AXI HWICAP v3.0

LogiCORE IP Product Guide

Vivado Design Suite

PG134 October 5, 2016

Table of Contents

Introduction

The Xilinx® AXI Hardware Internal Configuration Access Port (HWICAP) LogiCORE™ IP core for the AXI Interface enables an embedded microprocessor, such as the MicroBlaze™ processor, to read and write the FPGA configuration memory through the Internal Configuration Access Port (ICAPEn). This enables you to write software programs that modify the circuit structure and functionality during the operation of the circuit.

Features

- Supports resource reading
- Supports long frame reads
- AXI Interface is based on the AXI4-Lite specification
- Partial bitstream loading
- Enables Read/Write of Configurable Logic Block (CLB) Lookup Tables (LUTs)
- Enables Read/Write of CLB Flip-Flop properties
- ICAP arbitration interface to ease sharing of ICAP with other blocks and enable safe hand-off

Note: The ICAPE2 primitive is applicable for 7 series devices. The ICAPE3 primitive is applicable for UltraScale™ devices.

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale+™ UltraScale Zynq-7000 All Programmable SoC 7 Series
Supported User Interfaces	AXI4-Lite
Resources	See Table 2-2
Provided with Core	
Design Files	VHDL
Example Design	VHDL
Test Bench	VHDL
Constraints File	Xilinx Design Constraints (XDC)
Simulation Model	Not Provided
Supported S/W Driver	Standalone
Tested Design Flows[3]	
Design Entry	Vivado® Design Suite
Simulation	For a list of supported simulators, see the Xilinx Design Tools: Release Notes Guide
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx at the Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. For more information, see *7 Series FPGAs Overview Advanced Product Specification* (DS180) [Ref 1].
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The AXI HWICAP controller provides the interface necessary to transfer data to and from the ICAPEn. For writes to the ICAPEn, the required data is first stored within a Write FIFO, from where it can be sent to the ICAPEn. The AXI HWICAP also provides for read back of data from ICAPEn. In this case, the data is read into the Read FIFO.

The AXI HWICAP top-level block diagram is shown in Figure 1-1.

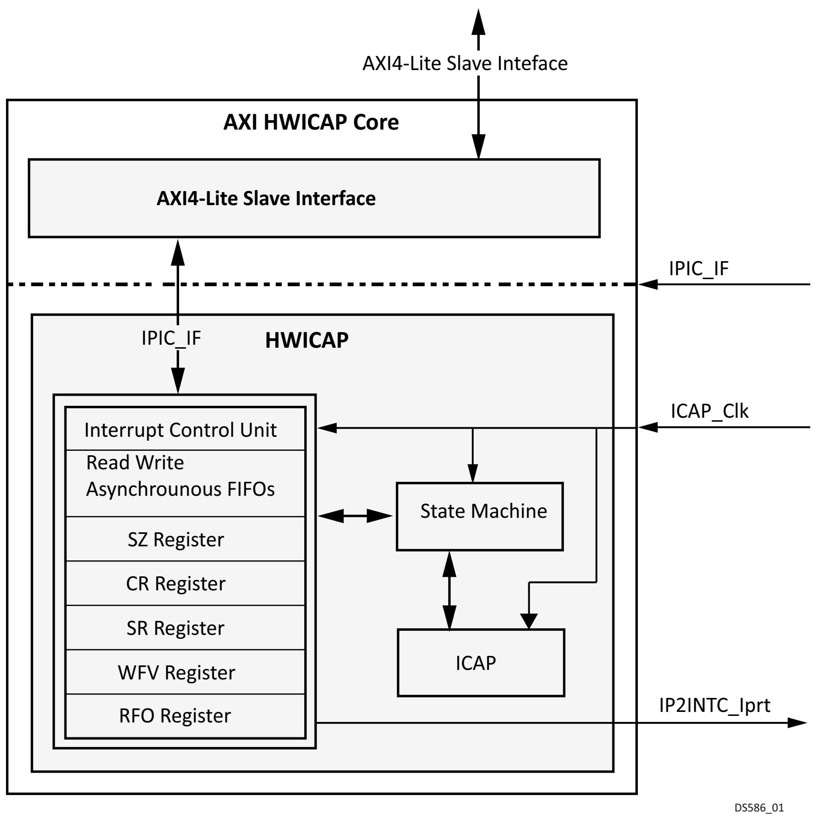


Figure 1-1: **Top Level Block Diagram for the AXI HWICAP Core**

Module Descriptions

AXI4-Lite Interface Module

The AXI4-Lite Interface Module provides the bidirectional interface between the HWICAP core and the AXI. The base element of the AXI Interface Module is the slave attachment, which provides the basic functionality of AXI slave operation.

IPIC_IF Module

The intellectual property interconnect interface (IPIC_IF) module incorporates logic to acknowledge the write and read transactions initiated by the AXI slave module to write or read the HWICAP module registers and FIFOs.

HWICAP Module

The HWICAP module provides the interface to the ICAPEn. It has a write FIFO, which stores the data locally. The data stored in the write FIFO is transferred to the ICAPEn The data that is read from the ICAPEn is stored in the Read FIFO. FIFO depth can be specified while customizing the IP.

ICAP Interface

The ICAP Interface is a point-to-point connection between the HWIPCA controller and the ICAP primitive. The ICAP primitive enables read and write access to the registers inside the FPGA configuration system. The ICAP primitive and the behavior of the signals on this interface are described in the *UltraScale Architecture Configuration User Guide (UG570)* [Ref 18].

This interface is exposed at the core level when the configuration primitives used by the IP (ICAP) are located outside the core (C_ICAP_EXTERNAL = 1).

Note: The ICAPE2 primitive is applicable for 7 series devices. The ICAPE3 primitive is applicable for UltraScale and later devices.

Switching Behavior of ICAP Interface Signals

The switching behavior is illustrated in Figure 1-2. When the IP needs to access the ICAP primitive (in response to AXI4-Lite transactions), the IP asserts the `cap_req` signal to request access to the ICAP primitive. It continuously asserts this signal as long as it needs access to the ICAP primitive. When the `cap_gnt` signal is High, the IP assumes that it has ICAP primitive control.

If the `cap_rel` signal is High, this indicates to the IP that it needs to release the ICAP primitive access. Subsequently, after finishing the ongoing transaction, the IP deasserts its `cap_req` signal. As shown in the figure, `cap_req` is asserted again (as per a new AXI4-Lite transaction request) only after three clock cycles have elapsed following `cap_rel` deassertion.

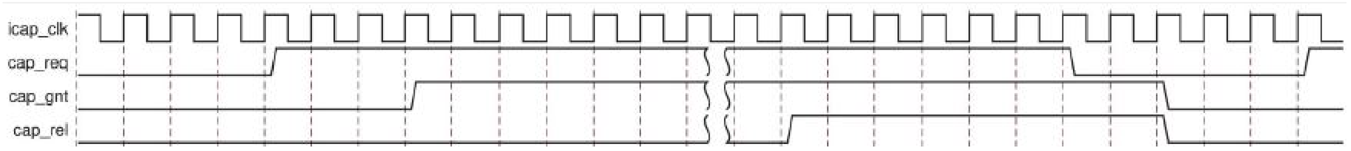


Figure 1-2: **Switching Behavior of ICAP Interface Signals**

Licensing and Ordering Information

This Xilinx LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado® Design Suite 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

Performance

The performance characterization numbers for the AXI HWICAP core are shown in Table 2-1. Performance characterization has been obtained using the margin system methodology. The details of the margin system characterization methodology are described in the "Vivado IP Optimization (Fmax Characterization)" appendix of the *Vivado Design Suite User Guide: Designing With IP (UG896)* [Ref 2].

Note: Performance numbers for Zynq®-7000 devices are expected to be similar to 7 series device numbers.

