



Watchdog Timer IP

IP Version: v1.6.1

User Guide

FPGA-IPUG-02097-1.9

June 2026

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language

This document was created consistent with Lattice Semiconductor's inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice's inclusive language [FAQ 6878](#) for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Contents	3
Abbreviations in This Document.....	6
1. Introduction	7
1.1. Overview of the IP	7
1.2. Quick Facts	7
1.3. IP Support Summary	7
1.4. Features	8
1.5. Licensing and Ordering Information	8
1.6. Minimum Device Requirements	8
1.7. Naming Conventions	8
1.7.1. Nomenclature	8
1.7.2. Signal Names.....	8
1.7.3. Attribute Names	8
2. Functional Descriptions	9
2.1. IP Architecture Overview	9
2.1.1. Watchdog Timer Modes	9
2.1.2. Reset Signals	9
2.1.3. Kick Signal	9
2.2. Clocking	10
2.3. Reset	10
2.4. User Interfaces	10
2.5. Programming Flow	11
2.5.1. Initialization	11
2.5.2. Watchdog Timer Non-APB Mode Operation	11
2.5.3. Watchdog Timer APB Mode Operation	11
2.6. Operation Details	11
3. IP Parameter Description.....	13
3.1. General.....	13
4. Signal Description	14
5. Register Description	15
5.1. WDT_MODE	15
5.2. WDT_CONTROL.....	15
5.3. TIMEOUT_VAL_1	16
5.4. TIMEOUT_VAL_2	16
5.5. STATUS_TIME	16
5.6. TIMEOUT_CNT_1.....	17
5.7. TIMEOUT_CNT_2.....	17
5.8. INT_STATUS_REG	17
5.9. INT_ENABLE_REG.....	18
5.10. INT_SET_REG.....	18
6. Example Design.....	19
6.1. Example Design Supported Configuration	19
6.2. Overview of the Example Design and Features	19
6.3. Example Design Components.....	20
6.4. Generating the Example Design	20
6.5. Simulating the Example Design	22
6.6. Hardware Testing	23
6.6.1. Hardware Testing Setup	23
6.6.2. Expected Output.....	23
7. Designing with the IP.....	24
7.1. Generating and Instantiating the IP	24
7.1.1. Generated Files and File Structure	26

7.2.	Design Implementation	26
7.3.	Timing Constraints	26
7.4.	Specifying the Strategy.....	27
7.5.	Running Functional Simulation	27
Appendix A. Resource Utilization		30
References		32
Technical Support Assistance		33
Revision History		34

Figures

Figure 2.1. Functional Block Diagram	10
Figure 2.2. Watchdog Timer Operation Timing Diagram	12
Figure 6.1. Watchdog Timer in Propel SoC Project	19
Figure 6.2. Watchdog Timer Example Design Block Diagram	20
Figure 6.3. Create Propel Project	21
Figure 6.4. Define Instance	21
Figure 6.5. Sample PDC File Content	22
Figure 6.6. Sample C Code for Watchdog Timer IP	22
Figure 6.7. Sample Waveform	23
Figure 7.1. Module/IP Block Wizard	24
Figure 7.2. IP Configuration	25
Figure 7.3. Check Generated Result	25
Figure 7.4. Simulation Wizard	27
Figure 7.5. Add and Reorder Source	28
Figure 7.6. Parse HDL Files for Simulation	28
Figure 7.7. Summary	29
Figure 7.8. Simulation Waveform	29

Tables

Table 1.1. Summary of the Watchdog Timer IP	7
Table 1.2. Watchdog Timer IP Support Readiness	7
Table 2.1. User Interfaces and Supported Protocols	10
Table 3.1. Attributes Table	13
Table 4.1. Ports Description	14
Table 5.1. Register Address Map	15
Table 5.2. Access Type Definition	15
Table 5.3. WDT_MODE	15
Table 5.4. WDT_CONTROL	15
Table 5.5. STATUS_CYCLE_1	16
Table 5.6. STATUS_CYCLE_2	16
Table 5.7. STATUS_TIME	16
Table 5.8. TIMEOUT_CNT_1	17
Table 5.9. TIMEOUT_CNT_2	17
Table 5.10. INT_STATUS_REG	17
Table 5.11. INT_ENABLE_REG	18
Table 5.12. INT_SET_REG	18
Table 6.1. Watchdog Timer IP Configuration Supported by the Example Design	19
Table 7.1. Generated File List	26
Table A.1. LFMXO5-25-9BBG400I Device Resource Utilization	30
Table A.2. LFMXO5-25-7BBG400I Device Resource Utilization	30
Table A.3. LAV-AT- E70-3LFG1156I Device Resource Utilization	31
Table A.4. LN2-CT-20-1ASG410I Device Resource Utilization	31

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
APB	Advanced Peripheral Bus
AXI	Advanced Extensible Interface
CPU	Central Processing Unit
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input/Output
HDL	Hardware Description Language
IP	Intellectual Property
LSE	Lattice Synthesis Engine
NMI	Non-Maskable Interrupt
OSC	Oscillator
PLIC	Platform-level Interrupt Controller
PLL	Phase-Locked Loop
RISC-V	Reduced Instruction Set Computer Five
RTL	Register Transfer Level
Rx	Receiver
SDK	Software Development Kit
SoC	System on Chip
TCM	Tightly-Coupled Memory
UART	Universal Asynchronous Receiver/Transmitter
WDT	Watchdog Timer

1. Introduction

1.1. Overview of the IP

The Watchdog Timer IP is designed for use as an indicator that corrective action is needed in response to a computer or processor malfunction. The design features a two-stage timer that supports non-maskable interrupt and a hard reset.

1.2. Quick Facts

Table 1.1. Summary of the Watchdog Timer IP

IP Requirements	Supported Devices	All
	IP Changes ¹	Refer to the Watchdog Timer IP Release Notes (FPGA-RN-02024) .
Resource Utilization	Supported User Interface	Advanced Peripheral Bus (APB)
	Resources	Refer to Appendix A. Resource Utilization .
Design Tool Support	Lattice Implementation	IP v1.6.1 – Lattice Radiant™ software 2026.1 and Lattice Propel™ Builder software 2026.1
	Synthesis	Lattice Synthesis Engine (LSE) Synopsys Synplify Pro® for Lattice
	Simulation	For a list of supported simulators, refer to the Lattice Radiant Software User Guide .

Note:

1. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

1.3. IP Support Summary

Table 1.2. Watchdog Timer IP Support Readiness

Lattice Device Family	Enable APB	Mode	Timer Stage 1	Timer Stage 2	Freq. Unit (Mode = Time)	Sys Clk Freq. (Mode = Time)	Radiant Timing Model
Avant™	False	Cycle	5000000	5000000	—	—	Preliminary
		Cycle	75000000	75000000	—	—	Preliminary
		Time	250	250	MHz	150	Preliminary
	True	Cycle	5000000	5000000	—	—	Preliminary
		Time	500	500	MHz	150	Preliminary
Certus™-NX, CertusPro™-NX, CrossLink™-NX, MachXO5™-NX, Certus-N2	False	Cycle	5000000	5000000	—	—	Preliminary
		Cycle	75000000	75000000	—	—	Preliminary
		Time	250	250	MHz	150	Preliminary
	True	Cycle	5000000	5000000	—	—	Preliminary
		Time	500	500	MHz	150	Preliminary

The example design for the Watchdog Timer IP allows for simulation and deployment to development boards for testing. Refer to the [Example Design](#) section for more details on how to run the example design on hardware and simulation

1.4. Features

Key features of the Watchdog Timer IP include:

- Seamlessly works with sleep mode.
- Programmable timer spec 50–500 ms (10 ms increments).
- Two-stage timer that supports warm boot and cold boot resets.
- You can specify in cycles or time.
- Supports trigger input, reset, and NMI output.
- Supports APB interface.

