XADC Wizard v3.3

LogiCORE IP Product Guide

Vivado Design Suite

PG091 October 5, 2016

XILINX
ALL PROGRAMMABLE™

Table of Contents

Introduction

The LogiCORE™ IP Xilinx® Analog-to-Digital Converter (XADC) Wizard generates an HDL wrapper to configure the XADC primitive for user-specified external channels, internal sensor channels, modes of operation, and alarms

Features

- Analog-to-digital conversion
- FPGA temperature and voltage monitoring
- Generate alarms based on user set parameters
- Optional AXI4-Lite interface based on the AXI4 specification
- Optional AXI4-Stream interface based on the AXI4-Stream specification
- Simple user interface
- Easy configuration of various modes and parameters
- Simple interface for channel selection and configuration
- Ability to select/deselect alarm outputs and set alarm limits
- Calculates all the parameters and register values

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1)	Zynq®-7000 All Programmable SoC, 7 Series
Supported User Interfaces	AXI4-Lite, AXI4-Stream
Resources	Performance and Resource Utilization web page
Provided with Core	
Design Files	Verilog and VHDL
Example Design	Verilog
Test Bench	Verilog
Constraints File	Xilinx Design Constraints (XDC)
Simulation Model	Not Provided
Supported S/W Driver(2)	Standalone
Tested Design Flows(3)	
Design Entry	Vivado® Design Suite
Simulation	For 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 listing of supported devices, see the Vivado IP catalog.
2. Standalone driver details can be found in the Software Development Kit (SDK) directory (<install_directory>/SDK/<release>/data/embeddedsw/doc/xilinx_drivers.htm). Linux OS and driver support information is available from the Xilinx Wiki page.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The XADC Wizard generates Verilog or VHDL Register Transfer Level (RTL) source code to configure the XADC primitive in Xilinx® 7 series FPGAs. An example design and simulation test bench demonstrate how to integrate the core into user designs.

The XADC Wizard is included with the Vivado® Design Suite. For information about system requirements and installation, see Installing the Wizard. Version 3.3 of the XADC Wizard product guide covers use of the wizard in the Vivado Design Suite only.

The top-level block diagram for the LogiCORE™ IP XADC core is shown in Figure 1-1.

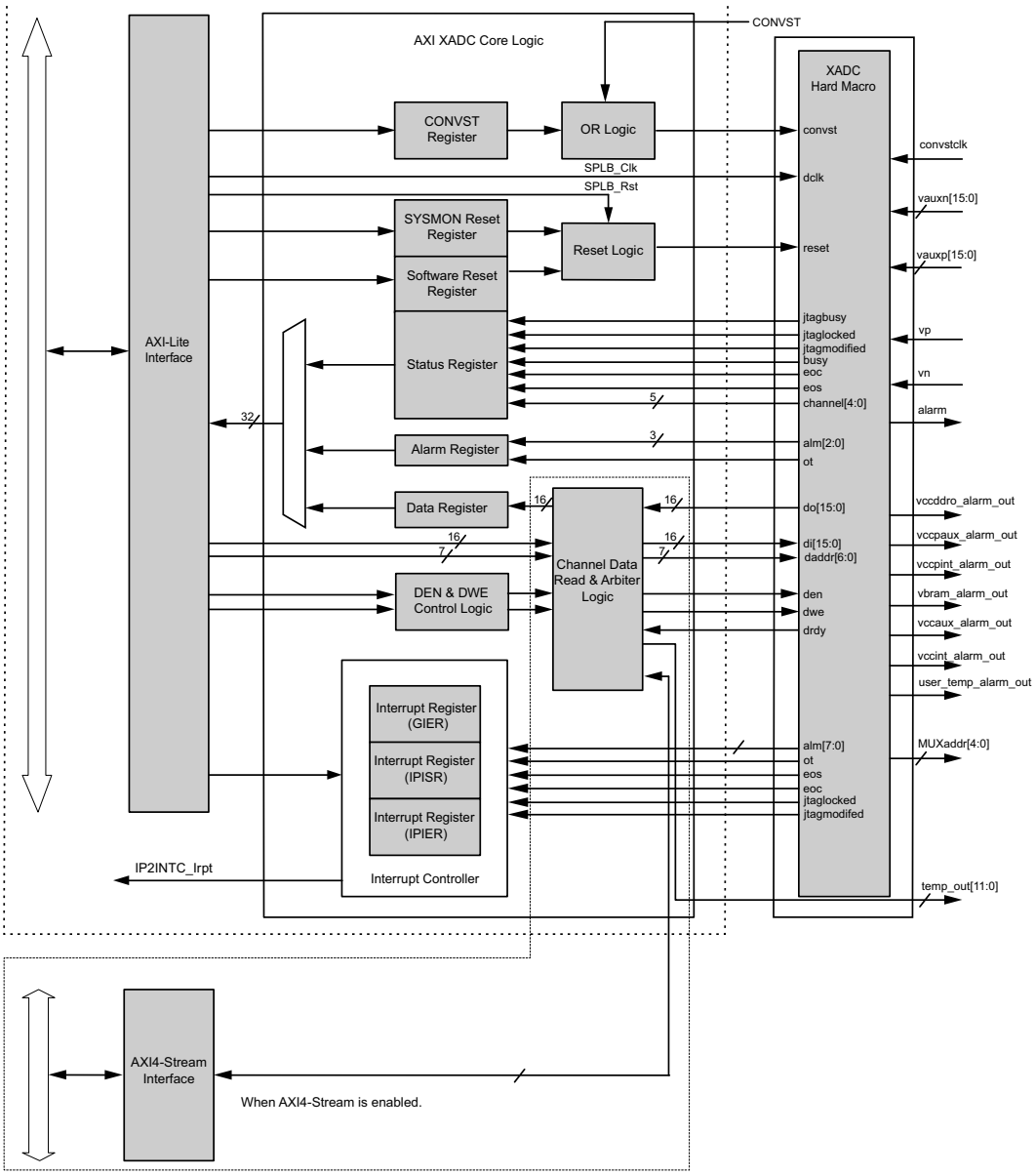

Figure 1-1: **XADC IP Core Block Diagram**

Operating System Requirements

For a list of system requirements, see the Xilinx Design Tools: Release Notes Guide.

Feature Summary

- Analog-to-digital conversion
- FPGA temperature and voltage monitoring
- Generate alarms based on user set parameters
- Optional AXI4-Lite interface based on the AXI4 specification
- Optional AXI4-Stream interface based on the AXI4-Stream specification
- Simple user interface
- Easy configuration of various modes and parameters
- Simple interface for channel selection and configuration
- Ability to select/deselect alarm outputs and set alarm limits
- Calculates all the parameters and register values

Applications

The XADC Wizard is ideally suited for high-volume applications such as multi-function printers, digital single-lens reflex (SLR) cameras, motor control, power conversion, touch/

gesture-based human machine interface (HMI), anti-tamper security, and system management. Figure 1-2 shows an application diagram.

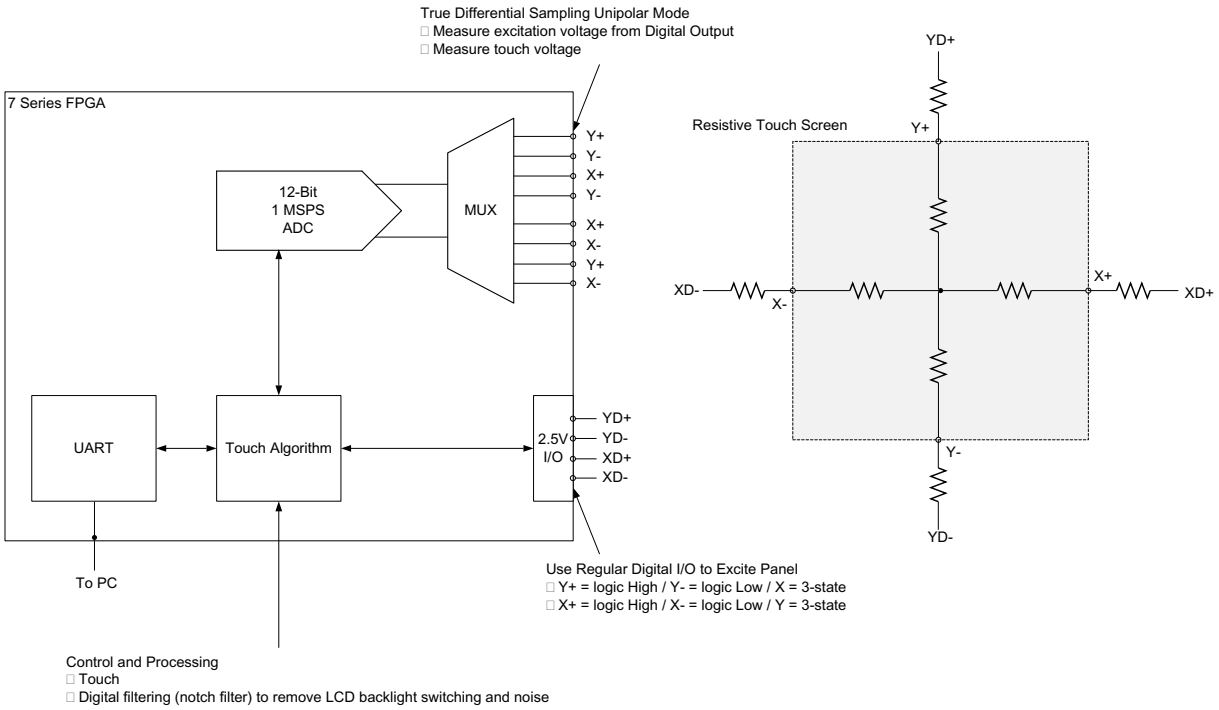

Figure 1-2: **Resistive Touch Interface**

Before You Begin

Before installing the Wizard, you must have a MySupport account. If you already have an account and have the software installed, go to Installing the Wizard; otherwise, click **Login** at the top of the Xilinx home page then follow the onscreen instructions to create a MySupport account.

Installing the Wizard

The XADC Wizard v3.3 is included with the Vivado Design Suite, and is accessed from the Vivado IP catalog. The Vivado Design Suite can be downloaded from the Xilinx Download Center.

For details, see the Vivado Design Suite Release Notes and Installation Guide.

Verifying Your Installation

Use the following procedure to verify that you have successfully installed the XADC Wizard in the Vivado IP catalog.

1. Start Vivado.
2. After creating a new 7 series family project or opening an existing one, the IP catalog appears at the right side of the window, as shown in Figure 1-3.

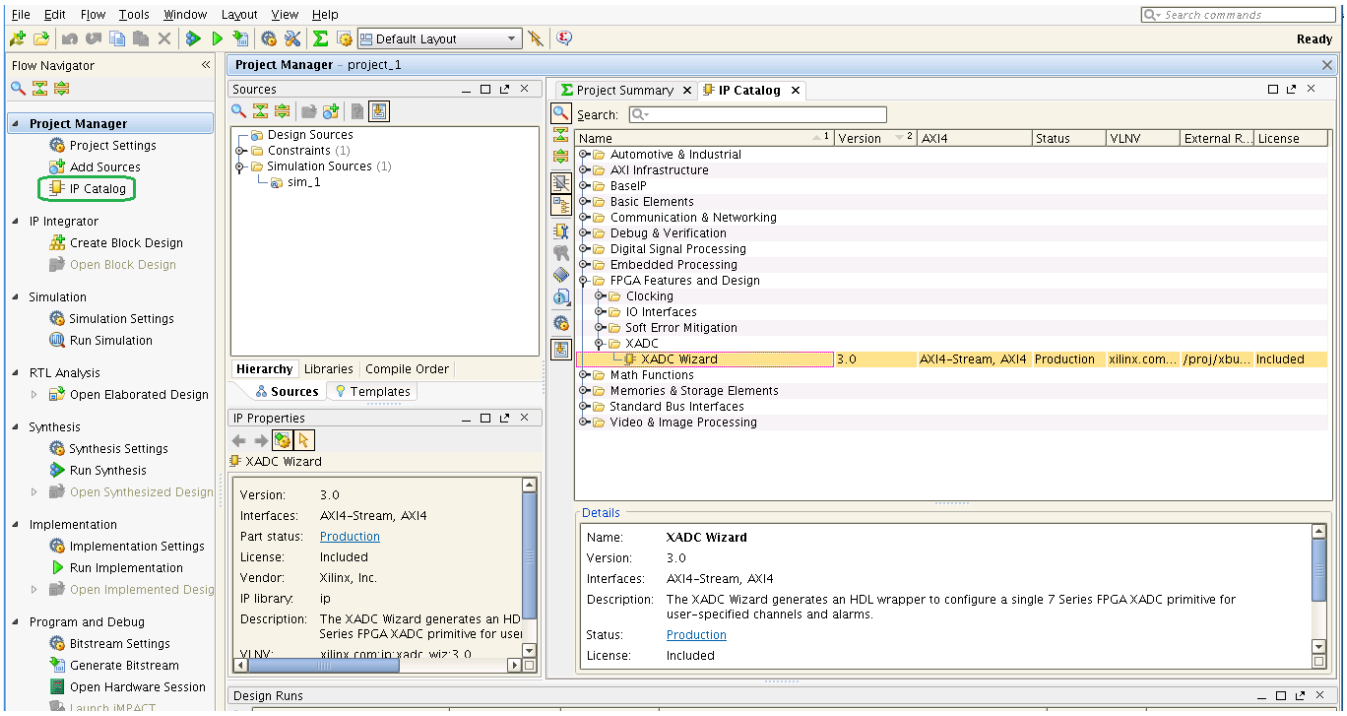

Figure 1-3: **Vivado IP Catalog**

3. Determine if the installation was successful by verifying that XADC Wizard appears at the following location in the catalog list:

   `/FPGA Features and Design/XADC`.

4. You can also generate the XADC Wizard core using the following command in the Tcl Console:

```
create_ip -name xadc_wiz -version 3.3 -vendor xilinx.com -library
ip -module_name xadc_wiz_0
```

Note: For more information about using the Vivado IP tools, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 1].

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 Agreement.

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.

License Checkers

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

- Vivado design 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.*

Product Specification

This chapter describes the Vivado® Integrated Design Environment (IDE) and follows the same flow required to set up the XADC primitive using v3.3 of the wizard.

TIP: *Tool tips are available in the Vivado IDE for most features; place your mouse over the relevant text, and additional information is provided in a popup dialog box.*

Functional Overview

The XADC Wizard is available in the Vivado IDE IP catalog that instantiates an XADC block configured to your requirements. Using the wizard, you can explicitly configure the XADC to operate in the desired mode. XADC Wizard allows you to select the channels, enable alarms, and set the alarm limits. XADC supports the following interfaces:

- AXI4-Lite
- Dynamic Reconfigurable Port (DRP)
- AXI4-Stream

XADC Functional Features

Major functional XADC features can be used to determine an appropriate mode of operation. These features include:

- Analog-to-digital conversion
- FPGA temperature and voltage monitoring
- Generate alarms based on user set parameters

Standards

The LogiCORE™ IP XADC Wizard core contains AXI4-Lite and AXI4-Stream interfaces, which are based on the AMBA® AXI4 specification [Ref 2].

Performance

If you enable averaging of the channel, data capture rate is reduced depending on the averaging selected. Choose the appropriate value to match your requirement. Analog Input noise from the supply or board can deviate from the expected 12-bit digital output.

Maximum Frequencies

Note: Maximum frequency numbers for Zynq®-7000 All Programmable SoC devices are expected to be similar to 7 series device numbers.

The maximum `s_axi_aclk/m_axis_aclk/s_axis_aclk/dclk` clock frequency supported is 250 MHz for the XADC primitive, but this might vary depending on the device and speed grade selected.

Resource Utilization

For details about resource utilization, visit Performance and Resource Utilization.

Port Descriptions

Table 2-1 describes the input and output ports provided from the XADC Wizard. Availability of ports is controlled by user-selected parameters. For example, when Dynamic Reconfiguration is selected, only ports associated with Dynamic Reconfiguration are exposed. Any port that is not exposed is tied off or connected to a signal labeled as unused in the delivered source code.

Table 2-1: **XADC I/O Signals**

Port	Direction	Description
DRP Ports		
di_in[15:0]	Input	Input data bus for the dynamic reconfiguration port (DRP).
do_out[15:0]	Output	Output data bus for the dynamic reconfiguration port.
daddr_in[6:0]	Input	Address bus for the dynamic reconfiguration port.
den_in	Input	Enable signal for the dynamic reconfiguration port.
dwe_in	Input	Write enable for the dynamic reconfiguration port.
dclk_in	Input	Clock input for the dynamic reconfiguration port.
drdy_out	Output	Data ready signal for the dynamic reconfiguration port.

