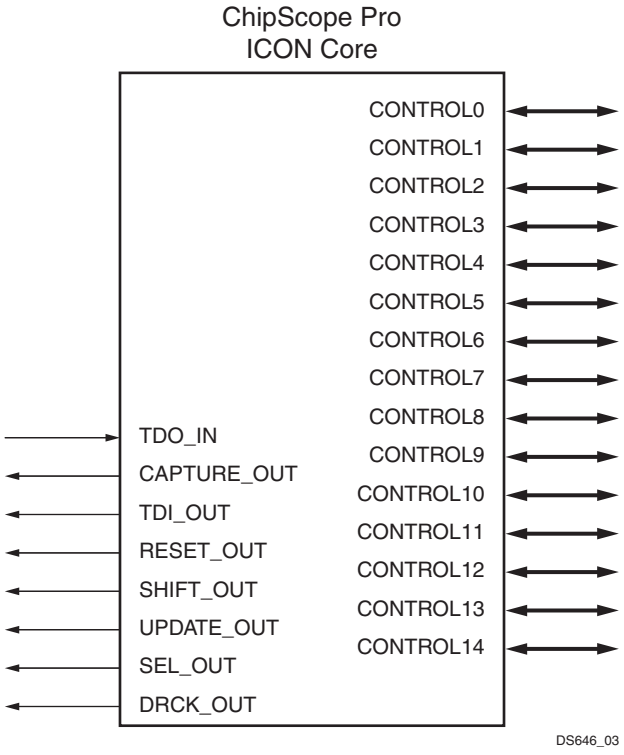

Figure 3: **ICON Core with Internal BSCAN Component and Exported Unused BSCAN Signals**

CORE Generator

The CORE Generator tool provides the ability to define and generate a customized ICON core to use with one or more ILA, VIO, or ATC2 capture cores in HDL designs. The control ports can be customized (that is, the number of cores to be connected to the ICON core); in addition, the use of the Boundary Scan primitive component can be customized (for example, BSCAN_VIRTEX5) that is used for JTAG communication.

Generating an Example Design

The ICON core generator normally generates standard Xilinx CORE Generator output files only, such as netlist and instantiation template files. To generate an example design that uses the ICON core, in addition to the normal generated files, select the Generate Example Design check box. This parameter is stored as example_design in the generated XCO parameter file.

Entering the Number of Control Ports

The ICON core can communicate with any combination of up to 15 ILA, VIO, or ATC2 capture core units at any given time by controlling each core with a distinct control port. The number of control portsYou can be selected from the Number of Control Ports pull-down list. This parameter is stored as number_control_ports in the generated XCO parameter file.

Disabling the Boundary Scan Component Instance

The Boundary Scan primitive component (for example, BSCAN_VIRTEX5) is used to communicate with the JTAG Boundary Scan logic of the target FPGA device. The Boundary Scan component extends the JTAG test access port (TAP) interface of the FPGA device so that up to four internal scan chains can be created. The Boundary Scan component is instantiated inside the ICON core by default. Use the Disable Boundary Scan Component Instance check box, stored as use_ext_bscan in the generated XCO parameter file, to disable the instantiation of the Boundary Scan component.

Selecting the Boundary Scan Chain

The Analyzer can communicate with the cores using either the USER1, USER2, USER3, or USER4 boundary scan chains. If the Boundary Scan component is instantiated inside the ICON core, the user can select the desired scan chain from the Boundary Scan Chain pull-down list. This parameter is stored as user_scan_chain in the generated XCO parameter file.

Enabling Unused Boundary Scan Ports

This feature is only available for Spartan®-3, Spartan-3E, Spartan-3A, and Spartan-3A DSP devices. The Boundary Scan primitive for Virtex®-4, Virtex-5, Virtex-6, and Spartan-6, Artix™-7, Kintex™-7, and Virtex-7 devices (including the variants of these families) can have only one of four sets of ports enabled at any given time.

The Boundary Scan primitive for Spartan-3, Spartan-3E, Spartan-3A, and Spartan-3A DSP devices (including the variants of these families) always has two sets of ports: USER1 and USER2. The ICON core uses only one of the USER* scan chain ports for communication purposes, therefore, the unused USER* port signals are available for use by other design elements, respectively. If the Boundary Scan component is instantiated inside the ICON core, selecting the Enable Unused Boundary Scan Ports check box provides access to the unused USER* scan chain interfaces of the Boundary Scan component. This parameter is stored as use_unused_bscan in the generated XCO parameter file.