1.5. Licensing and Ordering Information

The Watchdog Timer IP is provided at no additional cost with the Lattice Radiant software.

1.6. Minimum Device Requirements

There is no limitation in device speed grade for Watchdog Timer IP. See [Appendix A. Resource Utilization](#) for minimum required resources to instantiate this IP and maximum clock frequency supported.

1.7. Naming Conventions

1.7.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.7.2. Signal Names

Signal Names that end with:

- `_n` are active low
- `_i` are input signals
- `_o` are output signals

1.7.3. Attribute Names

Attribute names in this document are formatted in title case and italicized (*Attribute Name*).

2. Functional Descriptions

2.1. IP Architecture Overview

The Watchdog Timer IP performs the following functions:

- Send a non-maskable interrupt to the processor (`nmi_o`);
- Send a reset to the processor for system restart (`reset_o`).

The Watchdog Timer IP is a two-stage timer that starts to count depending on the status of the processor. The processor regularly sends a kick signal to the Watchdog Timer to indicate its active and *error-free* status, the absence of the kick signal means there is a hardware malfunction or program error in the processor. The processor is required to have custom application to send the kick signal regularly.

The Watchdog Timer IP uses an up-counter that counts from zero to a timeout value at a rate that is determined by the configurable mode and the clock frequency.

2.1.1. Watchdog Timer Modes

Cycle

When in cycle mode, you can configure the timeout in terms of the number of cycles for each stage of the timer. The number of cycles refers to the count the Watchdog Timer needs to reach, to assert the timeout signal.

Time

Using the time mode, you can configure the timeout in milliseconds and the clock frequency ranging from 10 kHz to 150 MHz.

2.1.2. Reset Signals

2.1.2.1. Non-APB Mode

Warm Boot (`nmi_o`)

The warm boot reset is a non-maskable interrupt which serves as a warning to the processor that a system reset is expected. When this port is asserted, the processor must either debug the program error or store the important files in running programs.

Cold Boot (`reset_o`)

The cold boot or the hard reset, as the name implies, resets the whole system when the second timeout value has been reached.

2.1.2.2. APB Mode

Interrupt Signal (`int_o`)

Interrupt signal serves as a timeout indicator. When `int_o` is set to 1, this implies that either the stage 1 or stage 2 timer has reached the timeout value that you have set. By this time, the processor must either debug the program error or store the important files in running programs.

Cold Boot (`reset_o`)

The cold boot or the hard reset, as the name implies, resets the whole system when the second timeout value has been reached.

2.1.3. Kick Signal

For APB Mode

The IP supports the use of APB interface. If APB mode is enabled, you must send the kick signal through the APB register (`WDT_CONTROL`), indicating that the processor is still active and in *error-free* status. This requires the processor to send a kick signal regularly.

For Non-APB Mode

If APB mode is not used in the operation, you must send the kick signal through the kick_i port, indicating that the processor is still active and in *error-free* status. This requires the processor to send a kick signal regularly through a standard GPIO port.

Figure 2.1 shows a functional diagram for the Watchdog Timer IP.

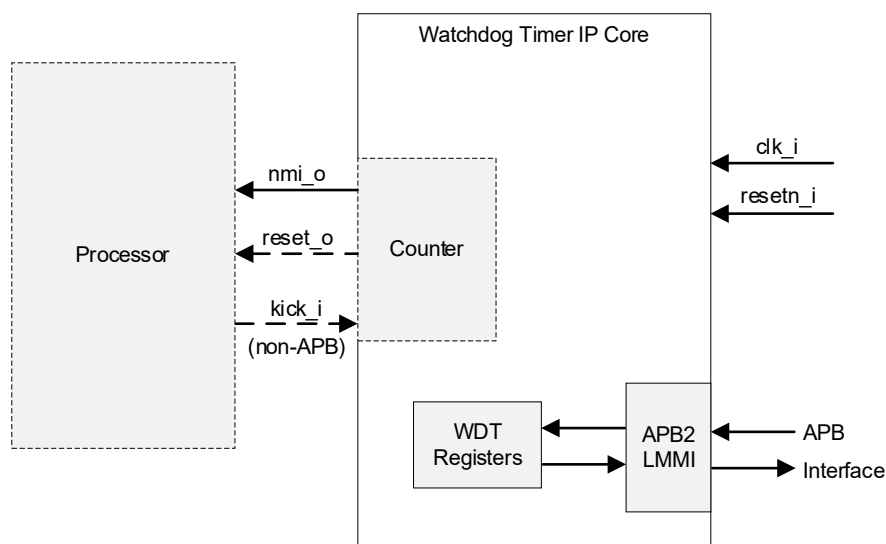


Figure 2.1. Functional Block Diagram

2.2. Clocking

The Watchdog Timer IP requires a system clock: clk_i. You need to provide this clock through an external source and route it to an appropriate pin on the FPGA device. The system clock (externally sourced) provides a timing reference and is used to operate the internal logic of the IP.

This IP supports two different frequency unit (kHz and MHz) ranging from 10 kHz to 150 MHz in the time mode.

2.3. Reset

The Watchdog Timer IP has one asynchronous active low reset: resetrn_i.

When asserted, the output ports and registers are forced to their default values. The reset assertion can be asynchronous but reset negation should be synchronous. The IP contains internal logic to synchronously de-assert the internal reset once resetrn_i is de-asserted, so you do not need to implement your own de-assertion logic.

2.4. User Interfaces

Table 2.1. User Interfaces and Supported Protocols

User Interface	Supported Protocols	Description
General – Watchdog Timer Register Access	APB	For APB interface, refer to the AMBA 3 APB Protocol v1.0 Specification for information and timing diagram of the APB interface.

2.5. Programming Flow

2.5.1. Initialization

Initial values for all registers come from the user interface. To change the default configuration, the following Watchdog Timer register should be set properly before programming the absence of a kick signal.

- INT_ENABLE_REG- Set 'h0 to disable reset interrupt, 'h1 to enable reset interrupt.

2.5.2. Watchdog Timer Non-APB Mode Operation

Below are the recommended steps for performing the Watchdog Timer operation when APB mode is disabled, this assumes that the module is not currently performing any operation or is in IDLE mode.

