LogiCORE IP AXI HWICAP (v2.00a)

DS817 June 22, 2011 **Product Specification**

Introduction

This product specification describes the functionality of the LogiCORE™ IP AXI HWICAP (Hardware ICAP) core for the AXI Interface. This core enables an embedded microprocessor, such as the MicroBlaze™ processor, to read and write the FPGA configuration memory through the Internal Configuration Access Port (ICAP). This enables a user 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 AXI4-Lite specification
- Partial bitstream loading
- Enables Read/Write of CLB LUTs
- Enables Read/Write of CLB Flip-Flop properties
- Supports the MicroBlaze embedded processor

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	Virtex-6[3], Spartan-6[4]
Supported User Interfaces	AXI4-Lite
Resources	
See Table 15 and Table 16.	
Provided with Core	
Documentation	Product Specification
Design Files	VHDL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	UCF (user constraints file)
Simulation Model	Not Provided
Tested Design Tools [2]	
Design Entry Tools	Xilinx Platform Studio (XPS)
Simulation	Mentor Graphic ModelSim
Synthesis Tools	Xilinx Synthesis Tool (XST)
Support	
Provided by Xilinx, Inc.	

Notes:

1. For a complete listing of supported devices, see the release notes for this core.
2. For a listing of the supported tool versions, see the ISE Design Suite 13: Release Note Guide.
3. For more information, see the DS150 Virtex-6 Family Overview Product Specification.
4. For more information, see DS160 Spartan-6 Family Overview Product Specification.

Functional Description

The AXI HWICAP controller provides the interface necessary to transfer bitstreams to and from the ICAP. For writes to the ICAP, the required bitstream data from main memory. Incoming data is stored within a Write FIFO, from where it can be sent to the ICAP. The AXI HWICAP also provides for read back of configuration resource states. In this case, the frames are read back into the Read FIFO one at a time and the CPU will then be able to read the frame data directly from the Read FIFO.

Sample applications

- A DSP system, such as software defined radio, where the filters and algorithms are modified at run time to receive and transmit at variable frequencies, or adapt to variable protocols.
- A debug system where trigger conditions are implemented as comparator circuits and modified at run time to enable variable trigger conditions. The system can also have counters to measure the amount of data sampled. The final counts of the counters can be modified to vary the amounts of data sampled.
- A crossbar switch where the fundamental switches are implemented using multiplexers in the routing fabric. The crossbar connections are reconfigured at run time by reconfiguring the routing multiplexers.

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

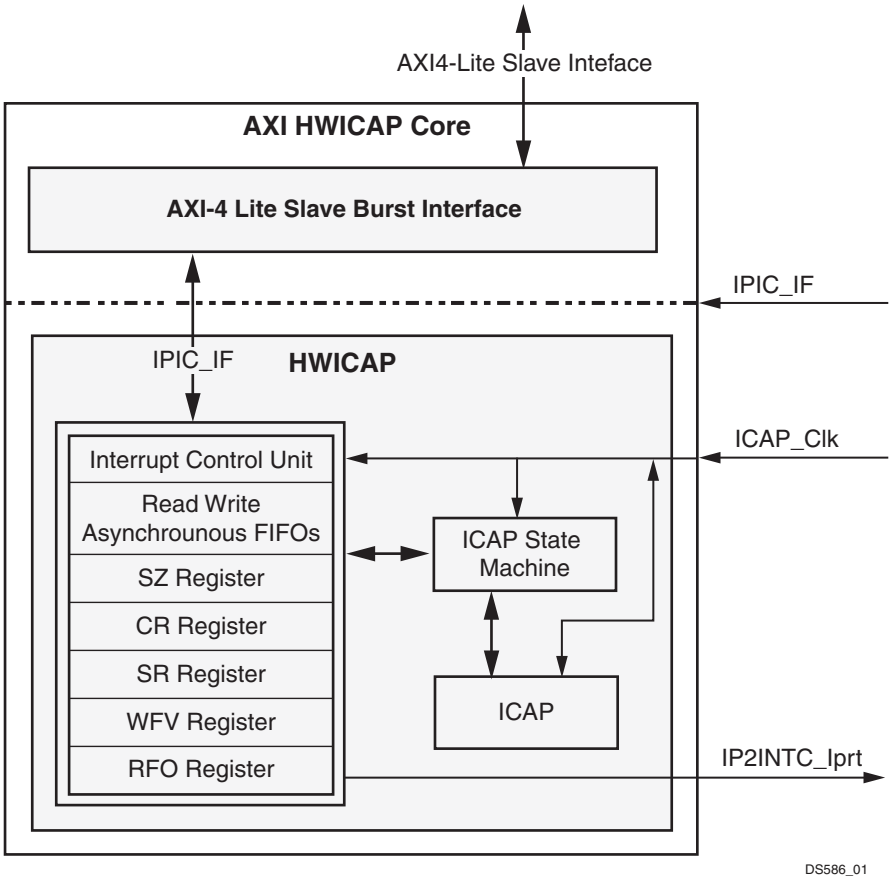

Figure 1: **Top Level Block Diagram for the AXI HWICAP Core**

AXI-Lite Interface Module

The AXI4-Lite Interface Module provides the bidirectional interface between 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 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 Internal Configuration Access Port (ICAP). It has a write FIFO, which stores the configuration locally. The processor writes the configuration in to the write FIFO. Simultaneously, the data stored in the write FIFO is transferred to the ICAP. The processor reads the configuration from the ICAP, which is stored inside the read FIFO. FIFOs are used because the rate of data flow from the processor interface is different from the ICAP interface. FIFO depth can be parameterizable using the generics C_WRITE_FIFO_DEPTH and C_READ_FIFO_DEPTH.