Table 2-1: **XADC I/O Signals *(Cont'd)***

Port	Direction	Description
Control and Reset Ports		
reset_in	Input	Reset signal for the XADC control logic and maximum/minimum registers.
convst_in	Input	Convert start input. This input is used to control the sampling instant on the ADC input and is only used in Event Mode Timing (see Event-Driven Sampling in the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].
convstclk_in	Input	Convert start input. This input is connected to a global clock input on the interconnect. Like CONVST, this input is used to control the sampling instant on the ADC inputs and is only used in Event Mode Timing.
External Analog Inputs		
vp_in vn_in	Input	One dedicated analog-input pair. The XADC has one pair of dedicated analog-input pins that provide a differential analog input.
vauxp15[15:0] vauxn15[15:0]	Inputs	16 auxiliary analog-input pairs. Also, the XADC uses 16 differential digital-input pairs as low-bandwidth differential analog inputs. These inputs are configured as analog during FPGA configuration.
Alarm and Status Ports		
user_temp_alarm_out	Output	XADC temperature-sensor alarm output.
vccint_alarm_out	Output	XADC VCCINT-sensor alarm output.
vccaux_alarm_out	Output	XADC VCCAUX-sensor alarm output.
ot_out	Output	Over-Temperature alarm output.
channel_out[4:0]	Outputs	Channel selection outputs. The ADC input MUX channel selection for the current ADC conversion is placed on these outputs at the end of an ADC conversion.
eoc_out	Output	End of Conversion signal. This signal transitions to an active-High at the end of an ADC conversion when the measurement result is written to the Status registers. For detailed information, see the XADC Timing section in the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].
eos_out	Output	End of Sequence. This signal transitions to an active-High when the measurement data from the last channel in the Channel Sequencer is written to the Status registers. For detailed information, see the XADC Timing section in the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].
busy_out	Output	ADC busy signal. This signal transitions High during an ADC conversion. This signal transitions High for an extended period during calibration.
jtaglocked_out	Output	Used to indicate that drp port has been locked by the JTAG interface.
jtagmodified_out	Output	Used to indicate that a JTAG write to the drp has occurred
jtagbusy_out	Output	Used to indicate that a JTAG drp transaction is in progress
vbram_alarm_out	Output	XADC VBRAM sensor alarm output.
vccpint_alarm_out	Output	XADC VCCPINT sensor alarm output.

Table 2-1: **XADC I/O Signals *(Cont'd)***

Port	Direction	Description
vccpaux_alarm_out	Output	XADC VCCPAUX sensor alarm output.
vccddro_alarm_out	Output	XADC VCCDDRO sensor alarm output.
muxaddr_out[4:0]	Output	Use in external multiplexer mode to decode external MUX channel.
alarm_out	Output	Logic OR of alarms. Can be used to flag occurrence of any alarm.
temp_out[11:0]	Output	12-bit temperature output bus for the Memory Interface Generator (MIG). This should be connected to xadc_device_temp_i_pin of MIG.
AXI4-Lite Ports		
s_axi_aclk	Input	AXI clock connects to DCLK of XADC primitive.
s_axi_aresetn	Input	AXI Reset, Active-Low
s_axi_awaddr[10:0]	Input	AXI Write address. The write address bus gives the address of the write transaction.
s_axi_awvalid	Input	Write address valid. This signal indicates that a valid write address and control information are available.
s_axi_awready	Output	Write address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
s_axi_wdata[31:0]	Input	Write data
s_axi_wstb[3:0]	Input	Write strobes. This signal indicates which byte lanes to update in memory.
s_axi_wvalid	Input	Write valid. This signal indicates that valid write data and strobes are available.
s_axi_wready	Output	Write ready. This signal indicates that the slave can accept the write data.
s_axi_bresp[1:0]	Output	Write response. This signal indicates the status of the write transaction 00 = OKAY (normal response) 10 = SLVERR (error condition) 11 = DECERR (not issued by core)
s_axi_bvalid	Output	Write response valid. This signal indicates that a valid write response is available.
s_axi_bready	Input	Response ready. This signal indicates that the master can accept the response information.
s_axi_araddr[10:0]	Input	Read address. The read address bus gives the address of a read transaction.
s_axi_arvalid	Input	Read address valid. This signal indicates, when High, that the read address and control information is valid and remains stable until the address acknowledgement signal, s_axi_arready, is High.
s_axi_arready	Output	Read address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
s_axi_rdata[31:0]	Output	Read data

Table 2-1: **XADC I/O Signals *(Cont'd)***

Port	Direction	Description
s_axi_rresp[1:0]	Output	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)
s_axi_rvalid	Output	Read valid. This signal indicates that the required read data is available and the read transfer can complete.
s_axi_rready	Input	Read ready. This signal indicates that the master can accept the read data and response information.
AXI4-Stream Ports		
m_axis_aclk	Input	The global clock signal. All signals until write interface of FIFO are sampled on the rising edge of m_axis_aclk.
s_axis_aclk	Input	The global clock signal. All streaming signals from Read interface of the FIFO are sampled on the rising edge of s_axis_aclk.
m_axis_resetn	Input	The global reset signal. m_axis_resetn is active-Low. Available when AXI4-Lite is not enabled
m_axis_tvalid	Output	Indicates that the master is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted.
m_axis_tready	Input	Indicates that the slave can accept a transfer in the current cycle.
m_axis_tdata[15:0]	Output	The primary payload that is used to provide the data that is passing across the interface. The width of the data payload is two bytes.
m_axis_tid[4:0]	Output	The data stream identifier that indicates analog channel number corresponding to the different streams of data.
ip2intc_irpt	Output	Interrupt signal. When one of the selected interrupts mentioned in the Interrupt Enable Register occurs, this signal goes High.

Notes:

1. AXI4-Lite ports are available only when Interface selection is AXI4-Lite.
2. drp, jtag, and reset_in ports are not available when AXI4-Lite interface is selected.
3. dclk_in, reset_in ports are not available when streaming interface is enabled. temp_out port is available only when streaming interface is enabled.

Register Space

XADC functionality is configured through control registers (See the Register File Interface sections in the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3]). Table 2-2 lists the attributes associated with these control registers. These control registers can be initialized using HDL by attaching HDL attributes to the XADC primitive instance and configuring them according to the information provided in Table 2-2. The control registers can also be initialized through the AXI4-Lite or DRP at run time. The XADC Wizard simplifies the initialization of these control registers in the HDL instantiation by automatically configuring them to implement the operating behavior you specify using the IP core in the Vivado IDE.

Table 2-2: **XADC Attributes**

Attribute	Name	Control Reg Address	Description
INIT_40	Configuration Register 0	40h	XADC configuration registers. For detailed information, see the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3]. For AXI4-Lite interface, 0x40 address on DRP is mapped to 0x300. AXI4-Lite addresses are offset by 0x4 for every next DRP address.
INIT_41	Configuration Register 1	41h	
INIT_42	Configuration Register 2	42h	
INIT_48 to INIT_4F	Sequence Registers	48h to 4Fh	Sequence registers used to program the Channel Sequencer function in the XADC. For detailed information, see the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].
INIT_50 to INIT_5F	Alarm Limits Registers	50h to 5Fh	Alarm threshold registers for the XADC alarm function. For detailed information, seethe *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].
SIM_MONITOR_FILE	Simulation Analog Entry File	–	This is the text file that contains the analog input stimulus. This is used for simulation.
SIM_DEVICE	Device Family Information	–	Specifies the device family. For 7 Series devices, this value is "7Series."

XADC Wizard Register Descriptions for AXI4-Lite Interface

Table 2-3 shows the XADC Wizard IP core registers and their corresponding addresses.

Table 2-3: **IP Core Registers**

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
XADC Wizard Local Register Grouping				
C_BASEADDR + 0x00	Software Reset Register (SRR)	N/A	W(1)	Software Reset Register
C_BASEADDR + 0x04	Status Register (SR)	N/A	R(2)	Status Register
C_BASEADDR + 0x08	Alarm Output Status Register (AOSR)	0x0	R(2)	Alarm Output Status Register
C_BASEADDR + 0x0C	CONVST Register (CONVSTR)	N/A	W(1)	Bit[0] = ADC convert start register(3) Bit[1] = Enable temperature update logic Bit[17:2] = Wait cycle for temperature update

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x10	XADC Reset Register (SYSMONRR)	N/A	W[1]	XADC Hard Macro Reset Register
XADC Wizard Interrupt Controller Register Grouping				
C_BASEADDR + 0x5C	Global Interrupt Enable Register (GIER)	0x0	R/W	Global Interrupt Enable Register
C_BASEADDR + 0x60	IP Interrupt Status Register (IPISR)	N/A	R/TOW[4]	IP Interrupt Status Register
C_BASEADDR + 0x68	IP Interrupt Enable Register (IPIER)	0x0	R/W	IP Interrupt Enable Register
XADC Wizard Hard Macro Register Grouping[5]				
C_BASEADDR + 0x200	Temperature	N/A	R[6]	The 12-bit Most Significant Bit (MSB) justified result of on-device temperature measurement is stored in this register.
C_BASEADDR + 0x204	V_{CCINT}	N/A	R[6]	The 12-bit MSB justified result of on-device V_{CCINT} supply monitor measurement is stored in this register.
C_BASEADDR + 0x208	V_{CCAUX}	N/A	R[6]	The 12-bit MSB justified result of on-device V_{CCAUX} Data supply monitor measurement is stored in this register.
C_BASEADDR + 0x20C	V_P/V_N	0x0	R/W[7]	When read: The 12-bit MSB justified result of A/D conversion on the dedicated analog input channel (Vp/Vn) is stored in this register. When written: Write to this register resets the XADC hard macro. No specific data is required.
C_BASEADDR + 0x210	V_{REFP}	0x0	R[6]	The 12-bit MSB justified result of A/D conversion on the reference input V_{REFP} is stored in this register.
C_BASEADDR + 0x214	V_{REFN}	0x0	R[6]	The 12-bit MSB justified result of A/D conversion on the reference input V_{REFN} is stored in this register.
C_BASEADDR + 0x218	V_{BRAM}	0x0	R[6]	The 12-bit MSB justified result of A/D conversion on the reference input V_{BRAM} is stored in this register.

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x21C	Undefined	Undefined	N/A	These locations are unused and contain invalid data.
C_BASEADDR + 0x220	Supply A Offset	N/A	R(6)	The calibration coefficient for the supply sensor offset of ADC A is stored in this register.
C_BASEADDR + 0x224	ADC A Offset	N/A	R(6)	The calibration coefficient for the ADC A offset calibration is stored in this register.
C_BASEADDR + 0x228	ADC A Gain Error	N/A	R(6)	The calibration coefficient for the gain error of ADC A is stored in this register.
C_BASEADDR + 0x22C to C_BASEADDR + 0x230	Undefined	Undefined	N/A	These locations are unused and contain invalid data.
C_BASEADDR + 0x234	Zynq-7000 Device Core Supply	N/A	R	The VCCINT of PSS core supply. Present only on Zynq-7000 devices.
C_BASEADDR + 0x238	Zynq-7000 Device Core Aux Supply	N/A	R	The VCCAUX of PSS core supply. Present only on Zynq-7000 devices.
C_BASEADDR + 0x23C	Zynq-7000 Device Core Memory Supply	N/A	R	The VCCMEM of PSS core supply. Present only on Zynq-7000 devices.
C_BASEADDR + 0x240	$V_{AUXP}[0]$/ $V_{AUXN}[0]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 0 is stored in this register.
C_BASEADDR + 0x244	$V_{AUXP}[1]$/ $V_{AUXN}[1]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 1 is stored in this register.
C_BASEADDR + 0x248	$V_{AUXP}[2]$/ $V_{AUXN}[2]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 2 is stored in this register.
C_BASEADDR + 0x24C	$V_{AUXP}[3]$/ $V_{AUXN}[3]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 3 is stored in this register.
C_BASEADDR + 0x250	$V_{AUXP}[4]$/ $V_{AUXN}[4]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 4 is stored in this register.
C_BASEADDR + 0x254	$V_{AUXP}[5]$/ $V_{AUXN}[5]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 5 is stored in this register.
C_BASEADDR + 0x258	$V_{AUXP}[6]$/ $V_{AUXN}[6]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 6 is stored in this register.

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x25C	$V_{AUXP}[7]$/ $V_{AUXN}[7]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 7 is stored in this register.
C_BASEADDR + 0x260	$V_{AUXP}[8]$/ $V_{AUXN}[8]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 8 is stored in this register.
C_BASEADDR + 0x264	$V_{AUXP}[9]$/ $V_{AUXN}[9]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 9 is stored in this register.
C_BASEADDR + 0x268	$V_{AUXP}[10]$/ $V_{AUXN}[10]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 10 is stored in this register.
C_BASEADDR + 0x26C	$V_{AUXP}[11]$/ $V_{AUXN}[11]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 11 is stored in this register.
C_BASEADDR + 0x270	$V_{AUXP}[12]$/ $V_{AUXN}[12]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 12 is stored in this register.
C_BASEADDR + 0x274	$V_{AUXP}[13]$/ $V_{AUXN}[13]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 13 is stored in this register.
C_BASEADDR + 0x278	$V_{AUXP}[14]$/ $V_{AUXN}[14]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 14 is stored in this register.
C_BASEADDR + 0x27C	$V_{AUXP}[15]$/ $V_{AUXN}[15]$	0x0	R(6)	The 12-bit MSB justified result of A/D conversion on the auxiliary analog input 15 is stored in this register.
C_BASEADDR + 0x280	Max Temp	N/A	R(6)	The 12-bit MSB justified maximum temperature measurement.
C_BASEADDR + 0x284	Max V_{CCINT}	N/A	R(6)	The 12-bit MSB justified maximum V_{CCINT} measurement.
C_BASEADDR + 0x288	Max V_{CCAUX}	N/A	R(6)	The 12-bit MSB justified maximum V_{CCAUX} measurement.
C_BASEADDR + 0x28C	Max V_{BRAM}	N/A	R(6)	The 12-bit MSB justified maximum V_{BRAM} measurement.
C_BASEADDR + 0x290	Min Temp	N/A	R(6)	The 12-bit MSB justified minimum temperature measurement
C_BASEADDR + 0x294	Min V_{CCINT}	N/A	R(6)	The 12-bit MSB justified minimum V_{CCINT} measurement
C_BASEADDR + 0x298	Min V_{CCAUX}	N/A	R(6)	The 12-bit MSB justified minimum V_{CCAUX} measurement.

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x29C	Min V_{BRAM}	N/A	R(6)	The 12-bit MSB justified minimum V_{BRAM} measurement.
C_BASEADDR + 0x2A0	Max V_{CCPINT}	N/A	R	The 12-bit MSB justified maximum V_{CCPINT} measurement. This register is only available in Zynq-7000 devices.
C_BASEADDR + 0x2A4	Max V_{CCPAUX}	N/A	R	The 12-bit MSB justified maximum V_{CCPAUX} measurement. This register is only available in Zynq-7000 devices.
C_BASEADDR + 0x2A8	Max V_{CCDDRO}	N/A	R	The 12-bit MSB justified maximum V_{CCDDRO} measurement. This register is only available in Zynq-7000 devices.
C_BASEADDR + 0x2AC	Undefined	Undefined	N/A	These locations are unused and contain invalid data.
C_BASEADDR + 0x2B0	Min V_{CCPINT}	N/A	R	The 12-bit MSB justified minimum V_{CCPINT} measurement. This register is only available in Zynq-7000 devices.
C_BASEADDR + 0x2B4	Min V_{CCPAUX}	N/A	R	The 12-bit MSB justified minimum V_{CCPAUX} measurement. This register is only available in Zynq-7000 devices.
C_BASEADDR + 0x2B8	Min V_{CCDDRO}	N/A	R	The 12-bit MSB justified minimum V_{CCDDRO} measurement. This register is only available in Zynq-7000 devices.
C_BASEADDR + 0x2BC	Undefined	Undefined	N/A	These locations are unused and contain invalid data.
C_BASEADDR + 0x2C0	Supply B Offset	N/A	R	The calibration coefficient for the supply sensor offset of ADC B is stored in this register.
C_BASEADDR + 0x2C4	ADC B Offset	N/A	R	The calibration coefficient for the ADC B offset calibration is stored in this register.
C_BASEADDR + 0x2C8	ADC B Gain Error	N/A	R	The calibration coefficient for the ADC B gain error is stored in this register.
C_BASEADDR + 0x2CC to C_BASEADDR + 0x2F8	Undefined	Undefined	N/A	These locations are unused and contain invalid data.
C_BASEADDR + 0x2FC	Flag Register	N/A	R(6)	The 16-bit register gives general status information of ALARM, Over Temperature (OT), Disable XADC information. Whether the XADC is using the internal reference voltage or external reference voltage is also provided.
C_BASEADDR + 0x300	Configuration Register 0	0x0	R/W(8)	XADC Configuration Register 0

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x304	Configuration Register 1	0x0	R/W[8]	XADC Configuration Register 1
C_BASEADDR + 0x308	Configuration Register 2	0x1E00	R/W[8]	XADC Configuration Register 2
C_BASEADDR + 0x30C to C_BASEADDR + 0x31C	Test Register 0 to 4	N/A	N/A	XADC Test Register 0 to 4 (For factory test only)
C_BASEADDR + 0x320	Sequence Register 0	0x0	R/W	XADC Sequence Register 0 (ADC channel selection)
C_BASEADDR + 0x324	Sequence Register 1	0x0	R/W	XADC Sequence Register 1 (ADC channel selection)
C_BASEADDR + 0x328	Sequence Register 2	0x0	R/W	XADC Sequence Register 2 (ADC channel averaging enable)
C_BASEADDR + 0x32C	Sequence Register 3	0x0	R/W	XADC Sequence Register 3 (ADC channel averaging enable)
C_BASEADDR + 0x330	Sequence Register 4	0x0	R/W	XADC Sequence Register 4 (ADC channel analog-input mode)
C_BASEADDR + 0x334	Sequence Register 5	0x0	R/W	XADC Sequence Register 5 (ADC channel analog-input mode)
C_BASEADDR + 0x338	Sequence Register 6	0x0	R/W	XADC Sequence Register 6 (ADC channel acquisition time)
C_BASEADDR + 0x33C	Sequence Register 7	0x0	R/W	XADC Sequence Register 7 (ADC channel acquisition time)
C_BASEADDR + 0x340	Alarm Threshold Register 0	0x0	R/W	The 12-bit MSB justified alarm threshold register 0 (Temperature Upper).
C_BASEADDR + 0x344	Alarm Threshold Register 1	0x0	R/W	The 12-bit MSB justified alarm threshold register 1 (V_{CCINT} Upper).
C_BASEADDR + 0x348	Alarm Threshold Register 2	0x0	R/W	The 12-bit MSB justified alarm threshold register 2 (V_{CCAUX} Upper).
C_BASEADDR + 0x34C	Alarm Threshold Register 3	0x0	R/W[8][9]	The 12-bit MSB justified alarm threshold register 3 (OT Upper).
C_BASEADDR + 0x350	Alarm Threshold Register 4	0x0	R/W	The 12-bit MSB justified alarm threshold register 4 (Temperature Lower).
C_BASEADDR + 0x354	Alarm Threshold Register 5	0x0	R/W	The 12-bit MSB justified alarm threshold register 5 (V_{CCINT} Lower).