1. Power up.
2. Reset de-assertion.
3. Timer counter stage 1 starts counting up.
4. If a processor is active, write 1 to kick_i to indicate that it is *alive*.
5. If nmi_o is asserted, this means that the processor did not write to kick for a longer time than the timeout 1 value. This will trigger the start of timer counter stage 2. Do step 4 if processor is active, else proceed to step 6.
6. If reset_o is asserted, this means that the processor did not write to kick for a longer time than the (timeout 1 + timeout 2) value.

2.5.3. Watchdog Timer APB Mode Operation

Below are the recommended steps for performing Watchdog Timer operation when APB mode is enabled, this assumes that the module is not currently performing any operation or is in IDLE mode.

1. Power up.
2. Reset de-assertion.
3. Timer counter stage 1 starts counting up. This can be monitored through the TIMEOUT_CNT_1 register.
4. You must monitor int_o by this time.
5. If a processor is active, write 1 to kick with the WDT_CONTROL register.
6. If the processor does not write to kick for a longer time than the timeout 1 value, this sets the INT_STATUS_REG (int_nmi_status) to 1 and interrupt status (int_o) to HIGH. Clear the interrupt status (int_o) by writing 1 to int_nmi_status. Moreover, this triggers the start of timer counter stage 2. Do step 5.
7. Timer counter stage 2 starts counting up. The Timer Counter 2 can be monitored through the TIMEOUT_CNT_2 register.
8. If reset_o is asserted, this means that the processor did not write to kick for a longer time than the (timeout 1 + timeout 2) value. Also, this sets the INT_STATUS_REG (int_reset_status) to 1 and interrupt status (int_o) to HIGH. Clear the interrupt status (int_o) by writing 1 to int_reset_status and int_nmi_status registers.

2.6. Operation Details

During normal operation, the connected processor regularly sends a *kick* signal to the Watchdog Timer using kick_i port. After IP reset, timer stage 1 begins to count. Timer stage 1 clears once the processor sends a *kick* signal. However, if Timer Stage 1 reaches the timeout value this asserts the nmi_o port. This port triggers the counting of Timer Stage 2 and simultaneously notifies the processor by means of the warm boot (*nmi_o*) that a reset is imminent. Timer Stage 2 continuous to count, unless the processor sends a *kick* signal which clears both Timer Stage 1 and 2. If the Timer Stage 2 reaches the timeout value, cold boot (*reset_o*) is asserted, which forcefully restart the processor. Refer to [Figure 2.2](#) for a sample Watchdog Timer operation timing diagram.

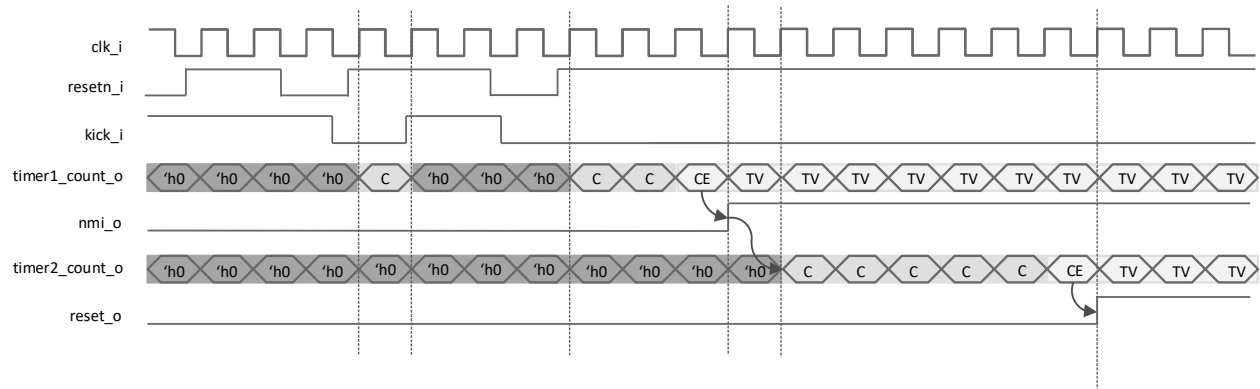


Figure 2.2. Watchdog Timer Operation Timing Diagram

3. IP Parameter Description

The configurable attributes of the Watchdog Timer IP are shown in the following tables. You can configure the IP by setting the attributes accordingly in the IP Module/IP Block Wizard of the Lattice Radiant Software. Wherever applicable, default values are in **bold**.

3.1. General

Table 3.1. Attributes Table

Attribute	Selectable Values	Description
General		
Enable APB	Checked, Unchecked	Specifies the use of APB Interface to access the available Watchdog Timer Registers.
Mode	Cycle , Time	Specifies the Watchdog Timer Mode.
Frequency Unit	kHz, MHz	Specifies the unit for <i>System Clock Frequency</i> . Active if <i>Watchdog Timer Mode</i> == Time.
System Clock Frequency	kHz: 10–999, 100 MHz: 1–150, 100	Specifies the user clock frequency. Active if <i>Watchdog Timer Mode</i> == Time, else uses the user testbench clock frequency. Selectable values are dependent on the <i>Frequency Unit</i> .
Timer Stage 1		
Timeout (number of cycles)	500–75000000, 5000000	Specifies the timeout of Timer Stage 1 in terms of number of cycles. Active if <i>Watchdog Timer Mode</i> == Cycle.
Timeout (ms)	50 –500	Specifies the timeout of Timer Stage 1 in terms of ms. Active if <i>Watchdog Timer Mode</i> == Time.
Timer Stage 2		
Timeout (number of cycles)	500–75000000, 5000000	Specifies the timeout of Timer Stage 2 in terms of number of cycles. Active if <i>Watchdog Timer Mode</i> == Cycle.
Timeout (ms)	50 –500	Specifies the timeout of Timer Stage 2 in terms of ms. Active if <i>Watchdog Timer Mode</i> == Time.

4. Signal Description

This section describes the Watchdog Timer IP signals and ports along with their descriptions.

Table 4.1. Ports Description

Port Name	Type	Width	Description
Clock and Reset			
clk_i	Input	1	System clock.
resetn_i	Input	1	Active low reset.
reset_o	Output	1	Cold boot output reset.
Interrupt Port			
nmi_o ¹	Output	1	Non-maskable interrupt signal.
int_o ²	Output	1	Bus for reset_o and nmi_o. Indicates if either reset_o and/or nmi_o is high.
Timer Count Ports¹			
timer1_count_o	Output	27	Timer count for timer stage 1.
timer2_count_o	Output	27	Timer count for timer stage 2.
Kick¹			
kick_i	Input	1	Kick signal indicating that the processor is active and <i>error-free</i> .
APB Ports²			
apb_psel_i	Input	1	APB Select Signal. Indicates that the device is selected, and a data transfer is required.
apb_penable_i	Input	1	APB Enable Signal. Indicates the second and subsequent cycles of an APB transfer.
apb_paddr_i	Input	8	APB Address signal.
apb_pwdata_i	Input	32	APB Write Data signal.
apb_pwrite_i	Input	1	APB Direction Signal. Indicates write or read access. 0 – Read 1 – Write
apb_prdata_o	Output	32	APB Read Data signal.
apb_pready_o	Output	1	APB Ready Signal. Indicates transfer completion. The IP uses this signal to extend an APB transfer.
apb_pslverr_o	Output	1	APB Error signal. Indicates a transfer failure.