Generating the Core

After entering the ICON core parameters, click Generate to create the ICON core files. After the ICON core has been generated, a list of the generated files will appear in a separate window called "Readme <corename>".

Using the ICON Core

To instantiate the example ICON core HDL files into the design, use the following guidelines to connect the ICON core port signals to various signals in your design:

- Connect one of the ICON core's unused CONTROL* port signals to a control port of only one ILA, VIO, or ATC2 core instance in the design.
- Do not leave any unused CONTROL* ports of the ICON core unconnected as this will cause the implementation tools to report an error. Instead, use an ICON core with the same number of CONTROL* ports as you have ILA, VIO or ATC2 cores

Example 1: ICON Connection in VHDL and Example 2: ICON Connection in Verilog show how the ICON core is connected in vhdl and verilog respectively. Note how the control bus control0 is attached to the control port of the VIO. Note how in the Verilog example an empty module declaration was created ICON and VIO. This is used as a black box declaration so that the synthesis tool properly accounts for the generated netlists.

Example 1: ICON Connection in VHDL

entity example_chipscope_icon is
end example_chipscope_icon;

```
architecture icon_arch of example_chipscope_icon iS
  ------------------------------------------------------------------
  --
  --  Component declarations
  --
  ------------------------------------------------------------------------
 component chipscope_icon
   port (
    CONTROL0 : inout std_logic_vector(35 downto 0));
 end component;
 component chipscope_vio
   port (
    CONTROL : inout std_logic_vector(35 downto 0);
    ASYNC_IN : in std_logic_vector(7 downto 0));
 end component;
 ------------------------------------------------------------------
 --  Local Signals
 ------------------------------------------------------------------
 signal control0 : std_logic_vector(35 downto 0);
begin
 ------------------------------------------------------------------
 --
 --  ICON Pro core instance
 --
 ------------------------------------------------------------------
 ICON_inst:  chipscope_icon
   port map (
     CONTROL0 => control0);

 ------------------------------------------------------------------
```

```
 --
 --  VIO Pro core instance
 --
 ------------------------------------------------------------------
 -- For Control Port0
 VIO_inst0:  chipscope_vio
   port map (
     CONTROL => control0,
     ASYNC_IN => "00000001");
end icon_arch;
```

Example 2: ICON Connection in Verilog

module example_chipscope_icon ();

```
  //------------------------------------------------------------------
  //  Local Signals
  //------------------------------------------------------------------
  wire [35: 0] control0;

  //------------------------------------------------------------------
  //
  //  ICON Pro core instance
  //
  //------------------------------------------------------------------

  chipscope_icon ICON_inst
    (
      .CONTROL0(control0));
  //------------------------------------------------------------------
  //
  //  VIO Pro core instance
  //
  //------------------------------------------------------------------
  // For Control Port0
  chipscope_vio VIO_inst0
    (
      .CONTROL(control0),
      .ASYNC_IN(8'd1));
endmodule

//------------------------------------------------------------------
//
//  ICON Pro core module declaration
//
//------------------------------------------------------------------
module chipscope_icon
  (
    inout [35:0] CONTROL0);
endmodule

//------------------------------------------------------------------
//
//  VIO Pro core module declaration
//
//------------------------------------------------------------------
```

```
module chipscope_vio
  (
    inout [35: 0] CONTROL,
    input [ 7: 0] ASYNC_IN);
endmodule
```

Xilinx Platform Studio

Using ICON Core in XPS

The ICON core can be inserted into an embedded processor design using the Xilinx Platform Studio (XPS). In this case, the ICON core depends on a BSCAN component instance whose interface is exported by the OPB_MDM peripheral component (see Figure 4).