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x358	Alarm Threshold Register 6	0x0	R/W	The 12-bit MSB justified alarm threshold register 6 (V_{CCAUX} Lower).
C_BASEADDR + 0x35C	Alarm Threshold Register 7	0x0	R/W	The 12-bit MSB justified alarm threshold register 7 (OT Lower).
C_BASEADDR + 0x360	Alarm Threshold Register 8	0x0	R/W	The 12-bit MSB justified alarm threshold register 8 (VBRAM Upper).
C_BASEADDR + 0x364	Alarm Threshold Register 9	0x0	R/W	The 12-bit MSB justified alarm threshold register 9 (V_{CCPint} Upper). This register is only on Zynq-7000 devices.
C_BASEADDR + 0x368	Alarm Threshold Register 10	0x0	R/W	The 12-bit MSB justified alarm threshold register 10 (V_{CCPaux} Upper). This register is only on Zynq-7000 devices.
C_BASEADDR + 0x36C	Alarm Threshold Register 11	0x0	R/W	The 12-bit MSB justified alarm threshold register 11 (V_{CCDDRO} Upper). This register is only on Zynq-7000 devices.
C_BASEADDR + 0x370	Alarm Threshold Register 12	0x0	R/W	The 12-bit MSB justified alarm threshold register 12 (VBRAM Lower).
C_BASEADDR + 0x374	Alarm Threshold Register 13	0x0	R/W	The 12-bit MSB justified alarm threshold register 13 (V_{CCPint} Lower). This register is only on Zynq-7000 devices.
C_BASEADDR + 0x378	Alarm Threshold Register 14	0x0	R/W	The 12-bit MSB justified alarm threshold register 14 (V_{CCPaux} Lower). This register is only on Zynq-7000 devices.
C_BASEADDR + 0x37C	Alarm Threshold Register 15	0x0	R/W	The 12-bit MSB justified alarm threshold register 15 (V_{CCDDRO} Lower). This register is only on Zynq-7000 devices.

Table 2-3: **IP Core Registers *(Cont'd)***

Base Address + Offset (hex)	Register Name	Reset Value (hex)	Access Type	Description
C_BASEADDR + 0x380 to C_BASEADDR + 0x3FC	Undefined	Undefined	N/A	Do not read/write these register.

Notes:

1. Reading of this register returns an undefined value.
2. Writing into this register has no effect.
3. Used in event-driven sampling mode only.
4. TOW = Toggle On Write. Writing a 1 to a bit position within the register causes the corresponding bit position in the register to toggle.
5. These are 16-bit registers internal to XADC. These are mapped to the lower halfword boundary on 32-bit XADC Wizard IP core registers.
6. Writing to this XADC hard macro register is not allowed. The XADC hard macro data registers are 16 bits in width. The XADC hard macro specification guarantees the first 12 MSB bits accuracy; so only these bits are used for reference.
7. Writing to this register resets the XADC hard macro. No specific data pattern is required to reset the XADC hard macro. Reading of this register gives the details of Vp/Vn port.
8. Read the XADC User Guide[Ref 3], for setting the different bits available in configuration registers for 7 series devices.
9. The OT upper register is a user-configurable register for the upper threshold level of temperature. If this register is left unconfigured, then the XADC considers 125°C as the upper threshold value for OT. While configuring this register, the last four bits must be set to 0011, that is, Alarm Threshold Register 3[3:0] = 0011. The upper 12 bits of this register are user configurable.

XADC Wizard Local Register Grouping for AXI4-Lite Interface

It is expected that the XADC Wizard IP core registers are accessed in their preferred-access mode only. If the write attempt is made to read-only registers, then there is not any effect on register contents. If the write-only registers are read, the result is undefined data. All the internal registers of the core have to be accessed in 32-bit format. If any other kind of access (like halfword or byte access) is done for the XADC Wizard IP core local 32-bit registers, the transaction is completed but with errors for the corresponding transaction.

Software Reset Register (SRR)

The Software Reset register permits you to reset the XADC Wizard IP core including the XADC hard macro output ports (except JTAG related outputs), independently of other IP cores in the systems. To activate software reset, the value 0x0000_000A must be written to the register. Any other access, read or write, has undefined results. The bit assignment in the Software Reset register is shown in Figure 2-1 and described in Table 2-4.

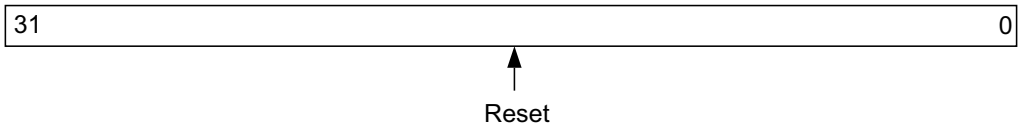

Figure 2-1: **Software Reset Register**

Table 2-4: **Software Reset Register Description (C_BASEADDR + 0x00)**

Bits	Name	Reset Value	Access Type	Description
31:0	Reset	N/A	W	The only allowed operation on this register is a write of 0x0000_000A, which resets the XADC Wizard IP Core. The reset is active only for 16 clock cycles.

Status Register (SR)

The Status register contains the XADC Wizard IP core channel status, End of Conversion (EOC), End of Sequence (EOS), and Joint Test Action Group (JTAG) access signals. This register is read only. Any attempt to write to the bits of the register would not change the bits. The Status register bit definitions are shown in Figure 2-2 and explained in Table 2-5.

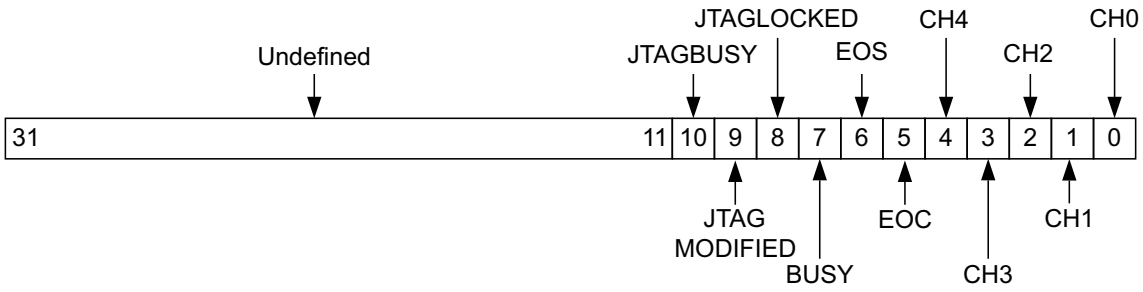

Figure 2-2: **Status Register**

Table 2-5: **Status Register (C_BASEADDR + 0x04)**

Bits	Name	Reset Value	Access Type	Description
31:11	Undefined	N/A	N/A	Undefined
10	JTAGBUSY	0	R	Used to indicate that a JTAG DRP transaction is in progress.
9	JTAG MODIFIED	0	R	Used to indicate that a write to DRP through JTAG interface has occurred. This bit is cleared when a successful DRP read/write operation through the FPGA logic is performed. The DRP read/write through the FPGA logic fails, if JTAGLOCKED = 1
8	JTAG LOCKED	0	R	Used to indicate that a DRP port lock request has been made by the Joint Test Action Group (JTAG) interface.
7	BUSY	N/A	R	ADC busy signal. This signal transitions High during an ADC conversion.
6	EOS	N/A	R	End of Sequence. This signal transitions to an active-High when the measurement data from the last channel in the auto sequence is written to the Status registers. This bit is cleared when a read operation is performed on Status register.
5	EOC	N/A	R	End of Conversion signal. This signal transitions to an active-High at the end of an ADC conversion when the measurement is written to the XADC hard macro Status register. This bit is cleared when a read operation is performed on Status register.
4:0	CHANNEL [4:0]	N/A	R	Channel selection outputs. The ADC input MUX channel selection for the current ADC conversion is placed on these outputs at the end of an ADC conversion.

Alarm Output Status Register (AOSR)

The Alarm Output Status register contains all the alarm outputs for the XADC Wizard IP core. This register is read-only. Any attempt to write to the bits of the register would not change the bits. The Alarm Output Status register bit definitions are shown in Figure 2-3 and explained in Table 2-6.

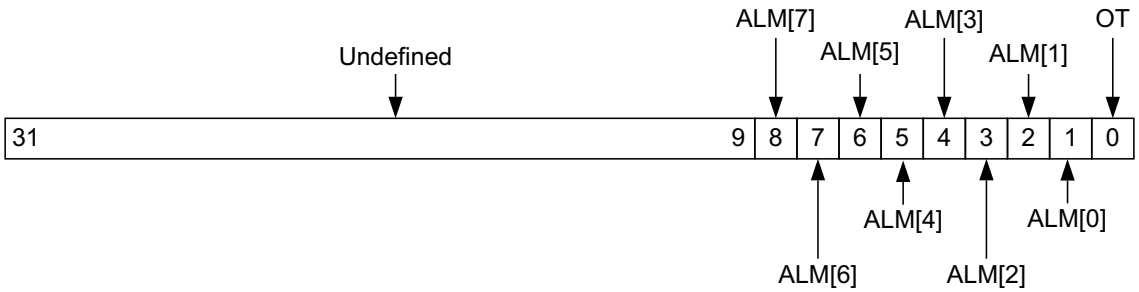

Figure 2-3: **Alarm Output Status Register**

Table 2-6: **Alarm Output Status Register (C_BASEADDR + 0x08)**

Bits	Name	Reset Value	Access Type	Description
31:9	Undefined	N/A	N/A	Undefined
8	ALM[7]	0	R	Logical ORing of ALARM bits 0 to 7. This is direct output from the XADC macro.
7	ALM[6]	0	R	**XADC V_{CCDDRO}-Sensor Status**. The XADC V_{CCDDRO}-sensor alarm output interrupt occurs when V_{CCDDRO} exceeds the user-defined threshold. This bit is only valid for Zynq-7000 devices.
6	ALM[5]	0	R	**XADC V_{CCPAUX}-Sensor Status**. The XADC V_{CCPAUX}-sensor alarm output interrupt occurs when V_{CCPAUX} exceeds the user-defined threshold. This bit is only valid for Zynq-7000 devices.
5	ALM[4]	0	R	**XADC V_{CCPINT}-Sensor Status**. The XADC V_{CCPINT}-sensor alarm output interrupt occurs when V_{CCPINT} exceeds the user-defined threshold. This bit is only valid for Zynq-7000 devices.
4	ALM[3]	0	R	**XADC V_{BRAM}-Sensor Status.** XADC V_{BRAM}-sensor alarm output interrupt occurs when V_{BRAM} exceeds user-defined threshold.
3	ALM[2]	0	R	**XADC V_{CCAUX}-Sensor Status.** XADC V_{CCAUX}-sensor alarm output interrupt occurs when V_{CCAUX} exceeds user-defined threshold.
2	ALM[1]	0	R	**XADC V_{CCINT}-Sensor Status.** XADC V_{CCINT}-sensor alarm output interrupt occurs when V_{CCINT} exceeds user-defined threshold.
1	ALM[0]	0	R	**XADC Temperature-Sensor Status.** XADC temperature-sensor alarm output interrupt occurs when device temperature exceeds user-defined threshold.
0	OT	0	R	**XADC Over-Temperature Alarm Status.** Over-Temperature alarm output interrupt occurs when the die temperature exceeds a factory set limit of 125°C.

CONVST Register (CONVSTR)

The CONVST register is used for initiating a new conversion in the event-driven sampling mode. The output of this register is logically ORed with the external CONVST input signal. This register also defines enable for the Temperature Bus update logic and the wait cycle count. The attempt to read this register results in undefined data. The CONVST register bit definitions are shown in Figure 2-4 and explained in Table 2-7.

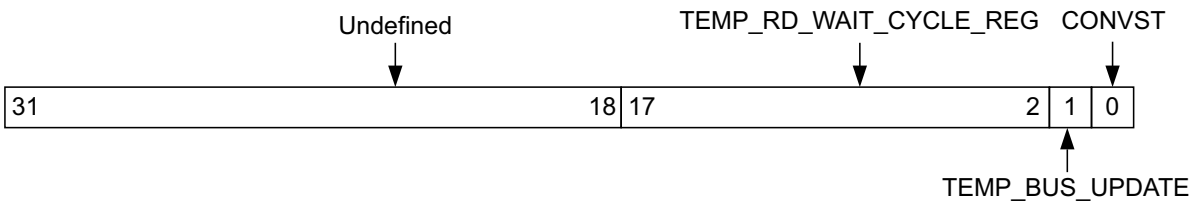

Figure 2-4: **CONVST Register**

Table 2-7: **CONVST Register (C_BASEADDR + 0x0C)**

Bits	Name	Reset Value	Access Type	Description
31:18	Undefined	N/A	N/A	Undefined
17:2	TEMP_RD_WAIT_CYCLE_REG	0x03E8	W	Wait cycle for temperature update. Temperature update logic waits for this count of the S_AXI_ACLK before reading the Temperature register. This value should be such that the period is more than the ADC conversion rate.
1	TEMP_BUS_UPDATE	0	W	Enable temperature update logic enables the temperature read from XADC and updates of TEMP_OUT port.
0	CONVST	0	W	A rising edge on the CONVST input initiates start of ADC conversion in event-driven sampling mode. For the selected channel the CONVST bit in the register needs to be set to 1 and again reset to 0 to start a new conversion cycle. The conversion cycle ends with EOC bit going High.

XADC Reset Register

The XADC Reset register is used to reset only the XADC hard macro. As soon as the reset is released the ADC begins with a new conversion. If sequencing is enabled this conversion is the first in the sequence. This register resets the OT and ALM[n] output from the XADC hard macro. This register does not reset the interrupt registers if they are included in the design. Also any reset from the FPGA logic does not affect the RFI (Register File Interface) contents of XADC hard macro. The attempt to read this register results in undefined data. The XADC Reset register bit definitions are shown in Figure 2-5 and explained in Table 2-8.

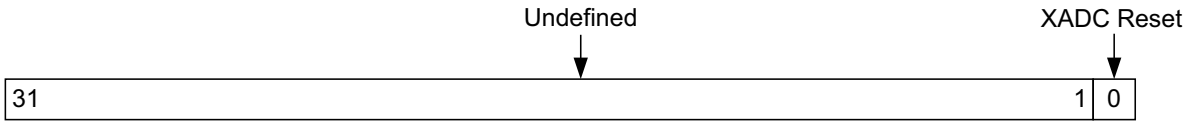

Figure 2-5: **XADC Reset Register**

Table 2-8: **XADC Reset Register (C_BASEADDR + 0x10)**

Bits	Name	Reset Value	Access Type	Description
31:1	Undefined	N/A	N/A	Undefined
0	XADC Reset	0	Write	Writing a 1 to this bit position resets the XADC hard macro. The reset is released only after 0 is written to this register.

Interrupt Controller Register Grouping for AXI4-Lite Interface

The Interrupt Controller Module is included in the XADC Wizard IP core design when C_INCLUDE_INTR = 1. The XADC Wizard has several distinct interrupts that are sent to the Interrupt Controller Module, which is one of the submodules of the XADC Wizard IP core. The Interrupt Controller Module allows each interrupt to be enabled independently (by the IP Interrupt Enable register (IPIER)). All the interrupt signals are rising-edge sensitive.

Interrupt registers are strictly 32-bit accessible. If byte/halfword or without byte enables access is made, the core behavior is not guaranteed.

The interrupt registers are in the Interrupt Controller Module. The XADC Wizard permits multiple conditions for an interrupt or an interrupt strobe which occurs only after the completion of a transfer.

Global Interrupt Enable Register (GIER)

The Global Interrupt Enable register is used to globally enable the final interrupt output from the Interrupt Controller as shown in Figure 2-6 and described in Table 2-9. This bit is a read/write bit and is cleared upon reset.

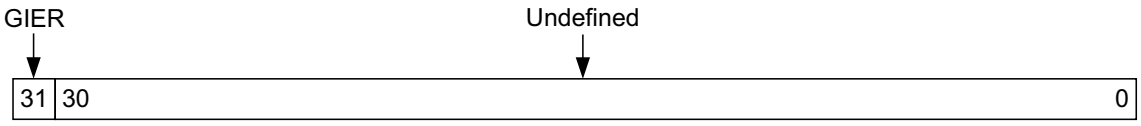

Figure 2-6: **Global Interrupt Enable Register (GIER)**

Table 2-9: **Global Interrupt Enable Register (GIER) Description (C_BASEADDR + 0x5C)**

Bits	Name	Reset Value	Access Type	Description
31	GIER	0	R/W	**Global Interrupt Enable Register.** It enables all individually enabled interrupts to be passed to the interrupt controller. 0 = Disabled 1 = Enabled
30:0	Undefined	N/A	N/A	Undefined.