The FIFO data width is dependent on the device family. For the Virtex-6 FPGA, the FIFO data width is 32 bit; for the Spartan-6 FPGA, the FIFO data width is 16 bit.

Note: With the Virtex-6 device, if the S_AXI_AClk is greater than 100 MHz, connect the ICAP_Clk to 100 MHz. If the SPLB_Clk is less than, or equal to, 100 MHz, connect the ICAP_Clk to the frequency equivalent to the SPLB_clk frequency.

Note: With the Spartan-6 device, because the maximum ICAP frequency 20 MHz, the ICAP_Clk *must* be connected to a frequency less than or equal to 20 MHz.

I/O Signals

The AXI4 HWICAP core signals are listed and described in Table 1

Table 1: **I/O Signals**

Port	Signal Name	Interface	I/O	Initial State	Description
ICAP Interface Signals					
P1	ICAP_Clk (1)	ICAP	I	-	ICAP clock
AXI Bus Request and Qualifier Signals					
P2	S_AXI_ACLK	AXI	I	-	AXI Clock
P3	S_AXI_ARESETN	AXI	I	-	AXI Reset; Active Low
P4	S_AXI_AWADDR(C_S_AXI_ADDR_WIDTH-1 :0)	AXI	I	-	AXI write address: The write address bus gives the address of the write location.
P5	S_AXI_AWVALID	AXI	I	-	AXI Write address valid: This signal indicates that a valid write address and control information are available
P6	S_AXI_AWREADY	AXI	O	-	Write address ready: This signal indicates that the slave is ready to accept an address and associated control signal
P7	S_AXI_WDATA[(C_S_AXI_DATA_WIDTH-1) : 0]	AXI	I	-	Write data.

Table 1: **I/O Signals *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P8	S_AXI_WSTRB[(C_S_AXI_ DATA_WIDTH/8)-1) :0]	AXI	I	-	Write strobes: This signal indicates which byte lanes to update in memory
P9	S_AXI_WVALID	AXI	I	-	Write valid: This signal indicates that valid write data and strobes are available
P10	S_AXI_WREADY	AXI	O	-	Write ready: This signal indicates that the slave can accept the write data
P11	S_AXI_BRESP[1:0]	AXI	O	0	Write response: This signal indicates the status of the write transaction "00" - OKAY (normal response) "10" - SLVERR (error response) "11" - DECERR (not issued by core)
P12	S_AXI_BVALID	AXI	O	0	Write response valid: This signal indicates that a valid write response is available
P13	S_AXI_BREADY	AXI	I	-	Response ready: This signal indicates that the master can accept the response information
P14	S_AXI_ARADDR[(C_S_AXI_ ADDR_WIDTH-1) : 0]	AXI	I	-	Read Address: The read address bus gives the address of a read transaction
P15	S_AXI_ARVALID	AXI	I		Read address valid: This signal indicates, when HIGH, that the read address and control information is valid and will remain stable until the address acknowledgement signal, S_AXI_ARREADY, is high
P16	S_AXI_ARREADY	AXI	O	1	Read address ready: This signal indicates that the slave is ready to accept and address and associated control signals
P17	S_AXI_RDATA[(C_S_AXI_ DATA_WIDTH-1) : 0]	AXI	O	0	Read data.
P18	S_AXI_RRESP[1:0]	AXI	O	0	Read response: This signal indicates the status of the read transfer "00" - OKAY (normal response) "10" - SLVERR (error condition) "11" - DECERR (not issued by core)
P19	S_AXI_RVALID	AXI	O	0	Read valid: This signal indicates that the master can accept the read data and response information
P20	S_AXI_RREADY	AXI	I	-	Read ready: This signal indicates that the master can accept the read data and response information
System Signals					
P21	IP2INTC_Irpt	AXI	O	0	AXI HWICAP Interrupt.

Notes:

1. ICAP_Clk must be less than or equal to 20 MHz, 12 MHz, or 4 MHz in Spartan-6 FPGAs. The Spartan-6 LX4 to LX75/T devices have a limiting frequency of 20 MHz. The Spartan-6 LX100/T to LX150/T devices have a limiting frequency of 12 MHz. The Spartan-6 Low Power devices have a limiting frequency of 4 MHz. The Virtex-6 devices have a limiting frequency of maximum 100 MHz on ICAP_Clk

Design Parameters

To allow the user to create a core that is uniquely tailored for the user's system, certain features are parameterizable in the design. This allows the user to have a design that utilizes only the resources required by the system and runs at the best possible performance. The features that are parameterizable in the core are as shown in Table 2.

Inferred Parameters

In addition to the parameters listed in Table 2, there are also parameters that are inferred for each AXI interface in the EDK tools. Through the design, these EDK-inferred parameters control the behavior of the AXI Interconnect. For a complete list of the interconnect settings related to the AXI interface, see the DS768 AXI Interconnect IP data sheet.

