AXI Timer v2.0

LogiCORE IP Product Guide

Vivado Design Suite

PG079 (v2.0) August 15, 2025

AMD

Table of Contents

Chapter 1

Introduction

The AMD LogiCORE™ IP AXI Timer/Counter is a 32/64-bit timer module that interfaces to the AXI4-Lite interface.

Features

- AXI interface based on the AXI4-Lite specification
- Two programmable interval timers with interrupt, event generation, and event capture capabilities
- Configurable counter width
- One Pulse Width Modulation (PWM) output
- Cascaded operation of timers in generate and capture modes
- Freeze input for halting counters during software debug

IP Facts

AMD LogiCORE™ IP Facts Table	
Core Specifics	
Supported Device Family[1]	AMD Versal™ adaptive SoCs, AMD UltraScale+™ families, AMD UltraScale™ architecture, AMD Zynq™ 7000, 7 series
Supported User Interfaces	AXI4-Lite
Resources	Performance
Provided with Core	
Design Files	VHDL
Example Design	VHDL
Test Bench	VHDL
Constraints File	XDC File
Simulation Model	Not Provided
Supported S/W Driver[2]	Standalone and Linux
Tested Design Flows[3]	
Design Entry	AMD Vivado™ Design Suite
Simulation	For supported simulators, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973).
Synthesis	Vivado Synthesis
Support	
All Vivado IP Change Logs	Master Vivado IP Change Logs: 72775
Support web page	

Notes:

1. For a complete list of supported devices, see the AMD Vivado™ IP catalog.
2. Standalone driver details can be found in <install_directory>/Vitis/<release>/data/embeddedsw/doc/xilinx_drivers_api_toc.htm. Linux OS and driver support is available from the Wiki page.
3. For the supported versions of third-party tools, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973).

Chapter 2

Overview

Navigating Content by Design Process

AMD Adaptive Computing documentation is organized around a set of standard design processes to help you find relevant content for your current development task. You can access the AMD Versal™ adaptive SoC design processes on the Design Hubs page. You can also use the Design Flow Assistant to better understand the design flows and find content that is specific to your intended design needs. This document covers the following design processes:

- **Hardware, IP, and Platform Development:** Creating the PL IP blocks for the hardware platform, creating PL kernels, functional simulation, and evaluating the AMD Vivado™ timing, resource use, and power closure. Also involves developing the hardware platform for system integration. Topics in this document that apply to this design process include:
  - Port Descriptions
  - Clocking
  - Resets
  - Customizing and Generating the Core
  - Chapter 6: Example Design

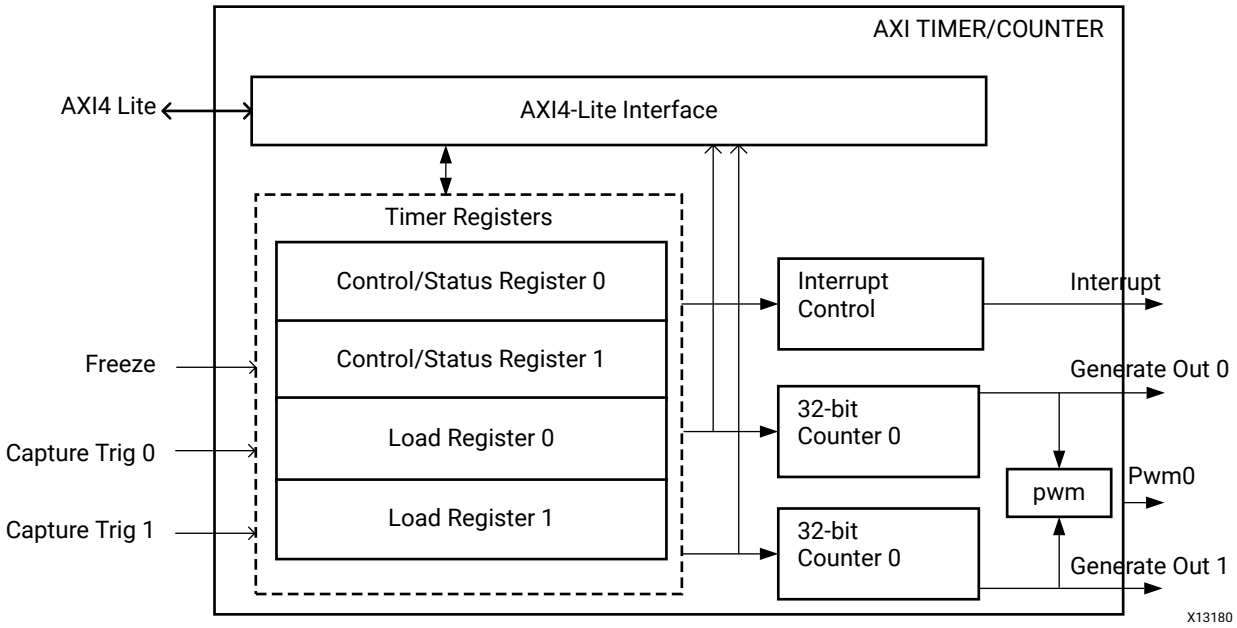
Functional Description

The AXI Timer is organized as two identical timer modules as shown in the following figure. Each timer module has an associated load register that is used to hold either the initial value for the counter for event generation or a capture value, depending on the mode of the timer.

The *generate value*, which is the value loaded into the load register, is used to generate a single interrupt at the expiration of an interval or a continuous series of interrupts with a programmable interval. The *capture value* is the timer value that has been latched on detection of an external event. The clock rate of the timer modules is `s_axi_aclk` (no prescaling of the clock is performed). All of the timer/counter interrupts are OR'ed together to generate a single external interrupt signal. The interrupt service routine reads the control/status registers to determine the source of the interrupt.

The block diagram of AXI Timer, also known as AXI Timer/Counter, is shown in the following figure.

Figure 1: **Block Diagram of AXI Timer**



Operation Overview

The AXI Timer/Counter modules provides an AXI4-Lite interface to communicate with the host. The timer/counter design has the following key modules:

- **AXI4-Lite Interface:** The AXI4-Lite Interface module implements an AXI4-Lite slave interface for accessing memory mapped Timer registers. For details about the AXI4-Lite slave interface, see the *LogiCORE IP AXI Lite IPIF (axi_lite_ipif) (v1.01a) Data Sheet (AXI)* (DS765).
- **Timer Registers:** The Register block implements a set of 32-bit registers for each timer/counter. This set of register contains load register, timer/counter register and control/status register.
- **32-bit Counters:** The Timer/Counter module has two 32-bit counters, each of which can be configured for up/down counts and can be loaded with a value from the load register.