IP Interrupt Status Register (IPISR)

The six unique interrupt conditions in the XADC Wizard IP core include:

- OT
- ALM[6:0]
- OT DEACTIVE,
- ALM[0] DEACTIVE,
- JTAG LOCKED/MODIFIED
- EOC/EOS

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 2-7 and described in Table 2-10. 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.

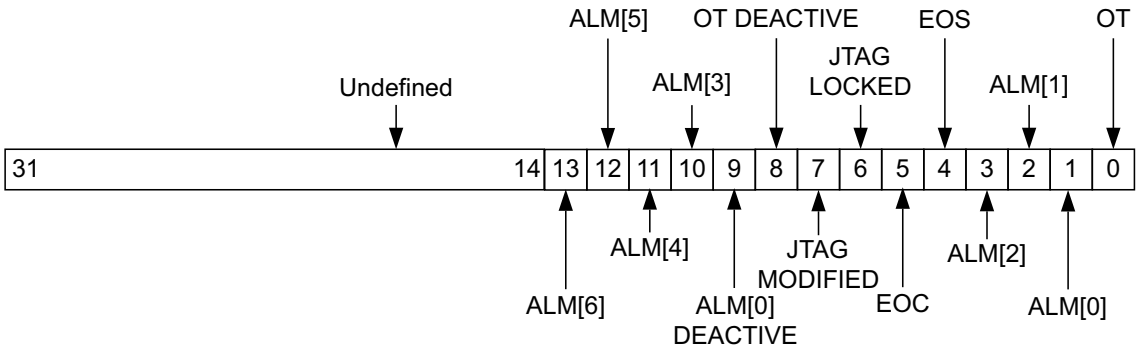

Figure 2-7: **IP Interrupt Status Register (IPISR)**

Table 2-10: **IP Interrupt Status Register (IPISR) Description (C_BASEADDR + 0x60)**

Bits	Name	Reset Value	Access Type	Description
31:14	Undefined	N/A	N/A	Undefined
13	ALM[6]	0	R/TOW(1)(2)	**XADC V_{CCDDRO}-Sensor Interrupt**. The XADC V_{CCDDRO}-sensor alarm output interrupt occurs when V_{CCDDRO} exceeds the user-defined threshold. This bit is only valid for Zynq-7000 devices.
12	ALM[5]	0	R/TOW(1)(2)	**XADC V_{CCPAUX}-Sensor Interrupt**. The XADC V_{CCPAUX}-sensor alarm output interrupt occurs when V_{CCPAUX} exceeds the user-defined threshold. This bit is only valid for Zynq-7000 devices.
11	ALM[4]	0	R/TOW(1)(2)	**XADC V_{CCPINT}-Sensor Interrupt**. The XADC V_{CCPINT}-sensor alarm output interrupt occurs when V_{CCPINT} exceeds the user-defined threshold. This bit is only valid for Zynq-7000 devices
10	ALM[3]	0	R/TOW(1)(2)	**XADC V_{BRAM}-Sensor Interrupt.** XADC V_{BRAM}-sensor alarm output interrupt occurs when V_{BRAM} exceeds user-defined threshold.
9	ALM[0] Deactive	0	R/TOW	**ALM[0] Deactive Interrupt.** This signal indicates that the falling edge of the Over Temperature signal is detected. It is cleared by writing a 1 to this bit position. The ALM[0] signal is generated locally from the core. This signal indicates that the XADC macro has deactivated the Over Temperature signal output.
8	OT Deactive	0	R/TOW(1)	**OT Deactive Interrupt.** This signal indicates that falling edge of the Over Temperature signal is detected. It is cleared by writing a 1 to this bit position. The OT Deactive signal is generated locally from the core. This signal indicates that the XADC macro has deactivated the Over Temperature signal output.
7	JTAG MODIFIED	0	R/TOW(1)(2)	**JTAGMODIFIED Interrupt.** This signal indicates that a write to DRP through the JTAG interface has occurred. It is cleared by writing a 1 to this bit position.
6	JTAG LOCKED	0	R/TOW(1)(2)	**JTAGLOCKED Interrupt.** This signal is used to indicate that a DRP port lock request has been made by the Joint Test Action Group (JTAG) interface.
5	EOC	N/A	R/TOW(1)(2)	**End of Conversion Signal Interrupt.** This signal transitions to an active-High at the end of an ADC conversion when the measurement is written to the XADC hard macro Status register.
4	EOS	N/A	R/TOW(1)(2)	**End of Sequence Interrupt.** This signal transitions to an active-High when the measurement data from the last channel in the auto sequence is written to the Status registers.
3	ALM[2]	0	R/TOW(1)(2)	**XADC V_{CCAUX}-Sensor Interrupt.** XADC V_{CCAUX}-sensor alarm output interrupt occurs when V_{CCAUX} exceeds the user-defined threshold.

Table 2-10: **IP Interrupt Status Register (IPISR) Description (C_BASEADDR + 0x60) *(Cont'd)***

Bits	Name	Reset Value	Access Type	Description
2	ALM[1]	0	R/TOW(1)(2)	**XADC V_{CCINT}-Sensor Interrupt.** XADC V_{CCINT}-sensor alarm output interrupt occurs when V_{CCINT} exceeds the user-defined threshold.
1	ALM[0]	0	R/TOW(1)(2)	**XADC Temperature-Sensor Interrupt.** XADC temperature-sensor alarm output interrupt occurs when device temperature exceeds the user-defined threshold.
0	OT	0	R/TOW(1)(2)	**Over-Temperature Alarm Interrupt.** Over-Temperature alarm output interrupt occurs when the die temperature exceeds a factory set limit of 125 °C.

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. This interrupt signal is directly generated from the XADC hard macro.

IP Interrupt Enable Register (IPIER)

The IPIER has an enable bit for each defined bit of the IPISR as shown in Figure 2-8 and described in Table 2-11. All bits are cleared upon reset.

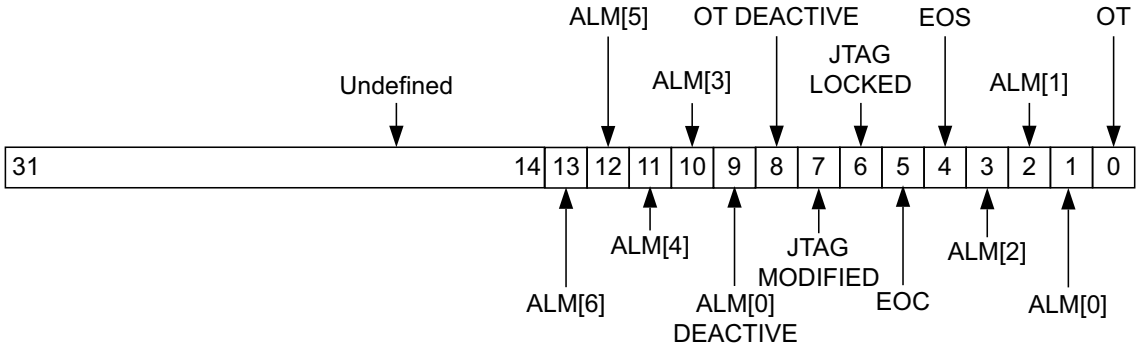

Figure 2-8: **IP Interrupt Enable Register (IPIER)**

Table 2-11: **IP Interrupt Enable Register (IPIER) Description (C_BASEADDR + 0x68)**

Bits	Name	Reset Value	Access Type	Description
31:14	Undefined	N/A	N/A	Undefined
13	ALM[6]	0	R/W	XADC $V_{CCPddro}$-Sensor Interrupt. 0 = Disabled 1 = Enabled
12	ALM[5]	0	R/W	XADC V_{CCPAUX}-Sensor Interrupt. 0 = Disabled 1 = Enabled

Table 2-11: **IP Interrupt Enable Register (IPIER) Description (C_BASEADDR + 0x68) *(Cont'd)***

Bits	Name	Reset Value	Access Type	Description
11	ALM[4]	0	R/W	XADC V_{CCPINT}-Sensor Interrupt. 0 = Disabled 1 = Enabled
10	ALM[3]	0	R/W	XADC V_{BRAM}-Sensor Interrupt 0 = Disabled 1 = Enabled
9	ALM[0] Deactive	0	R/W	ALM[0] Deactive Interrupt 0 = Disabled 1 = Enabled
8	OT Deactive	0	R/W	OT Deactive Interrupt 0 = Disabled 1 = Enabled
7	JTAG MODIFIED	0	R/W	JTAGMODIFIED Interrupt 0 = Disabled 1 = Enabled
6	JTAG LOCKED	0	R/W	JTAGLOCKED Interrupt 0 = Disabled 1 = Enabled
5	EOC	0	R/W	End of Conversion Signal Interrupt 0 = Disabled 1 = Enabled
4	EOS	0	R/W	End of Sequence Interrupt 0 = Disabled 1 = Enabled
3	ALM[2]	0	R/W	XADC V_{CCAUX}-Sensor Interrupt 0 = Disabled 1 = Enabled
2	ALM[1]	0	R/W	XADC V_{CCINT}-Sensor Interrupt 0 = Disabled 1 = Enabled
1	ALM[0]	0	R/W	XADC Temperature-Sensor Interrupt 0 = Disabled 1 = Enabled
0	OT	0	R/W	Over-Temperature Alarm Interrupt 0 = Disabled 1 = Enabled

More about Locally Generated Interrupt Bits in IPIER and IPISR

The interrupt bits ranging from Bit[16] to Bit[0] in IPISR as well as IPIER are direct output signals of the XADC hard macro. The signals like OT Deactive (Bit[8]), ALM[0] Deactive (Bit[9]), are locally generated in the core. These interrupts are generated on the falling edge of the Over Temperature and AML[0] signals. The falling edge of these signals can be used in controlling external things like controlling the fan or air-conditioning of the system.

Hard Macro Register (DRP Register) Grouping for AXI4-Lite Interface

The XADC hard macro register set consists of all the registers present in the XADC hard macro on 7 series FPGAs. The addresses of these registers are shown in Table 2-3. Because these registers are 16 bits wide but the processor data bus is 32 bits wide, the hard macro register data resides on the lower 16 bits of the 32-bit data bus. See Figure 2-9.

The 12-bit MSB aligned A/D converted value of different channels from the XADC hard macro are left-shifted and reside from bit position 15 to 6 of the processor data bus. The remaining bit positions from 5 to 0 should be ignored while considering the ADC data for different channels. Along with 16-bit data, the JTAGMODIFIED and JTAGLOCKED bits are passed that can be used by the software driver application for determining the validity of the DRP read data.

The JTAGMODIFIED bit is cleared when a DRP read/write operation through the FPGA logic is successful. If JTAGLOCKED = 1, a DRP read/write through the FPGA logic fails. The JTAGLOCKED signal is independently controlled through JTAG Test Access Port (TAP). These XADC hard macro registers should be accessed in their preferred access-mode only. The XADC Wizard IP core is not able to differentiate any non-preferred access to the XADC hard macro registers.

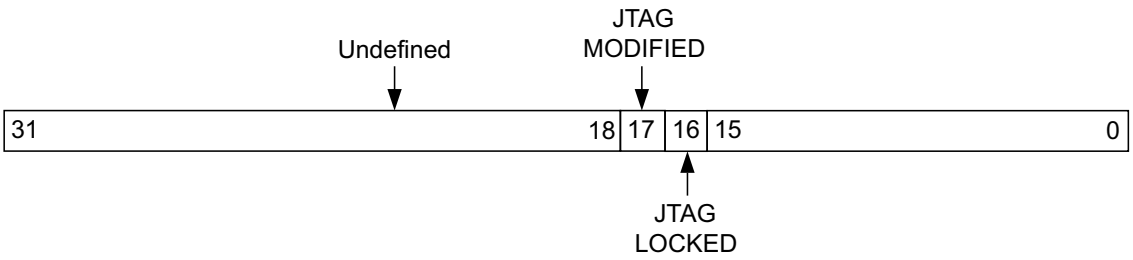

Figure 2-9: **XADC Hard Macro Register**

DRP registers are accessed as part of the core local registers.

IMPORTANT: *These registers must be accessed through the core local registers. Any attempt to access these registers in byte or halfword manner returns the error response from core.*

Designing with the Core

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

Clocking

The clock to XADC primitive is `DCLK`. When AXI4-Lite is selected as the Bus interface, `dclk` is connected to the `s_axi_aclk` clock. Hence the `adcclk` division factor must be programmed taking into consideration the `s_axi_aclk` frequency.

When DRP or None interface is selected, `dclk` clock is at the top-level of the IP and `adcclk` division factor must be programmed taking into consideration the `dclk` frequency.

When Streaming is enabled for DRP or None interface selection, `m_axis_aclk` is connected to `dclk`.

Resets

When AXI4-Lite is selected as the Bus interface, certain registers of the IP can be reset by writing a value 0xA to register 0x00. The AXI4-Lite and AXI4-Stream interfaces also have their own reset pins.

When DRP or None interface is selected, `reset_in` is the input port at the top-level of the IP.

Protocol Description

For more detailed information, see the AXI4-Lite protocol specifications. Figure 3-1 shows the simulation snapshots for Temperature value read from XADC register.

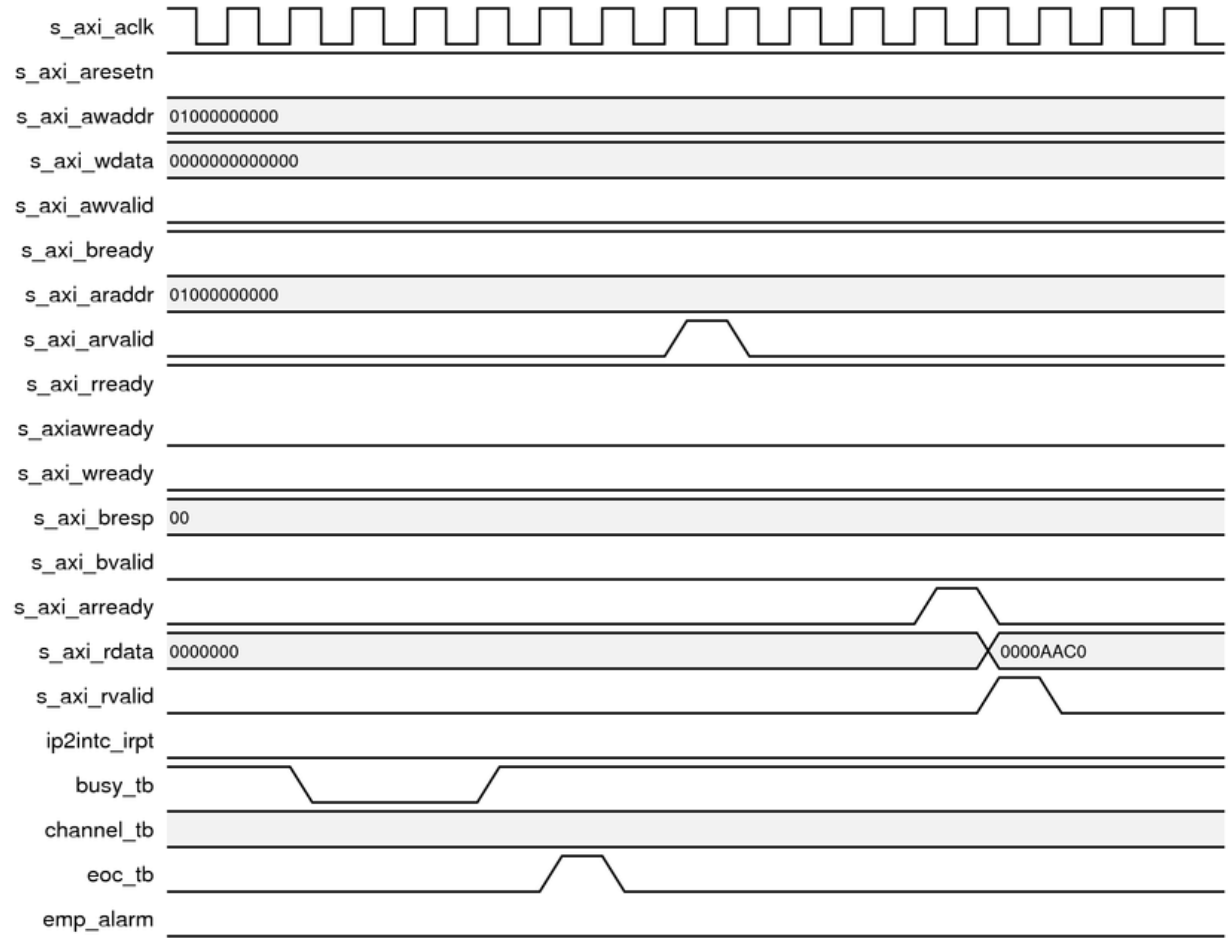

Figure 3-1: **AXI4-Lite Interface Reading Temperature Values in Simulation**

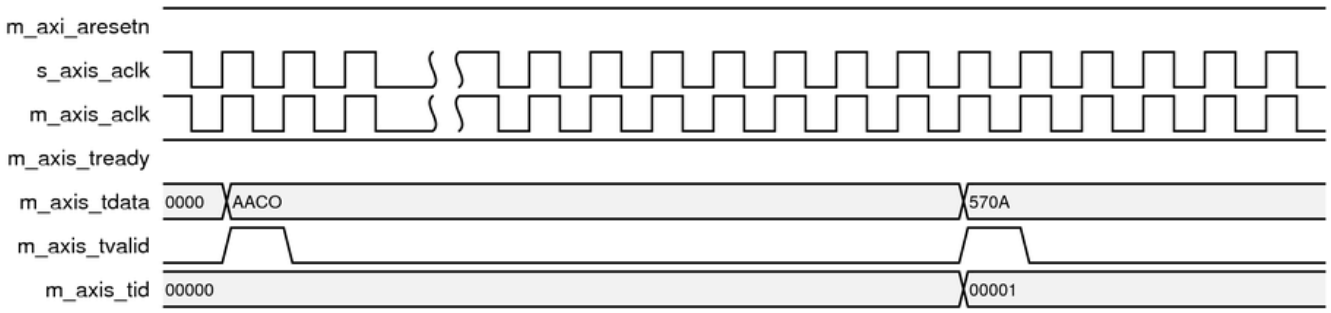

Figure 3-2: **Sequencer Mode Data on Streaming Interface**

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 Vivado 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 5]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 1]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 4]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 6]

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 5] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values 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. Select the IP from the IP catalog.
2. Double-click the selected IP or select the Customize IP command from the toolbar or right-click menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 1] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 4].

