LogiCORE IP Virtual Input/Output (VIO) v2.0

Product Guide for Vivado Design Suite

PG159 March 20, 2013

XILINX®

XILINX

Table of Contents

Appendix C: Additional Resources

Introduction

The LogiCORE™ IP Virtual Input/Output (VIO) core is a customizable core that can both monitor and drive internal FPGA signals in real time. The number and width of the input and output ports are customizable in size to interface with the FPGA design. Because the VIO core is synchronous to the design being monitored and/or driven, all design clock constraints that are applied to your design are also applied to the components inside the VIO core. Run time interaction with this core requires the use of the Vivado™ logic analyzer feature.

Features

- Provides virtual LEDs and other status indicators through input ports.
- Includes optional activity detectors on input ports to detect rising and falling transitions between samples.
- Provides virtual buttons and other controls through output ports.
- Includes custom output initialization that allows you to specify the value of the VIO core outputs immediately following device configuration and start-up.
- Run time reset of the VIO core to its initial values.

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1)	Virtex®-7, Kintex™-7, Artix™-7
Supported User Interfaces	N/A
Resources	See Table 2-1.
Provided with Core	
Design Files	Encrypted RTL
Example Design	Verilog and VHDL
Test Bench	Not Provided
Constraints File	XDC
Simulation Model	Not Provided
Supported S/W Driver	N/A
Tested Design Flows(2)	
Design Entry	Vivado™ Design Suite
Simulation	Not Provided
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete listing of supported devices, see the Vivado IP catalog.
2. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The VIO core is a customizable core that can both monitor and drive internal FPGA signals in real time. Unlike the ILA core, no on-chip or off-chip RAM is required.

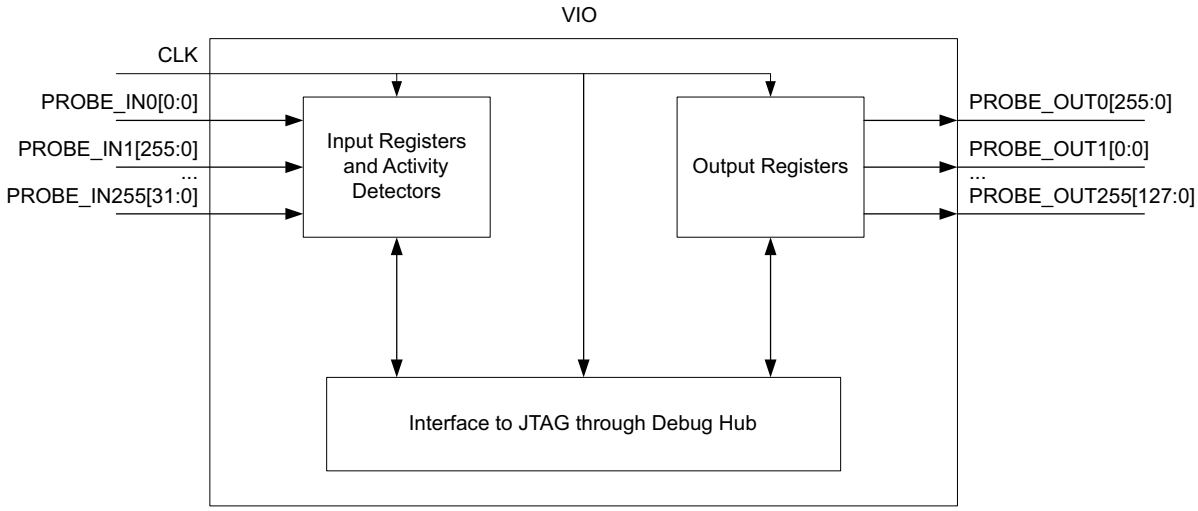

Figure 1-1: **VIO Block Diagram**

Feature Summary

Two types of signals are available in the VIO core:

- Input probes
- Output probes

Input Probes

These inputs to the VIO core registered using the design clock that is attached to the CLK input port. The input values are read back periodically and displayed in the Vivado™ logic analyzer feature.