Table 2-1: **7 Series Performance Characterizations**

Family	Speed Grade	Fmax (MHz) ICAP_Clk Clocks	Fmax (MHz) AXI4-Lite Clock
Virtex®-7	-1	100	180
Kintex®-7		100	180
Artix®-7		100	120
Virtex-7	-2	100	200
Kintex-7		100	200
Artix-7		100	140
Virtex-7	-3	100	220
Kintex-7		100	220
Artix-7		100	160

Maximum Frequencies

The target F_{MAX} is influenced by the switching characteristics of the ICAPEn primitive. This primitive cannot run on frequencies more than 100 MHz.

Resource Utilization

Table 2-2 shows the AXI HWICAP resource utilization for various parameter combinations for 7 Series FPGAs.

Note: Utilization numbers for Zynq-7000 and UltraScale™ devices are expected to be similar to 7 series device numbers.

Table 2-2: **Resource Utilization Numbers for 7 Series FPGAs**

Parameter Values				Device Resources		
Disable Write FIFO	**FIFO Type**	**Write FIFO Depth**	**Read FIFO Depth**	**Slices**	**Slice Flip-Flops**	**LUTs**
0	0	128	128	336	704	706
0	0	256	128	365	720	840
0	0	512	128	458	737	1048
0	1	512	128	277	672	544
0	1	1024	128	291	688	538
1	1	256	128	276	587	544

Port Descriptions

Input/Output Signals

The AXI4 HWICAP core Input/Output (I/O) signals are described in Table 2-3.

Table 2-3: **I/O Signals**

Signal Name	Interface	I/O	Initial State	Description
ICAP Interface Signals				
icap_clk [(1)]	ICAPEn	I	-	ICAPEn clock
eos_in[(2)]	NA	I	-	End of Start-up. This pins has to be connected to a valid EOS signal if Start-up primitive is not included in the HWICAP IP. This pin is ignored if the Start-up primitive is instantiated in AXI HWICAP.
AXI Bus Request and Qualifier Signals				
s_axi_aclk	AXI	I	-	AXI Clock
s_axi_aresetn	AXI	I	-	AXI Reset; Active-Low
s_axi_*	AXI	I	-	See Appendix A of the *Vivado AXI Reference Guide* (UG1037) [Ref 3] for AXI4 signals.

Table 2-3: ***(Cont'd)*****I/O Signals** ***(Cont'd)***

Signal Name	Interface	I/O	Initial State	Description
System Signals				
ip2intc_irpt	AXI	O	0	AXI HWICAP Interrupt
ICAP Signals[(3)]				
icap_csib	ICAP	O	0	Active-Low ICAP enable
icap_0	ICAP	O	-	ICAP data output bus
icap_i	ICAP	I	-	ICAP data input bus
icap_rdwrb	ICAP	O	0	ICAP Read Write select input
ICAP Arbiter Signals				
cap_gnt	ICAP_ARBITER	I	-	For use with an ICAP arbiter. This signal is asserted by the arbiter to inform the HW_ICAP controller that it has permission to access the ICAP. After cap_gnt is asserted, it should remain asserted until cap_req is deasserted. If arbitration is not required, tie this signal to a constant 1
cap_rel	ICAP_ARBITER	I	-	This signal indicates to the IP that it should relinquish control of the ICAP at the earliest safe opportunity. If arbitration is not required, tie this signal to a constant 0.
cap_req	ICAP_ARBITER	O	0	For use with an ICAP arbiter. This signal is asserted by the IP on every clock cycle where it has data to transfer to the ICAP.

Notes:

1. 7 series devices have a limiting frequency of maximum 100 MHz on `icap_clk`. In synchronous mode, this port is unused inside the core (i.e., in this mode, the core uses only `s_axi_aclk`). For more details, see the DC and Switching Characteristics documents [Ref 4] [Ref 5] [Ref 6].
2. This input port is used only in 7 series devices when STARTUPE2 is not included in the IP. In other configurations, this port is unused inside the core.
3. These signals are valid only if external ICAP is used (C_ICAP_EXTERNAL = 1).

Register Definition

The register address mapping of the AXI HWICAP are offset from the base address C_BASEADDR. The AXI HWICAP internal register set is described in Table 2-4.

Note: The AXI4-Lite write access register is updated by the 32-bit AXI Write Data (*_wdata) signal, and is not impacted by the AXI Write Data Strobe (*_wstrb) signal. For a Write, both the AXI Write Address Valid (*_awvalid) and AXI Write Data Valid (*_wvalid) signals should be asserted together.

Table 2-4: **Register Address Map**

Address Offset	Register Name	Access	Default Value	Description
1Ch	GIER	Read/Write	0x0	See Global Interrupt Enable Register.
20h	ISR	Read/Write	0x0	See Abort Status Register.
028h	IER	Read/Write	0x0	See IP Interrupt Enable Register.
100h	WF	Write Only	0x0	See Write FIFO Keyhole Register.
104h	RF	Read Only	0x0	See Read FIFO Keyhole Register.
108h	SZ	Write Only	0x0	See Size Register.
10Ch	CR	Read/Write	0x0	See Control Register.
110h	SR	Read Only	0x0	See Status Register.
114h	WFV	Read Only	(1)	See Write FIFO Vacancy Register.
118h	RFO	Read Only	0x0	See Read FIFO Occupancy Register.
11Ch	ASR	Read Only	0x0	See Abort Status Register.

Notes:

1. This value is based on the actual Write FIFO size. For example, if the **Write FIFO depth** is set to 1024 during customization, the actual FIFO depth is 1023 (or 0x3FF). This register reports the *actual* Write FIFO vacancy.

Global Interrupt Enable Register

The Global Interrupt Enable Register (GIER) register shown in Figure 2-1 and described in Table 2-5 is used to globally enable the final interrupt output from the Interrupt Controller. This bit is a read/write bit and is cleared upon reset.

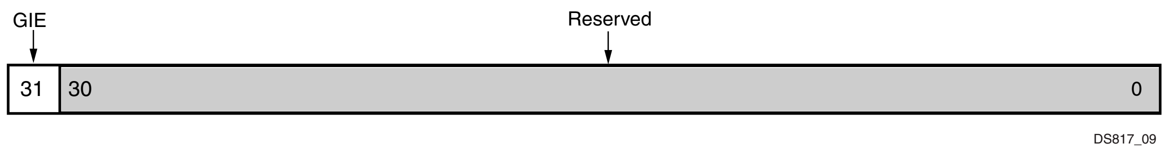


Figure 2-1: **Global Interrupt Enable Register (GIER)**

Table 2-5: **Global Interrupt Enable Register Bit Definitions (01Ch)**

Bits	Name	Access	Reset Value	Description
31	GIE	Read/Write	0	0 = Disabled 1 = Enabled
30–0	Reserved	N/A	N/A	Reserved

IP Interrupt Status Register

Four unique interrupt conditions are possible in the HWICAP core. The Interrupt Controller has a register that can enable each interrupt independently. Bit assignment in the Interrupt register for a 32-bit data bus is shown in Figure 2-2 and described in Table 2-6.

The interrupt register is a read/toggle on write register. By writing a 1 toggles the value of the bit. All register bits are cleared upon reset.