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

This section describes how to set up a project in the Vivado Design Suite flow. Before generating the example design, set up the project as described in Creating a Directory and Setting the Project Options of this guide.

Creating a Directory

To set up the example project, first create a directory using the following steps:

1. Change directory to the desired location. This example uses the following location and directory name:

   ```
   /Projects/xadc_example
   ```

2. Start Vivado Design Suite software.
3. Choose **File > New Project** (Figure 4-1).
4. Change the name of the .xpr file (optional).
5. Create project with default settings.

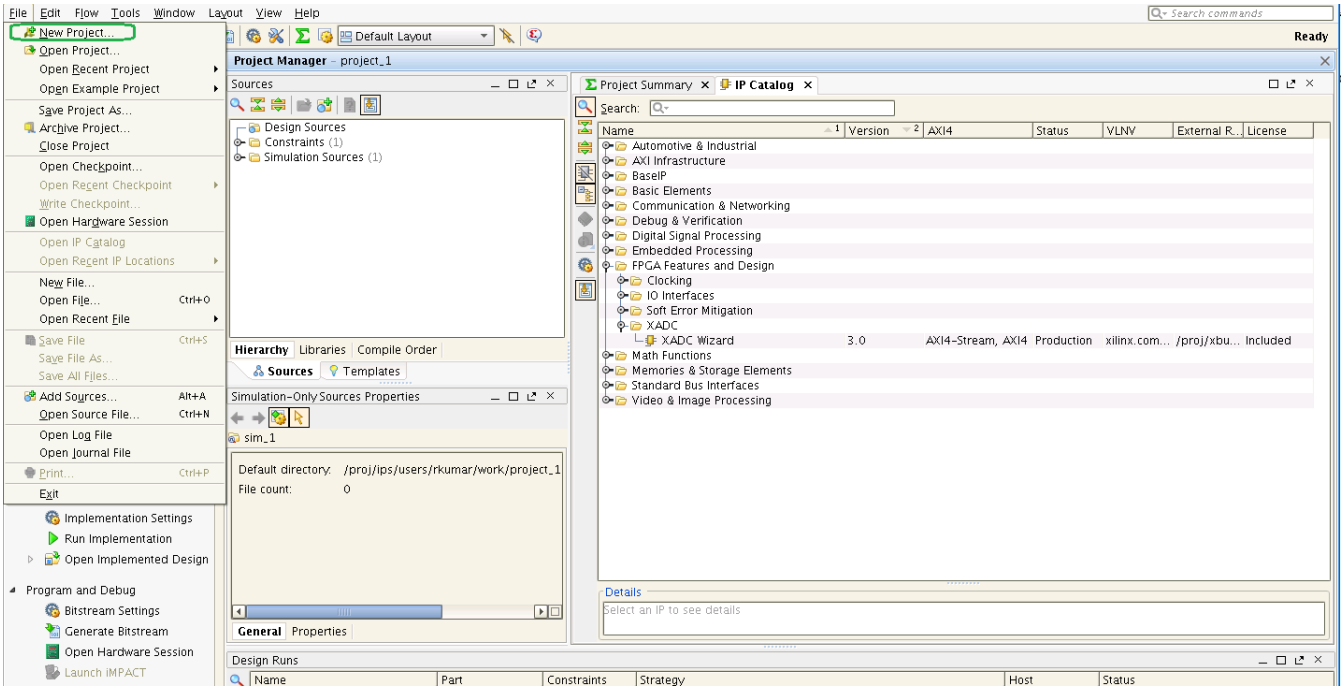

Figure 4-1: **New Project**

Setting the Project Options

Set the project options using the following steps:

1. Click **Project Part** in the option tree.
2. Select a **7 series FPGA** from the Family list.

3. Select a device from the Device list that support XADC primitive.
4. Select an appropriate package from the Package list. This example uses the XC7K235T device (see Figure 4-2).

   Note: If an unsupported silicon family is selected, the XADC Wizard remains light gray in the taxonomy tree and cannot be customized. Only devices containing the XADC are supported by the Wizard. See the *7 Series FPGAs Overview* (DS180) [Ref 7] for a list of devices containing XADC.
5. Select either Verilog or VHDL as the target language.
6. Click **OK**.

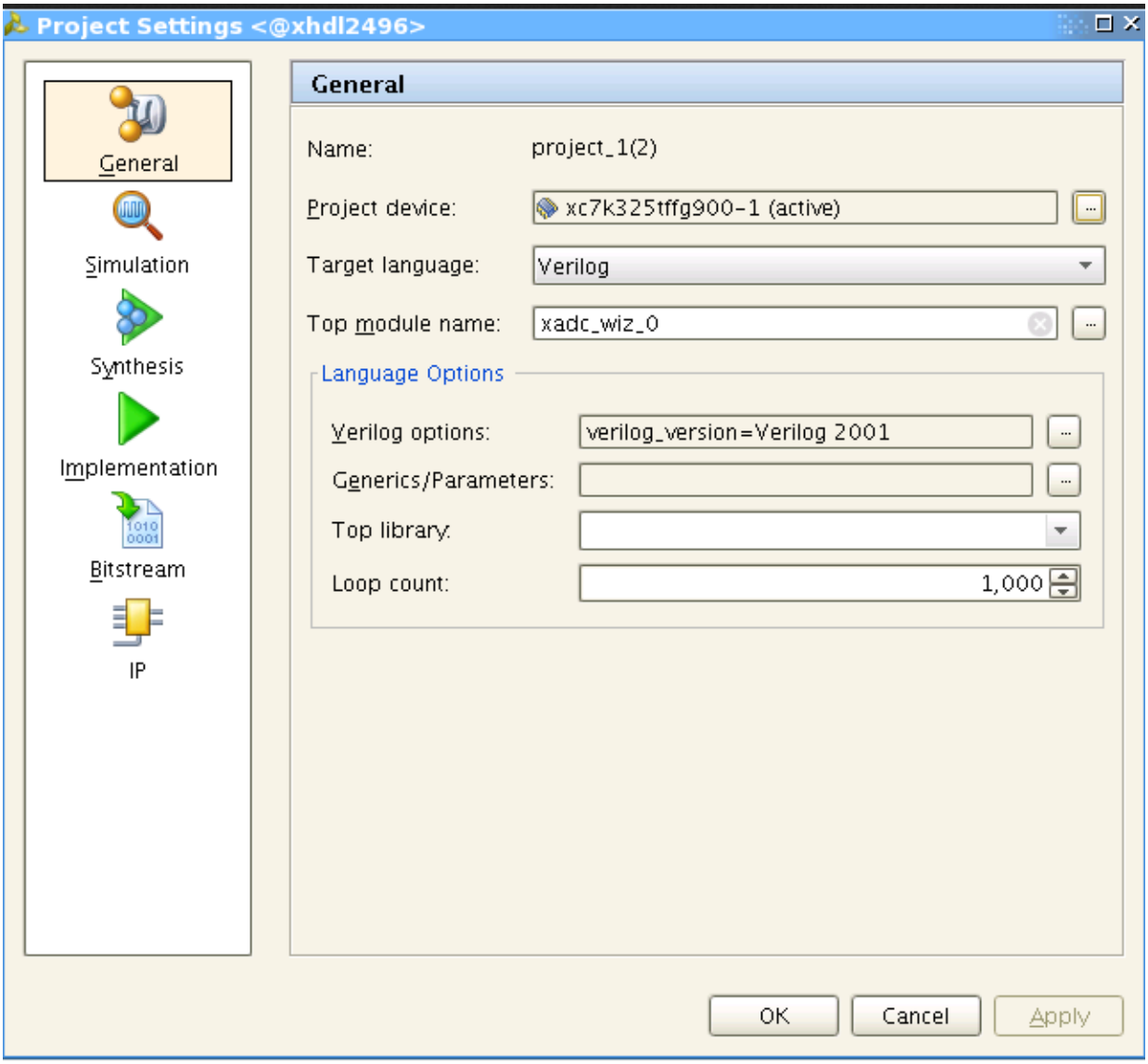

Figure 4-2: **Target Architecture Setting**

Generating the Core

This section provides instructions for generating an example XADC design using the default values. The wrapper and its supporting files, including the example design, are generated in the project directory. For additional details about the example design files and directories provided with the XADC Wizard, see Example Design.

1. Select the **IP Catalog** under the **Project Manager** tab to get the IP taxonomy view.
2. Locate the XADC Wizard in the taxonomy tree under:

   `/FPGA Features and Design/XADC`. (see Figure 4-1)

3. Double-click **XADC Wizard** to launch the Wizard.

After the wizard is launched, the IP catalog displays a series of screens that allow you to configure the XADC Wizard.

XADC Setup

The XADC Wizard screen (Figure 4-3) allows you to select the component name, interface type, startup channel mode, timing mode, analog stimulus file name, DRP timing options, and control and status ports.

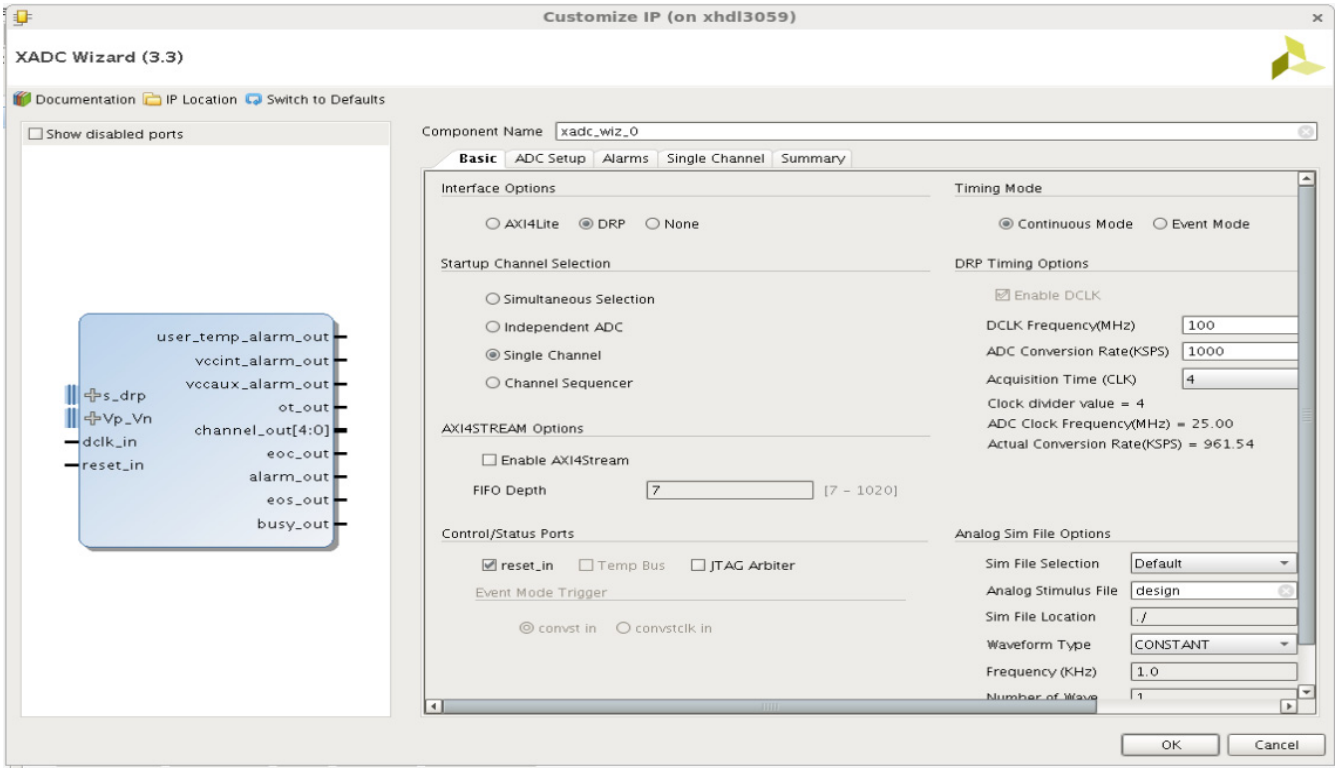

Figure 4-3: **XADC Basic Setup Tab**

- **Component Name** – User selectable component name is available. Component names must not contain any reserved words in Verilog or VHDL.

Basic Tab

The following describes the Basic options in the XADC Wizard core.

- **Interface Options** – Use this field to select the interface for the XADC Wizard. DRP is the default option. you can select **AXI4Lite, DRP**, or **None** option. DRP port is the FPGA logic interface for XADC. It facilitates access to the register file interface of the XADC. The XADC control registers can be read or written using this port. This port can only be enabled when `DCLK` clock is present.
- **Startup Channel Selection** – XADC can be configured in one of the four modes listed:
  - **Simultaneous Selection** – This mode allows you to monitor two external channels simultaneously. For more information about this mode see the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].
  - **Independent ADC** – This mode allows you to run the XADC in independent mode. Here, the XADC independently monitors the externals channels and at the same time monitors the FPGA voltages and temperature.
  - **Single Channel** – In this mode, you can select only one channel to monitor. If Calibration channel is selected, the example test bench does not support analog stimulus and the data comparison verifies this selection.
  - **Channel Sequencer** – Choosing this mode, allows you to select any number of channels to monitor. The channels to be used for this mode can be selected on Figure 4-9, page 48.
- **AXI4-Stream Options** – Use this field to enable or disable the AXI4-Stream interface.
  - **Enable AXI4-Stream** – You can enable or disable the AXI4-Stream interface. This is disabled by default.
  - **FIFO Depth** – FIFO Depth can be configured from 7 to 1,020. Here FIFO Depth refers to the maximum depth of the data which can be written into FIFO before ALMOSTFULL becomes High.
- **Timing Mode** – XADC can operate in two timing modes:
  - **Continuous Mode** – In this mode, the XADC continues to sample and convert the selected channel/channels.
  - **Event Mode** – This mode requires an external trigger event, `CONVST` or `CONVSTCLK`, to start a conversion on the selected channel. Event Mode should only be used with external channels.

- **DRP Timing Options** – The XADC clock (ADCCLK) is derived from the dynamic reconfiguration port (DRP) clock DCLK. The XADC supports a DRP clock frequency of up to 250 MHz. The XADC can also operate in absence of DCLK. For more information on the DRP see the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].

  The ADCCLK clock, should be in the range of 4–26 MHz. To support this lower frequency clock the XADC has an internal clock divider. The Vivado IDE allows an external DCLK frequency and required ADC conversion rate (maximum 1 Msps) to be specified. Based on the value of DCLK clock, the wizard then calculates the appropriate clock divider value based on the values of DCLK clock and ADC conversion.

  The wizard also displays the ADC Clock frequency value and the actual conversion rate of the ADC.

- **Analog Sim Options** – You can provide the relative or absolute path and update name of the Analog Stimulus File in this section.
  - **Sim File Selection** – By changing the default option to Relative path .txt in the drop-down option, the analog stimulus file path can be specified in the Sim File Location box.

    Default name and path for the analog stimulus is design.txt generated in the core simulation area.

    For converting the .csv file to the .txt file format, provide **Sim File Selection** as Relative path .csv, **Analog Stimulus File** name, and the file location. Here .csv to .txt conversion is performed automatically when the example design is opened for this IP. See Figure 4-4.

Provide stimulus in .xls and save the file in .csv format

	A	B	C	D	E	F	G	H	I	J	K	L
1	TIME	TEMP	VCCINT	VCCBRAM	VCCAUX	VP	VN	VAUXP[0]	VAUXN[0]	VAUXP[1]	VAUXN[1]	VAUXP
2	0	63	0.91	0.91	1.8	0.5	0	0.5	0	0.5	0	
3	6250	88	0.97	0.97	1.94	0.3	0	0.2	0	0.2	0	
4	11450	53	0.81	0.81	1.7	0.9	0	0.9	0	0.9	0	
5	22900	63	0.91	0.91	1.8	0.5	0	0.5	0	0.5	0	

.csv file example is here

```
TIME,TEMP,VCCINT,VCCBRAM,VCCAUX,VP,VN,VAUXP[0],VAUXN[0],VAUXP[1],VAUXN[1],VAUXP
0,63,0.91,0.91,1.8,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,
6250,88,0.97,0.97,1.94,0.3,0,0.2,0,0.2,0,0.2,0,0.2,0,0.2,0,0.2,0,0.2,0,0.2,0,0.
11450,53,0.81,0.81,1.7,0.9,0,0.9,0,0.9,0,0.9,0,0.9,0,0.9,0,0.9,0,0.9,0,0.9,0,0.
22900,63,0.91,0.91,1.8,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.5,0,0.
```

Wizard converts .csv to .txt file in required format

```
TIME TEMP VCCINT VCCBRAM VCCAUX VP VN VAUXP[0] VAUXN[0] VAUXP[1] VAUXN[1] VAUXP
0 63 0.91 0.91 1.8 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0
6250 88 0.97 0.97 1.94 0.3 0 0.2 0 0.2 0 0.2 0 0.2 0 0.2 0 0.2 0 0.2 0 0.2 0 0.
11450 53 0.81 0.81 1.7 0.9 0 0.9 0 0.9 0 0.9 0 0.9 0 0.9 0 0.9 0 0.9 0 0.9 0 0.
22900 63 0.91 0.91 1.8 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.5 0 0.
```

Figure 4-4: **Converting .csv to .txt File Format Example**

If the example project is not required, then run the `.csv` to `.txt` Tcl console script generated after the output products of the IP are generated. `<Component name>_csv_to_txt.tcl` conversion script is created in the Tcl console folder located in the IP path. This script should be sourced in the Vivado Tcl Console for conversion.