- **Interrupt Control:** The Interrupt control module generates a single interrupt depending on the mode of operation.
- **Pulse Width Modulation (PWM):** The PWM block generates a pulse signal, PWM0, with a specified frequency and duty factor. It uses Timer 0 for PWM0 period, and Timer 1 for PWM0 output width.

Timer Modes

Four modes can be used with the two Timer/Counter modules:

- Generate Mode
- Capture Mode
- Pulse Width Modulation Mode
- Cascade Mode

Generate Mode

In the Generate mode, the value in the load register is loaded into the counter. The counter, when enabled, begins to count up or down, depending on the selection of the Up/Down Count Timer (UDT) bit in the Timer Control Status Register (TCSR). See and . On transition of the carry out of the counter, the counter stops or automatically reloads the generate value from the load register and, after reaching the timeout value, continues counting as selected by the Auto Reload/Hold (ARHT) bit in the TCSR. The Timer Interrupt Status (TINT) bit is set in TCSR and, if enabled, the external `GenerateOut` signal is driven to 1 for one clock cycle.

If enabled, the interrupt signal for the timer is driven to 1 when reaching the timeout value. Clear the interrupt by writing a 1 to the Timer Interrupt register. Use this mode for generating repetitive interrupts or external signals with a specified interval.

Capture Mode

In Capture mode, the value of the counter is stored in the load register when the external capture signal is asserted. The TINT bit is also set in the TCSR on detection of the capture event. The counter can be configured as an up or down counter for this mode as determined by the selection of the UDT bit in TCSR. The Auto Reload/Hold (ARHT) bit controls whether the capture value is overwritten with a new capture value before the previous TINT flag is cleared. Use this mode for time-tagging external events while simultaneously generating an interrupt.

Pulse Width Modulation Mode

In Pulse Width Modulation (PWM) mode, two timer/counters are used as a pair to produce an output signal (`PWM0`) with a specified frequency and duty factor. Timer 0 sets the period and Timer 1 sets the high time for the `PWM0` output.

Cascade Mode

In the Cascade mode, the two timer/counters are cascaded to operate as a single 64-bit counter/timer. The cascaded counter can work in both generate and capture modes. TCSR0 acts as the control and status register for the cascaded counter. TCSR1 is ignored in this mode.

Use this mode when a timer/counter more than 32-bits wide is required. Cascaded operation requires using Timer 0 and Timer 1 together as a pair. The counting event for Timer 1 is when Timer 0 rolls over from all 1s to all 0s, or vice-versa when counting down.

Interrupts

The TC interrupt signals can be enabled or disabled with the Enabel Interrupt for Timer (ENIT) bit in the TCSR. The interrupt status bit (TINT) in the TCSR cannot be disabled and always reflects the current state of the timer interrupt. In Generate mode, a timer interrupt is caused by the counter rolling over (the same condition used to reload the counter when ARHT is set to 1). In Capture mode, the interrupt event is the capture event.

Feature Summary

The features of the AXI Timer/Counter core are as follows:

- 32-bit slave peripheral with AXI4-Lite Interface
- Two programmable interval timers with interrupt, event generation and capture capabilities
- Configurable counter width. Supports cascaded mode of timers in generate and capture modes.
- Supports programmable auto-reload & hold operation on each timer/counter.
- One Pulse Width Modulation (PWM) output
- Programmable interrupt generation to enable/disable the timer interrupt.
- Freeze input for halting counters during software debug

Applications

Applications of this core include:

- Clock generation with different time periods and duty cycle
- Pulse generation circuits

- Generating time-related interrupts
- Pulse Width Modulation signal for motor control

Licensing and Ordering

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

For more information about this core, visit the AXI Timer product web page.

Information about other AMD LogiCORE™ IP modules is available at the Intellectual Property page. For information about pricing and availability of other AMD LogiCORE IP modules and tools, contact your local sales representative.

License Checkers

If the IP requires a license key, the key must be verified. The AMD 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 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 contains specific details about the performance and resource utilization of the core.

Performance

Performance characterization of this core has been done using the margin system methodology. The details of the margin system characterization methodology is described in the .

The summary of performance F_{MAX} with this core performed on margin system is provided in the following table.

***Note*:** The performance numbers for AMD UltraScale™ architecture-based and AMD Zynq™ 7000 devices are expected to be similar to 7 series device numbers.

Table 1: **Maximum Frequencies**

Device Family	Speed Grade	F_{MAX} (MHz) AXI4-Lite
AMD Virtex™ 7	-1	180
AMD Kintex™ 7		180
AMD Artix™ 7		120
Virtex 7	-2	200
Kintex 7		200
Artix 7		140
Virtex 7	-3	220
Kintex 7		220
Artix 7		160

Resource Utilization

Resources required for the AXI Timer core has been estimated for the 7 series FPGAs (see the following table). These values were generated using the Vivado Design Suite.

The AXI Timer resource utilization for various parameter combinations measured on a 7 series device are detailed in the following table.

***Note*:** Resources numbers for UltraScale and Zynq 7000 devices are expected to be similar to 7 series device numbers.

Table 2: **Performance and Resource Utilization: 7 Series and Zynq 7000 Devices**

Parameter Values		Device Resources		
Width of Timer/ Counter	**Enable Timer2**	**Slices**	**Flip-Flops**	**LUTs**
8	False	49	53	96
16	False	61	69	120
32	False	84	101	181
8	True	50	74	123
16	True	74	106	161
32	True	97	170	256

Port Descriptions

The AXI Timer input/output (I/O) signals are listed and described in the following table.

Table 3: **I/O Signal Descriptions**

Signal Name	Interface	I/O	Initial State	Description
s_axi_aclk	Clock	I	NA	AXI Clock
s_axi_aresetn	Reset	I	NA	AXI Reset, active-Low
s_axi_*	S_AXI	NA	NA	AXI4-Lite Slave Interface signals. See Appendix A of the *AXI Reference Guide* (UG761) for AXI4-Lite signals.
capturetrig0	Timer	I	-	Timer 0 Capture Trigger Input
capturetrig1	Timer	I	-	Timer 1 Capture Trigger Input In cascade mode, this is not used.
freeze	Timer	I	-	
generateout0	Timer	O	0x0	Timer 0 Generate Output Asserts whenever the timer 0 wraps from all 0s to all 1s or vice-versa. In cascade mode, this signal is asserted when the 64-bit value wraps from all 0s to all 1s or vice-versa.