Notes:

1. These ports are available only if *Enable APB* is *Unchecked*.
2. These ports are available only if *Enable APB* is *Checked*.

5. Register Description

This section provides detailed descriptions of the Watchdog Timer IP registers.

The register address map, shown in [Table 5.1](#), specifies the available registers.

Table 5.1. Register Address Map

APB Offset	Register Name	Bit Width	Access Type	Description
0x00	WDT_MODE	32	RO	Watchdog Mode Register
0x04	WDT_CONTROL	32	RW	Watchdog Timer Control Register
0x08	TIMEOUT_VAL_1	32	RO	Timeout Value Status Register 1 (Cycle)
0x0C	TIMEOUT_VAL_2	32	RO	Timeout Value Status Register 2 (Cycle)
0x10	STATUS_TIME	32	RO	Timeout Value Status Register (Time)
0x14	TIMEOUT_CNT_1	32	RO	Timeout Counter Register 1
0x18	TIMEOUT_CNT_2	32	RO	Timeout Counter Register 2
0x1C	INT_STATUS_REG	32	RW1C	Interrupt Status Register
0x20	INT_ENABLE_REG	32	RW	Interrupt Enable Register
0x24	INT_SET_REG	32	WO	Interrupt Set Register

The behavior of registers to write and read access is defined by its access type, which is defined in [Table 5.2](#).

Table 5.2. Access Type Definition

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value.	Ignores write access.
WO	Returns 0.	Updates register value.
RW	Returns register value.	Updates register value.
RSVD	Ignore when read.	Ignores write access.
RW1C	Returns register value.	Writing 1'b1 on the register bit clears the bit to 1'b0. Writing 1'b0 on the register bit is ignored.

5.1. WDT_MODE

Table 5.3. WDT_MODE

Field	Name	Description	Access	Width	Reset
[31:1]	Reserved	—	RSVD	31	—
[0]	mode	Watchdog Timer Mode. Specifies the mode of the Watchdog Timer stages. 1'b0 – Cycle. 1'b1 – Time.	RO	1	Mode

5.2. WDT_CONTROL

Table 5.4. WDT_CONTROL

Field	Name	Description	Access	Width	Reset
[31:1]	Reserved	—	RSVD	30	—
[0]	kick	Watchdog Timer Kick Register. Specifies the external processor status connected to the Watchdog Timer. This register automatically clears after getting a value. 1'b0 – This triggers the timer to count up.	WO	1	'h0

Field	Name	Description	Access	Width	Reset
		1'b1 – Indicates that the external processor is active and in normal operation. This also resets the Timeout counter Stage 1 and/or Stage 2. This register is automatically cleared and returns to default value 0 after writing 1.			

5.3. TIMEOUT_VAL_1

Table 5.5. STATUS_CYCLE_1

Field	Name	Description	Access	Width	Reset
[31:27]	Reserved	—	RSVD	5	—
[26:0]	cycle_timeoutval_st1	Timeout Value Stage 1. Specifies the timeout value of Timer Stage 1 in Cycle Mode.	RO	27	Timeout (number of cycles)

5.4. TIMEOUT_VAL_2

Table 5.6. STATUS_CYCLE_2

Field	Name	Description	Access	Width	Reset
[31:27]	Reserved	—	RSVD	5	—
[26:0]	cycle_timeoutval_st2	Timeout Value Stage 2. Specifies the timeout value of Timer Stage 2 in Cycle Mode.	RO	27	Timeout (number of cycles)

5.5. STATUS_TIME

Table 5.7. STATUS_TIME

Field	Name	Description	Access	Width	Reset
[31:23]	Reserved	—	RSVD	9	—
[22:13]	clk_freq	Clock Frequency. Specifies the System Clock Frequency of the Watchdog Timer, allowed values are 10–999 kHz and 1–150 MHz.	RO	10	System Clock Frequency
[12]	freq_unit	Clock Frequency Unit. Specifies the Frequency unit of the clk_freq register field. 1'b0 – MHz 1'b1 – kHz	RO	1	Frequency Unit
[11:6]	time_timeoutval_st2	Timeout Value Stage 2. Specifies the timeout value of Timer Stage 2 in Time Mode 'b0000 0000 – 50 ms 'b0000 0001 – 60 ms 'b0000 0010 – 70 ms ... 'b0010 1100 – 490 ms 'b0010 1101 – 500 ms	RO	6	Timeout (ms)

Field	Name	Description	Access	Width	Reset
[5:0]	time_timeoutval_st1	Timeout Value Stage 1. Specifies the timeout value of Timer Stage 1 in Time Mode 'b0000 0000 – 50 ms 'b0000 0001 – 60 ms 'b0000 0010 – 70 ms ... 'b0010 1100 – 490 ms 'b0010 1101 – 500 ms	RO	6	Timeout (ms)

5.6. TIMEOUT_CNT_1

Table 5.8. TIMEOUT_CNT_1

Field	Name	Description	Access	Width	Reset
[31:27]	Reserved	—	RSVD	5	—
[26:0]	timer1_count	Timeout Counter Stage 1. Specifies the 27-bit counter that is incremented by one on the rising edge of the input clock. The counter is reset to zero when the kick is asserted.	RO	27	'h0

5.7. TIMEOUT_CNT_2

Table 5.9. TIMEOUT_CNT_2

Field	Name	Description	Access	Width	Reset
[31:27]	Reserved	—	RSVD	5	—
[26:0]	timer2_count	Timeout Counter Stage 2. Specifies the 27-bit counter that is incremented by one on the rising edge of the input clock. The counter is reset to zero when the kick is asserted.	RO	27	'h0

5.8. INT_STATUS_REG

Table 5.10. INT_STATUS_REG

Field	Name	Description	Access	Width	Reset
[31:2]	Reserved	—	RSVD	30	—
[1]	int_reset_status	Cold boot reset Interrupt Status Register. Specifies the reset_o interrupt status of the Watchdog Timer. 1'b0 – No interrupt. 1'b1 – This indicates that the processor has not written to kick register for some time and the timeout counter stage 2 has reached the specified timeout value/limit.	RW1C	1	'h0
[0]	int_nmi_status	Warm boot reset Interrupt Status Register. Specifies the nmi_o interrupt status of the Watchdog Timer. Note that this interrupt is non-maskable. 1'b0 – No interrupt.	RW1C	1	'h0

Field	Name	Description	Access	Width	Reset
		1'b1 – This indicates that the processor has not written to kick register for some time and the timeout counter stage 1 has reached the specified timeout value/limit.			

5.9. INT_ENABLE_REG

Table 5.11. INT_ENABLE_REG

Field	Name	Description	Access	Width	Reset
[31:2]	Reserved	—	RSVD	30	—
[1]	int_reset_en	Cold boot reset Interrupt Enable Register. When set to high, it enables the corresponding interrupt status signal (int_reset_status) to trigger the assertion of int_o port. 1'b0 – Interrupt disabled. 1'b1 – Interrupt enabled.	RW	1	'h0
[0]	Reserved	—	RSVD	1	—

5.10. INT_SET_REG

Table 5.12. INT_SET_REG

Field	Name	Description	Access	Width	Reset
[31:2]	Reserved	—	RSVD	30	—
[1]	int_reset_set	Cold boot reset Interrupt Set Register. You can set the bits in the int_reset_status register. This is intended for debugging and testing purposes, making sure int_o asserts if int_reset_status is set to 1. 1'b0 – Does nothing. 1'b1 – Sets the corresponding bits in the int_reset_status register.	WO	1	'h0
[0]	int_nmi_set	Warm boot reset Interrupt Set Register. You can set the bits in the int_nmi_status register. This is intended for debugging and testing purposes, making sure int_o asserts if int_nmi_status is set to 1. 1'b0 – Does nothing. 1'b1 – Sets the corresponding bits in the int_nmi_status register.	WO	1	'h0

6. Example Design

The Watchdog Timer IP example design allows you to compile, simulate, and test the Watchdog Timer IP on the following Lattice evaluation boards:

- LAV-E70-EVN-ES1

6.1. Example Design Supported Configuration

Table 6.1. Watchdog Timer IP Configuration Supported by the Example Design

Watchdog Timer IP GUI Parameter	Value
Enable APB	True
Mode	Cycle, Time
Frequency Unit (Time Mode)	MHz, kHz
System Clock Frequency (Time Mode)	1–150 (MHz Freq Unit), 10–999 (kHz Freq Unit)
Timer Stage 1	500–7.5e+07 (Cycle Mode), 50–500 (Time Mode)
Timer Stage 2	500–7.5e+07 (Cycle Mode), 50–500 (Time Mode)

6.2. Overview of the Example Design and Features

The example design discussed in this section is created using the RISC-V RX SoC Project template in the Lattice Propel Design Environment. The generated project includes the following components:

- Processor – RISC-V RX with TCMs and PLIC
- GPIO
- OSC and PLL
- UART – Serial port
- AXI Interconnect
- AXI to APB Bridge
- Glue Logic
- System Memory

Watchdog Timer IP is instantiated to the RISC-V SoC project template as shown in the figure below.

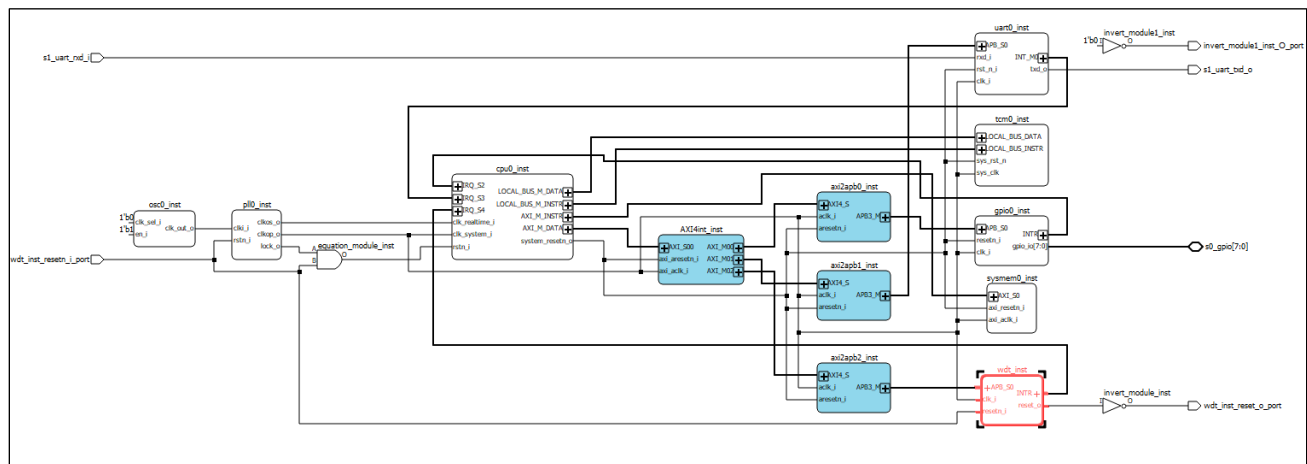


Figure 6.1. Watchdog Timer in Propel SoC Project

6.3. Example Design Components

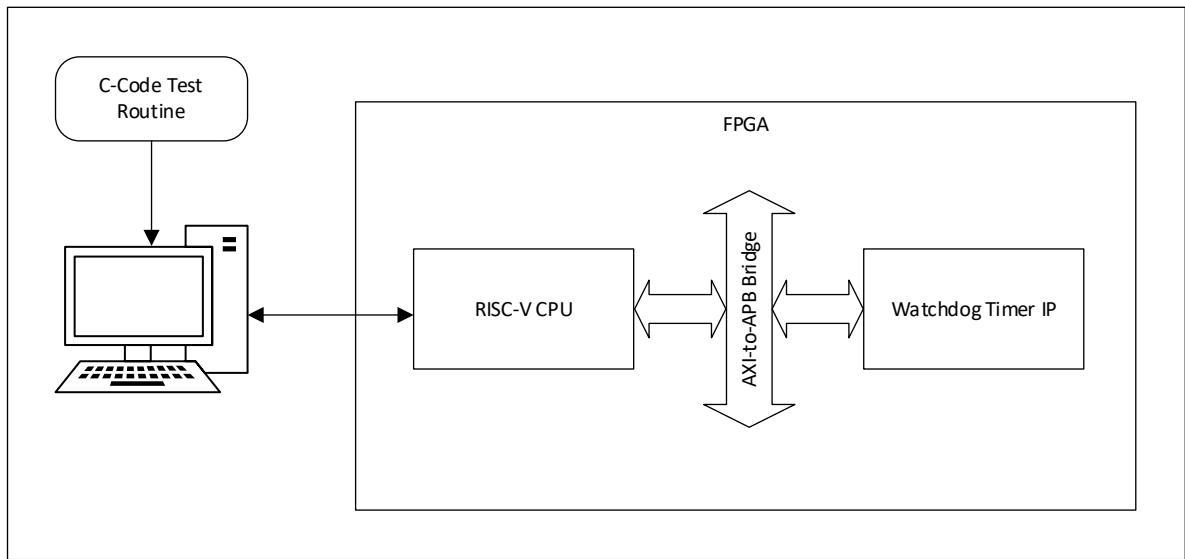


Figure 6.2. Watchdog Timer Example Design Block Diagram

The Watchdog Timer example design includes the following blocks:

- RISC-V CPU – Passes the C Code Test Routine through the APB Bridge and handles interrupts.
- AXI-to-APB Bridge – Read and write transfer on the AXI4 bus to APB.
- Watchdog Timer IP

6.4. Generating the Example Design

Refer to the [Lattice Propel SDK User Guide](#) for more details on the Lattice Propel software.

1. Launch Lattice Propel Builder software.
2. In the Propel Builder software, create a new Lattice Propel System Design Project by clicking the drop down icon and select *New Design*.
3. The Create System Design window opens.
 - For Project Type, select *SoC Project*. Enter a name and a location for the project. Click **Next**.
 - In Configure Propel Project window, refer to [Figure 6.3](#) and configure the settings based on the device or board that you use. [Figure 6.3](#) shows the configuration used when [LAV-AT-E70ES1](#) Development Board is used for the hardware testing. Click **Next**.
 - There will be a summary of the project configuration in the Project Information window. Click **Finish**.

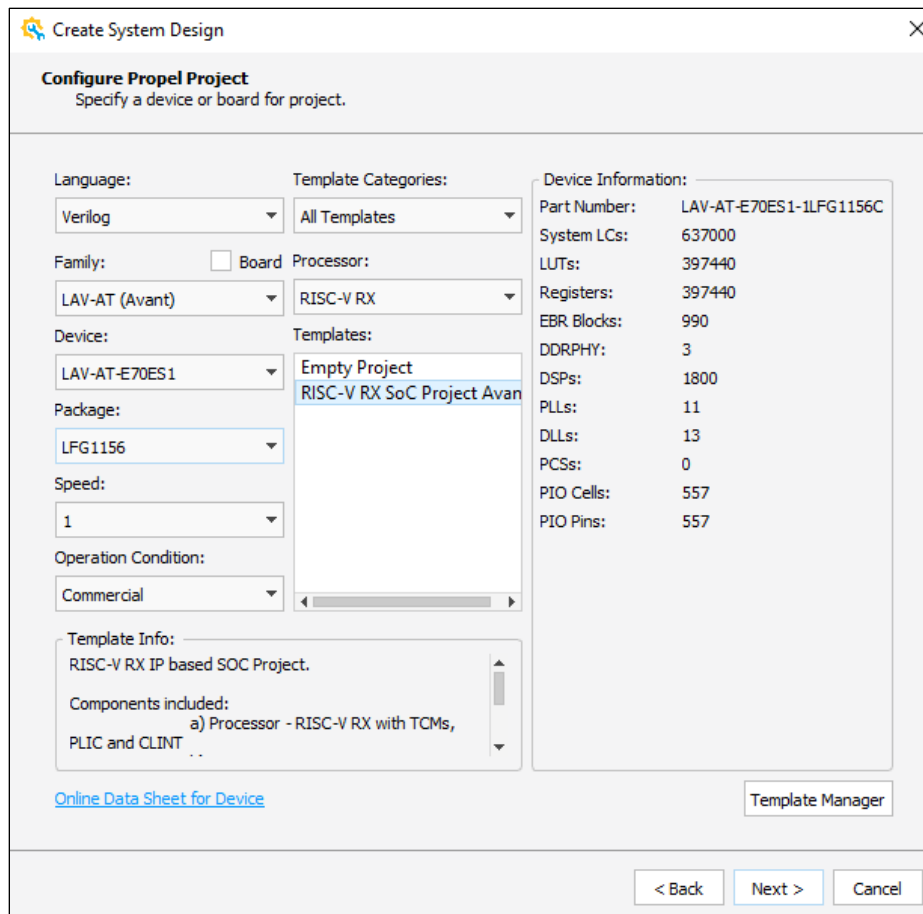


Figure 6.3. Create Propel Project

- From the **IP Catalog > IP on Server**, download the Watchdog Timer IP. Then, generate the IP. Refer to [Generating and Instantiating the IP](#) section for more details.
- After generating the Watchdog Timer IP, the **Define Instance** window opens. Modify the name if needed, then click **OK**.

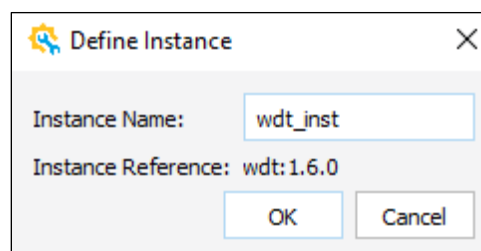


Figure 6.4. Define Instance

- Connect the instantiated IP to the system. Refer to [Figure 6.1](#) for the connections used in this IP. You need to update/add other components of the system for the clock and reset source, interrupts, and bus interface.
- Click the  icon to launch the Lattice Radiant software.
- Update your constraint file accordingly and generate the programming file. See [Figure 6.5](#) for a sample .pdc file content dedicated for LAV-AT-E70ES1.

```
# Reset in push button SW1
ldc_set_location -site {AC17} [get_ports rstn_i]

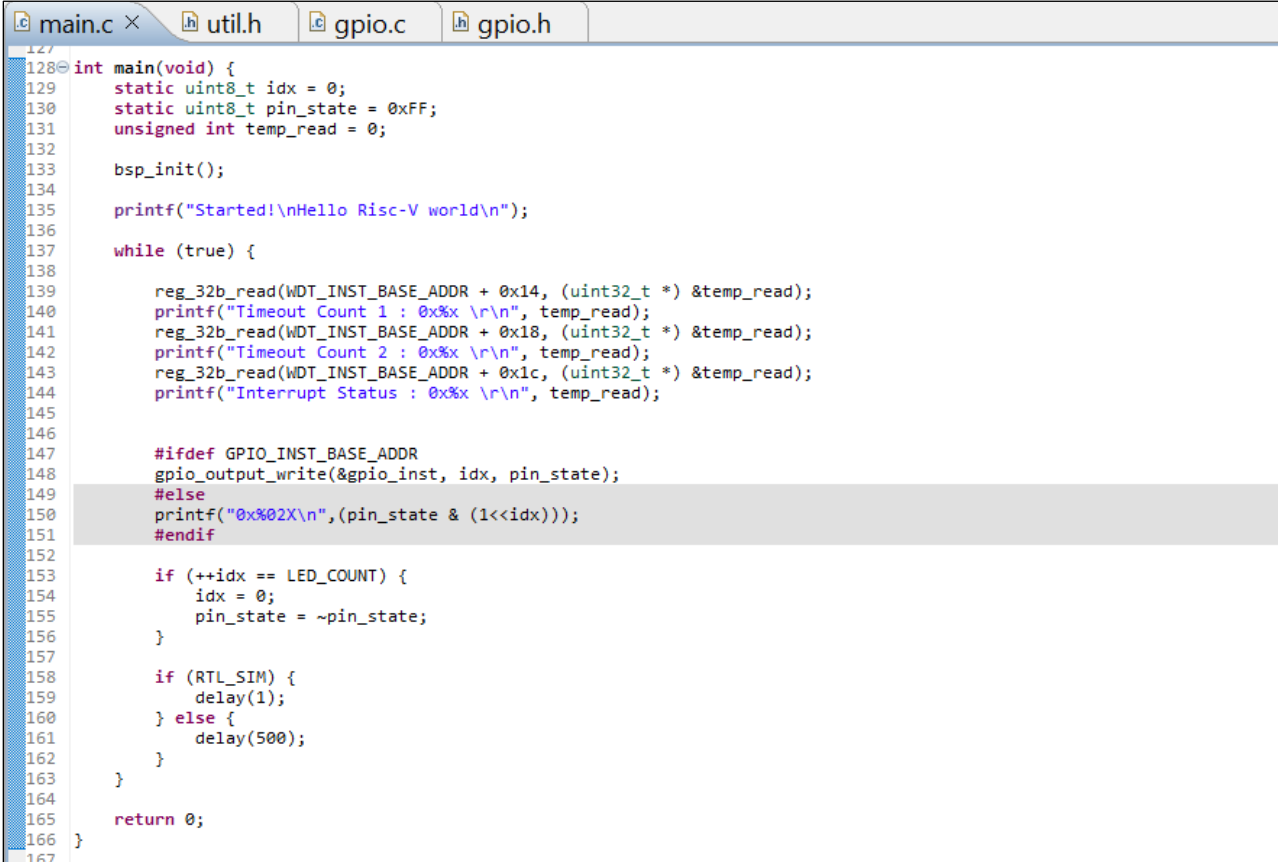
# Set UART
ldc_set_location -site {T3} [get_ports s1_uart_rxd_i]
ldc_set_location -site {R3} [get_ports s1_uart_txd_o]

# Set LED in GPIO[0:7]
ldc_set_location -site {N7} [get_ports {s0_gpio[0]}]
ldc_set_location -site {L7} [get_ports {s0_gpio[1]}]
ldc_set_location -site {L8} [get_ports {s0_gpio[2]}]

# WDT
ldc_set_location -site {N10} [get_ports wdt_inst_reset_o_port]
```