Output Probes

These outputs from the VIO core are driven to the surrounding user design. The output values are driven to combinations of 1s and 0s by the Vivado logic analyzer feature. Output probes can also be initialized to any desired value during generation time.

Activity Detectors

Every VIO core input has additional cells to capture the presence of transitions on the input. Because the design clock is most likely much faster than the sample period of the Analyzer, it is possible for the signal being monitored to transition many times between successive samples. The activity detectors capture this behavior and the results are displayed along with the value in the Vivado logic analyzer.

Licensing and Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx Vivado Design Suite tool under the terms of the Xilinx End User License.

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 IP modules and tools, contact your local Xilinx sales representative.

Product Specification

The VIO core is used to drive data into your design and read data from your design through the JTAG port. The core is designed to replace or augment board-level I/O components such as status indicators (for example, LEDs) and low-bandwidth controls (for example, buttons or DIP switches).

Performance

Maximum Frequency

The VIO core is designed to run at design clock frequencies up to 250 MHz, but maximum clock frequency could be limited by other factors in the design such as overall utilization or routing congestion.

Resource Utilization

Resources required for the VIO core have been estimated for the Kintex™-7 XC7K325T FPGA for several parameters, as shown in Table 2-1. These values were generated using Vivado™ IP Catalog. They are derived from post-synthesis reports and might change during implementation.

Table 2-1: **FPGA Resource Estimates**

Number of Ports	Width of Ports	Activity Detection	LUTs	Flip-Flops	Block RAMs	DSP Slices
1 input, 1 output	1-bit each	Enabled	75	108	0	0
1 input, 1 output	32 bits each	Enabled	187	361	0	0
4 inputs, 4 outputs	64 bits each	Enabled	964	1953	0	0
4 inputs, 4 outputs	64 bits each	Disabled	243	1439	0	0

Port Descriptions

Table 2-2 shows the VIO core I/O port signals.

Table 2-2: **VIO I/O Signals**

Signal Name	I/O	Description
CLK	I	Design clock used to register input ports and output ports. This is required.
PROBE_IN<m>[<n> - 1:0]	I	Input probe port number <m> (where <m> can be 0 to 256) of width <n> (where <n> can be 1 to 256). For a 1-bit wide port, use PROBE_IN<m>[0:0].
PROBE_OUT<m>[<n> - 1:0]	O	Output probe port number <m> (where <m> can be 0 to 256) of width <n> (where <n> can be 1 to 256). For a 1-bit wide port, use PROBE_OUT<m>[0:0].

Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the core.

General Design Guidelines

The VIO core can be used to replace or augment any board-level status indicators (such as LEDs) and controls (such as buttons or DIP switches).

Clocking

The CLK input port is the clock used by the VIO core to register input and output probe values. For best results, it should be the same clock signal that is synchronous to the design logic that is attached to the input and output probe ports of the VIO core.

Resets

The VIO core output probes can only be reset to their initial values by using the Vivado™ logic analyzer feature. Similarly, the VIO core input probe activity detectors can only be reset using the Vivado logic analyzer.

Customizing and Generating the Core

This chapter includes information about using Xilinx tools to customize and generate the core in the Vivado™ Design Suite environment.

Vivado Integrated Design Environment (IDE)

The VIO core can be found in `/Debug & Verification/Debug/` in the Vivado IP Catalog (Figure 4-1).

To access the core name, perform the following:

1. Open a project by selecting **File** then **Open Project** or create a new project by selecting **File** then **New Project** in Vivado.
2. Open the IP catalog and navigate to any of the taxonomies.
3. Double-click **VIO** to bring up the VIO Customize IP dialog box (Figure 4-1).