Table 2: **Design Parameters**

Generic	Parameter Description	Parameter Name	Allowable Values	Default Value	VHDL Type
AXI Parameters					
G1	AXI HWICAP Base Address	C_BASEADDR	Valid Word Aligned Address(1)	None(2)	std_logic_vector
G2	AXI HWICAP High Address	C_HIGHADDR	C_HIGHADDR -C_BASEADDR must be a power of 2 >= to C_BASEADDR+1FF(1)	None(2)	std_logic_vector
G3	AXI data bus width	C_S_AXI_DATA_WIDTH	32	32	integer
G4	AXI address Bus Width	C_S_AXI_ADDR_WIDTH	32	32	integer
G5	Write FIFO depth	C_WRITE_FIFO_DEPTH(3)(6)	64, 128, 256, 512,1024	64	integer
G6	Read FIFO depth	C_READ_FIFO_DEPTH(3)(6)	128, 256	128	integer
G7	Select FIFO type(4)	C_BRAM_SRL_FIFO_TYPE	0,1	1	integer
System Parameters					
G12	XILINX FPGA Family	C_FAMILY	spartan6, virtex6	spartan6	string
G13	Simulation(5)	C_SIMULATION	1: FIFO Model 2: UNISIM Model	2	integer

Notes:

1. Address range specified by C_BASEADDR and C_HIGHADDR. C_BASEADDR must be a multiple of the range, where the range is C_HIGHADDR - C_BASEADDR +1 and must be a power of 2 and greater than 0x1000. For example, if C_BASEADDR = 0x10000000, the C_HIGHADDR should be 0x10000FFF or more.
2. No default value will be specified to insure that the actual value is set. For example, if the value is not set, a compiler error will be generated.
3. This parameter must be set to 256 if the C_FAMILY = virtex6/spartan6, as the frame size in virtex6/spartan6 family is 162 bytes.
4. The parameter C_BRAM_SRL_FIFO_TYPE selects the FIFO type to be BRAM (1) or Distributed RAM (0).
5. The parameter C_SIMULATION must be set to 2 for simulations using the ICAP unisim model.
6. The actual depth of the Write/Read FIFO is one less than the parameter that defines the depth of the FIFOs (for example, either C_WRITE_FIFO_DEPTH or C_READ_FIFO_DEPTH).

Parameter - Port Dependencies

The dependencies between the AXI HWICAP core design parameters and the I/O signals are described in Table 3. When certain features are parameterized out of the design, the related logic will no longer be a part of the design. The unused input signals and related output signals are set to a specified value.

Table 3: **Parameter-Port Dependencies**

Generic or Port	Name	Affects	Depends	Relationship Description
Design Parameters				
G4	C_S_AXI_ADDR_WIDTH	P4, P14	-	Affects the number of bits in address bus
G3	C_S_AXI_DATA_WIDTH	P7, P8,P17	-	Affects the number of bits in data bus
I/O Signals				
P4	S_AXI_AWADDR[(C_S_AXI_ADDR_WIDTH-1) : 0]	-	G4	Width of the S_AXI_AWADDR varies with C_S_AXI_ADDR_WIDTH
P7	S_AXI_WDATA[(C_S_AXI_DATA_WIDTH - 1) : 0]	-	G3	Width of the S_AXI_WDATA varies according to C_S_AXI_DATA_WIDTH
P8	S_AXI_WSTRB[((C_S_AXI_DATA_WIDTH/8)-1) : 0]	-	G3	Width of the S_AXI_WSTRB varies according to C_S_AXI_DATA_WIDTH
P14	S_AXI_ARADDR[(C_S_AXI_ADDR_WIDTH -1) : 0]	-	G4	Width of the S_AXI_ARADDR varies with C_S_AXI_ADDR_WIDTH
P17	S_AXI_RDATA[(C_S_AXI_DATA_WIDTH-1) : 0]	-	G3	Width of the S_AXI_RDATA varies according to C_S_AXI_DATA_WIDTH

Register Definition

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

Table 4: **Internal Registers**

C_BASEADDR + Address	Register Name	Access	Default Value	Description
C_BASEADDR + 0x01C	Global Interrupt Enable Register (GIE)	Read/Write	0x0	Global interrupt enable register
C_BASEADDR + 0x020	IP Interrupt Status Register (IPISR)	Read/Write	0x0	IP interrupt status register
C_BASEADDR + 0x028	IP Interrupt Enable Register (IPIER)	Read/Write	0x0	IP interrupt enable register
C_BASEADDR + 0x100	Write FIFO Register (WF)	Write	0x0	Data write register
C_BASEADDR + 0x104	Read FIFO Register (RF)	Read	0x0	Data read register
C_BASEADDR + 0x108	Size Register (SZ)	Write	0x0	Size register
C_BASEADDR + 0x10C	Control Register (CR)	Read/Write	0x0	IP control register
C_BASEADDR + 0x110	Status Register (SR)	Read	0x0	Status register
C_BASEADDR + 0x114	Write FIFO Vacancy Register (WFV)	Read	(1)	Write FIFO vacancy register
C_BASEADDR + 0x118	Read FIFO Occupancy Register (RFO)	Read	0x0	Read FIFO occupancy register

Notes:

1. This value is based on the Write FIFO size. For example, if the Write FIFO is of 1024 depth, this value would be 3FF.

Write FIFO Register (WF)

The WF register shown in Figure 2 is a 32-bit Write FIFO. The bit definitions for the Write FIFO are shown in Table 5. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

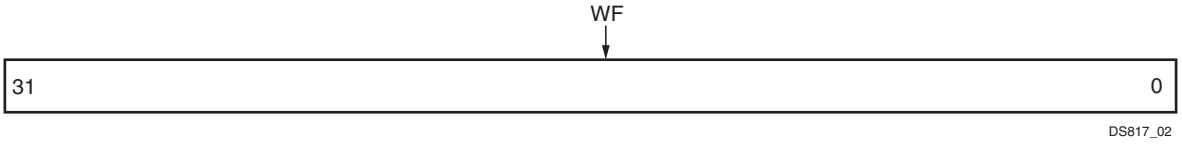

Figure 2: **Write FIFO (WF)**

Table 5: **Write FIFO Bit Definitions (C_BASEADDR + 0x100)**

Bit	Name	Access	Reset Value	Description
31 - 0	WF	Write	0	Data written into the FIFO