- **Analog Stimulus File** – Use this field to customize the name of the XADC Analog Stimulus File.
- **Sim File Location** – Enabled when Sim File Selection is not the default. Relative or absolute path of the analog stimulus can be provided in this box.
  Relative path is with respect to the simulation directory. If the example design behavioral simulation is run, the relative path is with respect to `project_1/xadc_wiz_0_example/xadc_wiz_0_example.sim/sim_1/behav` directory.
- **Waveform Type** – Choose CONSTANT, SINE, TRIANGLE, or SQUARE wave as stimulus on external channel analog inputs. See Figure 4-5.
- **Frequency** – Analog waveform frequency can be configured from 0.1 KHz to half of the ADC sampling rate.
- **Number of Waves** – 1 to 1,000 waves of the selected waveform type can be generated in the analog stimulus file.

Note:

1. `Sim_File_Rel_Path` parameter is relative to the directory where Vivado example design is evoked.
2. `Sim_File_Name` parameter must not include any file extensions (such as .txt or csv).
3. When you use the files in simulation, select the option of .txt for the txt file extension and .csv for csv file extension.

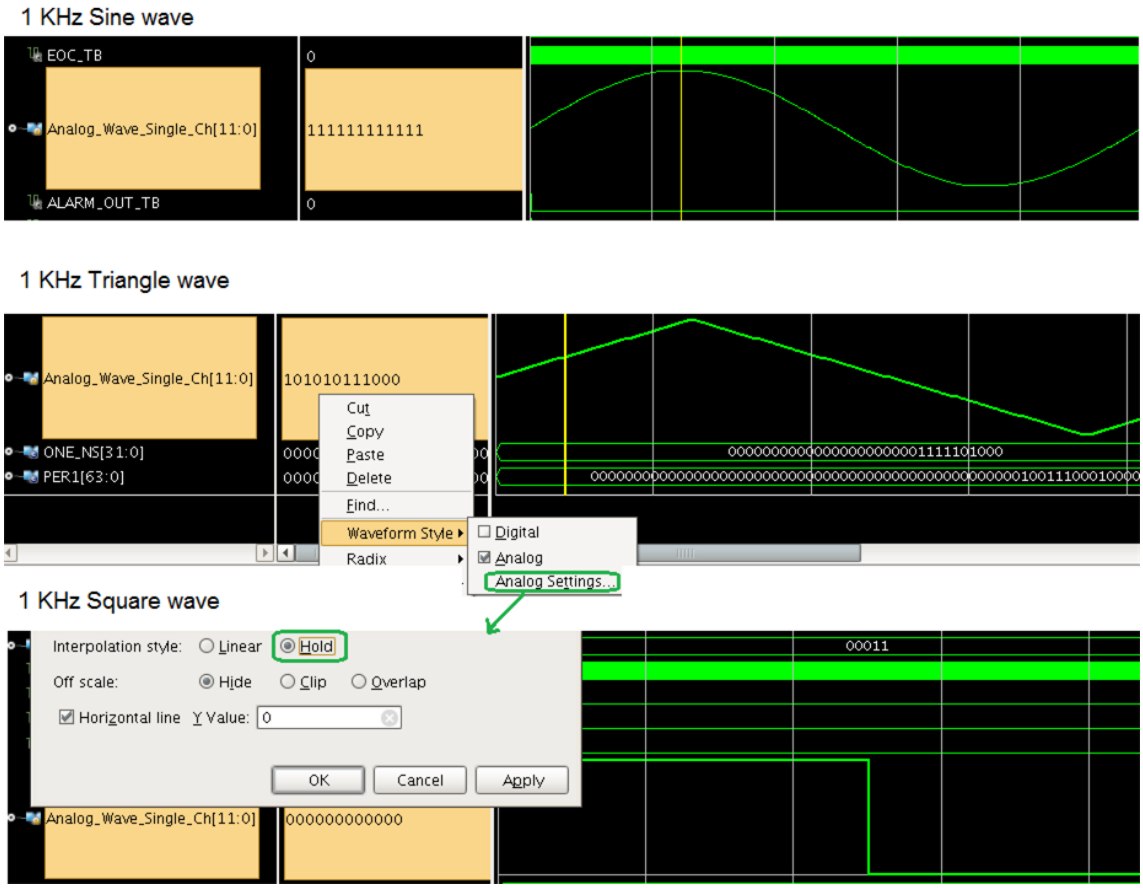

Figure 4-5: **Waveform Type**

Control/Status Ports

The Control/Status Port selection (Figure 4-3) allows you to select the I/O ports on the XADC primitive.

- **Control Ports** – This section allows you to select control input ports:
  - `reset_in` allows an external input reset signal to be connected to the XADC
  - `convst_in` or `convstclk_in` as trigger sources for Event Mode Timing
- **Temp Bus** – This option enables a special bus which updates the temperature for every given interval of time. There is only one XADC primitive available in a 7 series FPGA and

Zynq®-7000 AP SoC. If the XADC Wizard core is used in a system which uses MIG, the TEMP_OUT bus should be connected to the `xadc_device_temp_i` input port of the DDR3_SDRAM (MIG) block. This disables inference of the XADC hard block in DDR3_SDRAM. Enabling this provides the 12-bit TEMP_OUT port with the temperature update logic. This check box is available when the interface option is AXI4-Lite or AXI4-Stream is enabled. Selecting this option enables the temperature channel by default. Temperature channel is read periodically after the wait cycle count is over. This read value is loaded into TEMP_OUT port.

- **JTAG Arbiter** – This option enables the display of JTAG status ports (JTAGMODIFIED, JTAGLOCED, JTAGBUSY). These signals act as a reference to verify if primitive is accessed by JTAG.
- **Status Outputs** – Output status signals are also provided to facilitate interfacing of the XADC to a user design.

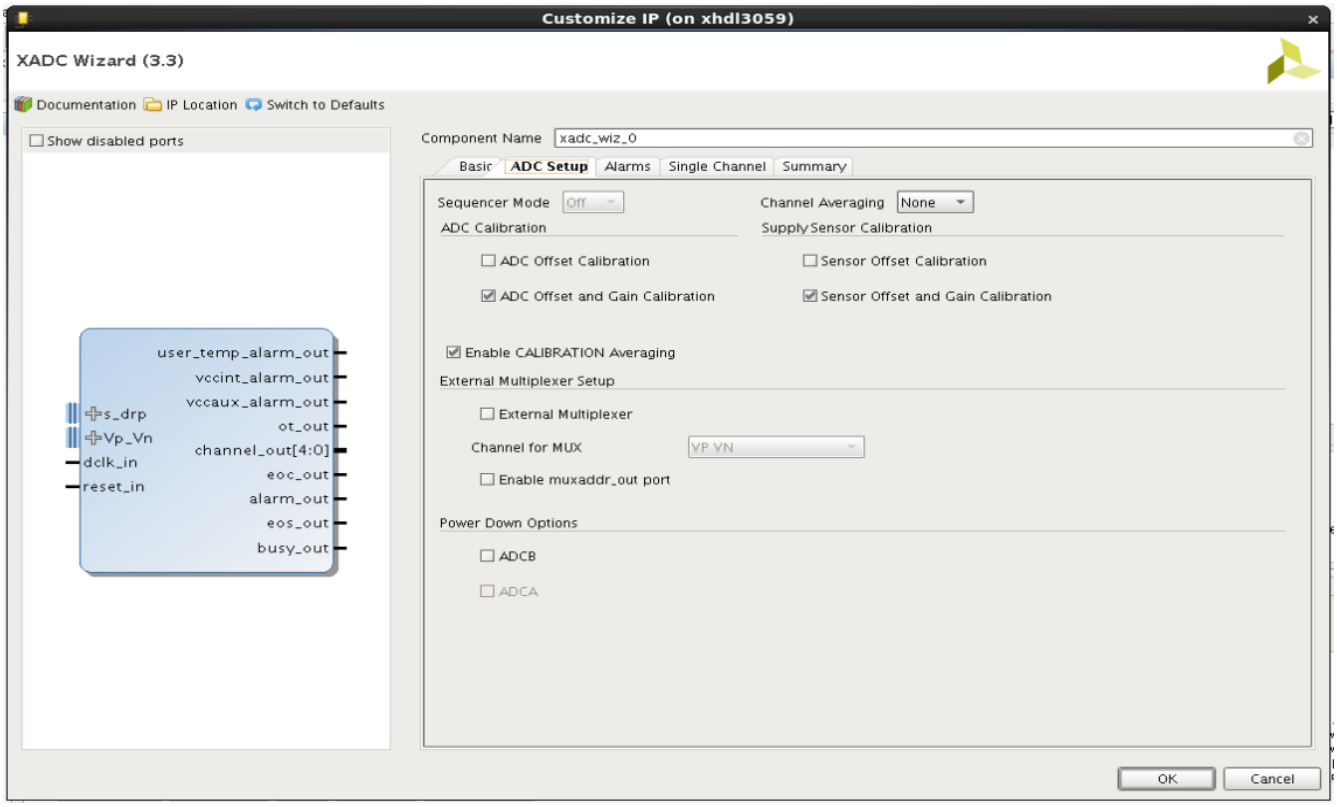

Figure 4-6: **ADC Setup Tab**

ADC Setup

If the XADC is configured for Channel Sequencer, Simultaneous Sampling or Independent ADC mode, you can choose the required sequencer mode. The available options are Continuous, One-pass or Default mode.

The Channel Averaging drop-down menu allows you to select the required averaging value. The available options are None, 16, 64, and 256.

You can select the type of ADC Calibration and/or Supply Sensor Calibration by checking the respective check boxes. Calibration Averaging is enabled by default in XADC. You can disable this by deselecting the box.

- **External Multiplexer Setup** – XADC supports a new timing mode that allows you to use an external analog multiplexer in situations where FPGA I/O resources might be limited or auxiliary analog I/O are more valuable when used to implement another interface.

  You can opt to use this feature by checking the box against Use External Multiplexer. If checked, it is necessary to specify the external channel to which the Multiplexer (MUX) connects. Select this channel using the drop-down menu.
  Enable `muxaddr_out` port option is provided for enabling the `muxaddr_outfor` external MUX mode using dynamic reconfiguration.

- **Power Down Options** – Analog-to-digital converter (ADC) controls (ADCB and ADCA) can be powered down when not in use. You can power down ADCB and use only ADCA. ADCA can be powered down only if ADCB is already powered down. This option is available to conserve power and the ADC does not generate any control and status signal when powered down. Wizard example simulation does not display any data when ADCs are powered down.

Alarm Setup

The Alarms (Figure 4-7) allows the alarm outputs to be enabled for the on-chip sensors. If a measurement of an on-chip sensor lies outside the specified limits, then a logic output goes active if enabled. For a detailed description of the alarm functionality see the *7 Series FPGAs XADC User Guide* (UG480) [Ref 3].

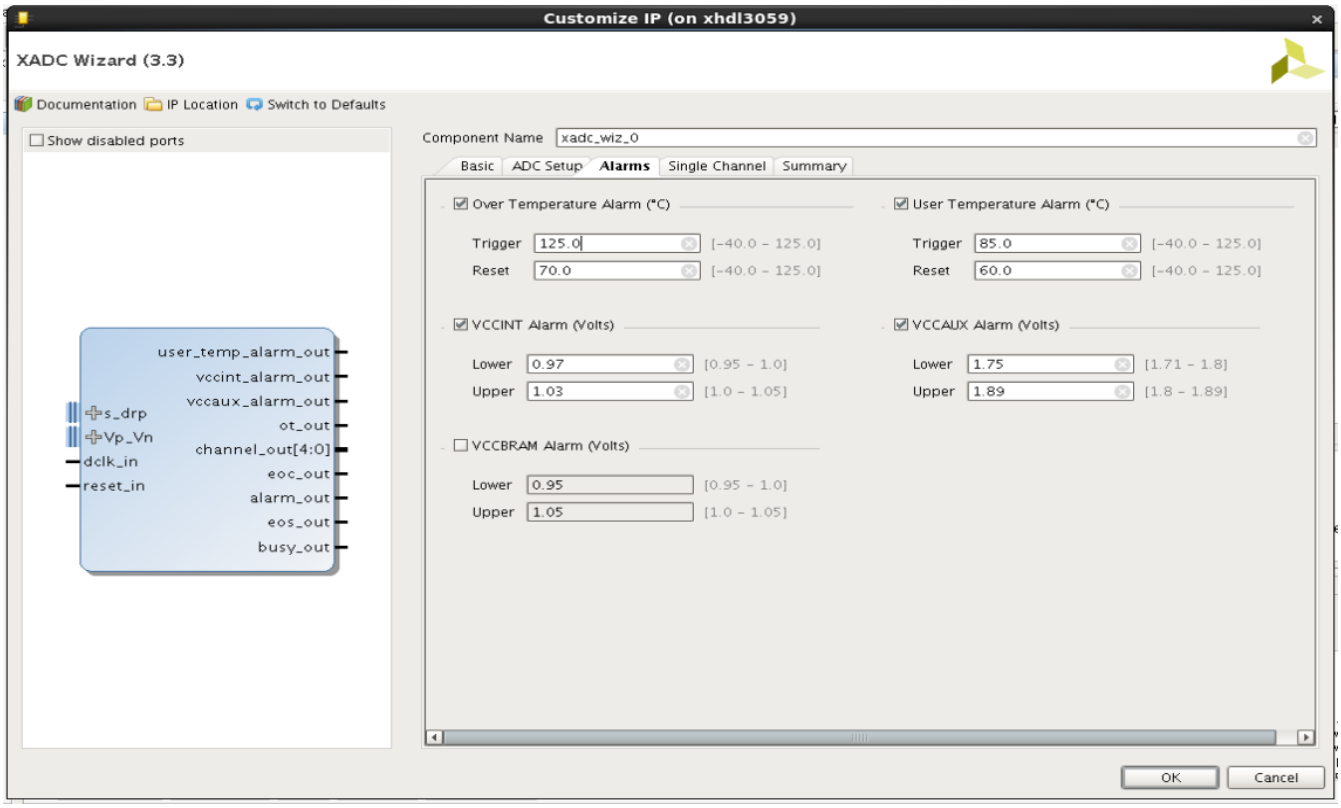

Figure 4-7: **Alarms Tab for 7 Series Devices**

- **Enable Alarms** – Use the check boxes to enable alarm logic outputs. The seven options are:
  - Over temperature alarm
  - User temperature alarm
  - V_{CCINT} alarm
  - V_{CCAUX} alarm
  - V_{BRAM} alarm
  - Additional alarms for Zynq-7000 devices
    - V_{CCPAUX} alarm
    - V_{CCDDRO} alarm
- **Temperature Alarm Limits** – Trigger and Reset levels for temperature alarm output can be entered using these fields. You can set both; the trigger as well as reset levels for the OT alarm.
- **V_{CCINT}, V_{CCAUX}, and V_{BRAM} Limits** – Both upper and lower alarm thresholds can be specified for the on-chip power supplies. If the measured value moves outside these limits the alarm logic output goes active. The alarm output is reset when a

measurement inside these limits is generated. The default limits in the Vivado IDE represent ±5% on the nominal supply value.