Table 3: **I/O Signal Descriptions** *(cont'd)*

Signal Name	Interface	I/O	Initial State	Description
generateout1	Timer	O	0x0	Timer 1 Generate Output Asserts when the timer1 wraps from all 0s to all 1s or vice-versa. In cascaded mode, this signal is asserted when lower 32-bit counter (timer 0) wraps from all 0s to all 1s or vice-versa.
interrupt		O	0x0	Indicates that the condition for an interrupt on this timer has occurred. If the timer mode is capture and the timer is enabled, this bit indicates a capture has occurred. If the mode is generate, this bit indicates the counter has rolled over. 0 = No interrupt has occurred 1 = Interrupt has occurred
pwm0	Timer	O	0x0	Pulse Width Modulation Output 0

Register Space

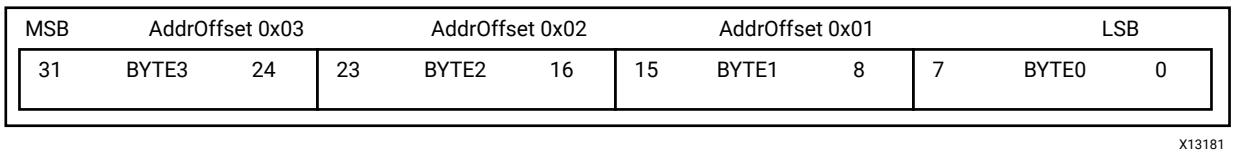
Timer Counter registers are accessed as one of these types:

- Byte (8 bits)
- Half word (2 bytes)
- Word (4 bytes)

The AXI Timer/Counter registers are organized as little-endian data. The bit and byte labeling for the little-endian data types is shown in the following figure.

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

Figure 2: **Little Endian Data Types**



The following table shows all the AXI Timer registers and their addresses. Accesses to addresses inside the core address range other than these registers return an OKAY response, with reads returning 0 data values and writes having no effect.

Table 4: **Register Overview**

Address Offset	Register Name	Description
0h	TCSR0	Timer 0 Control and Status Register
04h	TLR0	Timer 0 Load Register
08h	TCR0	Timer 0 Counter Register
0Ch-0Fh	RSVD	Reserved
10h	TCSR1	Timer 1 Control and Status Register
14h	TLR1	Timer 1 Load Register
18h	TCR1	Timer 1 Counter Register
1Ch-1Fh	RSVD	Reserved

Control/Status Register 0 (TCSR0)

The following figure and table show the Control/Status register 0. Control/Status Register 0 contains the control and status bits for timer module 0.

Figure 3: **Control/Status Register 0**

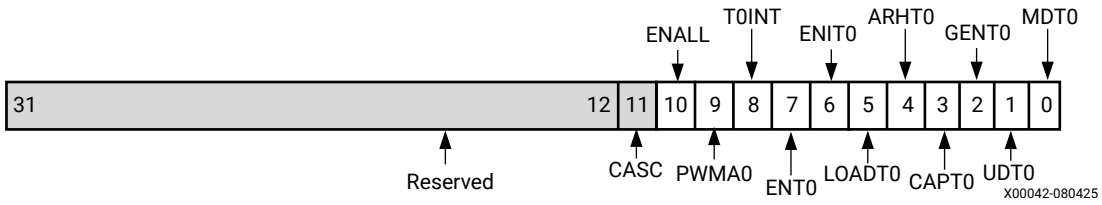


Table 5: **Control/Status Register 0 Bit Definitions**

Bit	Name	Access Type	Reset Value	Description
31 - 12	Reserved	N/A	-	Reserved
11	CASC	Read/Write	0	Enable cascade mode of timers 0 = Disable cascaded operation 1 = Enable cascaded operation Cascaded operation requires using Timer 0 and Timer 1 together as a pair. The counting event for the Timer 1 is when the Timer 0 rolls over from all 1s to all 0s or vice-versa when counting down. TLR0 and TLR1 are used for lower 32-bit and higher 32-bit respectively. Similarly, TCR0 contains lower 32-bits for the 64-bit counter and TCR1 contains the higher 32-bits. Only TCSR0 is valid for both the timer/counters in this mode. This CASC bit must be set before enabling the timer/counter.

Table 5: **Control/Status Register 0 Bit Definitions** *(cont'd)*

Bit	Name	Access Type	Reset Value	Description
10	ENALL	Read/Write	0	Enable All Timers 0 = No effect on timers 1 = Enable all timers (counters run) This bit is mirrored in all control/ status registers and is used to enable all counters simultaneously. Writing a 1 to this bit sets ENALL, ENT0, and ENT1. Writing a 0 to this register clears ENALL but has no effect on ENT0 and ENT1.
9	PWMA0	Read/Write	0	Enable Pulse Width Modulation for Timer 0 0 = Disable pulse width modulation 1 = Enable pulse width modulation PWM requires using Timer 0 and Timer 1 together as a pair. Timer 0 sets the period of the PWM output, and Timer 1 sets the high time for the PWM output. For PWM mode, MDT0 and MDT1 must be 0 and C_GEN0_ASSERT and C_GEN1_ASSERT must be 1.
8	T0INT	Read/Write	0	Timer 0 Interrupt Indicates that the condition for an interrupt on this timer has occurred. If the timer mode is capture and the timer is enabled, this bit indicates a capture has occurred. If the mode is generate, this bit indicates the counter has rolled over. Must be cleared by writing a 1. Read: 0 = No interrupt has occurred 1 = Interrupt has occurred Write: 0 = No change in state of T0INT 1 = Clear T0INT (clear to 0)
7	ENT0	Read/Write	0	Enable Timer 0 0 = Disable timer (counter halts) 1 = Enable timer (counter runs)
6	ENIT0	Read/Write	0	Enable Interrupt for Timer 0 Enables the assertion of the interrupt signal for this timer. Has no effect on the interrupt flag (T0INT) in TCSR0. 0 = Disable interrupt signal 1 = Enable interrupt signal