Read FIFO Register (RF)

The RF register shown in Figure 3 is a 32-bit Read FIFO. The bit definitions for the Read FIFO are shown in Table 6. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

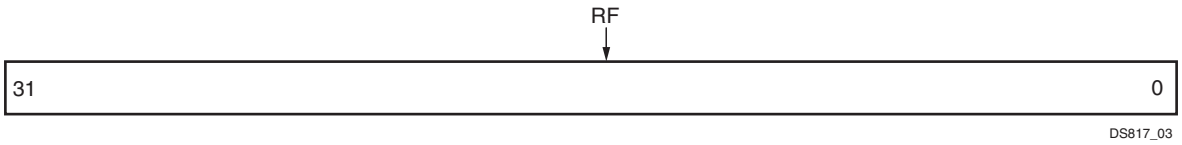

Figure 3: **Read FIFO (RF)**

Table 6: **Read FIFO Bit Definitions (C_BASEADDR + 0x104)**

Bit	Name	Access	Reset Value	Description
31 - 0	RF	Read	0	Data read from the FIFO

Size Register (SZ)

The SZ register shown in Figure 4 is a 12-bit write register that determines the number of words to be transferred from the ICAP to the read FIFO. The bit definitions for the register are shown in Table 7. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

Figure 4: **Size Register (SZ)**

Table 7: **Size Register Bit Definitions (C_BASEADDR + 0x108)**

Bit	Name	Access	Reset Value	Description
31 - 12	Reserved	N/A	0	Reserved bits
11 - 0	Size	Write	0	Number of words to be transferred from the ICAP to the FIFO

Control Register (CR)

The CR register shown in Figure 5 is a 32-bit write register that determines the direction of the data transfer. It controls whether a configuration or read back takes place. Writing to this register initiates the transfer. The bit definitions for the register are shown in Table 8. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

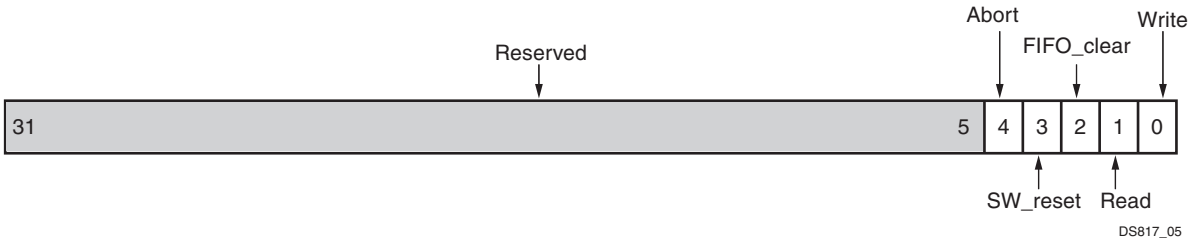

Figure 5: **Control Register (CR)**

Table 8: **Control Register Bit Definitions (C_BASEADDR + 0x10C)**

Bit	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 ICAP and clears the FIFOs
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 ICAP

Status Register (SR)

The SR register shown in Figure 6 is a 32-bit read register that contains the ICAP status bits. The bit definitions for the register are shown in Table 9. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

Figure 6: **Status Register (SR)**

Table 9: **Status Register Bit Definitions (C_BASEADDR + 0x110)**

Bit	Name	Access	Reset Value	Description
31 - 9	Reserved	N/A	0	Reserved bits
8	cfgerr_n	Read	1	Configuration error
7	dalign	Read	0	Data alignment, found syncword
6	rip	Read	0	Read back in progress

Table 9: **Status Register Bit Definitions (C_BASEADDR + 0x110) *(Cont'd)***

Bit	Name	Access	Reset Value	Description
5	in_abort_n	Read	1	Super8 (Select MAP) abort
4- 1	Always 1	Read	1	Always 1
0	Done	Read	1	AXI HWICAP done with configuration or read

Write FIFO Vacancy Register (WFV)

The WFV register shown in Figure 7 is a 32-bit read register that indicates the vacancy of the write FIFO. The actual depth of the Write FIFO is one less than the C_WRITE_FIFO_DEPTH. The bit definitions for the register are shown in Table 10. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

Figure 7: **Write FIFO Vacancy Register (WFV**

Table 10: **Write FIFO Vacancy Register Bit Definitions (C_BASEADDR + 0x114)**

Bit	Name	Access	Reset Value	Description
31 - 11	Reserved	NA	-	Reserved
10 - 0	WFV	Read	(1)	Write FIFO Vacancy

Notes:
1. This value is based on the Write FIFO size. For example, if the Write FIFO is of 1024 depth, this value would be 3FF.

Read FIFO Occupancy Register (RFO)

The RFO register shown in Figure 8 is a 32-bit read register that indicates occupancy of the read FIFO. The actual depth of the Read FIFO is one less than the C_READ_FIFO_DEPTH. The bit definitions for the register are shown in Table 11. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

Figure 8: **Read FIFO Occupancy Register (RFO)**

Table 11: **Read FIFO Occupancy Register Bit Definitions (C_BASEADDR + 0x118)**

Bit	Name	Access	Reset Value	Description
31 - 8	Reserved	N/A	-	Reserved
7 - 0	RFO	Read	0	Read FIFO Occupancy

Interrupt Descriptions

The interrupt signals generated by the AXI HWICAP are managed by the Interrupt Service Controller (ISC). This unit provides many of the features commonly provided for interrupt handling. Please refer to the *Processor IP Reference Guide* under Part 1 for a complete description of the GIE, IPISR and IPIER. The AXI HWICAP has four unique interrupts that are sent to the CPU.