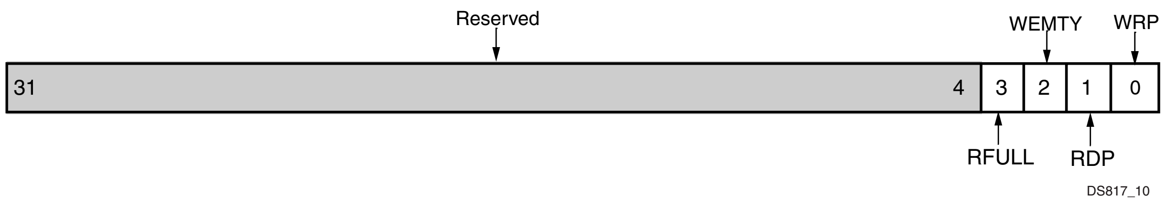


Figure 2-2: **IP Interrupt Status Register (IPISR)**

Table 2-6: **IP Interrupt Status Register Bit Definitions (020h)**

Bits	Name	Access	Reset Value	Description
31–4	Reserved	N/A	N/A	Reserved
3	RFULL	R/TOW(1)	0	Read FIFO full
2	WEMPTY(2)	R/TOW(1)	0	Write FIFO empty
1	RDP	R/TOW(1)	0	Interrupt set and remains set if the read FIFO occupancy is greater than half of the read FIFO depth.
0	WRP(2)	R/TOW(1)	0	Interrupt set and remains set if the write FIFO occupancy is less than half of the write FIFO depth.

Notes:

1. TOW = Toggle On Write. Writing a 1 to a bit position within the register causes the corresponding bit position in the register to toggle.
2. WRP and WEMTY bits are of no importance when Write FIFO is disabled. Xilinx recommends that these interrupts be disabled when HWICAP is configured in Lite Mode.

IP Interrupt Enable Register

The IP Interrupt Enable Register (IPIER) has an enable bit for each defined bit of the IPISR as shown in Figure 2-3 and described in Table 2-7. All bits are cleared upon reset.

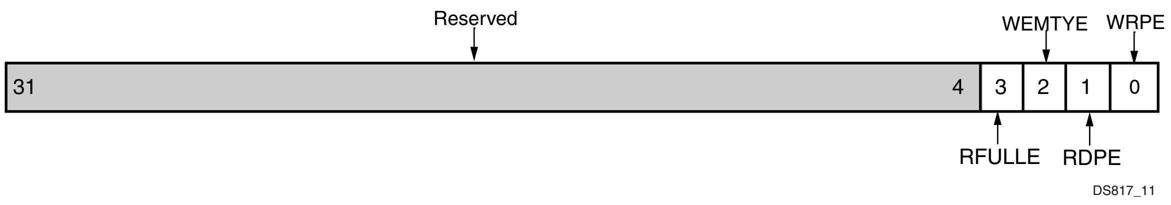


Figure 2-3: **IP Interrupt Enable Register (IPIER)**

Table 2-7: **IP Interrupt Enable Register Bit Definitions (028h)**

Bits	Name	Access	Reset Value	Description
31–4	Reserved	N/A	N/A	Reserved
3	RFULLE	Read/ Write(1)	0	Read FIFO full interrupt enable
2	WEMTYE(2)	Read/ Write(1)	0	Write FIFO empty interrupt enable
1	RDPE	Read/ Write(1)	0	Read FIFO occupancy greater than half of its depth interrupt enable
0	WRPE(2)	Read/ Write(1)	0	Write FIFO occupancy less than half of its depth interrupt enable

Notes:

1. Writing a 1 to this bit enables the particular interrupt. Writing 0 to this bit disables the particular interrupt.
2. WRP and WEMTY bits are of no importance when Write FIFO is disabled. Xilinx recommends that these interrupts be disabled when HWICAP is configured in Lite Mode.

Write FIFO Keyhole Register

The Write FIFO (WF) register shown in Figure 2-4 is a 32-bit keyhole register into a Write FIFO. The bit definitions for the Write FIFO are shown in Table 2-8.

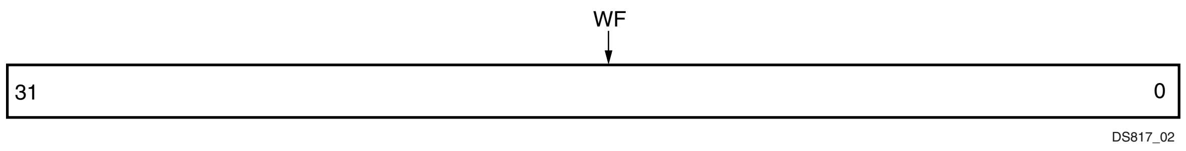


Figure 2-4: **Write FIFO (WF)**

Table 2-8: **Write FIFO Bit Definitions (100h)**

Bits	Name	Access	Reset Value	Description
31–0	WF	Write Only	0	Data written into the FIFO

Read FIFO Keyhole Register

The Read FIFO (RF) register shown in Figure 2-5 is a 32-bit keyhole register into a Read FIFO. The bit definitions for the Read FIFO are shown in Table 2-9.

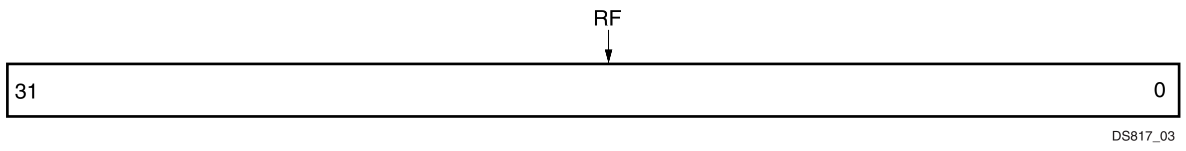


Figure 2-5: **Read FIFO (RF)**

Table 2-9: **Read FIFO Bit Definitions (104h)**

Bits	Name	Access	Reset Value	Description
31–0	RF	Read	0	Data read from the FIFO

Size Register

The Size (SZ) register shown in Figure 2-6 is a 12-bit write only register that determines the number of 32-bit words to be transferred from the ICAPEn to the read FIFO. (This means how many 32-bit data beats are expected.) The bit definitions for the register are shown in Table 2-10.

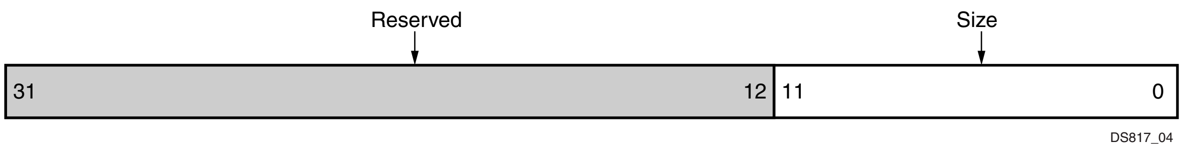


Figure 2-6: **Size Register (SZ)**

Table 2-10: **Size Register Bit Definitions (108h)**

Bits	Name	Access	Reset Value	Description
31–12	Reserved	N/A	0	Reserved bits
11–0	Size	Write Only	0	Number of words to be transferred from the ICAPEn to the FIFO

Control Register

The Control Register (CR) shown in Figure 2-7 is a 32-bit read/write register that determines the direction of the data transfer. It controls whether a configuration or readback takes place. Writing to this register initiates the transfer. The bit definitions for the register are shown in Table 2-11.