Table 5: **Control/Status Register 0 Bit Definitions** *(cont'd)*

Bit	Name	Access Type	Reset Value	Description
5	LOAD0	Read/Write	0	Load Timer 0 0 = No load 1 = Loads timer with value in TLR0 Setting this bit loads timer/counter register (TCR0) with a specified value in the timer/counter load register (TLR0). This bit prevents the running of the timer/counter; hence, this should be cleared alongside setting Enable Timer/Counter (ENT0) bit in TCSR0.
4	ARHT0	Read/Write	0	Auto Reload/Hold Timer 0 When the timer is in Generate mode, this bit determines whether the counter reloads the generate value and continues running or holds at the termination value. In Capture mode, this bit determines whether a new capture trigger overwrites the previous captured value or if the previous value is held. 0 = Hold counter or capture value. The TLR must be read before providing the external capture. 1 = Reload generate value or overwrite capture value
3	CAPT0	Read/Write	0	Enable External Capture Trigger Timer 0 0 = Disables external capture trigger 1 = Enables external capture trigger
2	GENT0	Read/Write	0	Enable External Generate Signal Timer 0 0 = Disables external generate signal 1 = Enables external generate signal
1	UDT0	Read/Write	0	Up/Down Count Timer 0 0 = Timer functions as up counter 1 = Timer functions as down counter
0	MDT0	Read/Write	0	Timer 0 Mode See Timer Modes. 0 = Timer mode is generate 1 = Timer mode is capture

Load Register (TLR0 and TLR1)

When the counter width has been configured as less than 32 bits, the load register value is right-justified in TLR0 and TLR1. The least-significant counter bit is always mapped to load register bit 0.

In cascade mode, TLR0 has the least significant 32-bits of the generate value and TLR1 should have the most significant bits of the generate value in generate mode. Similarly, in cascade mode TLR0 has the captured value from TCR0 and TLR1 from TCR1.

The following figure and table show the load register.

Figure 4: **Timer/Counter Load Register**

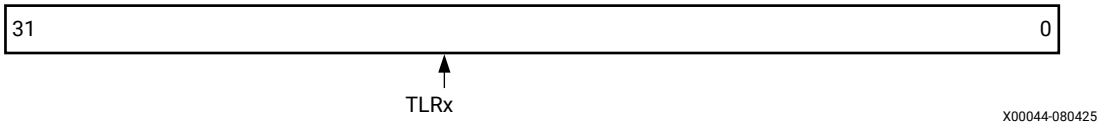


Table 6: **Load Register Bit Definitions**

Bit	Name	Access Type	Reset Value	Description
31-0	Timer/Counter Load Register	Read/Write	0x0	Timer/Counter Load register

Timer/Counter Register (TCR0 and TCR1)

When the counter width has been configured as less than 32 bits, the count value is right-justified in TCR0 and TCR1. The least-significant counter bit is always mapped to Timer/Counter Register bit 0. The following figure and table show the Timer/counter register. In cascade mode, TCR0 has the least significant 32-bits of the 64-bit counter, and TCR1 has the most significant bits.

Figure 5: **Timer/Counter Register**

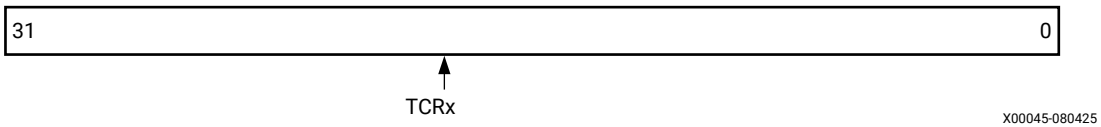


Table 7: **Timer/Counter Register Bit Definitions**

Bit	Name	Access Type	Reset Value	Description
31-0	Timer/Counter Register	Read	0x0	Timer/Counter register

Control/Status Register 1 (TCSR1)

The following figure and table show the Control/Status register 1. Control/Status Register 1 contains the control and status bits for timer module 1. This register is used for loading the TLR1 register in cascade mode.

Figure 6: **Control/Status Register 1**

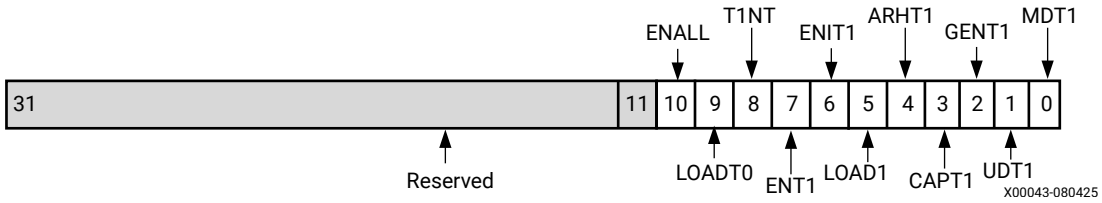


Table 8: **Control/Status Register 1 Bit Definitions**

Bit	Name	Access Type	Reset Value	Description
31 - 11	Reserved	N/A	-	Reserved
10	ENALL	Read/Write	0	Enable All Timers 0 = No effect on timers 1 = Enable all timers (counters run) This bit is mirrored in all control/status registers and is used to enable all counters simultaneously. Writing a 1 to this bit sets ENALL, ENT0, and ENT1. Writing a 0 to this register clears ENALL but has no effect on ENT0 and ENT1.
9	PWMA0	Read/Write	0	Enable Pulse Width Modulation for Timer1 0 = Disable pulse width modulation 1 = Enable pulse width modulation PWM requires using Timer 0 and Timer 1 together as a pair. Timer 0 sets the period of the PWM output, and Timer 1 sets the high time for the PWM output. For PWM mode, MDT0 and MDT1 must be 0.
8	T1INT	Read/Write	0	Timer1 Interrupt Indicates that the condition for an interrupt on this timer has occurred. If the timer mode is capture and the timer is enabled, this bit indicates a capture has occurred. If the mode is generate, this bit indicates the counter has rolled over. Must be cleared by writing a 1. Read: 0 = No interrupt has occurred 1 = Interrupt has occurred Write: 0 = No change in state of T1INT 1 = Clear T1INT (clear to 0)

Table 8: **Control/Status Register 1 Bit Definitions** *(cont'd)*

Bit	Name	Access Type	Reset Value	Description
7	ENT1	Read/Write	0	Enable Timer1 0 = Disable timer (counter halts) 1 = Enable timer (counter runs)
6	ENIT1	Read/Write	0	Enable Interrupt for Timer1 Enables the assertion of the interrupt signal for this timer. Has no effect on the interrupt flag (T1INT) in TCSR1. 0 = Disable interrupt signal 1 = Enable interrupt signal
5	LOAD1	Read/Write	0	Load Timer1 0 = No load 1 = Loads timer with value in TLR1 Setting this bit loads the timer/counter register (TCR1) with a specified value in the timer/counter load register (TLR1). This bit prevents running of timer/counter; hence, this should be cleared alongside setting Enable Timer/Counter (ENT1) bit in TCSR1.
4	ARHT1	Read/Write	0	Auto Reload/Hold Timer1 When the timer is in generate mode, this bit determines whether the counter reloads the generate value and continues running or holds at the termination value. In capture mode, this bit determines whether a new capture trigger overwrites the previous captured value or if the previous value is held until it is read. 0 = Hold counter or capture value 1 = Reload generate value or overwrite capture value
3	CAPT1	Read/Write	0	Enable External Capture Trigger Timer1 0 = Disables external capture trigger 1 = Enables external capture trigger
2	GENT1	Read/Write	0	Enable External Generate Signal Timer1 0 = Disables external generate signal 1 = Enables external generate signal
1	UDT1	Read/Write	0	Up/Down Count Timer1 0 = Timer functions as up counter 1 = Timer functions as down counter
0	MDT1	Read/Write	0	Timer1 Mode See Timer Modes. 0 = Timer mode is generate 1 = Timer mode is capture

Designing with the Core

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

Clocking

The AXI timer operates on `s_axi_aclk`.