Global Interrupt Enable Register (GIE)

The GIE register shown in Figure 9 and described in Table 12 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.

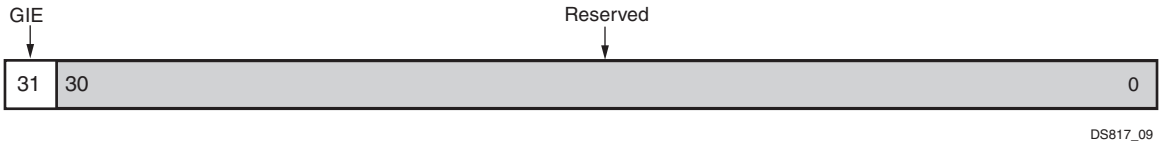

Figure 9: **Global Interrupt Enable Register (GIER)**

Table 12: **Global Interrupt Enable Register Bit Definitions (C_BASEADDR + 0x01C)**

Bit(s)	Name	Access	Reset Value	Description
31	GIE	R/W	'0'	'0' = Disabled '1' = Enabled
30 - 0	Reserved	N/A	N/A	Reserved

IP Interrupt Status Register (IPISR)

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 10 and described in Table 13. The interrupt register is a read/toggle on write register, and by writing a '1' to a bit position within the register causes the corresponding bit position in the register to 'toggle'. All register bits are cleared upon reset.

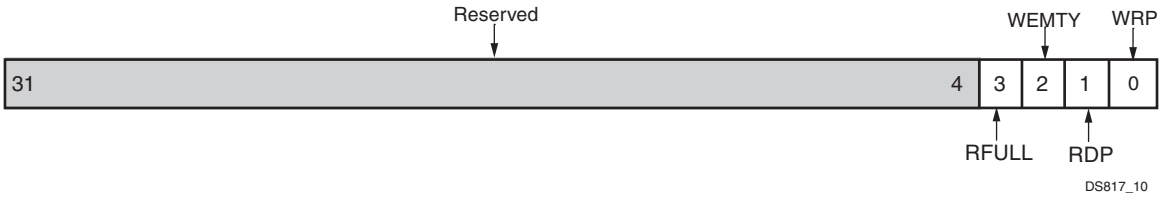

Figure 10: **IP Interrupt Status Register (IPISR)**

Table 13: **IP Interrupt Status Register Bit Definitions (C_BASEADDR + 0x020)**

Bit(s)	Name	Access	Reset Value	Description
31 - 4	Reserved	N/A	N/A	Reserved
3	RFULL	R/TOW(1)	'0'	Read FIFO full
2	WEMTY	R/TOW(1)	'0'	Write FIFO empty

Table 13: **IP Interrupt Status Register Bit Definitions (C_BASEADDR + 0x020)**

Bit(s)	Name	Access	Reset Value	Description
1	RDP	R/TOW(1)	'0'	Interrupt set and remains set if the read FIFO occupancy is greater than half of the read FIFO size
0	WRP	R/TOW(1)	'0'	Interrupt set and remains set if the write FIFO occupancy is less than half of the write FIFO size

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.

IP Interrupt Enable Register (IPIER)

The IPIER has an enable bit for each defined bit of the IPISR as shown in Figure 11 and described in Table 14. All bits are cleared upon reset.

Figure 11: **IP Interrupt Enable Register (IPIER)**

Table 14: **IP Interrupt Enable Register Bit Definitions (C_BASEADDR + 0x028)**

Bit(s)	Name	Access	Reset Value	Description
31 - 4	Reserved	N/A	N/A	Reserved
3	RFULLE	R/W(1)	'0'	Read FIFO full interrupt enable
2	WEMTYE	R/W(1)	'0'	Write FIFO empty interrupt enable
1	RDPE	R/W(1)	'0'	Read FIFO occupancy greater than half of its size interrupt enable
0	WRPE	R/W(1)	'0'	Write FIFO occupancy less than half of its size interrupt enable

Notes:

1. Writing '1' to this bit will enable the particular interrupt. Writing '0' to this bit will disable the particular interrupt.

Abort

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[7:4] pins over the next four clock cycles. The other icap_dout pins are always High. After the abort sequence finishes, the user can re-synchronize the configuration logic and resume configuration. Abort enable the user to know the status of the ICAP during the configuration or reading the configuration.

Software Support

Documentation for the associated software drivers for this hardware module are also available in EDK.

User Application Hints

The use of the AXI HWICAP is outlined in the steps below.