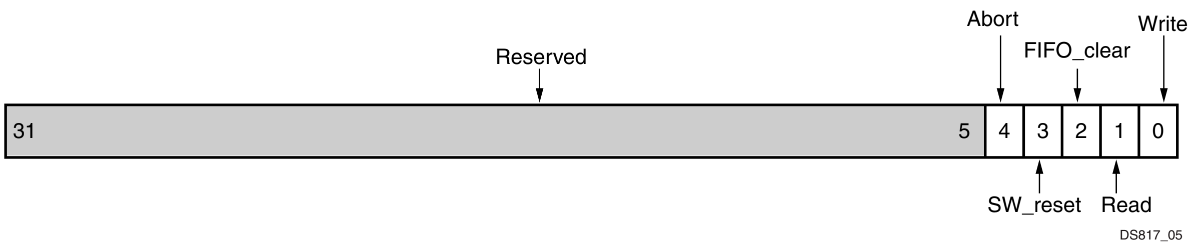


Figure 2-7: **Control Register (CR)**

Table 2-11: **Control Register Bit Definitions (10Ch)**

Bits	Name	Access	Reset Value	Description
31–5	Reserved	N/A	0	Reserved bits
4	Abort	Read/ Write	0	1 = Aborts the read or write of the ICAPEn and clears the FIFOs

Table 2-11: **Control Register Bit Definitions (10Ch) *(Cont'd)***

Bits	Name	Access	Reset Value	Description
3	SW_reset	Read/ Write	0	1 = Resets all the registers
2	FIFO_clear	Read/ Write	0	1 = Clears the FIFOs
1	Read	Read/ Write	0	1 = Initiates readback of bitstream in to the Read FIFO
0	Write	Read/ Write	0	1 = Initiates writing of bitstream in to the ICAPEn

Status Register

The Status Register (SR) shown in Figure 2-8 is a 32-bit read register that contains the ICAPEn status bits. The bit definitions for the register are shown in Table 2-12.

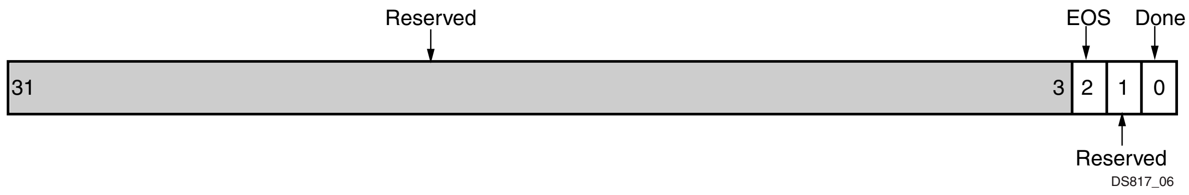


Figure 2-8: **Status Register (SR)**

Table 2-12: **Status Register Bit Definitions (110h)**

Bits	Name	Access	Reset Value	Description
31–3	Reserved	N/A	0	Reserved bits
2	EOS Bit	Read	1	Indicates that the EOS is complete. ICAPEn can be accessed only when this bit is 1.
1	Reserved	N/A	0	Reserved
0	Done	Read	1	AXI HWICAP done with configuration or read

Write FIFO Vacancy Register

The Write FIFO Vacancy (WFV) register shown in Figure 2-9 is a 32-bit read only register that indicates the vacancy of the Write FIFO. The actual depth of the Write FIFO is one less than the value specified during customization. For example, if the **Write FIFO depth** is set to 1024 during customization, the actual FIFO depth is 1023 (or 0x3FF). This register reports the *actual* Write FIFO vacancy (as described above). The bit definitions for the register are shown in Table 2-13.

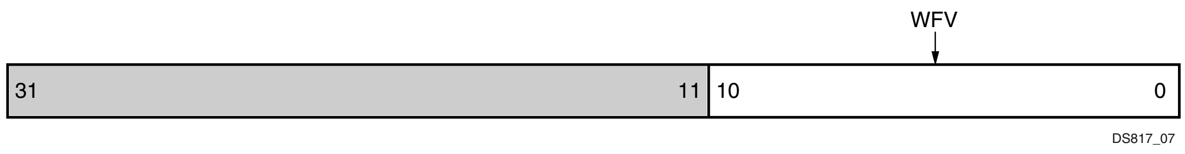


Figure 2-9: **Write FIFO Vacancy Register (WFV)**

Table 2-13: **Write FIFO Vacancy Register Bit Definitions (114h)**[(2)]

Bits	Name	Access	Reset Value	Description
31–11	Reserved	NA	-	Reserved
10–0	WFV	Read Only	[(1)]	Write FIFO Vacancy

Notes:
1. This value is based on the actual Write FIFO size as described above.
2. This register is of no importance when Write FIFO is disabled.

Read FIFO Occupancy Register

The Read FIFO Occupancy (RFO) register shown in Figure 2-10 is a 32-bit read only register that indicates occupancy of the Read FIFO. The actual depth of the Read FIFO is one less than the value specified during customization. For example, if the **Read FIFO depth** is set to 256 during customization, the actual FIFO depth is 255 (or 0xFF). This register reports the *actual* Read FIFO occupancy (as described above). The bit definitions for the register are shown in Table 2-14. This register is of no importance when disable Read FIFO is checked when the Read FIFO is disabled.

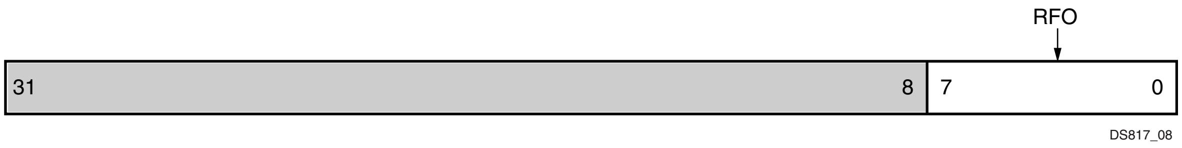


Figure 2-10: **Read FIFO Occupancy Register (RFO)**

Table 2-14: **Read FIFO Occupancy Register Bit Definitions (118h)[2]**

Bits	Name	Access	Reset Value	Description
31–8	Reserved	N/A	-	Reserved
7–0	RFO	Read Only	0[1]	Read FIFO Occupancy

Notes:

1. This value is based on the actual Write FIFO size as described above.
2. This register is of no importance when Write FIFO is disabled.

Abort Status Register

An abort is an interruption which occurs in the configuration or read-back sequence when the state of `icap_we` changes while `icap_ce` is asserted. During a Configuration Abort, internal status is driven onto the `icap_dout[7:0]` pins over the next four clock cycles. These four bytes are stored in the abort status register with the first status word available in the 31–24 bits while the last status word is in the 7–0 bits. After the abort sequence finishes, you can re-synchronize the configuration logic and resume configuration. Abort enables you to know the status of the ICAPEn during the configuration or reading the configuration.

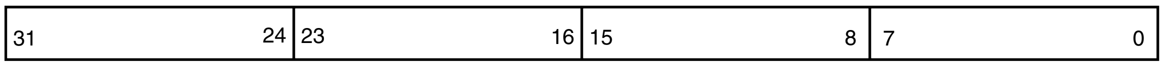


Figure 2-11: **Abort Status Register**

Table 2-15: **Abort Status Register (11Ch)**

Bits	Name	Access	Reset Value	Description
31–24	Status 0	Read	0	First Abort Status word
23–16	Status 1	Read	0	Second Abort Status word
15–8	Status 2	Read	0	Third Abort Status word
7–0	Status 3	Read	0	Fourth Abort Status Word

Designing with the Core

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

General Design Guidelines

The use of the AXI HWICAP is outlined in the following steps:

- Clocking and Resets
- Programming Sequence

Clocking

The core has two different clock domains: `s_axi_aclk` and `icap_clk`. In most cases both clocks can be connected to the same source. `icap_clk` can be up to 100 MHz.

Resets

The active-Low reset, `s_axi_aresetn`, is synchronized to `s_axi_aclk` clock.

Programming Sequence

Read or Write Configuration

1. Write the bitstream into the Write FIFO Register to configure.
2. Read the configuration bitstream from the Read FIFO register.
3. Write into the Control Register (CR) to initiate the read or write of a bitstream. The CR register determines the direction of the data transfer.

Writing 0x00000001 into the Control Register initiates the write configuration. Writing 0x00000002 into the Control Register initiates the read. This is shown as rnc[7:0].

4. Done bit in the Status Register indicates whether the ICAPEn interface is busy with writing or reading data from or to the ICAPEn bus. It does not indicate that the read or configuration with ICAPEn is completed successfully.
5. Hardware clears the Control Register bits after the successful completion of the read or configuration.
6. Software should not initiate another read or configuration to ICAPEn until the read or configuration bit in the Control Register is cleared.