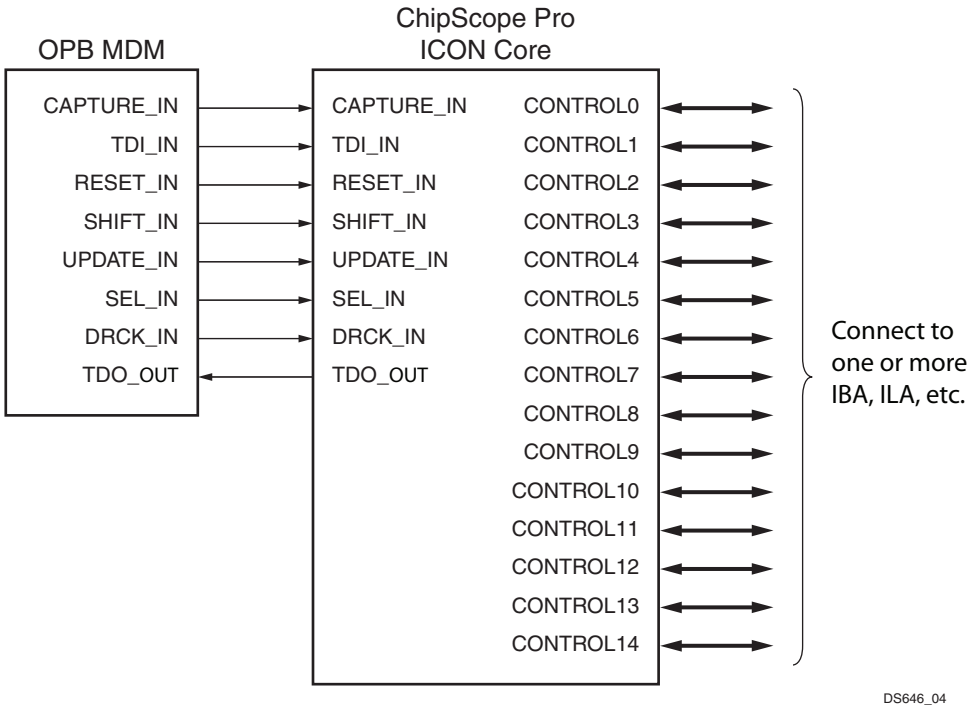

Figure 4: **ICON Core Component in EDK XPS Design**

In EDK, the ICON core is integrated into the tool using a Tcl script. When the EDK Hardware Platform Generator (Platgen) tool runs, the Tcl script is called and the script internally calls the CORE Generator™ (Coregen) tool in the command line mode. The Tcl script provides the CORE Generator software a parameters file (.xco) to generate the ICON core netlist. The Tcl script also generates an HDL wrapper to match the ICON ports based on the core parameters listed in Figure 4. The XST synthesis tool is used for synthesizing the wrapper HDL generated for the ICON core. The NGC netlist outputs from XST and ChipScope Pro Core Generator are subsequently incorporated into the Xilinx ISE® tool suite for actual device implementation.

Ports and Parameters

Ports

The ICON Interface Ports are shown and described in Table 1.

Table 1: **ICON Interface Ports**

Port Name	Direction	Description
CAPTURE_IN	IN	CAPTURE signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
CAPTURE_OUT	OUT	CAPTURE signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).
CONTROL0[35:0]	INOUT(1)	Provides control and status connection to the first ChipScope Pro target core. Mandatory.
CONTROL1[35:0]	INOUT(1)	Provides control and status connection to the second ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL2[35:0]	INOUT(1)	Provides control and status connection to the third ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL3[35:0]	INOUT(1)	Provides control and status connection to the fourth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL4[35:0]	INOUT(1)	Provides control and status connection to the fifth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL5[35:0]	INOUT(1)	Provides control and status connection to the sixth ChipScope Pro target core. Optional (depends on *number_control_ports*parameter).
CONTROL6[35:0]	INOUT(1)	Provides control and status connection to the seventh ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL7[35:0]	INOUT(1)	Provides control and status connection to the eighth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL8[35:0]	INOUT(1)	Provides control and status connection to the ninth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL9[35:0]	INOUT(1)	Provides control and status connection to the tenth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL10[35:0]	INOUT(1)	Provides control and status connection to the eleventh ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL11[35:0]	INOUT(1)	Provides control and status connection to the twelfth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL12[35:0]	INOUT(1)	Provides control and status connection to the thirteenth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL13[35:0]	INOUT(1)	Provides control and status connection to the fourteenth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
CONTROL14[35:0]	INOUT(1)	Provides control and status connection to the fifteenth ChipScope Pro target core. Optional (depends on *number_control_ports* parameter).
DRCK_IN	IN	DRCK signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
DRCK_OUT	OUT	DRCK signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).
RESET_IN	IN	RESET signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
RESET_OUT	OUT	RESET signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).

Table 1: **ICON Interface Ports *(Cont'd)***

Port Name	Direction	Description
SEL_IN	IN	SEL signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
SEL_OUT	OUT	SEL signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).
SHIFT_IN	IN	SHIFT signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
SHIFT_OUT	OUT	SHIFT signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).
TDI_IN	IN	TDI signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
TDI_OUT	OUT	TDI signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).
TDO_IN	IN	TDO signal to the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).
TDO_OUT	OUT	TDO signal to the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
UPDATE_IN	IN	UPDATE signal from the external BSCAN component. Optional (depends on *use_ext_bscan* parameter).
UPDATE_OUT	OUT	UPDATE signal from the unused USER scan chain of the internal BSCAN component. Optional (depends on *use_unused_bscan* parameter).