- Read or Configuration (write)
  - Write the bit-stream in to the Write FIFO Register (WF) to configure. Get the bit-stream from the Read FIFO Register (RF) to read.
  - Write in to the Control Register (CR) to initiate the read or write of bit-stream. The CR register determines the direction of the data transfer. Writing "0x00000001" in to the Control Register (CR) initiates the write configuration. Writing "0x00000002" in to the Control Register (CR) initiates the read. This is shown as rnc[7:0] in Figure 12, Figure 13 and Figure 14.
  - Done bit in the Status Register (SR) indicates whether the ICAP interface is busy with writing or reading data from or to the ICAP bus. It does not indicate that the read or configuration with ICAP is completed successfully.
  - Hardware clears the Control Register (CR) bits after the successful completion of the read or configuration.
  - Software should not initiate another read or configuration to ICAP until the read or configuration bit in the Control Register (CR) is cleared.
- Abort
  - Write in to the Control Register (CR) to initiate the read or write of bit-stream. The CR register determines the direction of the data transfer. Writing "0x00000001" in to the Control Register (CR) initiates the configuration. Writing "0x00000002" in to the Control Register (CR) initiates the read. This is shown as rnc[7:0] in Figure 12, Figure 13 and Figure 14.
  - Write the bit-stream in to the Write FIFO Register (WF) to configure. Get the bit-stream from the Read FIFO Register (RF) to read.
  - Write '1' in to the 5th bit of the Control Register (CR) to initiates abort.
  - Done bit in the Status Register (SR) indicates whether the ICAP interface is busy with writing/reading data from/to the ICAP bus. It does not indicate that the read/configuration with ICAP is completed successfully.
  - Hardware clears the Control Register (CR) bits after the successful completion of the abort-on read or abort-on configuration or normal abort.
  - Software should not initiate another read or configuration to ICAP until the read or configuration bit in the Control Register (CR) is cleared.

Timing Diagrams

The timing diagrams shown in Figure 12, Figure 13, and Figure 14 show the read and write cycles of AXI HWICAP core for the Virtex-6 and Spartan-6 FPGAs.

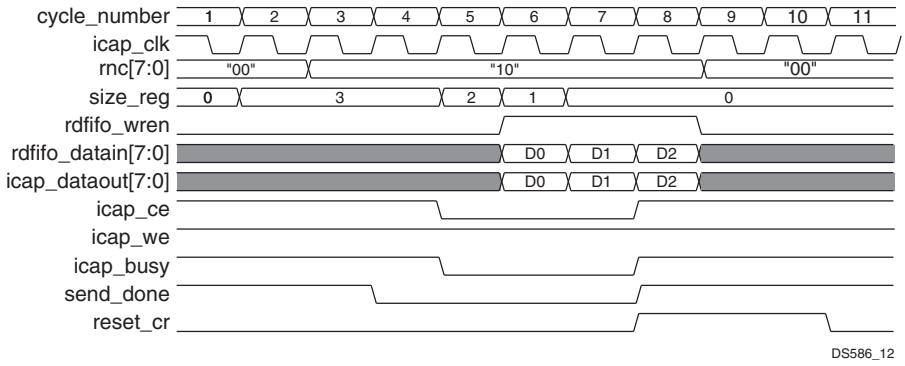

Figure 12: **Read Cycle for Virtex-6 Devices**

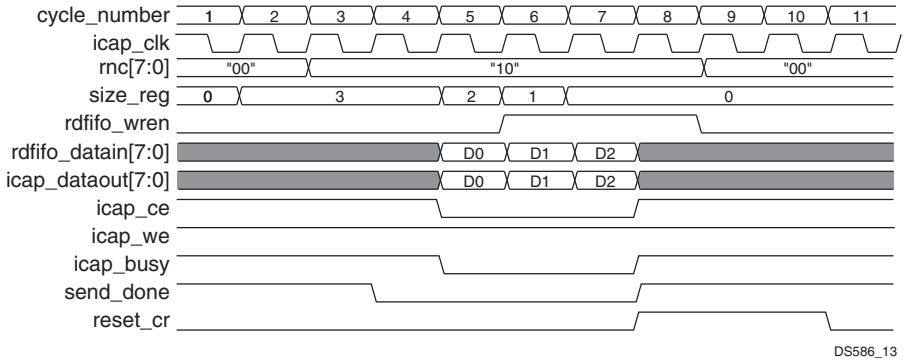

Figure 13: **Read Cycle for Spartan-6 Devices**

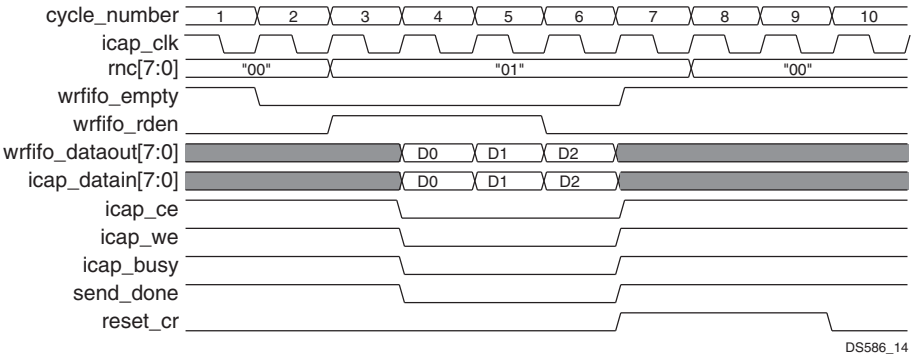

Figure 14: **Write Cycle**

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 are 1 bit wide and spans the full column of the FPGA. Configuration frames in the CLB space also contain 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 contains 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 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 will continue to operate unaffected.

There are two exceptions to this rule: 1) when the LUTs are configured in the Shift Register mode and 2) 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 will be interrupted and it will lose its state. To resolve the problem, 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 will maintain their state.