Resets

The AXI timer resets when `s_axi_aresetn` is asserted. This is an active-Low signal and should be synchronous to `s_axi_aclk`.

Programming Sequence

Generate Mode

The Generate mode has the following programming sequence:

- On start-up, the *generate* value in the load register is loaded into the counter by setting the Load bit in the TCSR. This applies whether the counter is set up to Auto Reload or Hold when the interval has expired. Setting the Load bit to 1 loads the counter with the value in the load register. For proper operation, the Load bit must be cleared before the counter is enabled or along with setting the enable bit. The timer/counter starts ticking when Enable is set (ENT).
- When the ARHT bit (Auto Reload/Hold) is set to 1 and the counter rolls over from all 1s to all 0s when counting up, or conversely from all 0s to all 1s when counting down, the generate value in the load register is automatically reloaded into the counter and the counter continues to count. If the `GenerateOut` signal is enabled (bit GENT in the TCSR), an output pulse is generated (one clock period in width). This is useful for generating a repetitive pulse train with a specified period.

- When the Auto Reload/Hold (ARHT) bit is set to 0 and the counter rolls over from all 1s to all 0s when counting up, or conversely, from all 0s to all 1s when counting down, the counter holds at the current value and does not reload the generate value. If the `GenerateOut` signal is enabled (bit GENT in the TCSR), an output pulse of one clock period in width is generated. This is useful for a one-shot pulse that is to be generated after a specified period of time.
- The counter can be set up to count either up or down as determined by the selection of the UDT bit in the TCSR. If the counter is set up as a down counter, the generate value is the number of clocks in the timing interval. The period of the `GenerateOut` signal is the generate value (value in load register TLRx) multiplied by the clock period.
- When the counter is set to count down,

  $$\text{TIMING\_INTERVAL} = (\text{TLRx} + 2) * \text{AXI\_CLOCK\_PERIOD}$$

- When the counter is set to count up,

  $$\text{TIMING\_INTERVAL} = (\text{MAX\_COUNT} - \text{TLRx} + 2) * \text{AXI\_CLOCK\_PERIOD}$$

where `MAX_COUNT` is the maximum count value of the counter, such as `0xFFFFFFFF` for a 32-bit counter.

Capture Mode

Capture mode has the following programming sequence:

- The capture signal can be configured to be low-true or high-true.
- When the capture event occurs, the counter value is written to the load register. This value is called the capture value.
- When the ARHT bit is set to 0 and the capture event occurs, the capture value is written to the load register, which holds the capture value until the load register is read. If the load register is not read, subsequent capture events do not update the load register and are lost.
- When the ARHT bit is set to 1 and the capture event occurs, the capture value is always written to the load register. Subsequent capture events update the load register and overwrite the previous value, whether it has been read or not.
- The counter can be set up to count either up or down as determined by the selection of the UDT bit in the TCSR.

Pulse Width Modulation Mode

Pulse Width Modulation (PWM) mode has the following programming sequence:

- The mode for both Timer 0 and Timer 1 must be set to Generate mode (bit MDT in the TCSR set to 0).
- The PWMA0 bit in TCSR0 and PWMB0 bit in TCSR1 must be set to 1 to enable PWM mode.

- The `GenerateOut` signals must be enabled in the TCSR (bit GENT set to 1). The `PWM0` signal is generated from the `GenerateOut` signals of Timer 0 and Timer 1, so these signals must be enabled in both timer/counters.
- The assertion level of the `GenerateOut` signals for both timers in the pair must be set to `Active High`.
- The counter can be set to count up or down.

Setting the PWM Period and Duty Factor

The PWM period is determined by the generate value in the Timer 0 load register (TLR0). The PWM high time is determined by the generate value in the Timer 1 load register (TLR1). The period and duty factor are calculated as follows:

When counters are configured to count up (UDT = 0):

PWM_PERIOD = (MAX_COUNT - TLR0 + 2) * AXI_CLOCK_PERIOD

PWM_HIGH_TIME = (MAX_COUNT - TLR1 + 2) * AXI_CLOCK_PERIOD

When counters are configured to count down (UDT = 1):

PWM_PERIOD = (TLR0 + 2) * AXI_CLOCK_PERIOD

PWM_HIGH_TIME = (TLR1 + 2) * AXI_CLOCK_PERIOD

where `MAX_COUNT` is the maximum count value for the counter, such as `0xFFFFFFFF` for a 32-bit counter.

Cascade Mode

In Cascade mode, the two timer/counters are cascaded to operate as a single 64-bit counter/timer. The cascaded counter can work in both generate and capture modes. TCSR0 acts as the control and status register for the cascaded counter. TCSR1 is ignored in this mode.

This mode is used when there is a requirement for a timer/counter more than 32 bits wide. Cascaded operation requires using Timer 0 and Timer 1 together as a pair. The counting event for the Timer 1 is when the Timer 0 rolls over from all 1s to all 0s or vice-versa when counting down.

The cascade mode has the following programming sequence:

- Parameter enable_timer2 (Enable Timer 2) should be set to 0 because both timers are required for a cascaded operation.
- The width of the AXI Timer should be 32 because it represents the width of each timer/counter in the core.