Figure 6.5. Sample PDC File Content

9. After adding the constraint file, you may now run the compilation process and generate the bit stream file.
10. In the Lattice Propel Builder software, run Lattice Propel by clicking the  icon. Then create a C/C++ project using the *Hello World Project* template.
11. Create the C code based on the main.c file accordingly. Figure 6.6 shows a sample C code implementation for the Watchdog Timer IP that displays the TIMEOUT_CNT_1, TIMEOUT_CNT_2, and INT_STATUS_REG through UART.



```
127
128 int main(void) {
129     static uint8_t idx = 0;
130     static uint8_t pin_state = 0xFF;
131     unsigned int temp_read = 0;
132
133     bsp_init();
134
135     printf("Started!\nHello Risc-V world\n");
136
137     while (true) {
138
139         reg_32b_read(WDT_INST_BASE_ADDR + 0x14, (uint32_t *) &temp_read);
140         printf("Timeout Count 1 : 0x%x \r\n", temp_read);
141         reg_32b_read(WDT_INST_BASE_ADDR + 0x18, (uint32_t *) &temp_read);
142         printf("Timeout Count 2 : 0x%x \r\n", temp_read);
143         reg_32b_read(WDT_INST_BASE_ADDR + 0x1c, (uint32_t *) &temp_read);
144         printf("Interrupt Status : 0x%x \r\n", temp_read);
145
146
147         #ifdef GPIO_INST_BASE_ADDR
148             gpio_output_write(&gpio_inst, idx, pin_state);
149         #else
150             printf("0x%02X\n", (pin_state & (1<<idx)));
151         #endif
152
153         if (++idx == LED_COUNT) {
154             idx = 0;
155             pin_state = ~pin_state;
156         }
157
158         if (RTL_SIM) {
159             delay(1);
160         } else {
161             delay(500);
162         }
163     }
164
165     return 0;
166 }
167
```

Figure 6.6. Sample C Code for Watchdog Timer IP

12. This environment is now ready to run your tests on the device. Refer to the *Propel Tutorial – Hello World* section of the [Lattice Propel SDK User Guide](#) for the step-by-step guide.

6.5. Simulating the Example Design

Refer to the *System Simulation Flow* section of the [Lattice Propel SDK User Guide](#).

6.6. Hardware Testing

6.6.1. Hardware Testing Setup

Download the generated bitstream file from the *Generating the Example Design* section to the [LAV-AT-E70ES1](#) Development Board from the Lattice Radiant Programmer.

6.6.2. Expected Output

Below is a sample waveform captured by Reveal Inserter and Reveal Analyzer tools. Refer to the relevant sections in the Lattice Radiant Software user guide for more information on how to use the Reveal Inserter and Reveal Analyzer tools.



Figure 6.7. Sample Waveform

7. Designing with the IP

This section provides information on how to generate the Watchdog Timer IP using the Lattice Radiant Software and how to run simulation and synthesis. For more details on the Lattice Radiant Software, refer to the Lattice Radiant Software user guide.

Note: The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the GUI, a screenshot may reflect an earlier version of the IP.

7.1. Generating and Instantiating the IP

The Lattice Radiant Software allows you to customize and generate modules and IPs and integrate them into the device architecture. The procedure for generating the Watchdog Timer IP in Lattice Radiant Software is described below.

To generate the Watchdog Timer IP:

1. Create a new Lattice Radiant Software project or open an existing project.
2. In the **IP Catalog** tab, double-click on **Watchdog Timer** under **IP, Processors, Controllers, and Peripherals** category. The **Module/IP Block Wizard** opens, as shown in [Figure 7.1](#). Enter values in the **Component name** and the **Create in** fields and click **Next**.

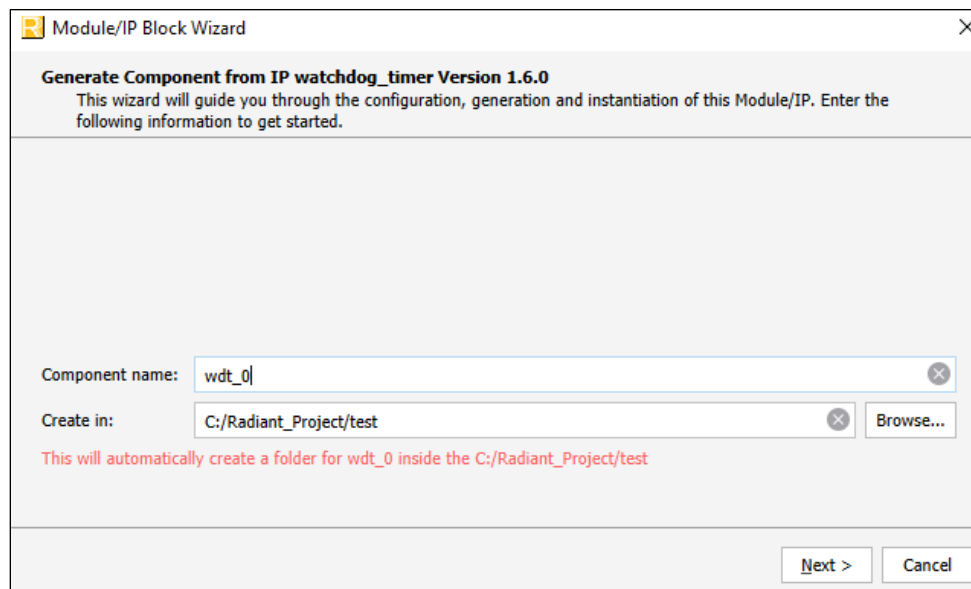


Figure 7.1. Module/IP Block Wizard

3. In the next **Module/IP Block Wizard** window, customize the selected Watchdog Timer IP using drop-down menus and check boxes. [Figure 7.2](#) shows an example configuration of the Watchdog Timer IP. For details on the configuration options, refer to the [IP Parameter Description](#) section.