For more information, see the Write FIFO Keyhole Register, Read FIFO Keyhole Register, Control Register and Status Register sections of this guide.

Write Sequence for Lite Mode

1. Write the instruction into the Write FIFO Register to configure.
2. Write into the Control Register to initiate the write of the instruction. The CR register determines the direction of the data transfer. Writing 0x00000001 into the Control Register initiates the write configuration.
3. Hardware clears the Control Register bits after the successful completion of the configuration.
4. Write the second instruction to Write FIFO Register. Write to Control Register to initiate the write to the ICAPEn.
5. Continue the process until all the instructions have been written to ICAPEn.

For more information, see the Write FIFO Keyhole Register, and Control Register sections of this guide.

Abort

1. Write into the Control Register to initiate the read or write of bitstream. The CR register determines the direction of the data transfer. Writing 0x00000001 into the Control Register initiates the configuration. Writing 0x00000002 into the Control Register initiates the read. This is shown as rnc[7:0].
2. Write the bitstream into the Write FIFO Register to configure. Get the bitstream from the Read FIFO register to read.
3. Write a 1 into the fifth bit of the Control Register to initiate abort.
4. Done bit in the Status Register indicates whether the ICAPEn interface is busy with writing/reading data from/to the ICAPEn bus. It does not indicate that the read/configuration with ICAPEn is completed successfully.

5. Hardware clears the Control Register bits after the successful completion of the abort-on read or abort-on configuration or normal abort.
6. Software should not initiate another read or configuration to ICAPEn until the read or configuration bit in the Control Register is cleared.

For more information, see the Control Register, Write FIFO Keyhole Register, Read FIFO Keyhole Register, and Status Register sections of this guide.

Limitations

A frame is the smallest granularity in which the FPGA allows configuration data to be read and written. A configuration frame is a collection of data that is 1-bit wide and spans the full column of the FPGA. Configuration frames in the CLB space also contain Input Output Block (IOB) configuration data at the top and bottom, which configure the IOBs at the top and bottom of the FPGA. A single column of CLBs contain multiple configuration frames.

Although a single CLB LUT or flip-flop can be modified, the underlying mechanism requires that the full column be read into Block Random Access Memory (RAM). This implies that other logic in the same column can be modified. In most cases, this effect can be ignored. When the frame is written back to the configuration memory, the sections of the column that were not modified are written with the same data. Because the FPGA memory cells have glitch less transitions, when rewritten, the unmodified logic continues to operate unaffected.

The two exceptions to this rule are:

- When the LUTs are configured in the Shift Register mode.
- When the LUTs are configured as a RAM.

If a LUT is modified or just read back in a column that also has a LUT RAM or LUT shift register, then the LUT or shift register is interrupted and it loses its state. To resolve the issue, the LUT shift registers and LUT RAMs should be placed in columns that are not read back or modified. If the LUT RAMs or shift register in a column do not change state during the read back or modification, then they maintain their state.

Important Notes

- The HWICAP core uses the ICAPE2 primitive found in 7 series devices or the ICAPE3 primitive found in UltraScale™ devices. The ICAPEn port interface is similar to the SelectMAP interface but is accessible from general interconnects rather than the device pins. The Joint Test Action Group (JTAG) or Boundary Scan configuration mode pin setting (M2:M0 = 101) disables the ICAPEn interface. If JTAG is used as the primary configuration method, another mode pin setting must be selected to avoid disabling the ICAPEn interface.

  JTAG configuration remains available because it overrides other means of configuration, and the HWICAP core functions as intended. Besides being disabled by the Boundary-Scan mode pin setting, the ICAPEn is also disabled if the persist bit in the device configuration logic control register is set.
- There can be cases when you access the ICAPEn as soon as JTAG programming is over. In such cases, the ICAPEn is not accessible as JTAG might be performing some other tasks.

RECOMMENDED: *Allow sufficient time before initiating any transaction to ICAPEn, or be sure to enable the use of the STARTUPE2 primitive within the IP.*

Design Flow Steps

This chapter describes customizing and generating the core, constraining the core, and the simulation, synthesis and implementation steps that are specific to this IP core. More detailed information about the standard Vivado® design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 7]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 9]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 10]

Customizing and Generating the Core

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

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 7] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl console.

You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps.

1. Open an existing project or create a new project in the Vivado Design Suite.
2. Open the **IP catalog** and navigate to **Embedded_Processing** > **AXI_Peripheral** > **Low_Speed_Peripheral**.
3. Double-click **AXI HWICAP** to open the Customize IP dialog box for AXI HWICAP.

4. In the Customize IP dialog box, you can:
   - Display information about the core
   - Configure of the core
   - Generate the core

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 9].

The Customize IP dialog box for AXI HWICAP contains a Basic Tab and an Advanced Tab. In most cases the AXI HWICAP can be configured with the option specified on the **Basic** tab.

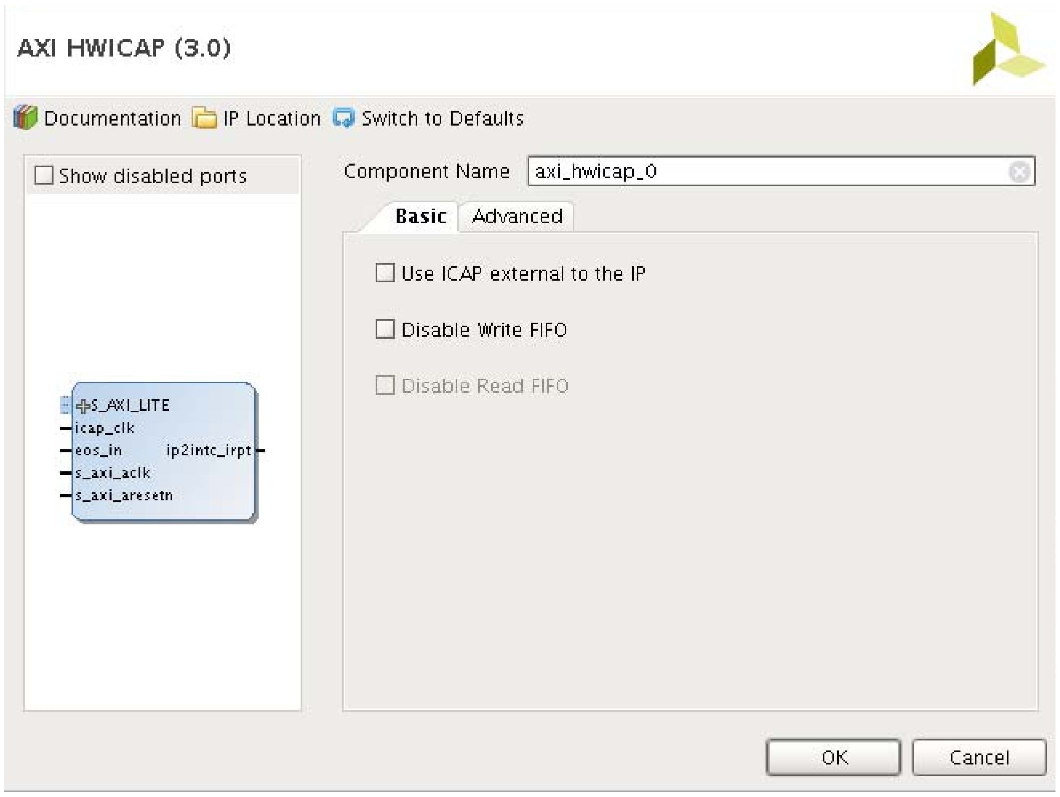
Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). The layout depicted here might vary from the current version.

Component Name

The base name of the output files generated for the core. Names must begin with a letter and can be composed of any of the following characters: a to z, 0 to 9, and "_".