Important Notes:

1. The HWICAP core uses the ICAP found inside Virtex-6 and Spartan-6 devices. The ICAP port interface is similar to the SelectMAP interface, but is accessible from general interconnects rather than the device pins. The JTAG or "Boundary Scan" configuration mode pin setting (M2:M0 = 101) will disable the ICAP interface. If JTAG will be used as the primary configuration method, another mode pin setting must be selected to avoid disabling the ICAP interface. JTAG configuration will remain available because it overrides other means of configuration, and the HWICAP core will function as intended. Besides being disabled by the Boundary Scan mode pin setting, the ICAP will also be disabled if the persist bit in the device configuration logic's control register is set. When using bitgen, the Persist option must be set to No, which is the default. This option is generally specified in the bitgen.ut file in the etc./subdirectory of the EDK project. The maximum frequency of operation for ICAP on the Virtex-6 device is 100 MHz. In the case of the Spartan-6 device, the maximum frequency of operation for ICAP is 20 MHz.
2. In the case of the Virtex-6 device, if the AXI operates at less than 100MHz, the ICAP clock must be given frequency equivalent to AXI clock frequency. For example, if the AXI frequency is 90 MHz, the ICAP clock should also be 90 MHz. It is suggested to derive two independent clocks from the clock generator even the frequencies are same. If the AXI operates greater than 100 MHz, the ICAP clock must be fixed at 100 MHz.
3. In case of the Spartan-6 device, the ICAP clock must be connected to 20 MHz.

Design Constraints

Timing Constraints on the clocks:

The core has two different clock domains: S_AXI_ACLK and ICAP_Clk. A timing ignore constraint should be added to isolate these two clock domains. The constraints given below can be used with the AXI HWICAP core.

Period Constraints for Clock Nets

If the clock generator instantiated as below.

BEGIN clock_generator

PORT CLKOUT1 = S_AXI_ACLK_125_0000MHz

PORT CLKOUT2 = ICAP_CLK_100MHz

END

Then the following constraints are required in the UCF.

NET "AXI_Clk_125_0000MHz" TNM = "AXICLK_GRP";

NET "ICAP_CLK_100MHz" TNM = "ICAPCLK_GRP";

TIMESPEC TS_TIG0 = FROM "AXICLK_GRP" TO "ICAPCLK_GRP" TIG;

TIMESPEC TS_TIG1 = FROM "ICAPCLK_GRP" TO "AXICLK_GRP" TIG;

Timing Constraints on the ICAP interface:

If the Internal Configuration Access Port (ICAP) is used as the configuration port for partially reconfiguring the FPGA, timing constraints can be very useful to understand the potential performance of this interface. It is important to understand that the paths to the ICAP and from the ICAP are not covered by PERIOD constraints. The ICAP inputs and outputs are not considered synchronous by TRCE. This means that the inputs to and the outputs from the ICAP must be constrained using the exception constraint: NET MAXDELAY.

The following MAXDELAY constraints are required in the UCF, after adjusting the left-most instance names:

NET "axi_hwicap_0/axi_hwicap_0/HWICAP_CTRL_I/icap_statemachine_I1/Icap_datain<*>" MAXDELAY = 2 ns;

NET "axi_hwicap_0/axi_hwicap_0/HWICAP_CTRL_I/icap_statemachine_I1/Icap_ce" MAXDELAY = 2 ns;

NET "axi_hwicap_0/axi_hwicap_0/HWICAP_CTRL_I/icap_statemachine_I1/Icap_we" MAXDELAY = 2 ns;

The asterisk represents the entire bus (that is, 0, 1, 2, …). The NET MAXDELAY constraint constrains only the net delay. It does not take the setup time or clock-to-out time into consideration. The ICAP component cannot be added to time groups because it is not considered a synchronous element. Therefore, the ICAP cannot be made a synchronous component by use of a TPSYNC constraint.

Design Implementation

Target Technology

The target technology is an FPGA listed in the Supported Device Family field in the LogiCORE IP Facts Table.

Device Utilization and Performance Benchmarks

Because the AXI HWICAP core will be used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the AXI HWICAP core is combined with other designs in

the system, the utilization of FPGA resources and timing of the AXI HWICAP design will vary from the results reported here.

Table 15 lists the AXI HWICAP resource utilization data for various parameter combinations measured with the Virtex-6 FPGA as the target device.

Table 15: **Performance and Resource Utilization Benchmarks on the Virtex-6 FPGA (xc6vlx240t)**

Parameter Values				Device Resources			Performance	
FIFO_TYPE	C_S_AXI_WIDTH	C_WRITE_FIFO_DEPTH	C_READ_FIFO_DEPTH	Slices	Slice Flip-Flops	LUTs	AXI f_{MAX} (MHz)	ICAP f_{MAX} (MHz)
0	32	128	128	86	187	278	160	80
0	32	256	128	115	186	377	160	80
0	32	512	128	179	189	605	160	80
1	32	512	128	67	161	186	160	80
1	32	1024	128	65	164	184	160	80

Table 16 lists the AXI HWICAP resource utilization data for various parameter combinations measured with the Spartan-6 FPGA as the target device.

Table 16: **Performance and Resource Utilization Benchmarks on the Spartan-6 FPGA (xc6slx45t-fgg484-1)**

Parameter Values				Device Resources			Performance	
FIFO_TYPE	C_S_AXI_WIDTH	C_WRITE_FIFO_DEPTH	C_READ_FIFO_DEPTH	Slices	Slice Flip-Flops	LUTs	AXI f_{MAX} (MHz)	ICAP f_{MAX} (MHz)
0	32	128	128	83	139	211	100	80
0	32	256	128	84	141	244	100	20
0	32	512	128	127	144	367	100	20
1	32	512	128	86	217	227	100	20
1	32	1024	128	91	229	241	100	20

System Performance

To measure the system performance (F_{MAX}) of the AXI Sysmon ADC core, it was added as the Device Under Test (DUT) to a Virtex-6 FPGA system as shown in Figure 15.

Because the core is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When this core is combined with other designs in the system, the utilization of FPGA resources and timing of the design will vary from the results reported here.