- Load Registers of both timer/counters are used (TLR0 and TLR1 - TLR1 for higher 32-bit and TLR0 for lower 32-bit). The value loaded into the load registers is called the generate value in generate mode. And the capture value is captured in these load registers in capture mode.
- Timer/counter 0 control register TCSR0, `GenerateOut0`, Capture event 0 are valid in this mode. Timer 1 related signals are invalid, that is, TCSR1, `GenerateOut1` and `CaptureEvent1` are not used. TCSR1 is used only for loading the TLR1 register.
- CASC bit in Timer Control Status Register 0 (TCSR0) must be set for the counters to be in cascade mode. This bit must be set before enabling the timer/counter.
- The sequence of accesses for generate and capture modes are as mentioned in previous sections.
- In generate mode, when the counter is set to count down,

  $$\text{TIMING\_INTERVAL} = (\text{TLR} + 4) * \text{AXI\_CLOCK\_PERIOD}$$

  where *TLR* is the concatenated value of TLR1 and TLR0 (TLR = {TLR1, TLR0}).
- In generate mode, when the counter is set to count up,

  $$\text{TIMING\_INTERVAL} = (\text{MAX\_COUNT} - \text{TLR} + 4) * \text{AXI\_CLOCK\_PERIOD}$$

  where `MAX_COUNT` is the maximum count value of the counter, such as `0xFFFFFFFFFFFFFFFF` for a 64-bit counter, and *TLR* is the concatenated value of TLR1 and TLR0 (TLR = {TLR1, TLR0}).

The following are the steps for running the 64-bit counter/timer in generate mode:

1. Clear the timer enable bits in control registers (TCSR0 and TCSR1).
2. Write the lower 32-bit timer/counter load register (TLR0).
3. Write the higher 32-bit timer/counter load register (TLR1).
4. Set the CASC bit in Control register TCSR0.
5. Set other mode control bits in control register (TCSR0) as needed.
6. Enable the timer in Control register (TCSR0).

The following are the steps for reading the 64-bit counter/timer:

1. Read the upper 32-bit timer/counter register (TCR1).
2. Read the lower 32-bit timer/counter register (TCR0).
3. Read the upper 32-bit timer/counter register (TCR1) again. If the value is different from the 32-bit upper value read previously, go back to previous step (reading TCR0). Otherwise 64-bit timer counter value is correct.

Interrupts

The programming requirements to enable/disable interrupts and the programming sequence are as follows:

- Interrupt events can only occur when the timer is enabled. In Capture mode, this prevents interrupts from occurring before the timer is enabled.
- The interrupt signal goes high when the interrupt condition is met and the interrupt is enabled in the TCSR. The interrupt is asserted when the interrupt signal is High.
- A single interrupt signal is provided. The interrupt signal is the OR of the interrupts from the two counters. The interrupt service routine must poll the TCSRs to determine the source or sources of the interrupt.
- The interrupt status bit (TINT in the TCSR) can only be cleared by writing a 1 to it. Writing a 0 to it has no effect on the bit. Because the interrupt condition is an edge (the counter rollover or the capture event), it can be cleared at any time and does not indicate an interrupt condition until the next interrupt event.
- In cascade mode, only Timer 0 interrupt events occur. There are no interrupts from Timer 1.

Design Flow Steps

This section 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 AMD Vivado™ design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
- *Vivado Design Suite User Guide: Designing with IP* (UG896)
- *Vivado Design Suite User Guide: Getting Started* (UG910)
- *Vivado Design Suite User Guide: Logic Simulation* (UG900)

Customizing and Generating the Core

This section includes information about using AMD tools to customize and generate the core in the AMD 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) for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl console.

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

1. 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.

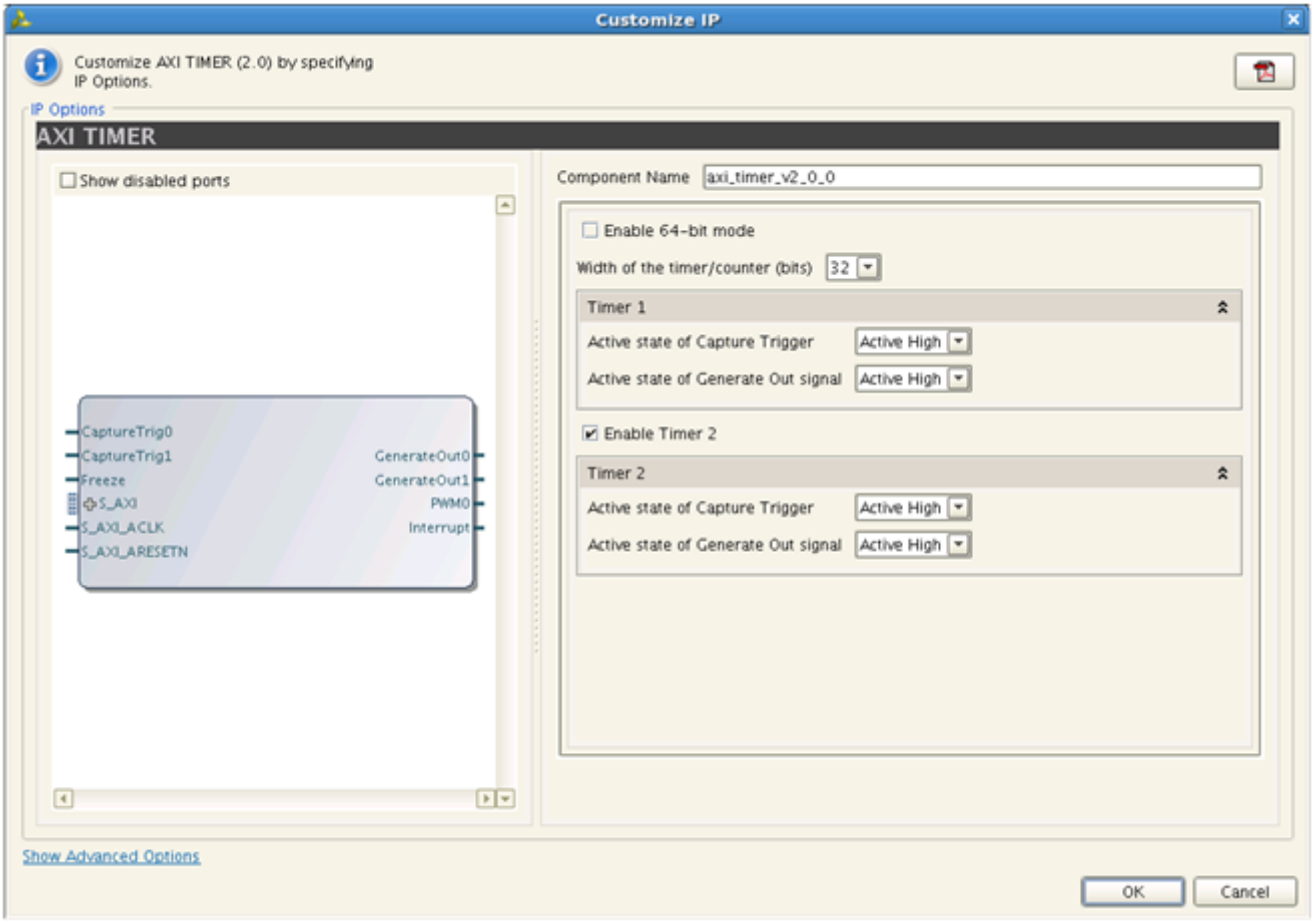
For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) and the *Vivado Design Suite User Guide: Getting Started* (UG910).