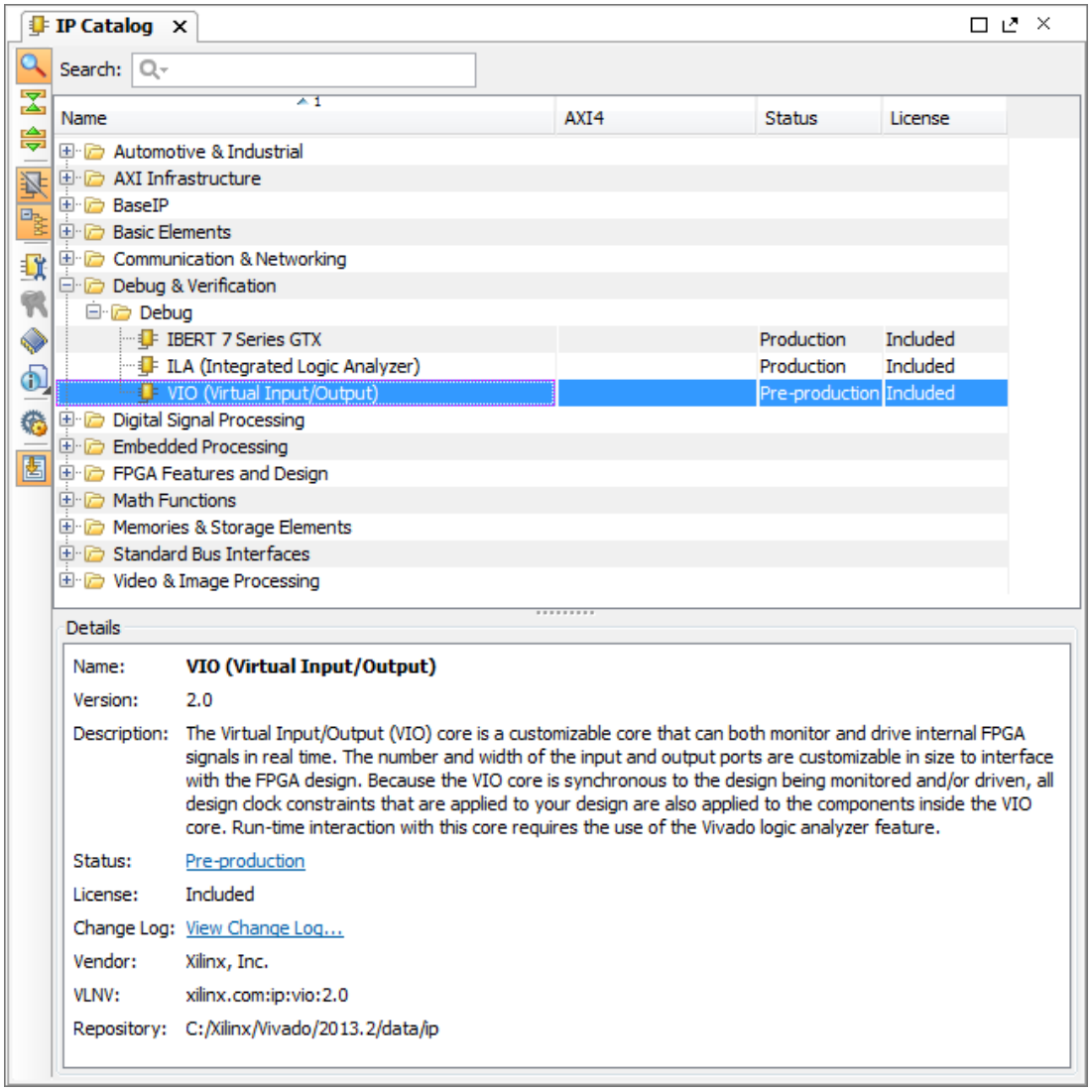

Figure 4-1: **VIO Core in Vivado IP Catalog**

General Options Panel

- **Component Name** – Use this text field to provide a unique module name for the VIO core.
- **Input Probe Count** – Use this text field to select the number of input probe ports on the VIO core. The valid range used in the GUI is 0 to 64. If you need more than 64 input probe ports, you need to use the Tcl command flow to generate the VIO core.