Basic Tab

The fundamental, or basic, options that affect the AXI HWICAP core are shown in Figure 4-1 and are described in this section.


AXI HWICAP (3.0)

Component Name axi_hwicap_0

- [ ] Use ICAP external to the IP
- [ ] Disable Write FIFO
- [ ] Disable Read FIFO

Figure 4-1: **Basic Tab**

Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). This layout might vary from the current version.

Disable Write FIFO

Checking this option disables the instantiation of Write FIFO thereby reducing the utilization. This can be disabled only when `icap_clk` and `s_axi_aclk` are connected to the same clock source.

Disable Read FIFO

Checking this option disables the Read FIFO. This option is available only when instantiation of Write FIFO is disabled. Reading back the data from ICAPEn is not possible when this is selected. Reading the Read FIFO register return all zeros.

ICAPEn Data Width

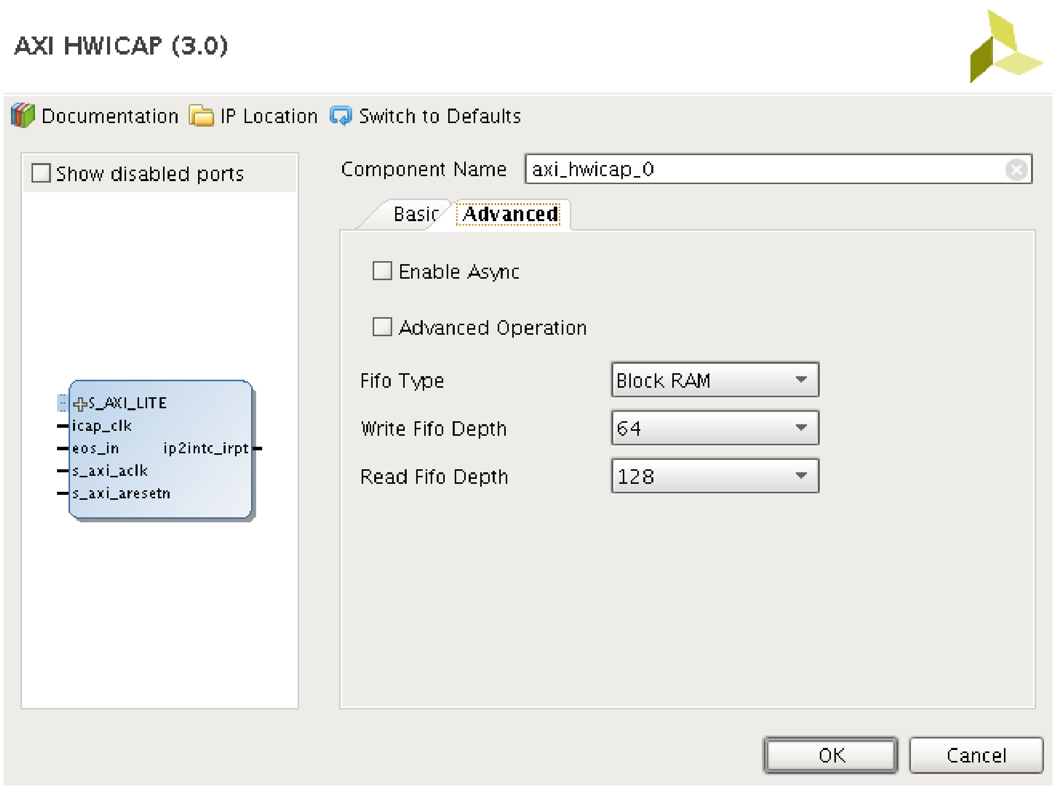
Select the width of the ICAPEn primitive. Available options are 8, 16, 32. By default this is set to 32.

Instantiate STARTUPE2 Primitive

Select this box to instantiate the STARTUPE2 primitive. The EOS output from STARTUPE2 is used to control access to ICAPE2. EOS output signifies the end of device configuration. ICAPE2 can be accessed only after EOS goes High.

Advanced Tab

The advanced options that affect the AXI HWICAP core are shown in Figure 4-2 and Figure 4-3 (for Vivado IP integrator). The options are described in this section.


AXI HWICAP (3.0)

Component Name: axi_hwicap_0

Enable Async: ☐

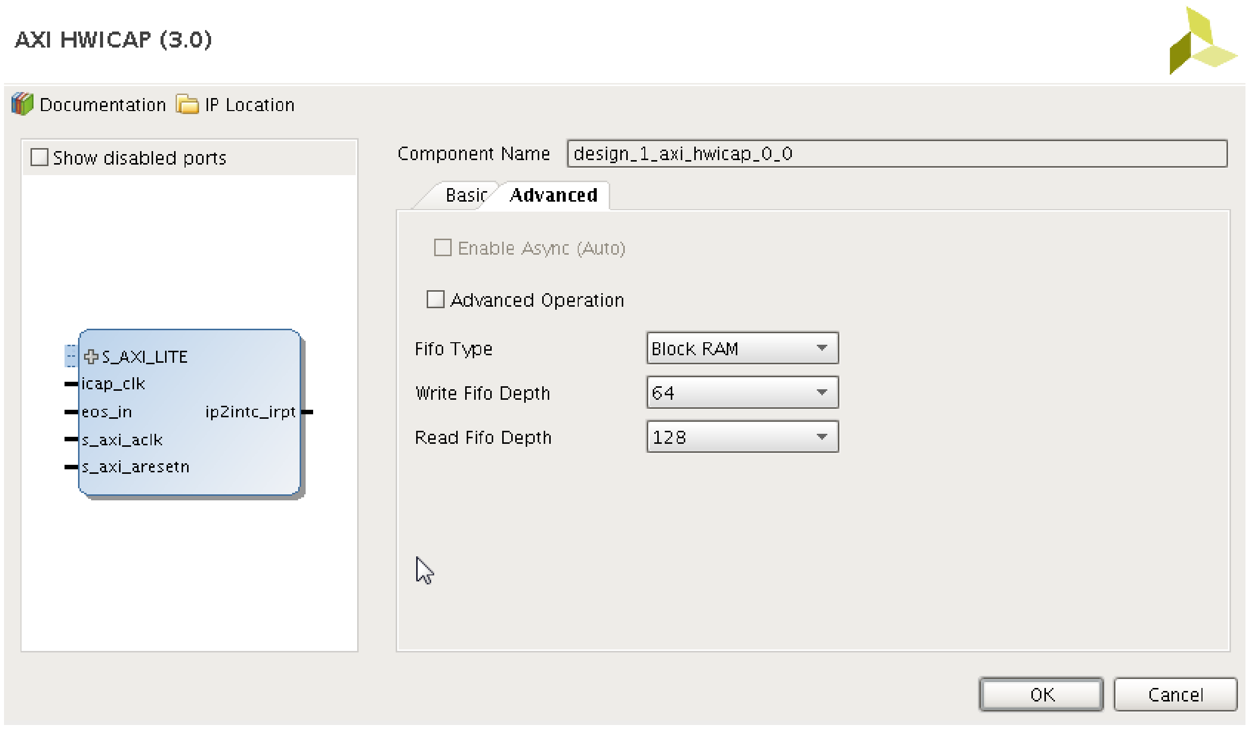
Advanced Operation: ☐

Fifo Type: Block RAM

Write Fifo Depth: 64

Read Fifo Depth: 128

Figure 4-2: **Advanced Tab**



Component Name: design_1_axi_hwicap_0_0

☐ Enable Async (Auto)

☐ Advanced Operation

Fifo Type	Block RAM
Write Fifo Depth	64
Read Fifo Depth	128

Figure 4-3: **Advanced Tab (IP Integrator)**

Enable Aysnc

Select this box if the `icap_clk` and AXI Clocks are asynchronous. In synchronous mode, only `s_axi_aclk` is used. This parameter is automatically updated when the IP is used in the IP integrator.