Notes:

1. For projects created using Platform Studio, the direction for CONTROL ports is OUT.

Parameters

CORE Generator Parameters

The ICON XCO parameters are shown and described in Table 2.

Table 2: **ICON XCO Parameters**

Parameter Name	Allowable Values	Default Value	Description
component_name	String with A-z, 0-9, and _ (underscore)	icon	Name of instantiated component
number_control_ports	1 to 15	1	Number of ChipScope Pro target cores to be connected to this ICON core.
use_ext_bscan	False = Use internal True = Use external	False	Use an external or internal instance of the BSCAN primitive component
use_unused_bscan	False = Don't bond out True = Bond out	0	Indicates whether or not unused internal BSCAN component signals are bonded out to ports.
user_scan_chain	USER1, USER2, USER3, USER4	USER1	BSCAN USER scan chain number to be used by the ICON core (1)
example_design	False - do not generate, True - Generate Example	false	Enable generation of an example design for the core.

Notes:

1. Only USER1 and USER2 are supported for Spartan-3/XA, Spartan-3E/XA, and Spartan-3A/3AN/3A DSP/XA. All other device families support USER1, USER 2, USER3 and USER4.

XPS Parameters

The EDK-specific parameters are listed and described in Table 3.

Table 3: **ICON EDK-Specific Parameters**

Parameter Name	Allowable Values	Default Value	Description
c_family	virtex4, virtex-5, virtex6, virtex6l, virtex7, kintex7, spartan3, spartan3a, spartan3adsp, spartan3e, spartan6, spartan6l, aspartan3, aspartan3a, aspartan3adsp, aspartan3e, aspartan6, avirtex4, qspartan6, qspartan6l, qvirtex4, qspartan6, qspartan6l, qvirtex4, qvirtex5, qvirtex6	N/A	Target FPGA device family.
c_force_bscan_user_port	Integer: 1, 3, or 4 (port 2 is used by OPB_MDM)	1	BSCAN USER scan chain number to be used by the ICON core.
c_num_control_ports	Integer: 1-15	1	Number of ChipScope Pro target cores to be connected to this ICON core.
c_system_contains_mdm	Integer: 1 = system contains MDM 0= system does not contain MDM	0 (automatically calculated by tools)	Indicates whether or not the system containing the ICON core also contains the OPB_MDM peripheral. This parameter dictates whether a BSCAN component should be instantiated.

Performance and Resource Utilization

The configuration details for various devices and set ups is shown in Table 4.

Table 4: **Configuration Details**

Configuration Name	Device	ICON Setup
Config1	Xc7v450t-2ffg1761	Control Ports (1)
Config2	Xc7v450t-2ffg1761	Control Ports (8)
Config3	Xc7v450t-2ffg1761	Control Ports (16)

Verification

Xilinx has verified the ICON core in a proprietary test environment using an internally developed bus functional model.

References

1. More information on the ChipScope Pro software and cores is available in the Software and Cores User Guide, located at http://www.xilinx.com/documentation.
2. Information about hardware debugging using ChipScope Pro in EDK is available in the Platform Studio online help, located at http://www.xilinx.com/documentation.
3. Information about hardware debugging using ChipScope Pro in System Generator for DSP is available in the Xilinx System Generator for DSP User Guide, located at http://www.xilinx.com/documentation.

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx ISE® Design Suite Embedded Edition software under the terms of the Xilinx End User License. The core is generated using the Xilinx ISE Design Suite software. For more information, visit the Chipscope ICON page.

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE modules and software, please contact your local Xilinx sales representative.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
03/24/08	1.0	Release 10.1 (Initial Xilinx release).
09/19/08	1.1	Release 10.1, Service Pack 3 changes.
04/07/09	2.0	Release 11.1.
06/24/09	2.1	Release 11.2.
6/22/11	2.2	Updated to v1.05a for the 13.2 release.

Notice of Disclaimer

Xilinx is providing this product documentation, hereinafter "Information," to you "AS IS" with no warranty of any kind, express or implied. Xilinx makes no representation that the Information, or any particular implementation thereof, is free from any claims of infringement. You are responsible for obtaining any rights you may require for any implementation based on the Information. All specifications are subject to change without notice. XILINX EXPRESSLY DISCLAIMS ANY WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE INFORMATION OR ANY IMPLEMENTATION BASED THEREON, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF INFRINGEMENT AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Except as stated herein, none of the Information may be copied, reproduced, distributed, republished, downloaded, displayed, posted, or transmitted in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx.