

AXI Timebase Watchdog Timer v3.0

LogiCORE IP Product Guide

Vivado Design Suite

PG128 October 4, 2017

Table of Contents

IP Facts

Chapter 1: Overview

Functional Description.....	5
Feature Summary.....	8
Licensing and Ordering	9

Chapter 2: Product Specification

Performance.....	10
Resource Utilization.....	11
Port Descriptions	11
Register Space	13

Chapter 3: Designing with the Core

Clocking.....	23
Resets	23
Programming Sequence for Legacy WDT Mode.....	23
Programming Sequence for WWDT Mode.....	24
Protocol Description	26

Chapter 4: Design Flow Steps

Customizing and Generating the Core	29
Constraining the Core	32
Simulation	33
Synthesis and Implementation	33

Chapter 5: Example Design

Implementing the Example Design.....	35
Simulating the Example Design.....	36

Chapter 6: Test Bench

Appendix A: Upgrading

Migrating to the Vivado Design Suite.....	39
Upgrading in the Vivado Design Suite	39

Appendix B: Debugging

Finding Help on Xilinx.com	40
Debug Tools	42
Interface Debug	42

Appendix C: Additional Resources and Legal Notices

Xilinx Resources	43
Documentation Navigator and Design Hubs	43
References	43
Revision History	45
Please Read: Important Legal Notices	46

Introduction

The Xilinx® LogiCORE™ IP AXI4-Lite Timebase Watchdog Timer (WDT) is a 32-bit peripheral that provides a 32-bit free-running timebase and watchdog timer.

Features

- Connects as a 32-bit slave on a AXI4-Lite interface
- Programmable watchdog timer width
- Watchdog timer with selectable timeout period and interrupt
- Configurable WDT enable: enable-once or enable-repeatedly
- One 32-bit free-running timebase counter with rollover interrupt-dual control register
- Configurable windowing mode with the following features:
 - Window Watchdog capability
 - Program Sequence Monitoring (PSM)
 - Second Sequence Timer (SST)
 - Fail counter (FC)

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family ⁽¹⁾	UltraScale+™, UltraScale™ Zynq®-7000 All Programmable SoC 7 Series
Supported User Interfaces	AXI4-Lite
Resources	See Table 2-2 .
Provided with Core	
Design Files	VHDL
Example Design	VHDL
Test Bench	VHDL
Constraints File	Xilinx Design Constraints (XDC)
Simulation Model	Not Provided
Supported S/W Driver	Standalone and Linux
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 list of supported derivative devices, see the Vivado IP catalog.
2. Standalone driver details can be found in the software development kit (SDK) directory <install_directory>/SDK.
3. For the supported versions of the tools, see the [Xilinx Design Tools: Release Notes Guide](#).

Overview

Functional Description

The top-level block diagram for the AXI Timebase Watchdog Timer (WDT) is shown in [Figure 1-1](#).

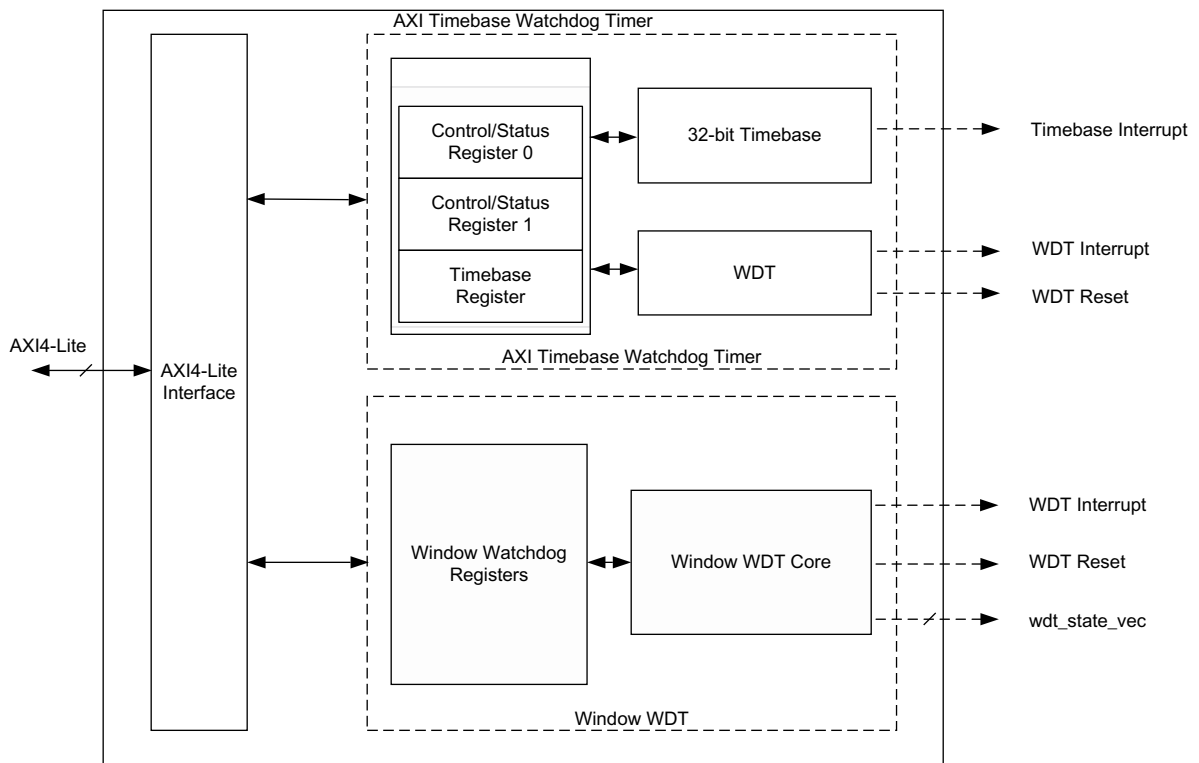


Figure 1-1: AXI Timebase Watchdog Timer Block Diagram

- AXI4-Lite Interface** – AXI4-Lite Interface module implements AXI4-Lite slave interface for accessing memory mapped Timebase WDT registers. For additional information about the AXI4-Lite slave interface, see the specification usage section of the *LogiCORE IP AXI4-Lite IPIF Product Guide* (PG155) [\[Ref 1\]](#).
- Timebase Watchdog Timer Registers** – Timebase Watchdog Timer register module includes all memory-mapped registers (as shown in [Figure 1-1](#)). It consists of a 32-bit Control/Status register 0, a 32-bit Control/Status register 1 and a 32-bit Timebase register (TBR).

- **Timebase Watchdog Mode**

- **32-bit Timebase** – 32-bit timebase consists of a free-running 32-bit timebase counter.
- **WDT** – WDT block provides watchdog functionality and generates reset and/or interrupt.

WDT uses a dual-expiration architecture. After one expiration of the timeout interval, an interrupt is generated and the WDT state bit is set to one in the status register. If the state bit is not cleared (by writing a one to the state bit) before the next expiration of the timeout interval, a WDT reset is generated.

When WDT reset is generated, the WDT reset status bit in Status register is set. This bit is used to determine if the last system reset was due to WDT reset or not.

WDT can only be disabled by writing two distinct addresses, reducing the possibility of inadvertently disabling the WDT in application code. After WDT expires, it can be restarted only by asserting the `s_axi_aresetn` pin.

- **Window Watchdog Timer** – A window watchdog timer (WWDT) starts with an adjustable period called "close (first) window time" followed by another period called "open (second) window time." The WWDT has to be restarted within the open window time. If software tries to restart WWDT outside of the open window time period, it generates a reset. In this method, WWDT includes more sticker requirement on the software execution and provides more predictability. If software is running out of control, there is a greater probability that it fails to fulfill a strict timing requirement of the WWDT restart and hence can reduce the probability of errant restart of the watchdog.

Therefore, WWDT can increase the probability of recovering from anomalies where WDT might not be able to help.

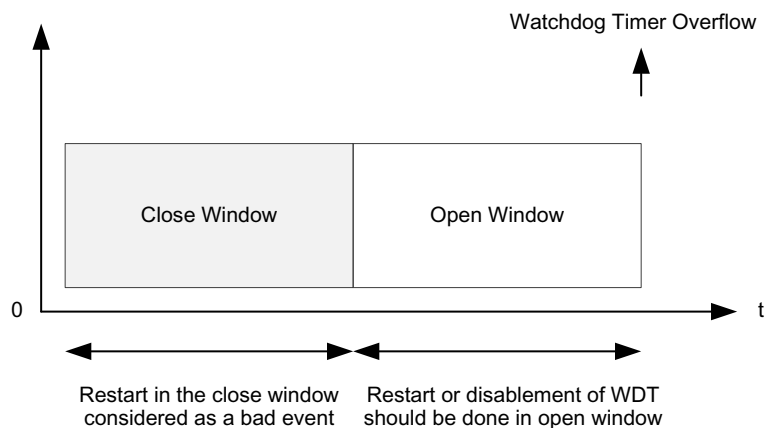


Figure 1-2: Window Watchdog Close and Open Period

- **Window Watchdog Mode** – When Window WDT is enabled in this mode, the software must restart or disable it in the open (second) window duration only. If software is successful, it is considered a good event.

If Window WDT gets a restart attempt or disablement attempt in the close (first) window, it is considered a bad event. In this scenario, Window WDT is not disabled.

Also, if Window WDT does not get any restart or disablement before the second window expires, it is considered a bad event.

- **Program Sequence Monitor** – The Program Sequence Monitor (PSM) feature provides additional health check if the software task has been executed in expected manner without error. With this feature, two Task Signature registers (TSR0 and TSR1) are added.

PSM can be enabled using the register control bit on top of the basic WWDT mode.

Software initializes the TSR0 and if Task0 completes successfully, it writes the Task0 signature in the TSR1. WWDT compares the TSR0 and TSR1 at the restart kick time and if they do not match, it is considered a bad event.

- **Second Sequence Timer** – The Second Sequence Timer (SST) is an additional timer that can be used to delay the inevitable WWDT reset. This delay can be useful in applications where software needs to log the data for later debugs before the watchdog driven reset arrives.

SST can be enabled using the register control bit in both WWDT modes.

- **Fail Counter** – When Fail Counter (FC) is enabled, every good event decrements the fail counter by 1 (unless it is 0). Every bad event increments the fail counter by 1 (unless it is 7).

Post reset value (default value) of fail counter is 5.

When fail counter is 7 and the bad event occurs, it leads to `wdt_reset` generation either immediately or after SST count rolls over.

If fail counter is disabled, a single bad event leads to `wdt_reset` generation.

Operation Overview

Timebase Operation

The timebase is a 32-bit counter that is incremented by one on the rising edge of the input clock. The counter is reset to zero when the reset is asserted or when the WDT is enabled. The Timebase register (TBR) contains the full timebase count value of 32 bits.

The Control/Status register 0 (TWCSR0) contains the most-significant 28 bits of the timebase count, as well as the WDT enable and status bits. The timing resolution from the upper 28 bits of the timebase count is $T_{clk} \times 16$ (T_{clk} is the period of the input clock). As a result, a single access can be used to read the state of the watchdog timer, as well as a reduced resolution version of the timebase.

An interrupt signal (Timebase Interrupt) is provided that pulses High for one clock period as the counter rolls over from 0xFFFFFFFF to 0x00000000. This interrupt can be used by the software to keep track of how many timebase rollovers have occurred.

WDT Operation

The WDT timeout interval is configured by a parameter to be $2^{\text{Width of WDT}}$ clock cycles, where width of WDT parameter is any integer from 8 to 31. The WDT interval is set at core generation time and cannot be modified dynamically through a Control register. Figure 1-3 shows the WDT state diagram.

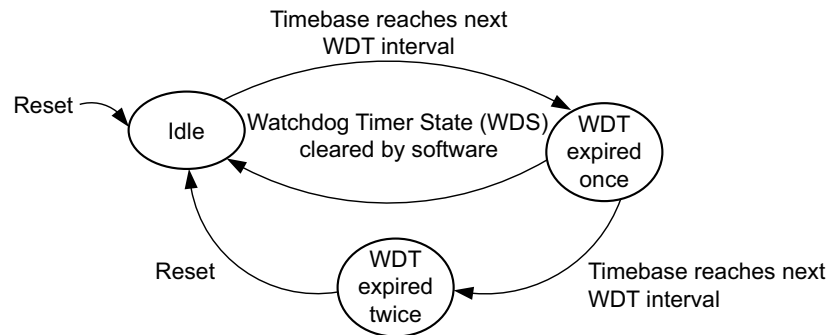


Figure 1-3: Timebase Watchdog Timer State Diagram

Feature Summary

- 32-bit slave device on a AXI4-Lite interface
- Window watchdog feature with programmable open and close windows
- Additional features for better controllability such as PSM, SST, and FC
- Watchdog timer with configurable timeout period and interrupt
- Configurable WDT enable: enable-once or enable-repeatedly
- One 32-bit free-running timebase counter with rollover interrupt
- WDT with dual-expiration architecture, where WDT reset is generated if WDT timeout occurs before previous WDT timeout interrupt is cleared by software
- Disabled only with two distinct writes, reducing the possibility of inadvertently disabling

Licensing and Ordering

This Xilinx® LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado Design Suite under the terms of the [Xilinx End User License](#).

Information about this and other Xilinx LogiCORE IP modules is available at the [Xilinx Intellectual Property](#) page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your [local Xilinx sales representative](#).

Product Specification

Performance

Performance characterization of this core was performed using margin system methodology, described in the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Note: System performance numbers for UltraScale™ architecture and Zynq®-7000 AP SoC devices are expected to be similar to 7 series device numbers.

Table 2-1: AXI Timebase Watchdog Timer System Performance

FPGA	Speed Grade	F _{Max} (MHz)
		AXI4-Lite
Virtex-7	-1	180
Kintex-7		180
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

Resource requirements for the AXI Timebase Watchdog Timer core have been estimated for 7 series and Zynq-7000 devices (Table 2-2). These values were generated using the Vivado[®] Design Suite.

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

Table 2-2: Device Utilization – 7 Series and Zynq-7000 Devices

Parameter Values		Device Resources		
		Flip-Flops	Slices	LUTs
Window WDT = 0	Enable repeatedly	94	39	105
	Enable only once	94	39	105
	Enable only once	94	40	106
Window WDT = 1	Width of second sequence timer = 8	309	108	327
	Width of second sequence timer = 31	309	120	347

Port Descriptions

The AXI Timebase Watchdog Timer I/O signals are listed and described in Table 2-3.

Table 2-3: AXI Timebase WDT I/O Signals

Signal Name	Interface	I/O	Initial State	Description
s_axi_aclk	Clock	I	–	AXI Clock
s_axi_aresetn	Reset	I	–	AXI Reset, active-Low
s_axi_*	S_AXI	I	–	AXI4-Lite Slave Interface signals. See Appendix A of the <i>Vivado AXI Reference Guide</i> (UG1037) [Ref 3] for AXI4, AXI4-Lite, and AXI Stream Signals.
wdt_reset		O	0	Window WDT Reset (active-High) In WWDT mode, after asserting reset, core stops running (auto disables, that is, WEN clears) and wait to receive input AXI reset. In Legacy WDT mode, assert on second expiration of the WDT timeout interval.
wdt_reset_pending		O	0	Active-High output. Applicable to WWDT mode only. This indicates Window WDT reset is asserted after the SST count expires. This is linked to WRP bit. This deasserts when WRP bit is cleared.

Table 2-3: AXI Timebase WDT I/O Signals (Cont'd)

Signal Name	Interface	I/O	Initial State	Description
wdt_interrupt		O	0	Active-High output. In WWDT mode this is linked to WINT bit. This deasserts when WINT bit is cleared. In Legacy WDT mode, assert High until WDS bit is cleared in TWCSR0 register.
wdt_state_vec[6:0]		O	0x50	Active-High output. Applicable to WWDT mode only. Bit[0] = wdt_on (Active-High as long as WEN bit is High) Bit[1] = wdt_wsw (Active-High. Indicates Window WDT is in second window) Bit[2] = Watchdog mode (0 = basic mode, 1 = Reserved bit (Q and A mode)) Bit[3] = Fail counter enabled Bits[6:4] = Fail counter value
freeze		I	–	Applicable to Legacy WDT mode only. Active-High. Stalls the WDT counters.

Register Space

Table 2-4 shows all the AXI Timebase Watchdog Timer registers and their addresses.

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

Table 2-4: AXI Timebase WDT Register Map

Base Address + Offset	Register Name	Access Type	Default Value (Hex)	Description
BASEADDR + 0x00 ⁽¹⁾	TWCSR0	R/W	0x00000000	Control/Status Register 0
BASEADDR + 0x04 ⁽¹⁾	TWCSR1	W ⁽²⁾	0x00000000	Control/Status Register 1, state is mirrored in TWCSR0 for read.
BASEADDR + 0x08 ⁽¹⁾	TBR	R ⁽³⁾	0x00000000	Timebase Register
BASEADDR + 0x0C ⁽⁵⁾	MWR	R/W	0x00000000	When the Enabled Window WDT is enabled, this register is used as the Master Write Control Register. When it is disabled, it is used to program the width of the watchdog timer.
BASEADDR + 0x10 ⁽⁴⁾	ESR	R/W	0x00000000	Enable and Status Register
BASEADDR + 0x14 ⁽⁴⁾	FCR	R/W	0x00000000	Function Control Register
BASEADDR + 0x18 ⁽⁴⁾	FWR	R/W	0x00000000	First Window Configuration Register
BASEADDR + 0x1C ⁽⁴⁾	SWR	R/W	0x00000000	Second Window Configuration Register
BASEADDR + 0x20 ⁽⁴⁾	TSR0	R/W	0x00000000	Task Signature Register 0
BASEADDR + 0x24 ⁽⁴⁾	TSR1	R/W	0x00000000	Task Signature Register 1
BASEADDR + 0x28 ⁽⁴⁾	STR	R	0x00000000	Second Sequence Timer Register

Notes:

1. Present when Enable Window WDT = 0. In this case, register space 0x0C to 0x30 is reserved.
2. Reading of this register returns undefined value.
3. Writing into this register has no effect.
4. Present when Enable Window WDT = 1. In this case, register space 0x00 to 0x08 is reserved.
5. Register 0xC is available in two modes. However, the functionality is different based on the mode selection.

Control/Status Register 0 (TWCSR0)

Control/Status register 0 contains the watchdog timer reset status, watchdog timer state, and watchdog timer enables. The TWCSR0 bit definitions are explained in [Table 2-5](#).

Table 2-5: Control/Status Register 0 (0x00)

Bits	Name	Reset Value	Access Type	Description
31:4	TBR	0	R	Timebase Register (Most significant 28 bits) This read-only field contains the most significant 28 bits of the timebase register. The timebase register is mirrored here so that a single read can be used to obtain the count value and the watchdog timer state if the upper 28 bits of the timebase provide sufficient timing resolution.
3	WRS	0	R/W	Watchdog Reset Status Indicates the WDT reset signal was asserted. This bit is not cleared by a system reset so that it can be read after a system reset to determine if the reset was caused by a watchdog timeout. This bit can be cleared by applying reset to the IP (asserting the <code>s_axi_aresetn</code>) followed by writing 1 to this bit. Writing a 0 to this bit has no effect. 0 = WDT reset has not occurred 1 = WDT reset has occurred
2	WDS	0	R/W	Watchdog Timer State Indicates the WDT period has expired. The <code>wdt_reset</code> signal is asserted if the WDT period expires again before this bit is cleared by software. Writing a 1 to this bit clears the watchdog timer state. Writing a 0 to this bit has no effect. 0 = WDT period has not expired 1 = WDT period has expired, reset occurs on next expiration
1	EWDT1	0	R/W	Enable Watchdog Timer (Enable 1) This bit must be used in conjunction with the EWDT2 bit in the TWCSR1 register. Both bits must be 0 to disable the WDT. 0 = Disable WDT function if EWDT2 also equals 0 1 = Enable WDT function
0	EWDT2	0	R	Enable Watchdog Timer (Enable 2) This bit is read-only and is the only place to read back a value written to Bit[0] of TWCSR1.

Control/Status Register 1 (TWCSR1)

Control/Status register 1 contains the second WDT enable bit. The WDT enable must be cleared in both TWCSR0 and TWCSR1 to disable the WDT. If the WDT is configured as enable-once, then the WDT cannot be disabled after it has been enabled. The TWCSR1 bit definitions are explained in [Table 2-6](#).

Table 2-6: Control/Status Register 1 (0x04)

Bits	Name	Reset Value	Access Type	Description
31:1	Reserved	N/A	N/A	Reserved
0	EWDT2	0	W ⁽¹⁾	Enable Watchdog Timer (Enable 2) This bit must be used in conjunction with the EWDT1 bit in the TWCSR0 register to disable the WDT. Both bits must be 0 to disable the WDT. The value of EWDT2 can be read back only in TWCSR0. 0 = Disable WDT function if EWDT1 also equals 0 1 = Enable WDT function

Notes:

1. Reading of this register returns undefined value.

Timebase Register (TBR)

The Timebase register is the output of a free-running incrementing counter that clocks at the input clock rate (no prescaling of the clock is done for this counter). This register is read-only and is reset by the following:

- A system reset
- Enabling the WDT after power on reset
- Enabling the WDT after the WDT has been disabled. EWDT1 and EWDT2 must both be zero to disable the WDT. The WDT is enabled when either EWDT1 or EWDT2 are set to one. Note that when the WDT mode is enable-once, the TBR can only be reset when the WDT is first enabled.

Table 2-7: Timebase Register (0x08)

Bits	Name	Reset Value	Access Type	Description
31:0	TBR	0	R ⁽¹⁾	Timebase Register This register indicates the free-running incrementing counter value.

Notes:

1. Writing into this register has no effect.

Master Write Control Register (MWR)

This register is applicable when the Enable Window WDT is enabled. This register is available in two modes and its functionality is different based on the mode selection.

Table 2-8: Master Write Control Register (0x0C) with Window Mode Enabled

Bits	Name	Reset Value	Access Type	Description
31:2	Reserved	0	–	Reserved
1	AEN	0	R/W*	WWDT Always Enable This bit provides extra safeguard (if needed) against unintentional clear of WEN bit. 0 = WEN bit can be cleared, depending on WDP value 1 = WEN bit cannot be cleared After this is set, this bit can be cleared only by applying reset. If required, this bit should be set before Window WDT is enabled.
0	MWC	1	R/W	Master Write Control Master write access control bit for the Window Watchdog. 0 = Window WDT register space is read only 1 = Window WDT register space is writable In basic WWDT mode, this bit auto clears when WEN is changed from 0 to 1. This bit controls the write access to the complete register space. When this bit is 0, writes to any register are ignored by all means (that is, it does not lead to good or bad event generation).

Notes:

1. This register is available when Enable Window WDT is disabled.
2. This register should be programmed before the timer is started.

Table 2-9: Master Write Control Register (0x0C) with Window Mode Disabled

Bits	Name	Reset Value	Access Type	Description
31:5	Reserved	0	–	Reserved
4:0	Width of Watchdog timer	Width of timer as configured in GUI	R/W	Program the register to change the width of watchdog timer. Valid values are 8-31. Programming any other value can result in undefined behavior

Notes:

1. This register is available when Enable Window WDT is disabled.
2. This register should be programmed before the timer is started.

Enable and Status Register (ESR)

Table 2-10: Enable and Status Register (0x10)

Bits	Name	Reset Value	Access Type	Description
31:27	Reserved	0	–	Reserved
26:24	LBE[2:0]	0	RW1C*	<p>Last Bad Event</p> <p>This status field provides the scratch bit functionality. This bit is not cleared by a system reset so that it can be read after a system reset to determine the reason for the reset generation.</p> <p>Until wdt_reset is not asserted, these bits can be cleared by writing 111.</p> <p>After wdt_reset is asserted by the core, this bit can be cleared only by applying reset to the IP (asserting AXI reset) followed by writing 111 to this field. Writing any other pattern has no effect.</p> <p>Basic Mode</p> <p>000 = No bad event</p> <p>001 = Restart kick in first window or disable attempt in first window</p> <p>010 = TSR mismatch or disable attempt in second window when FC is enabled and its value is non-zero</p> <p>011 = Second window overflow</p>
23	Reserved	0	–	Reserved
22:20	FCV[2:0]	101	RO	<p>Fail Counter Value</p> <p>Watchdog fail counter value (initialized to 5).</p> <p>A good event decrements FCV by 1 unless it is 0. A bad event increments FCV unless it is 7. If the FCV = 7 and another bad event happens, wdt_reset is generated (either immediately or after SST count rolls over).</p>
19:18	Reserved	0	–	Reserved
17	WRP	0	RW1C	<p>Watchdog Reset Pending</p> <p>This status bit asserts High when SST counter starts.</p> <p>It is a Write 1 to Clear bit. When this bit is cleared, wdt_reset_pending deasserts.</p> <p>This bit and the output wdt_reset_pending is provided to indicate that wdt_reset is asserted when SST count rolls over and can be used as an another independent interrupt from the core.</p>
16	WINT	0	RW1C	<p>Watchdog Interrupt</p> <p>This status bit asserts High in the next clock cycle, after watchdog reaches to the interrupt programmed point in second window. Restart of timer automatically clears WINT bit.</p> <p>It is a Write 1 to Clear bit. When this bit is cleared, wdt_interrupt deasserts.</p>

Table 2-10: Enable and Status Register (0x10) (Cont'd)

Bits	Name	Reset Value	Access Type	Description
15:9	Reserved	0	–	Reserved
8	WSW	0	RW1C/ RO	Window WDT in Second Window This bit is a status bit and tracks when Window WDT is in second window. 0 = Window WDT is not in Second Window 1 = Window WDT is in Second Window This bit is RW1C and is used by software to provide restart kick to Window WDT (that is, when this bit is 1, Software writes 1 to clear it and it is considered as restart kick). If software writes 1 on this bit when this bit is 0, it is considered a bad event (value of WSW does not change).
7:2	Reserved	0	–	Reserved
1	WCFG	0	RW1C	Wrong Configuration This is a status bit that indicates wrong configuration as follows. This bit gets set if second window count is set as 0 (when WEN bit is made 1).
0	WEN	0	R/W	Window WDT Enable This bit is the enable bit for Window WDT. 0 = Window WDT is disabled 1 = Window WDT is enabled Disabling watchdog in first window duration is considered a bad event and disablement is not honored. Watchdog can be disabled only in the second window duration (If fail counter is enabled, watchdog can be disabled only when fail counter is 0. If fail counter is not enabled, it can be cleared any time in second window.), If WEN clear and WSW clear events happen at same time, then the core considers only WEN. If wrong configuration is detected or wdt_reset is generated, this bit auto clears (irrespective of WDP settings).

Function Control Register (FCR)

Note: FCR register value can only be changed when WEN = 0.

Table 2-11: Function Control Register (0x14)

Bits	Name	Reset Value	Access Type	Description
31:16	Reserved	0	–	Reserved
15:8	SBC[7:0]	0	R/W	Selected Byte Count This field (along with BSS[1:0]) is used to determine the interrupt assertion point in the second window configuration. SBC[7:0] provides the count value of selected byte segment.
7:6	BSS[1:0]	0	R/W	Byte Segment Selection (of Second Window Count) This field (along with SBC[7:0]) is used to determine the interrupt assertion point in the second window configuration. BSS[1:0] provides Byte Segment selection in Second Window Count as: 00 = SW Byte0 selected (that is, SBC[7:0] are compared with SW[7:0]) 01 = SW Byte1 selected (that is, SBC[7:0] are compared with SW[15:8]) 10 = SW Byte2 selected (that is, SBC[7:0] are compared with SW[23:16]) 11 = SW Byte3 selected (that is, SBC[7:0] are compared with SW[31:24]) For example: <ul style="list-style-type: none"> • If BSS = 00 and SWC = 0xAA then WDT asserts interrupt when second window reaches 0x000000AA that is when SW[7:0] == SBC[7:0] and other bits are zeros • If BSS = 01 and SWC = 0xAA then WDT asserts interrupt when second window reaches 0x0000AA00 that is when SW[15:8] == SBC[7:0] and other bits are zeros
5	Reserved	0	–	Reserved
4	SSTE	0	R/W	Second Sequence Timer Enable This bit enables second sequence timer function. 0 = SST disabled 1 = SST enabled This option provides additional time to software by delaying the inevitable wdt_reset assertion/generation by SC count delay. SC count delay time in the STR register is controlled through GUI option. This is an independent function and can be enabled in any WWDT mode with/without other options).

Table 2-11: Function Control Register (0x14) (Cont'd)

Bits	Name	Reset Value	Access Type	Description
3	PSME	0	R/W	<p>Program Sequence Monitor Enable</p> <p>This bit enables Task Signature register comparison in basic WWDT mode.</p> <p>0 = PSM disable 1 = PSM enabled</p> <p>When this function is enabled, the core checks and compares the contents of TSR0 and TSR1 registers at the restart kick/disablement of Window WDT in second window. If they match, no effect. If they do not match, wdt_reset is generated (either immediately when SSTE is disabled or after SC count delay when SSTE is enabled).</p>
2	FCE	0	R/W	<p>Fail Counter Enable</p> <p>This bit provides option to disable the fail counter in basic WWDT mode.</p> <p>0 = Fail counter disabled 1 = Fail counter enabled</p> <p>When disabled, one bad event triggers wdt_reset generation (either immediately when SSTE is disabled or after SC count delay when SSTE is enabled).</p> <p>When fail counter is enabled, wdt_reset is generated when fail counter is 7 and another bad event happens.</p>
1	Reserved	0	–	Reserved
0	WDP	0	R/W	<p>WWDT Disable Protection</p> <p>This bit provides extra safeguard (if needed) against unintentional clear of WEN bit.</p> <p>0 = WEN bit can be cleared 1 = WEN bit cannot be cleared</p> <p>If required, this bit should be set before Window WDT is enabled.</p>

First Window Count Register (FWR)

Note: FWR register value can only be changed when WEN = 0.

Table 2-12: First Window Count Register (0x18)

Bits	Name	Reset Value	Access Type	Description
31:0	FW[31:0]	0	R/W	<p>First Window Count</p> <p>This field provides the count value for the first window and is valid for both WWDT modes.</p> <p>FW[31:0] = First Window Count value</p> <p>FW Counter is a down counter and it starts from the programmed FWR value and ends at 0.</p> <p>First window continue. When completed, it is followed by the second window.</p> <p>This field can be set as 0. In this case, it achieves “close” window absent case.</p> <p>Xilinx recommends that the minimum non-zero value should be 15 or more.</p>

Second Window Count Register (SWR)

Note: SWR register value can only be changed when WEN = 0.

Table 2-13: Second Window Count Register (0x1C)

Bits	Name	Reset Value	Access Type	Description
31:0	SW[31:0]	0	R/W	<p>Second Window Count</p> <p>This field provides the count value for the second window and is valid for both WWDT modes.</p> <p>SW[31:0] = Second Window Count value</p> <p>SW Counter is a down counter and it starts with the programmed SWR value and ends at 0.</p> <p>SW[31:0] is used to place the interrupt assertion with the help of BSS[1:0] and SBC[7:0].</p> <p>Any good or bad event ends the second window.</p> <p>Absence of a good or bad event allows the second window timeout. This is considered a bad event.</p> <p>This field cannot be set as 0. Setting this field as 0 causes “Wrong Config” status bit set which disables WWDT (by clearing WEN bit irrespective of WDP settings).</p> <p>Xilinx recommends that the minimum value should be sufficiently large to complete the required AXI4-Lite write transactions at system-level.</p>

Task Signature Register 0 (TSR0)

Table 2-14: Task Signature Register 0 (0x20)

Bits	Name	Reset Value	Access Type	Description
31:0	TSR0[31:0]	0	R/W	Task Signature 0 This register function is controlled through the PSME bit. Software writes a signature into this register. The core makes comparison with TSR1 at watchdog restart kick point. If TSR1 != TSR0 at restart point, this is considered a bad event. If TSR1 = TSR0 at restart point, this is considered a good event. The core does not track writes to this register and it makes comparison at restart time.

Task Signature Register 1 (TSR1)

Table 2-15: Task Signature Register 1 (0x24)

Bits	Name	Reset Value	Access Type	Description
31:0	TSR1[31:0]	0	R/W	Task Signature 1 This register function is controlled through the PSME bit. Software writes a signature into this register. The core makes comparison with TSR0 at watchdog restart kick point. If TSR1 != TSR0 at restart point, this is considered a bad event. If TSR1 = TSR0 at restart point, this is considered a good event. The core does not track writes to this register and it makes comparison at restart time.

Second Sequence Timer Register (STR)

Table 2-16: Second Sequence Timer Register (0x28)

Bits	Name	Reset Value	Access Type	Description
SST Count width – 1:0	SC [SST Count width – 1:0]	0	R	SST Count This is free running down counter starts from $2^{(SST\ WIDTH - 1)}$. Function of this counter is to provide extra delay before inevitable wdt_reset is generated. Width of the counter is provided as a GUI option from 8 to 31. This counter runs only if SSTE = 1.

Designing with the Core

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

Clocking

The `axi_timebase_wdt` operates on the `s_axi_aclk` clock.

Resets

AXI Timebase WDT resets on `s_axi_aresetn`, which is active-Low and synchronous to `s_axi_aclk`.

Programming Sequence for Legacy WDT Mode

The following is the programming sequence to reset the AXI Timebase WDT.

1. Set the EWDT1 in TWCSR0 register to enable Timebase WDT.
2. With the WDT period expired, writing a one to WDS bit clears the WDT timer expiry status.
3. With the WDT reset state, writing a one to WRS bit clears the watchdog reset status bit. Writing a zero to this bit has no effect.
4. Clear both the EWDT1 in TWCSR0 register and EWDT2 in TWCSR1 register to disable the Timebase WDT.

Programming Sequence for WWDT Mode

1. If required, set the WDP or AEN bit (to enable protection against accidental clearing).
2. Configure the First and Second Window Count registers as per close/open window requirements/constraints.
3. Set the interrupt position in the second window as per requirements (SBC[7:0] and BSS[1:0]).
4. If required, enable the Fail Counter (FC), Program Sequence Monitor (PSM), and Second Sequence Timer (SST) functions.
5. If PSM is enabled, write TSR0.
6. Enable the watchdog (WEN bit). This generates the first kick and starts the first window. This step (WEN 0->1) auto clears the MWC bit to make the address space read only.
7. After completing the first window, watchdog enters in the second window period and the core sets the WSW bit. Software might generate the next restart kick (or might disable the watchdog) any time after the WSW bit is set.
8. TSR1 can be written any time irrespective whether WSW is set or not (enable MWC, write TSR1, and disable MWC). TSR0 and TSR1 comparison is done at restart kick/disable event if PSM is enabled.
9. Wait for the `wdt_interrupt`.
10. Enable MWC and restart/disable watchdog as per requirement (clear WINT, WSW, or clear WINT, WSW, WDP, ad WEN).
 - a. If software attempts to restart or disable the watchdog in the first window, it is considered a bad event. Disable request is not honored.
 - b. If software does not restart or disable the watchdog before the second window rolls over, it is considered a bad event.
 - c. If PSM enabled and TSR mismatch was found at restart/disable time in the second window, it is considered a bad event. If PSM is disabled, TSR values are not compared.
 - d. If fail counter is disabled, a single bad event leads to `wdt_reset` generated. Scratch bits store the last bad event. Scratch bits can be cleared post AXI reset.
 - e. If SST is disabled, `wdt_reset` is generated immediately.
 - f. If SST is enabled, `wdt_reset` is generated after SC count expires.
 - g. If software restarts/disables the watchdog in the second window, it is considered a good event.
 - h. If fail counter is enabled, the good event decrements the fail counter by 1 unless it is 0 and the bad event increments the fail counter by 1 unless it is 7.

- i. If fail counter is 7 and a bad event occurs, then it leads to a `wdt_reset` generation based on SST.
 - j. Fail counter status can be tracked through the ESR register or through `wdt_stat_vec` output port.
11. If the watchdog is restarted, it starts with a new cycle with the first window.
12. After it is enabled in Basic mode, Window WDT can be disabled only when fail counter is zero.
13. After generating the `wdt_reset`, the watchdog stops running and the WEN bit auto clears.
14. If the core detects a wrong configuration, it auto clears the WEN. You have to clear WCFG first, then reconfigure the window sizes, and enable the WEN.
15. Latencies:
 - a. Restarting WDT takes one clock latency.
 - b. `wdt_interrupt` and `wdt_reset` assertion have one clock cycle latency.

Protocol Description

Timing Diagrams

Figure 3-1 shows the operation performed where WDS bit is cleared by the software before the second expiration occurs. Figure 3-2 shows the operation performed where WDS bit is not cleared and WDT expired twice state is reached.

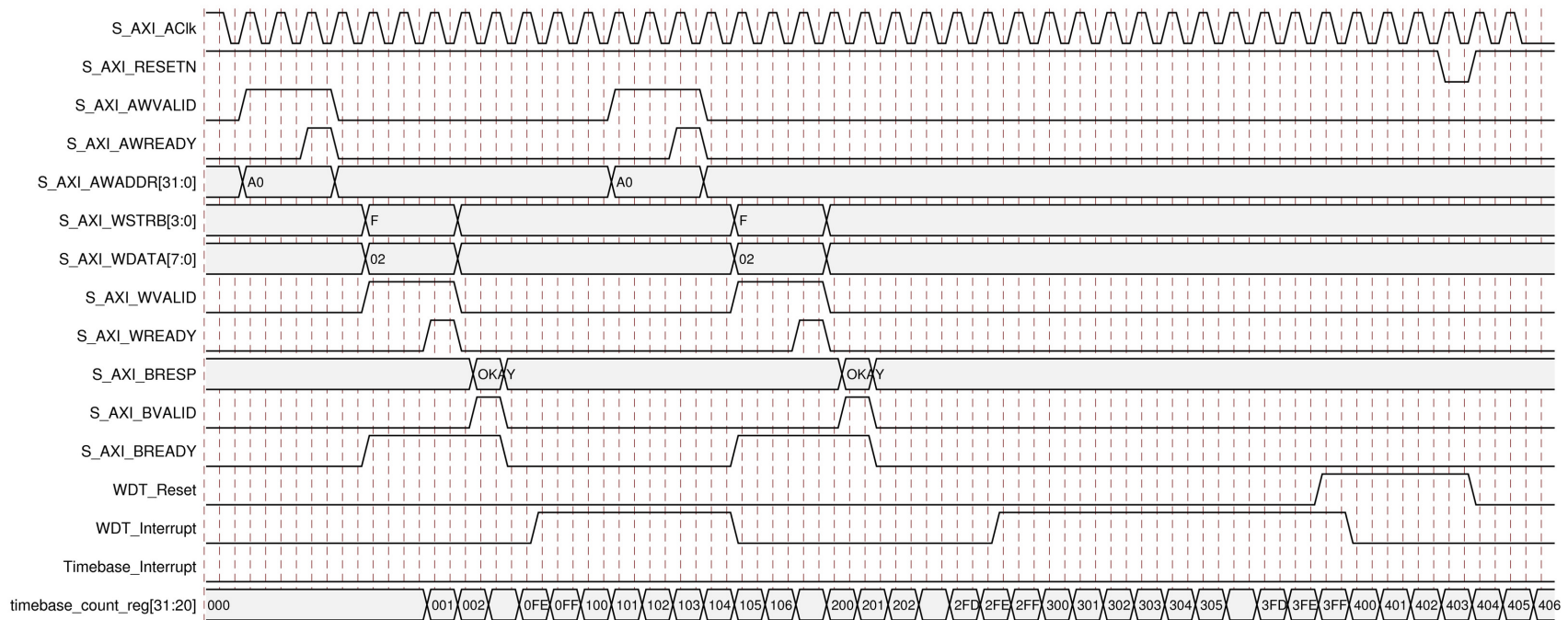


Figure 3-1: WDT Expired Once Operation Waveform

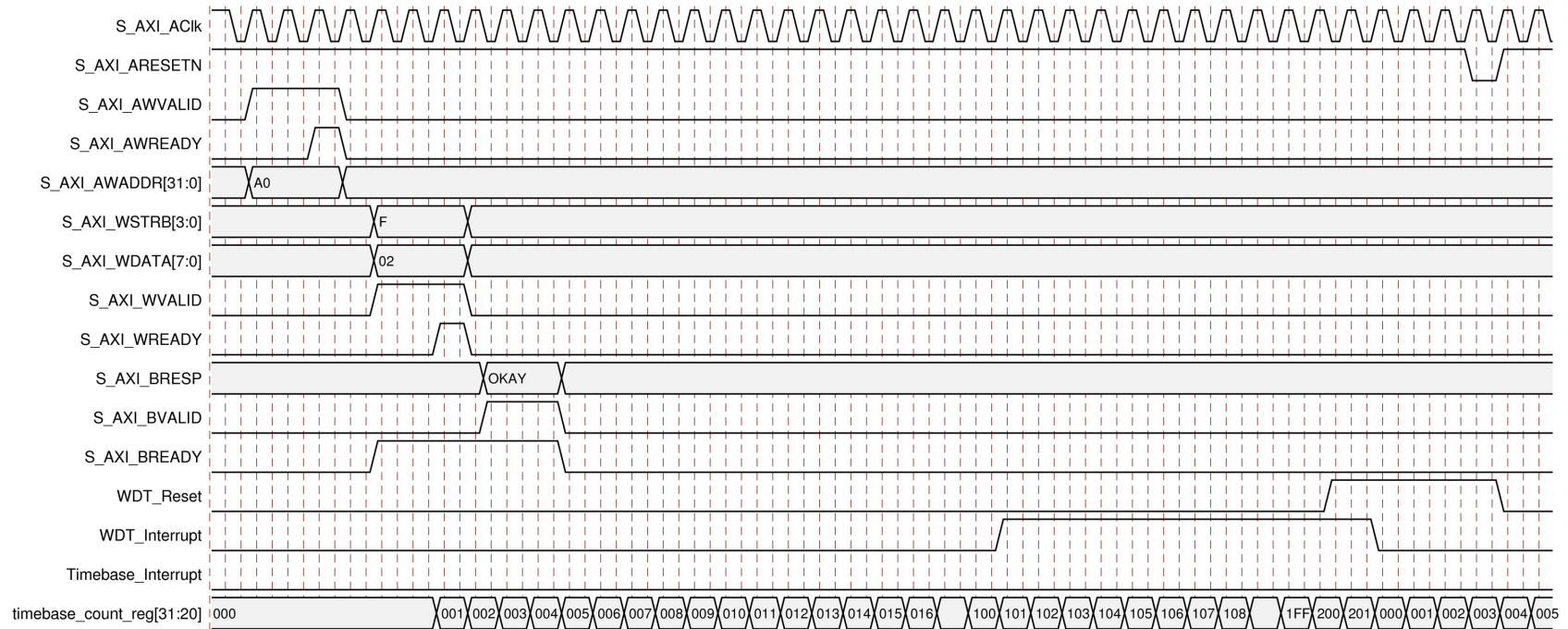


Figure 3-2: WDT Expired Twice Operation Waveform

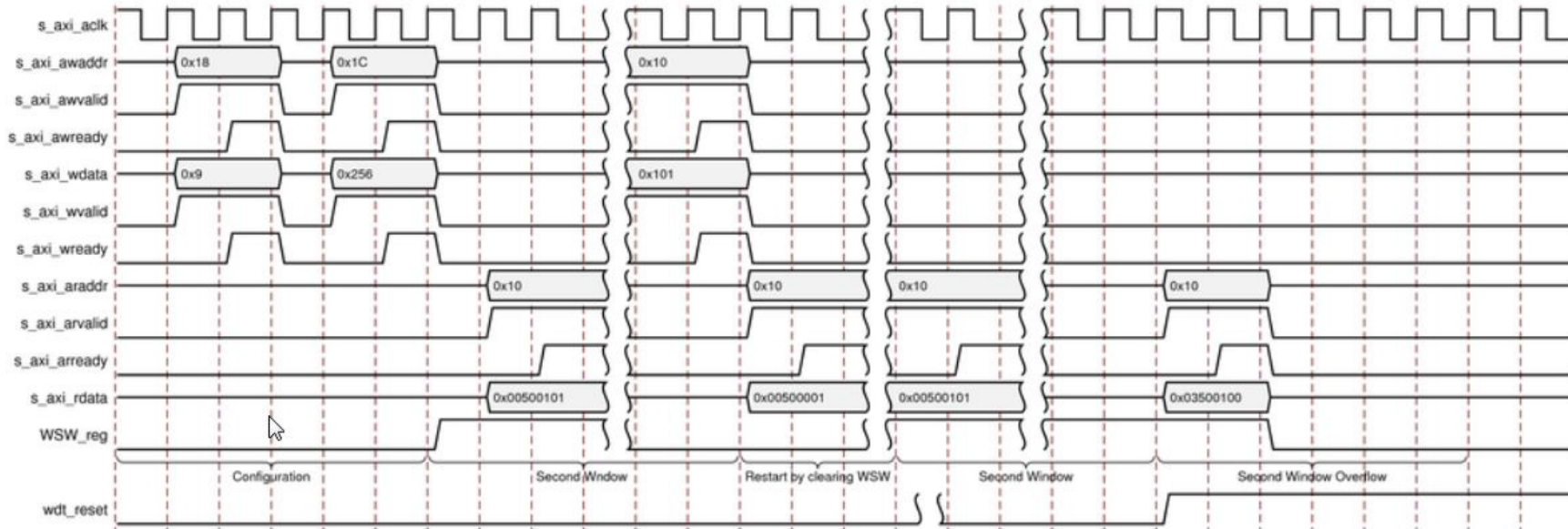


Figure 3-3: Window WDT Operation Waveform

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) [\[Ref 4\]](#)
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [\[Ref 2\]](#)
- *Vivado Design Suite User Guide: Getting Started* (UG910) [\[Ref 5\]](#)
- *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 IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [\[Ref 4\]](#) 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.

Vivado Integrated Design Environment

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) [\[Ref 2\]](#) and the *Vivado Design Suite User Guide: Getting Started* (UG910) [\[Ref 5\]](#).

Note: Figure in this chapter is an illustration of the Vivado IDE. This layout might vary from the current version.

Figure 4-1 shows the Customize IP window settings for AXI Timebase WDT IP core.

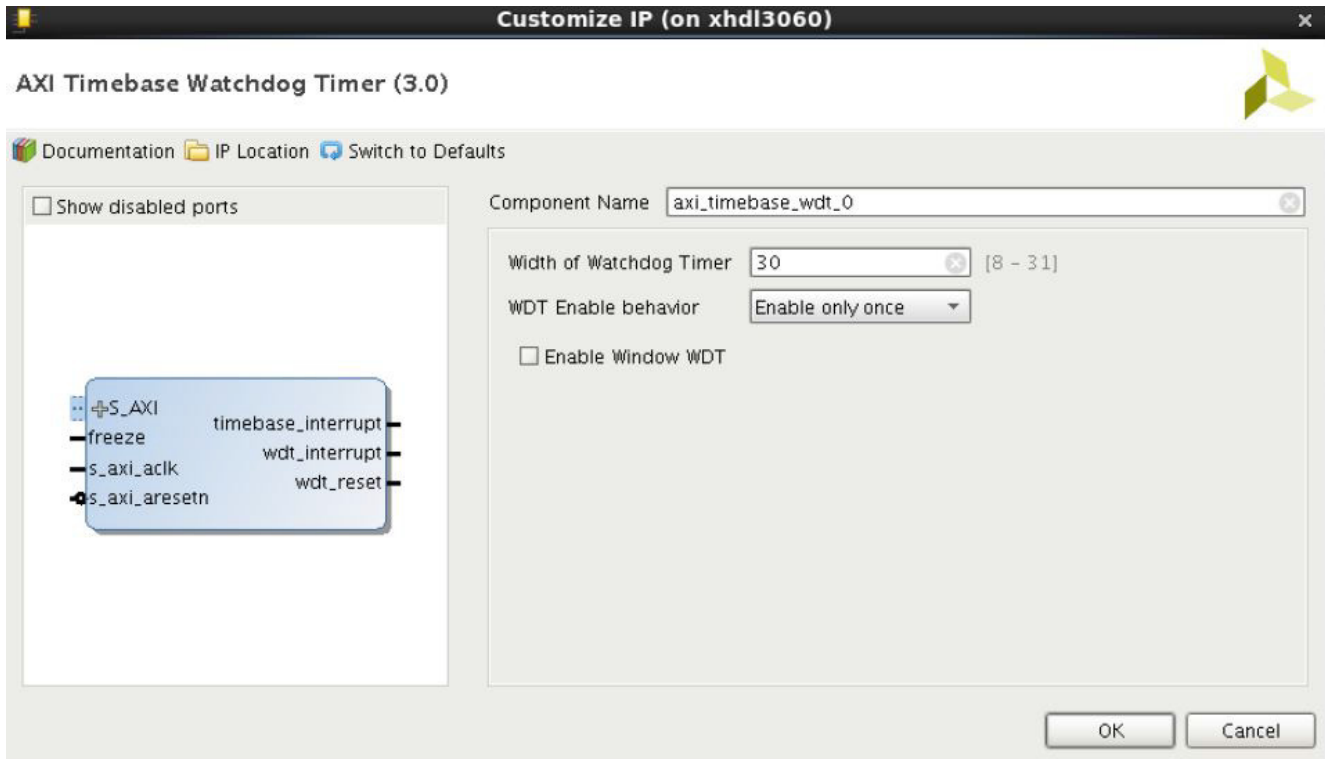


Figure 4-1: Vivado Customize IP Dialog Box

Figure 4-2 shows the Customize IP window settings with the Window WDT enabled..

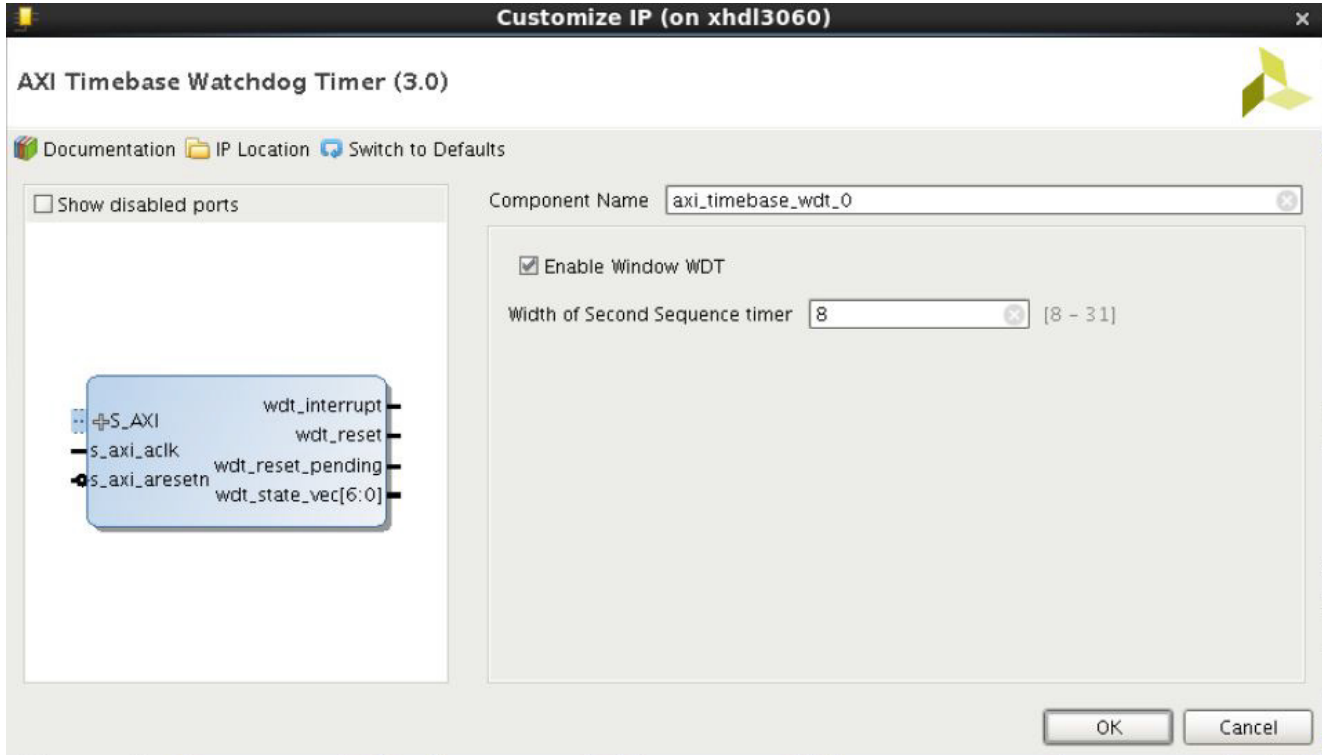


Figure 4-2: Vivado Customize IP Dialog Box

The AXI Timebase WDT Customize IP dialog box includes the following options:

- **Width of Watchdog Timer** – This indicates exponent for setting the length of the WDT interval. $WDT\ interval = 2^{C_WDT_INTERVAL} \times T_{clk}$. Values for this selection can be in range from 8 to 31. Default values is 30. This value can be changed at runtime by programming register 0xC. This is applicable to the legacy WDT mode only.
- **WDT Enable Behavior** – Enable only once = WDT can be enabled only once after reset, cannot disable it. Enable Repeatedly = WDT can be enabled/disabled by software. Default value is “Enable only once.” This is applicable to the legacy WDT mode only.
- **Enable Window WDT** – Default value is 0 with a range of 0 and 1. Default selection generates fully backward-compatible core. This is a parent parameter of all sub-features of the Window WDT.
- **Max Count Width** – This is the maximum value of the first timer. Default value is 8 with a range of 8 to 31.
- **SST Count Width** – Second sequence timer. SST count width should be \leq first counter maximum value. Default value is 8 with a range of 8 to 31.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Constraining the Core

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

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

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].

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 2].



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

Example Design

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

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

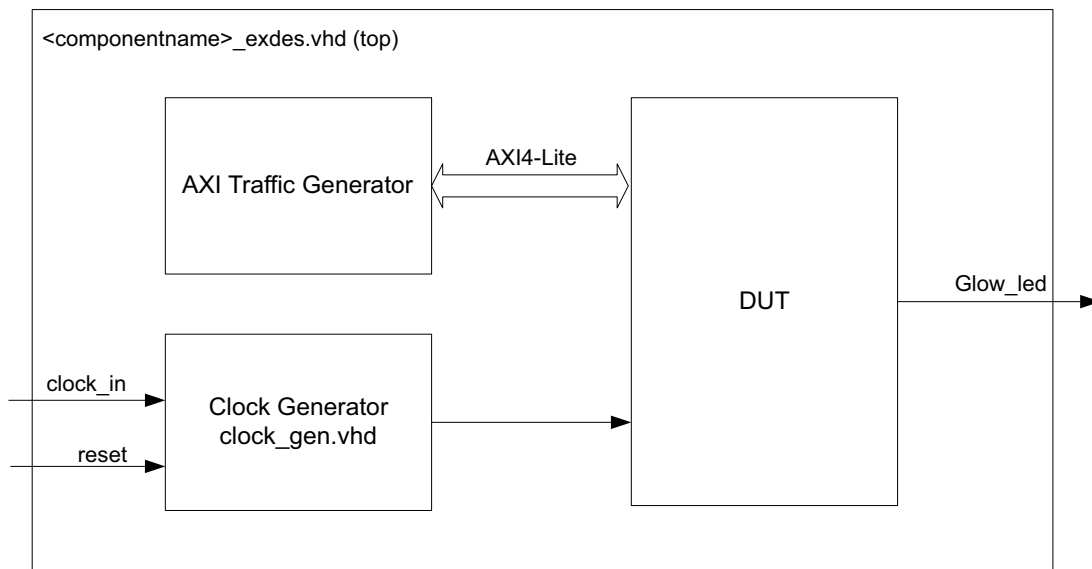


Figure 5-1: AXI Timebase WDT Example Design Block Diagram

- **Clock Generator** – MMCME2 generates the clocks for the example design. MMCME2 generates 100 MHz clock for `s_axi_aclk`. DUT and other modules of the example design are kept under reset until MMCME2 is locked.
- **AXI Traffic Generator (ATG)** – This module (IP) is configured in System Test Mode. All of the `AXI_TIMEBASE_WDT` related AXI4-Lite transactions are stored in the `coe/mif` file. For more information on AXI Traffic Generator, see *LogiCORE IP AXI Traffic Generator* (PG125) [Ref 8]. 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, page 29](#) to generate 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 it is opened in a new Vivado window.
4. In the Flow Navigator (left-side pane), click **Run Implementation** and follow the directions.

ATG writes value "2" to x"00 register and this starts the WDT timer. When timer value has reached a certain count, `Glow_led` output drives to 1.