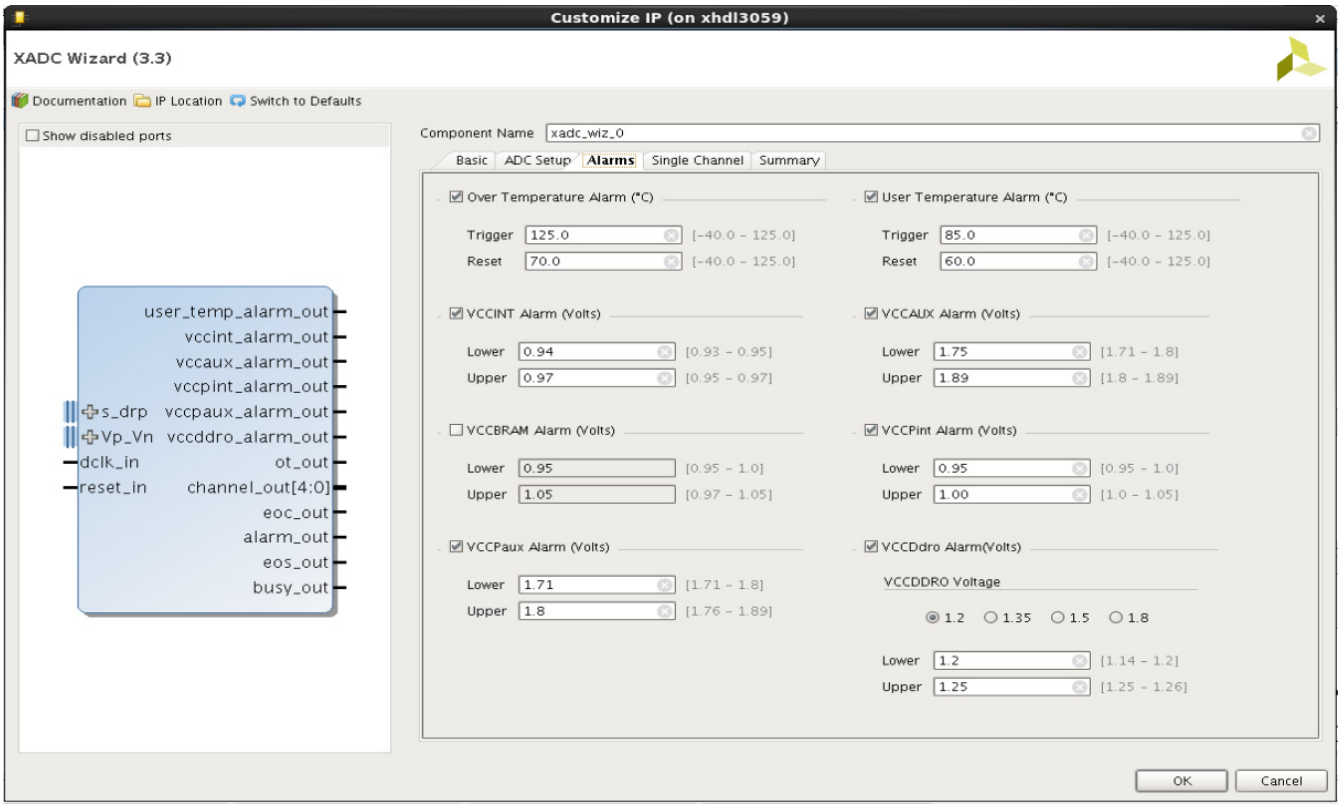

Figure 4-8: **Alarms Tab for Zynq-7000 AP SoC**

Channel Sequencer Setup

Channel Sequencer (Figure 4-9) is used to configure the XADC sequence registers when the XADC is configured in Channel Sequencer, Simultaneous sampling, or Independent ADC mode. All the possible channels that can be included in the sequence are listed in the Channel Sequencer table of the Wizard shown in Figure 4-9.

- Use the Channel Sequencer Setup screen to select Channels for monitoring, enable Averaging for selected channels, enable Bipolar mode for external channels and increase the Acquisition time for the selected channels.
- In the case of Simultaneous sampling mode, selecting channel Vauxp[0]/Vauxn[0] would automatically select channel Vauxp[8]/Vauxn[8]. Similarly selecting channel Vauxp[1]/Vauxn[1] would select channel Vauxp[9]/Vauxn[9] etc.
- In case of Independent ADC mode, only external channels are listed and can be user-selected.

For more information about the simultaneous sampling mode and Independent ADC mode, see the *7 Series FPGAs XADC User Guide* [Ref 3].

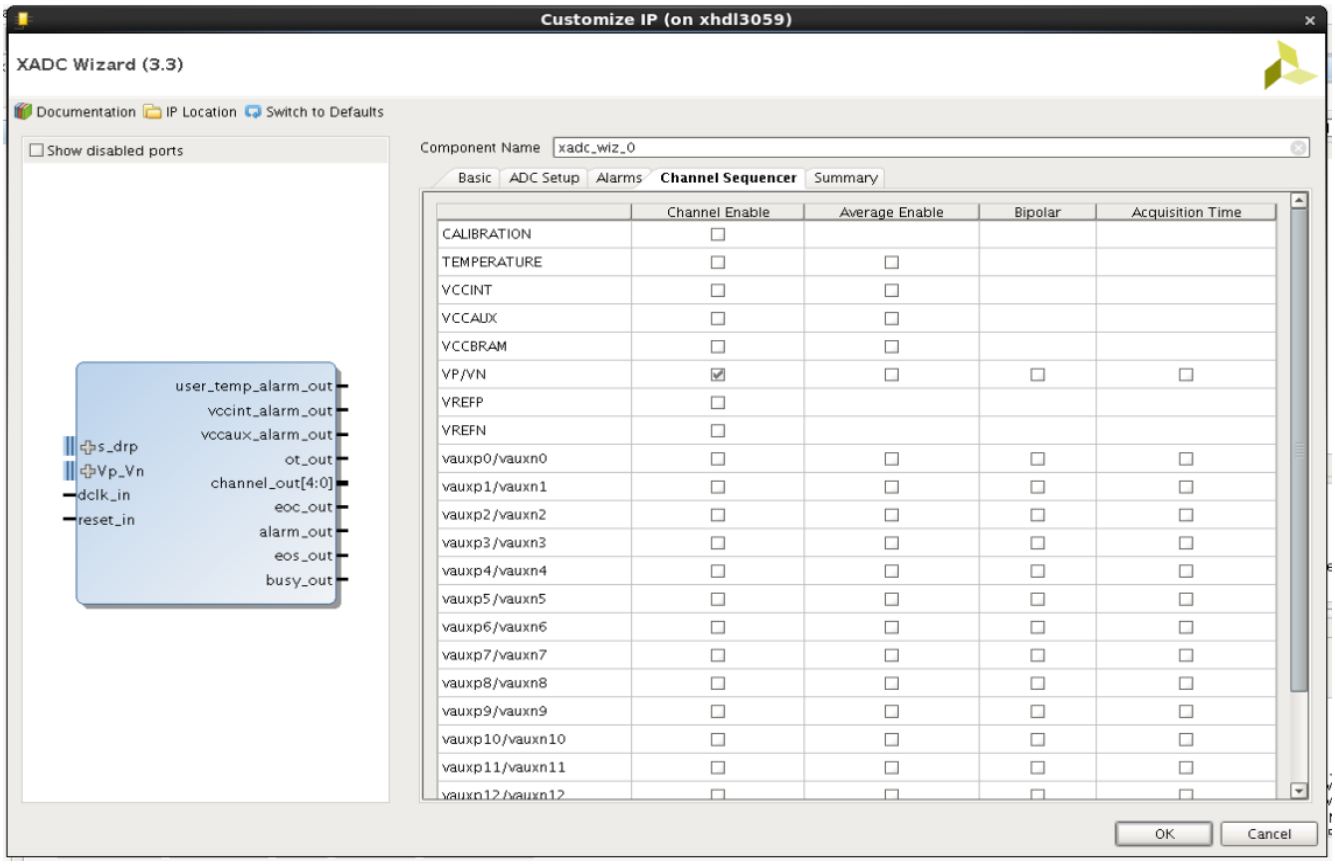

Figure 4-9: **Channel Sequencer Tab**

Single Channel Mode

If the Single Channel Mode operation (see Startup Channel Selection in Basic Tab) is selected, the Single Channel Setup is displayed on Figure 4-10. The Single Channel allows you to select the channel for measurement and the analog input mode if the channel is an external analog input (that is, unipolar or bipolar).

The columns Channel Enable, Average Enable, and Increase Acquisition Time are disabled and are shown only information and ease.

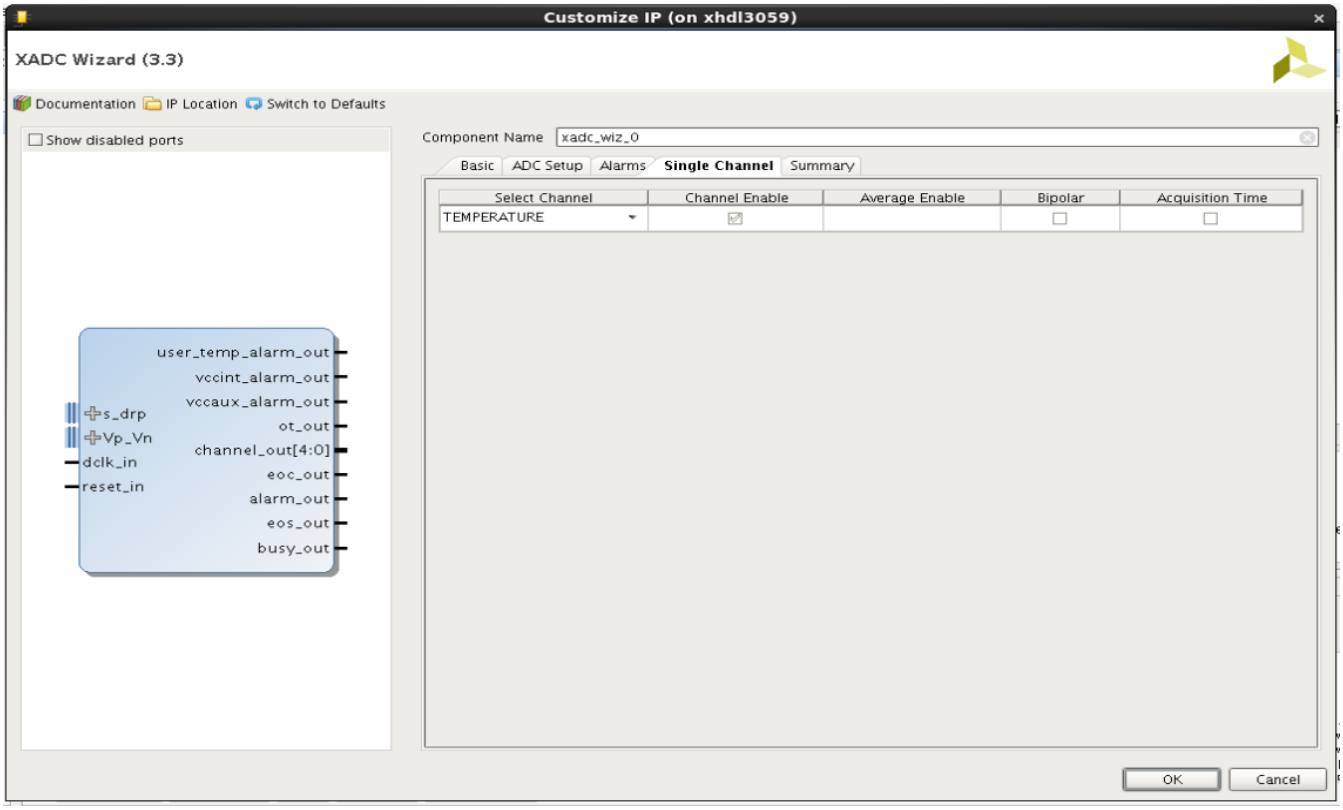

Figure 4-10: **Single Channel Tab**

Summary

The summary tab lists all the key parameters, such as the interface selected, XADC operating mode, Timing Mode, etc.

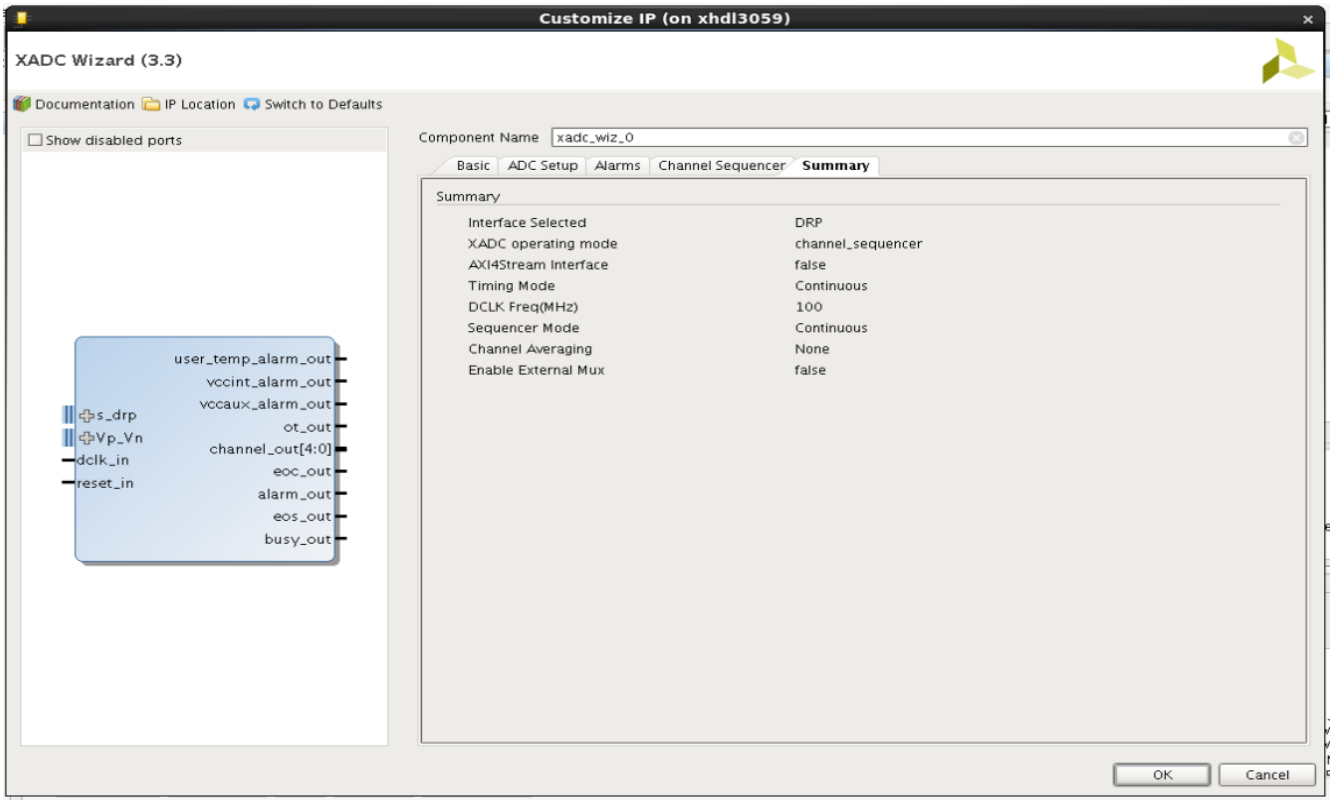

Figure 4-11: **Summary Tab**

In IP integrator, the default interface for XADC is AXI4-Lite because most of the systems are based on the AXI4 interface (Figure 4-12).

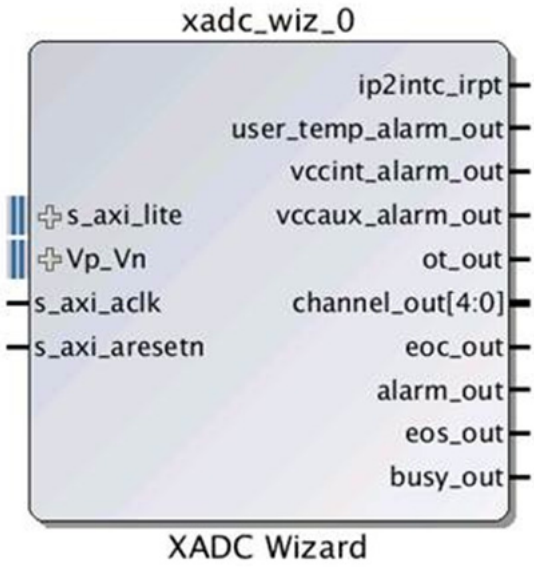

Figure 4-12: **XADC Wizard IP Symbol**

Generating the HDL Wrapper

After selecting the configuration options, click **OK** on the Wizard screen to generate the HDL wrapper and other Wizard outputs.

The output files are placed in the `<project_name>/<project_name>.srcs/sources_1/ip/<component_name>/` directory you selected or created when setting up a new Vivado project.

Output Generation

For details, see "Generating IP Output Products" in the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 1].

User Parameters

Table 4-1 shows the relationship between the fields in the Vivado IDE and the User Parameters (which can be viewed in the Tcl Console).