  Note: At least one input or output probe port needs to be specified.
- **Output Probe Count** – Use this text field to select the number of output probe ports on the VIO core. The valid range used in the GUI is 0 to 64. If you need more than 64 output probe ports, you need to use the Tcl command flow to generate the VIO core.

  Note: At least one input or output probe port needs to be specified.

- **PROBE_IN Ports Panels** – Each PROBE_IN Ports panel has up to 16 ports.
- **Probe Width** – Use the Probe Width field to set the width of each probe port. The valid width range is 1 to 256 bits wide.
- **PROBE_OUT Ports Panels** – Each PROBE_OUT Ports panel has up to 16 ports.
- **Probe Width** – Use the Probe Width field to set the width of each probe port. The valid width range is 1 to 256 bits wide.
- **Initial Value** – Use the Initial Value field to set the initial value of the output probe port to a specific value. The value is to be specified in hexadecimal format with a "0x" prefix.

Output Generation

This section describes the files and directory structure generated by the IP. For the purposes of this document, assume the name of the project is the default "project_1."

- **project_1/project_1.srcs**
  Top-level project directory; name is user-defined
  - sources_1/ip/<component name>
    - constraints
      - vio.xdc
    - labtools_general_components_lib_v2_0/*
    - labtools_xsdb_slave_lib_v2_1/*
    - synth
      - <component name>.v/.vhd
    - vio_v2_0/hdl
      - decoder.v
      - probe_in_one.v
      - probe_out_all.v
      - probe_out_one.v
      - probe_width.v
      - vio_include.v
      - vio.v
    - <component name>.veo/.vho
    - <component name>.xci
    - <component name>.xml

project_1/project_1.srcs/sources_1/ip

This directory contains the source files needed to synthesize the VIO core whose name is `<component name>`.

Table 4-1: **VIO Source Files**

Name	Description
constraints/vio.xdc	Constraints file for the VIO core.
labtools_general_components_lib_v2_0/*	Common components used by all lab tools cores.
labtools_xsdb_slave_lib_v2_1/*	Common interface library used by all lab tools to connect to the dbg_hub core.
synth/<component name>.v/.vhd	Verilog (.v) or VHDL (.vhd) file used by synthesis.
vio_v2_0/hdl/*.v	Verilog source files used to describe the VIO v2.0 core.
<component name>.veo/.vho	Verilog (.veo) or VHDL (.vho) file used to describe the instantiation template for the VIO core.
<component name>.xci	Core description file for the VIO core.
<component name>.xml	Component XML file for the VIO core.

Constraining the Core

This chapter contains information about constraining the core in the Vivado™ Design Suite environment.

Required Constraints

The VIO core includes an XDC file that contains appropriate false path constraints to prevent the over-constraining of clock domain crossing synchronization paths. It is also expected that the clock signal connected to the CLK input port of the VIO core is properly constrained in your design constraints.

Detailed Example Design

This chapter contains information about the provided example design in the Vivado™ Design Suite environment.

Directory and File Contents

<component name>_example/<component name>_example.srcs/
Top-level project directory; name is user-defined

- constrs_1/imports/<component name>/
  - example_<component name>.xdc
- sources_1/imports/<component name>/
  - example_<component name>.v/.vhd
- sources_1/ip/<component name>
  - See Output Generation, page 12 section for details on the files in this directory.

<component name>_example/<component name>_example.srcs/

This directory contains the source files needed to synthesize the VIO core whose name is `<component name>`.