Figures in this chapter are illustrations of the Vivado IDE. The layout depicted here might vary from the current version.

Configuring the Core Parameters

The Vivado IDE for the AXI Timer for the AXI4-Lite interface is shown in the following figure.

Figure 7: **AXI Timer Configuration**



The following are the core parameters:

- **Enable 64-bit Mode:** Check this box if an AXI Timer with a 64-bit width is needed. This option disables the Timer 2 option because a 64-bit AXI Timer is created by cascading 2 timers inside the IP core.
- **Width of the Timer/Counter (bits):** Select 8, 16, 32 bit count widths based on user requirements.
- Timer 1
  - **Capture Trigger:** Select to configure this signal as active-High or active-Low. Default is active-High
  - **Generate Out:** Select to configure this signal as active-High or active-Low. Default is active-High.

- **Enable Timer 2:** Select if Timer 2 is needed.

- Timer 2
  - **Capture Trigger:** Select to configure this signal as active-High or active-Low. Default is active-High
  - **Generate Out:** Select to configure this signal as active-High or active-Low. Default is active-High.

Constraining the Core

Required Constraints

This section is not applicable for this IP core.

Device, Package, and Speed Grade Selections

This section is not applicable for this IP core.

Clock Frequencies

This section is not applicable for this IP core.

Clock Management

This section is not applicable for this IP core.

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

For comprehensive information about AMD Vivado™ simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900).

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896).

Chapter 6

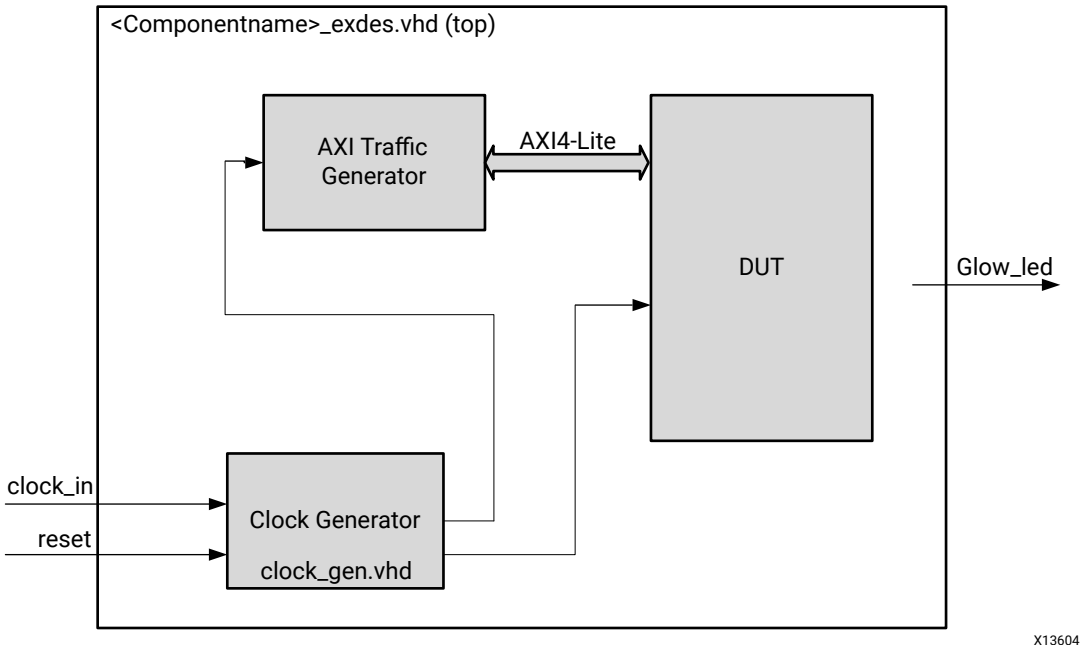
Example Design

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

Overview

The top module instantiates all components of the core and example design that are needed to implement the design in hardware, as shown in the following figure. This includes the clock generator (MMCME2), register configuration, data generator and data checker modules.

Figure 8: **Example Design Block Diagram**



This example design includes two modules:

- **Clock Generator:** MMCME2 is used to generate the clocks for the example design. MMCME2 is used to generate 100 MHz clock for `s_axi_aclk`. The DUT and other modules of the example design are kept under reset until MMCME2 is locked.
- **AXI Traffic Generator (ATG):** This module (IP) is configured in the System Test mode. All the AXI Timer-related AXI4-Lite transactions are stored in the COE file. For more information on the AXI Traffic Generator, see the . The ATG automatically starts the AXI4-Lite transaction after coming out of reset.

Implementing the Example Design

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

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

The ATG writes value 0x45 to 0x00 register. This starts the timer counters. When the timer value is reached certain count, the glow_led output drives 1. The glow_led output of the example design is connected to the GPIO_LED7 of the KC705 evaluation board and indicates the status of the example design. On successful completion of ATG write transactions to the timer registers and counters initiation, the eighth LED is lit. If an error occurs, the GPIO_LED7 is not lit.

Example Design Directory Structure

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

The following table lists the files delivered in `<component_name>_example/<component_name>_example.srcs/sources_1/imports/example_design`.

Table 9: **Example Design Directory**

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

The `<component_name>_example/<component_name>_example.srcs/sources_1/sim_1/imports/simulation` directory contains the test bench file for the example design: `<component_name>_exdes_tb.vhd`.

The `<component_name>_example/<component_name>_example.srcs/sources_1/constrs_1/imports/example_design` directory contains the top-level constraints file for the example design: `<component_name>_exdes.xdc`.

The XDC has all the necessary constraints needed to run the example design on the KC705 board. All the IO constraints are commented in the XDC file. Uncomment these constraints before implementing the design on the KC705 board.

Simulating the Example Design

The AXI Timer example design quickly simulates and demonstrates the behavior of the AXI Timer.