Advanced Operation

Select this box only when performing large frame reads from the ICAPEn. This instantiates an extra BUFGCTRL within the IP.

FIFO Type

Choose the type of FIFO to be instantiated. You can choose between Block RAM or Distributed RAM type of FIFO.

Write FIFO Depth

Select the depth of the Write FIFO. Available values are 64, 128, 256, 512 and 1,024. The actual depth of the Write FIFO is one less than the one specified during core customization.

Read FIFO Depth

Select the depth of the Read FIFO. Available values are 128 and 256. The actual depth of the Read FIFO is one less the value specified during core customization.

User Parameters

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship**

Vivado IDE Parameter/Value	User Parameter/Value Default Value	Default Value
Use ICAP external to the IP	C_ICAP_EXTERNAL	0
Disable Write FIFO	C_MODE	0
Disable Read FIFO	C_NOREAD	0
ICAPE2 Data Width	C_ICAP_DWIDTH	32
Include STARTUPE2 primitive	C_INCLUDE_STARTUP	0
Enable Async	C_ENABLE_ASYNC	0
Advanced Operation	C_OPERATION	0
Fifo Type	C_BRAM_SRL_FIFO_TYPE	1
Write Fifo Depth	C_WRITE_FIFO_DEPTH	64
Read Fifo Depth	C_READ_FIFO_DEPTH	128
Share the unused STARTUP ports	C_SHARED_STARTUP	0
Use ICAP external to the IP	C_ICAP_EXTERNAL	0

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8].

Constraining the Core

The necessary Xilinx design constraints (XDCs) are delivered when the core is generated.

Simulation

- For comprehensive information about Vivado simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 10].
- For information about simulating the example design, see Simulating the Example Design.

IMPORTANT: *For cores targeting 7 series or Zynq®-7000 devices, UNIFAST libraries are not supported. Xilinx IP is tested and qualified with UNISIM libraries only.*

Synthesis and Implementation

- For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8].
- For information about synthesizing and implementing the example design, see Implementing the Example Design.

Example Design

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

Overview

The top module instantiates all components of the core and example design that are needed to implement the design in hardware, as shown in Figure 5-1. This includes the clocking wizard (MMCME2), register configuration, data generator and data checker modules.

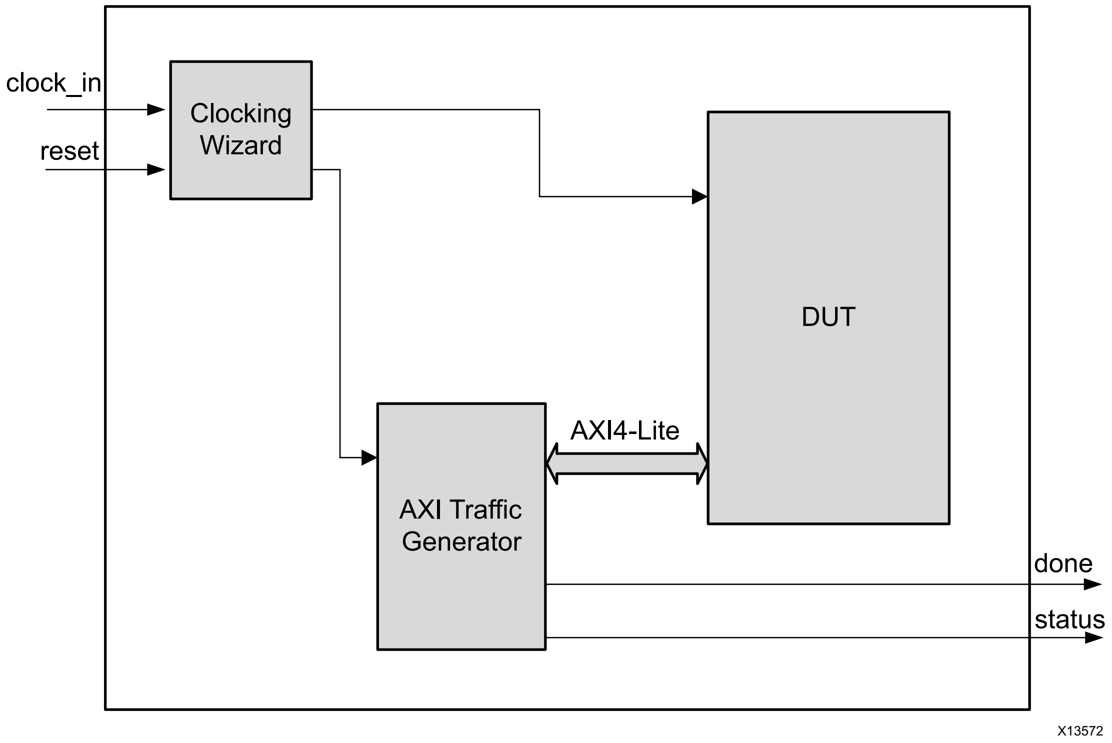


Figure 5-1: **Block Diagram of Example Design**

This example design demonstrates transactions on AXI4-Lite interfaces of the DUT to read the IDCODE from ICAPEn.

- **Clocking Wizard:** MMCME2 is used to generate the clocks for the example design. When DUT is in synchronous mode then MMCME2 generates a 50 MHz clock for the AXI interfaces in the example design. When DUT is in asynchronous mode then MMCME2 generates a 50 MHz clock for the `icap_clk` interface and a 100 MHz clock for `s_axi_aclk`. DUT and other modules of the example design are kept under reset until MMCME2 is locked.
- **AXI Traffic Generator (ATG):** This module (IP) is configured in System Test Mode. All the AXI_HWICAP related AXI4-Lite transactions are stored in the coe/mif file. For more information, see the *AXI Traffic Generator LogiCORE IP Product Guide* (PG125) [Ref 16]. The ATG automatically starts the AXI4-Lite transaction after coming out of reset. The ATG asserts the done pin along with the necessary status bits after the IDCODE is read from the ICAPEn.

A successful completion of test is determined by the `Done` and `Status` pins going High. These two pins can be connected to LEDs to know the status of the test.

In case of a failure, only one pin (`Done`) goes High.

This test starts soon after the bitfile is programmed. In such a case, the ICAPEn cannot be accessed while it is still busy completing the FPGA configuration.

RECOMMENDED: *Access ICAPEn after sufficient time has passed. A counter has been provided in the Example Design top-level file. Update the counter value to give ICAPEn sufficient time to come out of configuration state. This counter has been set to count a value of 2 to reduce the simulation run time.*

Implementing the Example Design

After following the steps described in Chapter 4, Customizing and Generating the Core, implement the example design as follows:

1. Right-click the core in the Hierarchy window, and select **Open IP Example Design**.
2. A new window pops up asking you to specify a directory for the example design. Select a new directory, or keep the default directory.
3. A new project is created in the selected directory and it opens in a new window.
4. In the Flow Navigator (left side pane), click **Run Implementation**.

Example Design Directory Structure

In the current project directory, a new project with name `<component_name>_example` is created and the files are delivered in the `<component_name>_example/<component_name>_example.srcs/` directory. This directory and its subdirectories contain all the source files that are required to create the AXI HWICAP controller example design.

Example Design Directory

Table 5-1 shows the files delivered in the `<component_name>_example/<component_name>_example.srcs/sources_1/imports/example_design/` directory.

Table 5-1: **Example Design Directory**

Name	Description
<component_name>_exdes.vhd	Top-level HDL file for the example design.
clock_gen.vhd	Clock generation module for example design.
atg_addr.coe	COE file of address. This file contains the axi_hwicap register address.
atg_data.coe	COE file of data. This file contains the data to be written/read from the axi_hwicap registers.
atg_mask.coe	COE file to mask certain reads.
atg_ctrl.coe	COE file that contains control information of ATG.