Table 6-1: **VIO Example Design Source Files**

Name	Description
constrs_1/imports/<component name>/	
example_<component name>.xdc	Constraints file for the example design
sources_1/imports/<component name>/	
example_<component name>.v/.vhd	Verilog (.v) or VHDL (.vhd) source file for the example design

Implementation

To implement the example design, select **Run Implementation** in the **Vivado Project Manager** window. For further details on setting up the implementation, see the *Vivado Design Suite User Guide, Implementation* (UG904) [Ref 4].

Migrating

For information on migrating to the Vivado™ Design Suite, see *Vivado Design Suite Migration Methodology Guide* (UG911) [Ref 1].

Port Changes

The CLK port is now required. The ASYNC_IN and ASYNC_OUT ports are no longer available on the VIO v2.0 core.

The SYNC_IN and SYNC_OUT ports are now similar to the PROBE_IN<m> and PROBE_OUT<m> ports, except that VIO v2.0 now allows up to 256 of each kind of port.

Functionality Changes

The VIO v2.0 core no longer has asynchronous input or output ports. All input and output probe ports are assumed to be synchronous to the CLK input port.

IMPORTANT: *The VIO v2.0 core is not compatible with the legacy ChipScope™ Pro Analyzer tool. The VIO v2.0 core requires the Vivado logic analyzer feature for run time interaction.*

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools. In addition, this appendix provides a step-by-step debugging process to guide you through debugging the VIO core.

The following topics are included in this appendix:

Finding Help on Xilinx.com

To help in the design and debug process when using the VIO, the Xilinx Support web page (www.xilinx.com/support) contains key resources such as product documentation, release notes, answer records, information about known issues, and links for opening a Technical Support WebCase.

Documentation

This product guide is the main document associated with the VIO. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page (www.xilinx.com/support) or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Design Tools tab on the Downloads page (www.xilinx.com/download). For more information about this tool and the features available, open the online help after installation.

Known Issues

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core are listed below, and can also be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the VIO

AR 54606

Contacting Technical Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE™ IP 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.

To contact Xilinx Technical Support:

1. Navigate to www.xilinx.com/support.
2. Open a WebCase by selecting the WebCase link located under Support Quick Links.

When opening a WebCase, include:

- Target FPGA including package and speed grade.
- All applicable Xilinx Design Tools and simulator software versions.
- Additional files based on the specific issue might also be required. See the relevant sections in this debug guide for guidelines about which file(s) to include with the WebCase.

Debug Tools

There are many tools available to address VIO design issues. It is important to know which tools are useful for debugging various situations.

Example Design

The VIO is delivered with an example design that can be synthesized, complete with functional test benches. Information about the example design can be found in *Chapter 6, Detailed Example Design*.

Vivado Lab Tools

Vivado inserts logic analyzer and virtual I/O cores directly into your design. Vivado Lab Tools allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature represents the functionality in the Vivado IDE that is used for logic debugging and validation of a design running in Xilinx FPGA devices in hardware.

The Vivado logic analyzer is used to interact with the logic debug LogiCORE IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

License Checkers

If the IP requires a license key, the key must be verified. The Vivado design tools have several license check points for gating licensed IP through the flow. If the license check succeeds, the IP can continue generation. Otherwise, generation halts with error. License checkpoints are enforced by the following tools:

- Vivado Synthesis
- Vivado Implementation
- write_bitstream (Tcl command)

IMPORTANT: *IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.*

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. The Vivado logic analyzer tool is a valuable resource to use in hardware debug.

General Checks

Ensure that all the timing constraints for the core were properly incorporated from the example design and that all constraints were met during implementation.

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the LOCKED port.

Appendix C

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website at:

www.xilinx.com/support.

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm.

References

These documents provide supplemental material useful with this product guide:

1. Vivado™ Design Suite user documentation
2. *Vivado Design Suite User Guide, Programming and Debugging* (UG908)
3. *Vivado Design Suite User Guide, Designing with IP* (UG896)
4. *Vivado Design Suite User Guide, Implementation* (UG904)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/20/2013	1.0	Initial Xilinx release of the product guide and replaces DS284.

Notice of Disclaimer