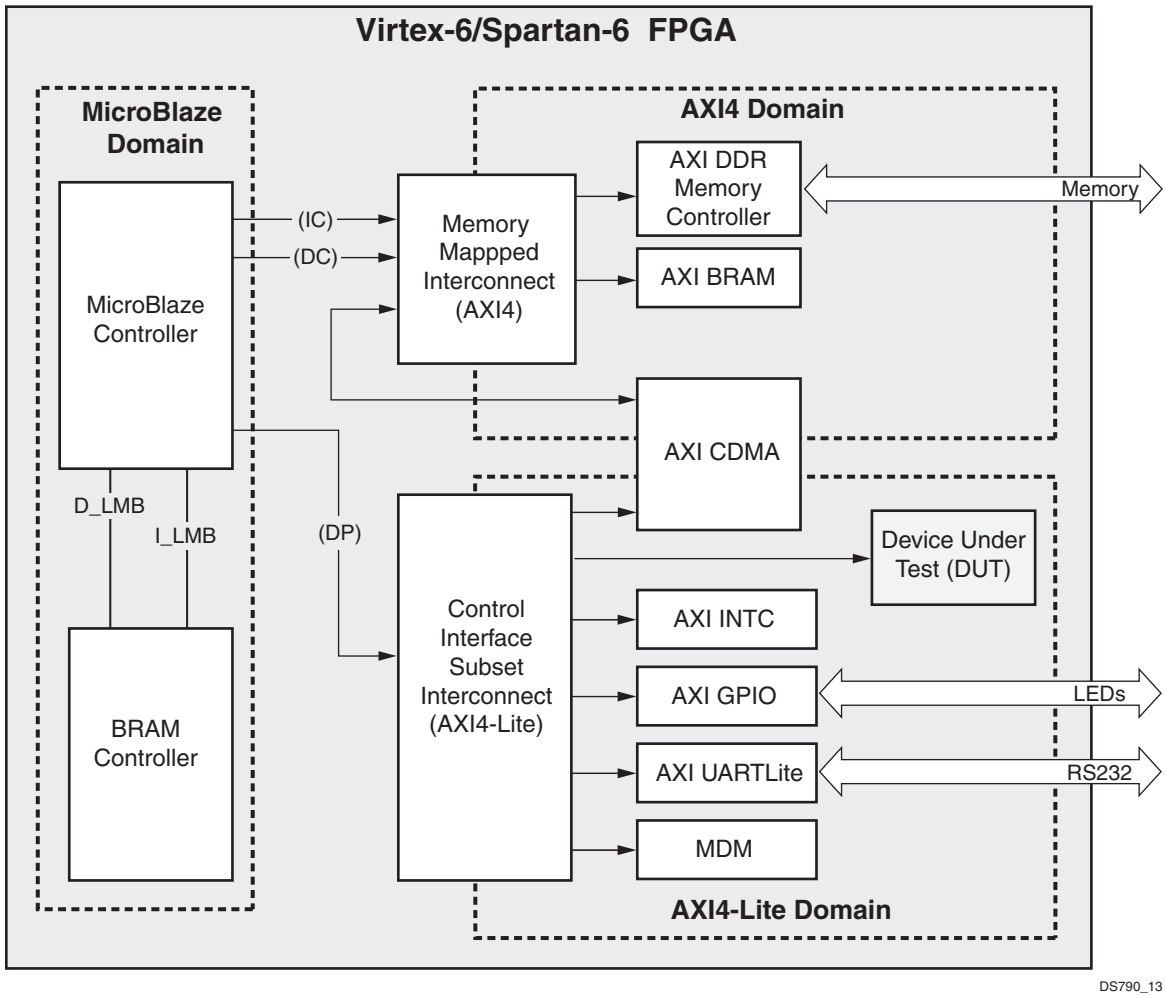

Figure 15: **Virtex-6 and Spartan-6 FPGA System with the AXI HWICAP core as the DUT**

The target FPGA was then filled with logic to drive the LUT and BRAM utilization to approximately 70% and the I/O utilization to approximately 80%. Using the default tool options and the slowest speed grade for the target FPGA, the resulting target F_{MAX} numbers are shown in Table 17.

Table 17: **System Performance**

Target FPGA	Target F_{MAX} (MHz)		
	AXI4	AXI4-Lite	MicroBlaze
xc6slx45t (1)	90 MHz	120 MHz	80
xc6vlx240t (2)	135 MHz	180 MHz	135

Notes:

1. Spartan-6 FPGA LUT utilization: 60%; Block RAM utilization: 70%; IO utilization: 80%; MicroBlaze Controller not AXI4 interconnect; AXI4 interconnect configured with a single clock of 120MHz.
2. Virtex-6 FPGA LUT utilization: 70%; Block RAM utilization: 70%; IO utilization: 80%.

The target F_{MAX} is influenced by the exact system and is provided for guidance. It is not a guaranteed value across all systems.

Reference Documents

The following document contains reference information important to understanding the AXI HWICAP core:

1. AXI4 AMBA® AXI Protocol Version: 2.0 Specification (ARM IHI 0022C)
2. DS765 LogiCORE IP AXI-Lite IPIF (v1.01a) Data Sheet
3. DS160 Spartan-6 Family Overview
4. DS150 Virtex-6 Family Overview

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 Embedded Edition software (EDK).

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

Date	Version	Revision
12/14/10	1.0	Initial release.
3/1/2011	1.1	Updated to v2.00a for the 13.1 release.
6/22/2011	1.2	Updated for the 13.2 release.

Notice of Disclaimer