Simulation Directory

Table 5-2 shows the files delivered in the `<component_name>_example/<component_name>_example.srcs/sources_1/sim_1/imports/simulation/` directory.

Table 5-2: **Simulation Directory**

Name	Description
<component_name>_exdes_tb.vhd	Test Bench for example design.

Constraints Directory

Table 5-3 shows the files delivered in the `<component_name>_example/<component_name>_example.srcs/sources_1/constrs_1/imports/example_design/` directory.

Table 5-3: **Constraints Directory**

Name	Description
<component_name>_exdes.xdc	Top-level constraints file for the example design.

The delivered XDC file has I/O constraints for a KC705 board. These constraints are commented out by default.

IMPORTANT: *Uncomment the constraints before proceeding with synthesis and implementation for a KC705 board.*

Simulating the Example Design

Using the example design delivered as part of the AXI HWICAP, you can quickly simulate and observe the behavior of the AXI HWICAP.

Setting up the Simulation

The Xilinx simulation libraries must be mapped into the simulator. To set up the Xilinx simulation models, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 10]. To switch simulators, click **Simulation Settings** in the Flow Navigator (left pane). In the Simulation options list, change **Target Simulator**.

The example design supports functional (behavioral) and post-synthesis simulations. For how to run simulation, see the *Vivado Design Suite User Guide: Logic Simulation (UG900)* [Ref 10].

Simulation Results

The simulation script compiles the AXI HWICAP example design, and supporting simulation files. It then runs the simulation and checks to ensure that it completed successfully.

If the test fails, the following message is displayed.

```
Test Failed !!!
```

If the test passes, the following message is displayed:

```
Test Completed Successfully
```

If the test hangs, the following message is displayed.

```
Test Failed !! Test Timed Out.
```

Test Bench

This chapter contains information about the test bench provided in the Vivado® Design Suite.

Figure 6-1 shows test bench for AXI HWICAP example design. The top-level test bench generates a 200 MHz clock and drives the initial reset to the example design.

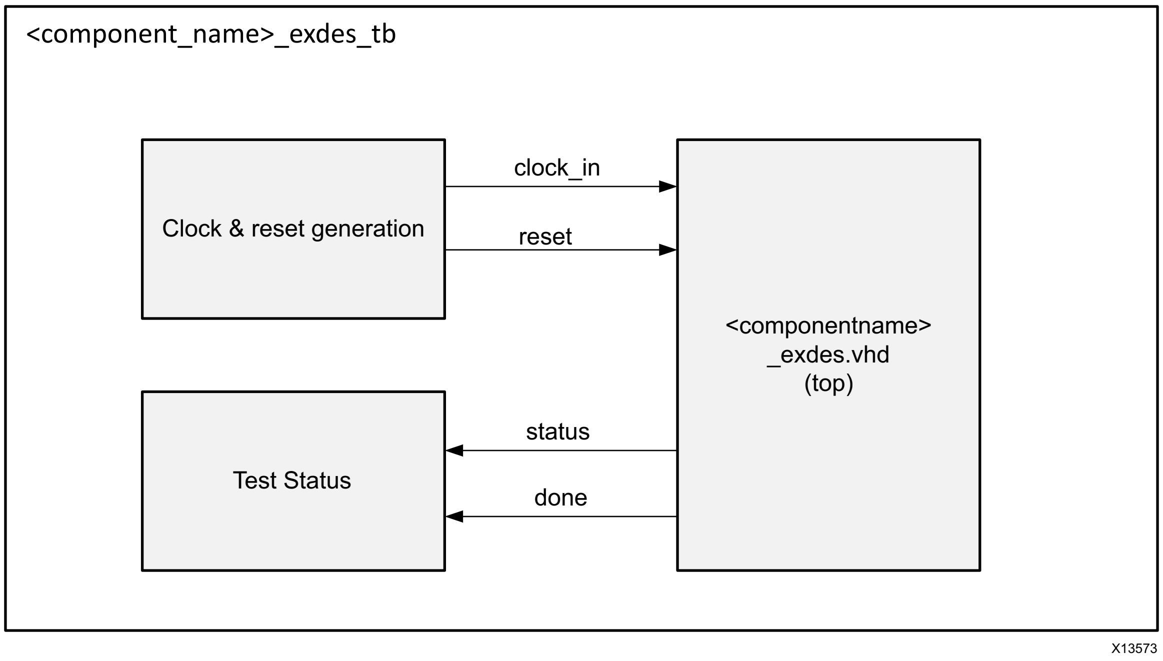


Figure 6-1: **AXI HWICAP Example Design Test Bench**

Migrating and Upgrading

This appendix contains information about upgrading to a more recent version of the IP core.

Migrating to the Vivado Design Suite

For information about migrating to the Vivado Design Suite, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 11].

Upgrading in the Vivado Design Suite

There are no port or parameter changes.

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

Finding Help on Xilinx.com

To help in the design and debug process when using the AXI HWICAP, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the AXI HWICAP. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

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 can 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 AXI HWICAP

AR: 54402

Technical Support

Xilinx provides technical support at the Xilinx Support web page for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support, navigate to the Xilinx Support web page.

Debug Tools

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

Vivado Design Suite Debug Feature

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

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

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

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 12].

Some important signals have been marked as `debug` in the register transfer level (RTL). These can be easily added to logic analyzer for debugging.

- Signals, such as size register, control register, write fifo vacancy and read fifo occupancy, have been added.
- The I/O signal of the ICAPEn interface, such as `ce`, `we`, `busy`, `datain` and `dataout`, have been added.

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

References

These documents provide supplemental material useful with this product guide:

1. *7 Series FPGAs Overview Advanced Product Specification* (DS180)
2. *Vivado Design Suite User Guide: Designing With IP* (UG896)
3. *Xilinx Vivado AXI Reference Guide* (UG1037)
4. *Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics* (DS183)
5. *Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics* (DS182)
6. *Artix-7 FPGAs Data Sheet: DC and Switching Characteristics* (DS181)
7. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
8. *Vivado Design Suite User Guide: Designing with IP* (UG896)
9. *Vivado Design Suite User Guide: Getting Started* (UG910)
10. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
11. *ISE to Vivado Design Suite Migration Guide* (UG911)
12. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
13. *Vivado Design Suite User Guide: Implementation* (UG904)
14. *AXI Interconnect LogiCORE IP Product Guide* (PG059)
15. *AXI4-Lite IPIF LogiCORE IP Product Guide* (PG155)
16. *AXI Traffic Generator LogiCORE IP Product Guide* (PG125)
17. *AMBA AXI and ACE Protocol Specification* (ARM IHI 0022D)
18. *UltraScale Architecture Configuration User Guide* (UG570)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/05/2016	3.0	• Added a note about the AXI4-Lite write access register to the beginning of the Register Space section. • Added the Automotive Applications Disclaimer. • Added User Parameters section in Chapter 4.
11/18/2015	3.0	Added support for UltraScale+ families.
09/30/2015	3.0	• Added support for ICAPE3 primitive (UltraScale™ architecture). • Added ICAP, and ICAP arbiter interface signals. • Added descriptions for ICAP interface module, and switching behavior of ICAP interface signals. • Clarified how *actual* Read FIFO Depth is calculated. • Updated the example design block diagram.
12/18/2013	3.0	Added UltraScale architecture support.
10/02/2013	3.0	Re-release of core v3.0 product guide, with the following updates: • Updated version in this table to align to core version. • Added example design and test bench information. • Changed signal names to lowercase. • Added information about Vivado IP integrator support.
03/20/2013	1.0	Initial release as a product guide. Replaces *LogiCORE IP AXI HWICAP Data Sheet* (DS817).

Please Read: Important Legal Notices