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

Vivado IDE Parameter/Value(1)	User Parameter/Value(1)	Default Value(1)
Interface Options	INTERFACE_SELECTION	DRP
Startup Channel Selection	STARTUP_CHANNEL_SELECTION	Single Channel
Enable AXI4-Stream	C_HAS_AXI4STREAM	FALSE

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship *(Cont'd)***

Vivado IDE Parameter/Value[1]	User Parameter/Value[1]	Default Value[1]
FIFO Depth	C_FIFO_DEPTH	7
Reset In	RESET_IN	TRUE
Temperature Bus	ENABLE_TEMP_BUS	FALSE
JTAG Arbiter	ENABLE_JTAG_ARBITER	FALSE
Convst In	C_HAS_CONVST	Not Available
Convstclk In	C_HAS_CONVSTCLK	Not Available
Timing Mode	C_CONFIGURATION_R0	Continuous Mode
Enable DCLK	C_HAS_DCLK	Not Available
DCLK Frequency	C_DCLK_FREQUENCY	100
ADC Conversion Rate	ADC_CONVERSION_RATE	1000
Acquisition Time	SINGLE_CHANNEL_ACQUISITION_TIME	4
Clock Divider Value	MIN_CLK_DIV_FACTOR	Not Available
ADC Clock Frequency	ADC_CLK_FREQ_VALUE	25
Actual Conversion Rate	ACTUAL_CONVERSION_RATE_VALUE	961.54
Sim File	SIM_FILE_SEL	Default
Analog Stimulus File	SIM_FILE_NAME	./
Sim File Location	SIM_FILE_REL_PATH	./
Sequencer Mode	SEQUENCER_MODE	Not Available
ADC Offset Calibration	ADC_OFFSET_CALIBRATION	FALSE
ADC Offset and Gain Calibration	ADC_OFFSET_AND_GAIN_CALIBRATION	FALSE
Channel Averaging	CHANNEL_AVERAGING	None
Sensor Offset Calibration	SENSOR_OFFSET_CALIBRATION	FALSE
Sensor Offset and Gain Calibration	SENSOR_OFFSET_AND_GAIN_CALIBRATION	FALSE
Enable Calibration Averaging	ENABLE_CALIBRATION_AVERAGING	TRUE
External Multiplexer	ENABLE_EXTERNAL_MUX	FALSE
Channel for Multiplexer	EXTERNAL_MUX_CHANNEL	VP, VN
ADCB	POWER_DOWN_ADCB	FALSE
ADCA	POWER_DOWN_ADCA	FALSE
Over Temperature Alarm	C_HAS_OT_ALARM	TRUE
Over Temperature Alarm Trigger	TEMPERATURE_ALARM_OT_TRIGGER	125
Over Temperature Alarm Reset	TEMPERATURE_ALARM_OT_RESET	70
VCCINT Alarm	C_HAS_VCCINT_ALARM	TRUE
VCCINT Alarm Lower	VCCINT_ALARM_LOWER	0.97
VCCINT Alarm Upper	VCCINT_ALARM_UPPER	1.03
VCCBRAM Alarm	C_HAS_VBRAM_ALARM	

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship *(Cont'd)***

Vivado IDE Parameter/Value[1]	User Parameter/Value[1]	Default Value[1]
VCCBRAM Lower	VCCBRAM_ALARM_LOWER	0.95
VCCBRAM Upper	VCCBRAM_ALARM_UPPER	1.05
User Temperature Alarm	C_HAS_USER_TEMP_ALARM	TRUE
User Temperature Alarm Trigger	TEMPERATURE_ALARM_TRIGGER	85
User Temperature Alarm Reset	TEMPERATURE_ALARM_RESET	60
VCCAUX Alarm	C_HAS_VCCAUX_ALARM	TRUE
VCCAUX Alarm Lower	VCCAUX_ALARM_LOWER	1.75
VCCAUX Alarm Upper	VCCAUX_ALARM_UPPER	1.89
Select Channel	MUX_SELECTED_CHANNEL_LIST	VP, VN
Channel Sequencer	CHANNEL_SEQUENCER	FALSE

1. Parameter values are listed in the table where the Vivado IDE parameter value differs from the user parameter value. Such values are shown in this table as indented below the associated parameter.

Constraining the Core

This section contains information about constraining the core in the Vivado Design Suite.

Required Constraints

The core generates IP level constraints required for the selected configuration. The following constraints need to be added at system/top-level design:

- For AXI4-Lite interface, `create_clock` -period <clock period in ns> [`get_ports s_axi_aclk`].
- For DRP, `create_clock` -period <clock period in ns> [`get_ports dclk_in`].

Device, Package, and Speed Grade Selections

The XADC Wizard core supports all parts and packages.

Clock Frequencies

The clock frequencies supports from 8 to 250 MHz.

Clock Management

Depending on configuration, ADC clock is internally divided by the XADC primitive to achieve the desired sampling rate.

Clock Placement

This section is not applicable for this IP core.

Banking

This section is not applicable for this IP core.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

This section is not applicable for this IP core.

Simulation

This section contains information about simulating IP in the Vivado Design Suite. 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 6].

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

Analog waveform simulation is performed using the `design.txt` file which contains the time reference and the analog values for a selected channel. This file is generated by default. The analog and its digital equivalence comparison is in the example design test bench to verify the XADC behavior.

You can provide your own waveform in a file using the Relative path option in the Vivado IDE. In this case, the comparison values should be updated with respect to the analog stimulus to complete the example design simulation without error.

Simulation of Channel Averaging is not supported in the XADC Wizard example design test bench.

Synthesis and Implementation

This section contains information about synthesis and implementation in the Vivado Design Suite. For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 1].

VP/VN and 16 VAUXP/VAUXN pin pairs need LOC constraints to be specified in XDC.

Example Design

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

Directory and File Contents

For more information on file and directory structure, see Output Generation.

Top-Level Example Design

The following file describes the top-level example design for the XADC Wizard core.

Verilog

```
<project_name>/<project_name>.srcs/sources_1/ip/<component_name>/
example_design/<component_name>_exdes.v
```

The example design, instantiates the XADC core that is generated by the wizard.

Open Example Project Flow

In the Vivado tools, the command

```
open_example_project [get_ips <component_name>]
```

in the Tcl Console invokes a separate example design project where it creates `<component_name>_exdes` as the top module for synthesis and `<component_name>_tb` as the top module for simulation. Implementation or simulation of the example design can be run from the example project.

Test Bench

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

Demonstration Test Bench

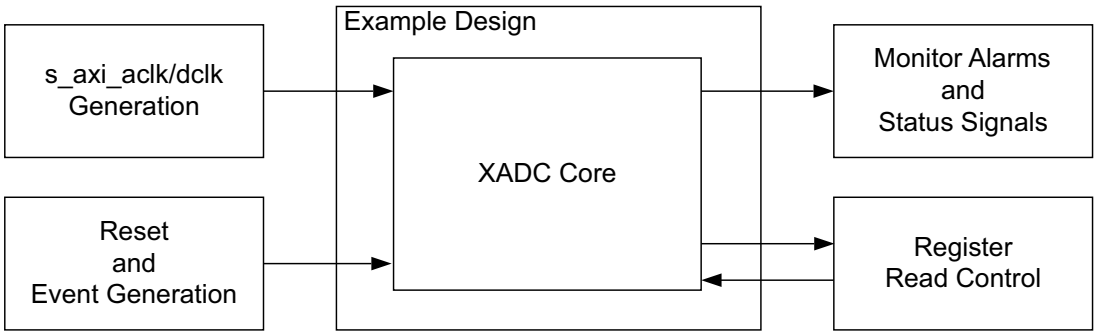

Figure 6-1: **Demonstration Test Bench for the XADC Wizard and Example Design**

The following file describe the demonstration test bench.

Verilog

```
<project_name>/<project_name>.srcs/sources_1/ip/<component_name>/
simulation/<component_name>_tb.v
```

The demonstration test bench is a simple Verilog program to exercise the example design and the core.

The demonstration test bench performs the following tasks:

- Generates the input `s_axi_aclk/dclk` clock signal
- Applies a reset to the example design
- Monitors the alarms and other status outputs
- Reads the respective registers when a conversion is complete

Migrating and Upgrading

This appendix contains information about migrating a design from ISE® to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating to the Vivado Design Suite

For information on migrating to the Vivado Design Suite, see *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 8].

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations that take place when you upgrade to a more current version of this IP core in the Vivado Design Suite.

Parameter Changes in the XCI File

- `Enable_Temp_Bus` parameter added for enabling 12-bit `TEMP_OUT` port.
- `Enable_AXI4STREAM` parameter is used for enabling streaming interface. `FIFO_Depth` parameter is for the configuration of FIFO depth.
- `POWER_DOWN_ADCA` and `POWER_DOWN_ADCB` parameters are added to provide Vivado IDE option to power down `ADCB` or both.
- `SIM_FILE_SEL` and `SIM_FILE_REL_PATH` parameters are added to provide absolute/relative path for analog stimulus file.

Port Changes

AXI4-Stream interface ports are added. For port details, see Table 2-1, page 12.

Functionality Changes

AXI4-Stream supports added with configurable FIFO depth. You can enable `TEMP_OUT` port when AXI4-Stream is enabled.

Debugging

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

TIP: *If the IP generation halts with an error, there might be a license issue. See License Checkers in Chapter 1 for more details.*

Finding Help on Xilinx.com

To help in the design and debug process when using the XADC Wizard, 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 XADC Wizard. 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 are listed below, and 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 the XADC Wizard

AR: 56583

Technical Support

Xilinx provides technical support in 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 XADC Wizard design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

Vivado® Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. Vivado Design Suite debug feature 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 *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 9].

Reference Boards

Various Xilinx development boards support the XADC Wizard. These boards can be used to prototype designs and establish that the core can communicate with the system.

- 7 series evaluation boards
  - KC705
  - KC724

Simulation Debug

The simulation debug flow for Mentor Graphics Questa Simulator ® (QuestaSim) is illustrated below. A similar approach can be used with other simulators.

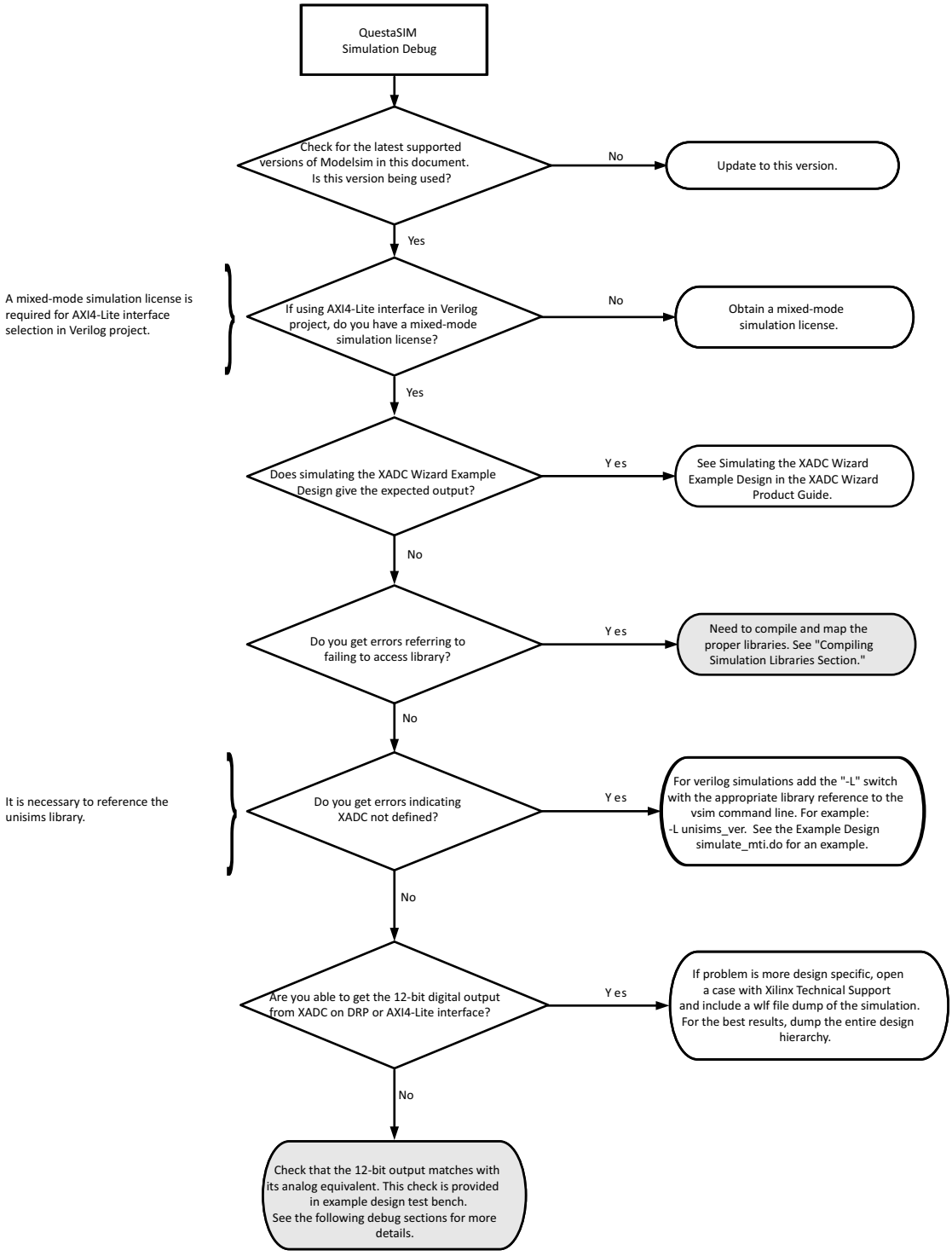

Figure B-1: **QuestaSIM Debug Flow Diagram**

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 Design Suite debug feature is a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the Vivado Design Suite debug feature for debugging the specific problems.

Many of these common issues can also be applied to debugging design simulations. 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 mixed-mode clock managers (MMCMs) in the design, ensure that all MMCMs have obtained lock by monitoring the `locked` port.
- If your outputs go to 0, check your licensing.

Interface Debug

AXI4-Lite Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. Output `s_axi_arready` asserts when the read address is valid, and output `s_axi_rvalid` asserts when the read data/response is valid. If the interface is unresponsive, ensure that the following conditions are met:

- The `s_axi_aclk and aclk` inputs are connected and toggling.
- The interface is not being held in reset, and `s_axi_areset` is an active-Low reset.
- The interface is enabled, and `s_axi_aclken` is active-High (if used).
- Ensure that the main core clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or Vivado Design Suite debug feature captures that the waveform is correct for accessing the AXI4-Lite interface.

AXI4-Stream Interfaces

- Ensure that the `m_axis_aclk` or `s_axi_aclk` (when AXI4-Lite interface is used) and `s_axis_aclk` are toggling.

- Ensure that `m_axis_tvalid` signal transitions from Low to High.
- Ensure that `m_axis_tready` signal transitions from Low to High.

Additional Resources and Legal Notices

Xilinx Resources

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

References

Unless otherwise noted, IP references are for the product documentation page. These documents provide supplemental material useful with this product guide:

1. *Vivado Design Suite User Guide: Designing with IP* (UG896)
2. *AMBA AXI4-Stream Protocol Specification*
3. *7 Series FPGAs XADC User Guide* (UG480)
4. *Vivado Design Suite User Guide: Getting Started* (UG910)
5. *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994)
6. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
7. *7 Series FPGAs Overview* (DS180)
8. *ISE to Vivado Design Suite Migration Guide* (UG911)
9. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
10. XADC Wizard Release Notes
11. *LogiCORE IP XADC Wizard User Guide* (UG772)
12. *LogiCORE IP AXI4-Lite IPIF Product Guide* (PG155)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/05/2016	3.3	• Updated the voltage alarm limits
04/06/2016	3.3	• Revision number advanced to 3.3 to align with core version number. • Updated IP Core Registers and User Parameters section.
09/30/2015	3.2	• Removed the VHDL support for example design and test bench. • Revision number advanced to 3.2 to align with core version number.
06/24/2015	3.1	Revision number advanced to 3.1 to align with core version number.
04/01/2015	3.0	• Added ip2intc_irpt signal.
10/01/2014	3.0	• Document updates only for revision change. • Added Fig. 1-2: Resistive Touch Interface. • Updated Maximum Frequencies section. • Updated description in 7 Series and Zynq-7000 Devices section. • Added cell heading descriptions to Table 2-2: XADC I/O Signals. • Updated Bits[17:2] description in CONVST Register. • Updated description in IP Interrupt Status Register section. • Updated Basic Tab and Control/Status Ports sections in Design Flow chapter. • Updated Figs. 4-3 to 4-11. • Added User Parameters section. • Updated Constraining the Core and Synthesis and Implementation sections.
10/02/2013	3.0	Revision number advanced to 3.0 to align with core version number. • Added description in Verifying Installation in Overview chapter. • Added License Checkers in Overview chapter. • Updated descriptions in Functional Overview. • Updated to V_{CCDDRO} in Product Specification chapter. • Updated Figs. 4-2 to 4-9 in Customizing and Generating the Core chapter. • Updated descriptions in Basic Tab section. • Updated descriptions in Control/Status Ports section. • Added new description in Output Generation. • Added new Simulation, Synthesis, and Test Bench chapters. • Added new parameters in Parameter Changes in the XCI File section. • Updated Migrating Appendix. • Updated Debug Appendix and AR.

Date	Version	Revision
03/20/2013	2.1	• Updated core v3.0 for Vivado Design Suite only. Removed ISE. • Added AXI4-Stream content. • Updated Figs. 1-1, 1-2, and Feature Summary. • Updated Table 2-1 7 Series FPGAs Resource Estimates. • Updated Table 2-2 with additional AXI4-Stream signals. • Updated Timing diagrams in Figs. 3-1 to 3-2. • Updated Figs. 4-1 to 4-8 and added AXI4-Stream Options. • Updated Output Generation section. • Updated Migrating section. • Updated Debug section with AXI4-Stream Interfaces section. • Updated to Questa SIM.
12/18/2012	2.0	• Updated core v2.4 and Vivado Design Suite v2012.4. Removed Zynq references. • Updated block diagram in Fig. 1-1. • GUIs updated to v2.4. • Updated Table 2-1: 7 Series Devices Resource Estimates. • Updated Table 2-2: XADC I/O Signals. • Updated description to “configuring HDL” in Register Space. • Updated CONVST register (CONVSTR) Bits[17:1]. • Updated Designing with the Core chapter. • Added IP Integrator. • Updated all figures and FIFO Depth in Customizing and Generating the Core chapter. • Added temp_rd_arbiter.vhd, drp_arbiter.vhd, and drp_rdwr_fsm.vhd to Table 6-2. • Updated Simulation chapter. • Updated Migration Appendix. • Added new Debug Appendix.
10/16/2012	1.0	• Initial Xilinx release of Product Guide and replaces UG772 • Added Vivado and Zynq-7000 information in IP Facts Table • Added support for AXI4-Lite interface • Added Fig. 1-1 block diagram and Vivado GUIs throughout book • Added Product Specification and Designing with the Core sections • Added Section II: Vivado Design Suite and Section III: Appendices

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos.

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.