Simulation Results

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

If test passes, the following message is displayed:

```
Test Completed Successfully
```

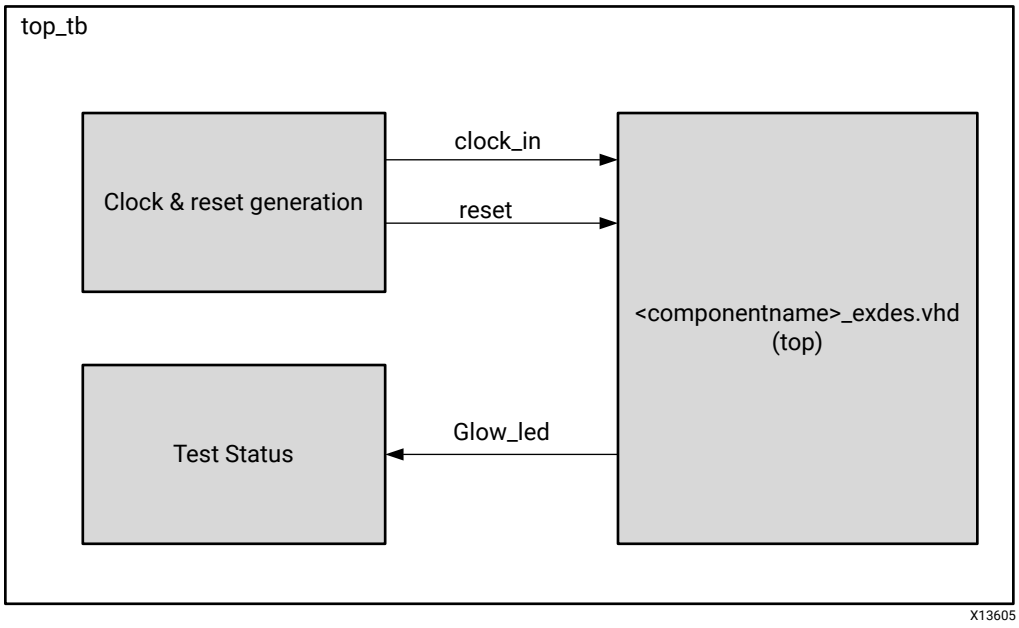
If the test fails or does not end, the following message is displayed:

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

Test Bench

This chapter contains information about the test bench provided in the AMD Vivado™ Design Suite. The following figure shows the test bench for AXI Timer example design. The top-level test bench generates a 200 MHz clock and drives the initial reset to the example design.

Figure 9: **Test Bench Block Diagram**



Appendix A

Debugging

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

If the IP requires a license key, the key must be verified. The AMD 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 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.*

Finding Help with AMD Adaptive Computing Solutions

To help in the design and debug process when using the core, the 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. The Community Forums are also available where members can learn, participate, share, and ask questions about AMD Adaptive Computing solutions.

Documentation

This product guide is the main document associated with the core. This guide, along with documentation related to all products that aid in the design process, can be found on the AMD Adaptive Support web page or by using the AMD Adaptive Computing Documentation Navigator. Download the 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 an AMD Adaptive Computing product. Answer Records are created and maintained daily to ensure that users have access to the most accurate information available.

Answer Records for this core can be located by using the Search Support box on the main AMD Adaptive Support web page. To maximize your search results, use 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 Core

AR 54438.

Technical Support

AMD Adaptive Computing provides technical support on the Community Forums for this AMD LogiCORE™ IP product when used as described in the product documentation. AMD Adaptive Computing 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 ask questions, navigate to the Community Forums.

Debug Tools

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

Vivado Design Suite Debug Feature

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

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

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

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

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

Many of these common issues can also be applied to debugging design simulations. Details are provided on:

- General Checks

General Checks

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

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the `locked` port.
- 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` input is connected and toggling.
- The interface is not being held in reset, and `s_axi_aresetn` is an active-Low reset.
- If the simulation has been run, verify in simulation to capture that the waveform is correct for accessing the AXI4-Lite interface.

Appendix B

Upgrading

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

Port Changes

There are no port changes.

Parameter Changes

There are no parameter changes.

Appendix C

Additional Resources and Legal Notices

Finding Additional Documentation

Technical Information Portal

The AMD Technical Information Portal is an online tool that provides robust search and navigation for documentation using your web browser. To access the Technical Information Portal, go to https://docs.amd.com.

Documentation Navigator

Documentation Navigator (DocNav) is an installed tool that provides access to AMD Adaptive Computing documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the AMD Vivado™ IDE, select **Help → Documentation and Tutorials**.
- On Windows, click the **Start** button and select **Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

***Note*:** For more information on DocNav, refer to the *Documentation Navigator User Guide* (UG968).

Design Hubs

AMD Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- Go to the Design Hubs web page.

Support Resources

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

References

These documents provide supplemental material useful with this guide:

1. *ARM® AMBA® AXI and ACE Specification* (ARM IHI 0022D)
2. *AXI Reference Guide* (UG761)
3. *AXI Interconnect LogiCORE IP Product Guide* (PG059)
4. *AXI4-Lite IPIF LogiCORE IP Product Guide* (PG155)
5. *LogiCORE IP AXI Lite IPIF (axi_lite_ipif) (v1.01a) Data Sheet (AXI)* (DS765)
6. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
7. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
8. *Vivado Design Suite User Guide: Implementation* (UG904)
9. *AXI Traffic Generator LogiCORE IP Product Guide* (PG125)
10. *ISE to Vivado Design Suite Migration Guide* (UG911)
11. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
12. *Vivado Design Suite User Guide: Getting Started* (UG910)
13. *Vivado Design Suite User Guide: Designing with IP* (UG896)
14. *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
08/15/2025 Version 2.0	
IP Facts	Added Versal device support.
Timer/Counter Register (TCR0 and TCR1)	Updated topic.
Cascade Mode	Updated topic.

Section	Revision Summary
10/05/2016 Version 2.0	
N/A	• Added note in Register Space. • Updated Automotive Disclaimer in Please Read: Important Legal Notices
11/18/2015 Version 2.0	
N/A	Added support for UltraScale+ families.
04/02/2014 Version 2.0	
N/A	General document updates.
12/18/2013 Version 2.0	
N/A	Added support for UltraScale architecture.
10/02/2013 Version 2.0	
N/A	• Added Chapter 6: Example Design and Chapter 7: Test Bench. • Added IP integrator support. • Changed all signals/ports to lower case.
03/20/2013 Version 2.0	
N/A	• Updated for core v2.0, and for Vivado Design Suite-only support. • Added list of modules, and updated the I/O signal descriptions, and register overview. • Major revisions made to the Debugging Appendix.
10/16/2012 Version 1.0	
N/A	Initial Xilinx release as a product guide. Based on DS764, *LogiCORE IP AXI Timer*. Updated core to v1.03a.

Please Read: Important Legal Notices

DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

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.

Copyright