The `Glow_led` outputs of the WDT example design is connected to the `GPIO_LED_7` LED of the KC705 board for the example design status. On successful completion of ATG write transactions to WDT registers and counters initiation, the `GPIO_LED_7` LED of KC705 would glow. In case of a failure, the `GPIO_LED_7` LED of KC705 would not glow.

Example Design Directory Structure

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

[Table 5-1](#) shows the files delivered in this `<component_name>_example/
<component_name>_example.srcs/sources_1/imports/example_design/` directory.

Table 5-1: Example Design Directory

Name	Description
<code><component_name>_exdes.vhd</code>	Top-level HDL file for the example design.
<code>clock_gen.vhd</code>	Clock generation module for example design.
<code>atg_addr.coe</code>	COE file of address. This file contains the <code>axi_timebase_wdt</code> register address.
<code>atg_data.coe</code>	COE file of data. This file contains the data to be written/read from the <code>axi_timebase_wdt</code> registers.

Table 5-1: Example Design Directory (Cont'd)

Name	Description
atg_mask.coe	COE file to mask certain reads.
atg_ctrl.coe	COE file that contains control information of ATG.

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

Table 5-2: Simulation Directory

Name	Description
<code><component_name>_exdes_tb.vhd</code>	Test bench for the example design.

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

Table 5-3: Constraints Directory

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

The XDC has all the necessary constraints needed to run the example design on the KC705 board. All I/O constraints are commented in the XDC file.



IMPORTANT: *Uncomment before implementing the design for KC705 board.*

Simulating the Example Design

Using the AXI Timebase WDT example design (delivered as part of the AXI Timebase WDT), you can quickly simulate and observe the behavior of the AXI Timebase WDT.

Setting Up the Simulation

The Xilinx[®] simulation libraries must be mapped into the simulator. If the libraries are not set for your environment, see the *Vivado Design Suite User Guide: Logic Simulation (UG900)* [Ref 6] for assistance compiling Xilinx simulation models and setting up the simulator environment. To switch simulators, click **Simulation Settings** in the Flow Navigator (left pane). In the Simulation options list, change **Target Simulator**.

Simulation Results

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

If the test passes, then the following message is displayed:

```
Test Completed Successfully
```

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

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

Test Bench

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

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

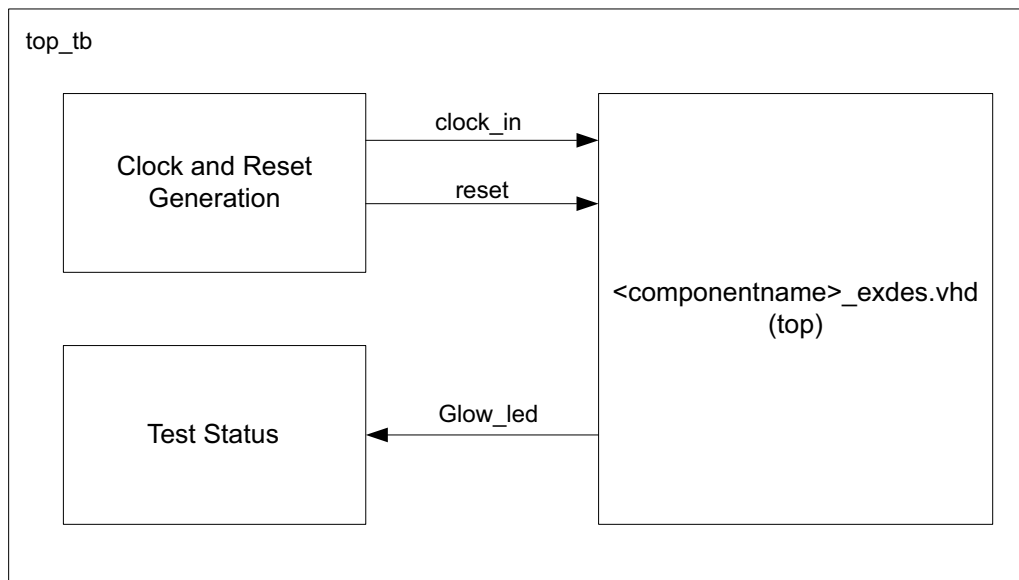


Figure 6-1: AXI Timebase WDT Example Design Test Bench

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 the *ISE to Vivado Design Suite Migration Guide* (UG911) [[Ref 7](#)].

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.

Debugging

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

Finding Help on Xilinx.com

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

Documentation

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

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

Answer Records

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

Answer Records for this core can be located by using the Search Support box on the main [Xilinx support web page](#). To maximize your search results, use proper keywords such as:

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

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

Master Answer Record for the AXI Timebase WDT

AR: [54444](#)

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 AXI Timebase WDT design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

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

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

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

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 9].

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. See [Figure 3-1](#) for a read timing diagram. 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.
- The main core clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or a Vivado Design Suite debug feature capture that the waveform is correct for accessing the AXI4-Lite interface.

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see [Xilinx Support](#).

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open the Xilinx Documentation Navigator (DocNav):

- From the Vivado® IDE, select **Help > Documentation and Tutorials**.
- On Windows, select **Start > All Programs > Xilinx Design Tools > DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx 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 the Xilinx Documentation Navigator, click the **Design Hubs View** tab.
- On the Xilinx website, see the [Design Hubs](#) page.

Note: For more information on Documentation Navigator, see the [Documentation Navigator](#) page on the Xilinx website.

References

These documents provide supplemental material useful with this product guide:

1. *LogiCORE™ IP AXI4-Lite IPIF Product Guide* ([PG155](#))
2. *Vivado® Design Suite User Guide: Designing with IP* ([UG896](#))

3. *Vivado AXI Reference Guide* ([UG1037](#))
4. *Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator* ([UG994](#))
5. *Vivado Design Suite User Guide: Getting Started* ([UG910](#))
6. *Vivado Design Suite User Guide: Logic Simulation* ([UG900](#))
7. *ISE[®] to Vivado Design Suite Migration Guide* ([UG911](#))
8. *LogiCORE IP AXI Traffic Generator Product Guide* ([PG125](#))
9. *Vivado Design Suite User Guide: Programming and Debugging* ([UG908](#))
10. *ARM[®] AMBA[®] AXI4 Protocol Version: 2.0 Specification*
<http://www.arm.com/products/system-ip/amba/amba-open-specifications.php>

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/04/2017	3.0	<ul style="list-style-type: none"> Updated the Timer window to be programmable.
10/05/2016	3.0	<ul style="list-style-type: none"> Added a note in the Register Space section under Chapter 2, Product Specification. Added the Automotive Applications Disclaimer in Please Read: Important Legal Notices.
04/06/2016	3.0	<ul style="list-style-type: none"> Added description in IP Facts section. Updated Functional Description section. Updated Features section. Updated Product Specification chapter. Added Programming Sequence for WWDT Mode section. Added Fig. 3-3: Window WDT Operation Waveform. Updated Design Flow Steps chapter.
11/18/2015	2.0	Added support for UltraScale+ families.
04/02/2014	2.0	<ul style="list-style-type: none"> Updated AXI Timebase Watchdog Timer Block Diagram. Added PG155 reference in Overview section. Updated 32-bit Timebase and WDT descriptions in Functional Description section. Updated description in Timebase Operation section. Updated Bit[0] description in Control/Status Register 0. Updated descriptions in Clocking and Resets sections.
12/18/2013	2.0	Added UltraScale support.
10/02/2013	2.0	<ul style="list-style-type: none"> Revision number advanced to 2.0 to align with core version number 2.0. Added IP Integrator. Updated Functional Description. Updated Table 2-1 AXI Timebase Watchdog Timer System Performance. Updated Performance and Resource Utilization sections. Updated Table 2-7 Control/Status Register 0 Bit[3] description. Updated Resets section. Updated Timing Diagrams section. Added Simulation, Synthesis, Example Design, and Test Bench chapters. Updated Migrating Appendix.
03/20/2013	1.0	<p>Initial Xilinx release of the product guide and replaces DS763.</p> <ul style="list-style-type: none"> Updated Table 2-1 AXI Timebase Watchdog Timer System Performance. Updated Table 2-2 to 2-4 Resource Estimates. Updated Table 2-5 AXI Timebase WDT I/O Signals.

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 <https://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 <https://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.

© Copyright 2013–2017 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.