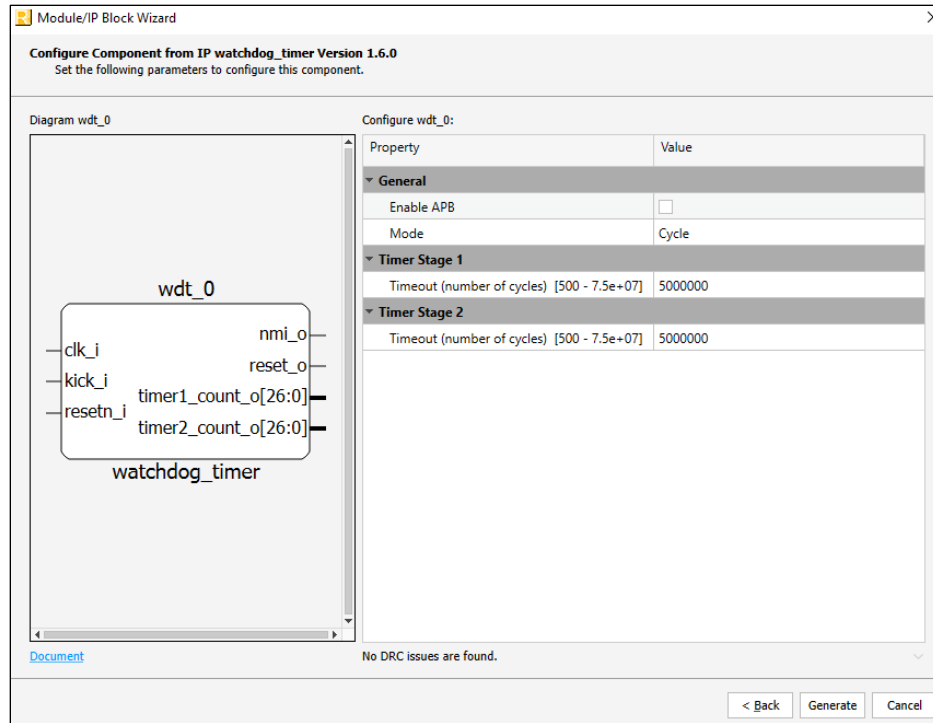


Figure 7.2. IP Configuration

- Click **Generate**. The **Check Generated Result** dialog box opens, showing design block messages and results as shown in Figure 7.3.

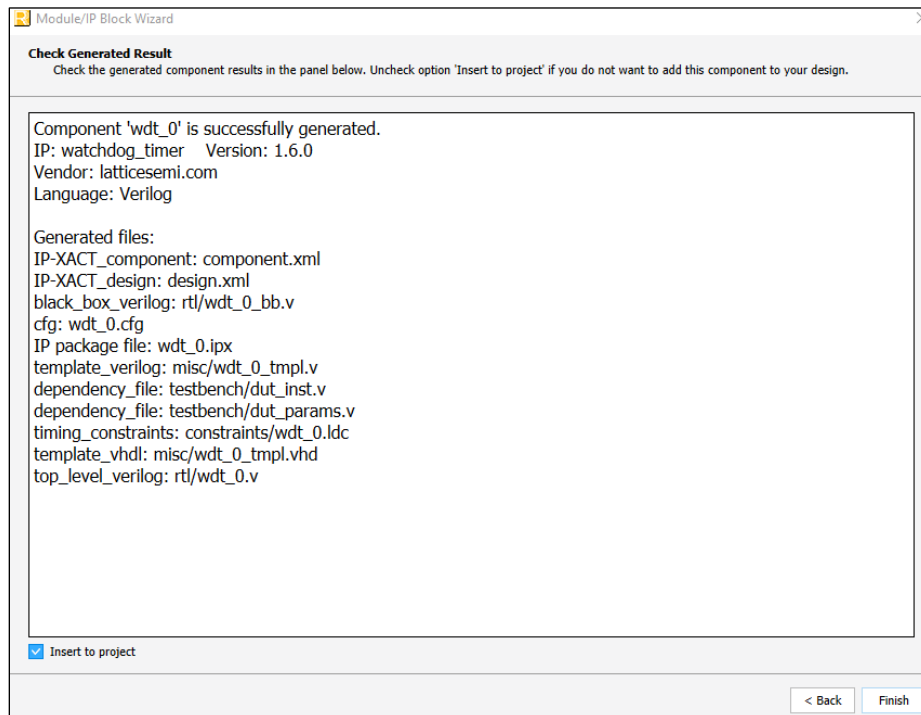


Figure 7.3. Check Generated Result

- Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields, as shown in Figure 7.1.

7.1.1. Generated Files and File Structure

The generated Watchdog Timer IP package includes the closed-box (<Component Name>_bb.v) and instance templates (<Component Name>_tpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component Name>.v) that can be used as an instantiation template for the IP is also provided. You may also use this top-level reference as the starting template for the top-level of your complete design. The generated files are listed in [Table 7.1](#).

Table 7.1. Generated File List

Attribute	Description
constraints/<Component Name>.ldc	This file contains the Pre-Synthesis Constraints for this IP.
misc/<Component Name>_tpl.v	These files provide instance templates for the IP.
misc/<Component Name>_tpl.vhd	These files provide instance templates for the IP.
rtl/<Component Name>.v	This file provides an example RTL top file that instantiates the IP.
rtl/<Component Name>_bb.v	This file provides the synthesis closed box.
testbench/dut_inst.v	This file contains the instance of the generated IP.
testbench/dut_params.v	This file contains the parameter settings of the generated IP.
testbench/tb_apb_mst.v	This file contains additional testbench RTL for APB and is imported in testbench top module.
testbench/tb_top.v	This file is the testbench top module for the IP.
component.xml	Contains the ipxact: component information of the IP.
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format.
<Component Name>.ipx	This file contains the information on the files associated to the generated IP.
<Component Name>.cfg	This file contains the parameter values used in IP configuration.

7.2. Design Implementation

Completing your design includes additional steps to specify analog properties, pin assignments, and timing and physical constraints. You can add and edit the constraints using the Device Constraint Editor or by manually creating a PDC File.

Post-Synthesis constraint files (.pdc) contain both timing and non-timing *constraint.pdc* source files for storing logical timing/physical constraints. Constraints that are added using the Device Constraint Editor are saved to the active .pdc file. The active post-synthesis design constraint file is then used as input for post-synthesis processes.

Refer to the relevant sections in the Lattice Radiant Software user guide for more information on how to create or edit constraints and how to use the Device Constraint Editor.

7.3. Timing Constraints

You need to provide proper timing and physical design constraints to ensure that your design meets the desired performance goals on the FPGA device. Add the content of the following IP constraint file to your design constraints:

<Component Name>/constraints/<Component Name>.ldc

The above constraint file has been verified during IP evaluation with the IP instantiated directly at the top-level module. You can modify the constraints in this file provided you understand the effect of each constraint.

To use this constraint file, copy the contents of <Component Name>.ldc to the top-level design constrain for pre-synthesis.

Refer to the [Lattice Radiant Timing Constraints Methodology \(FPGA-AN-02059\)](#) for details on how to constrain your design.

7.4. Specifying the Strategy

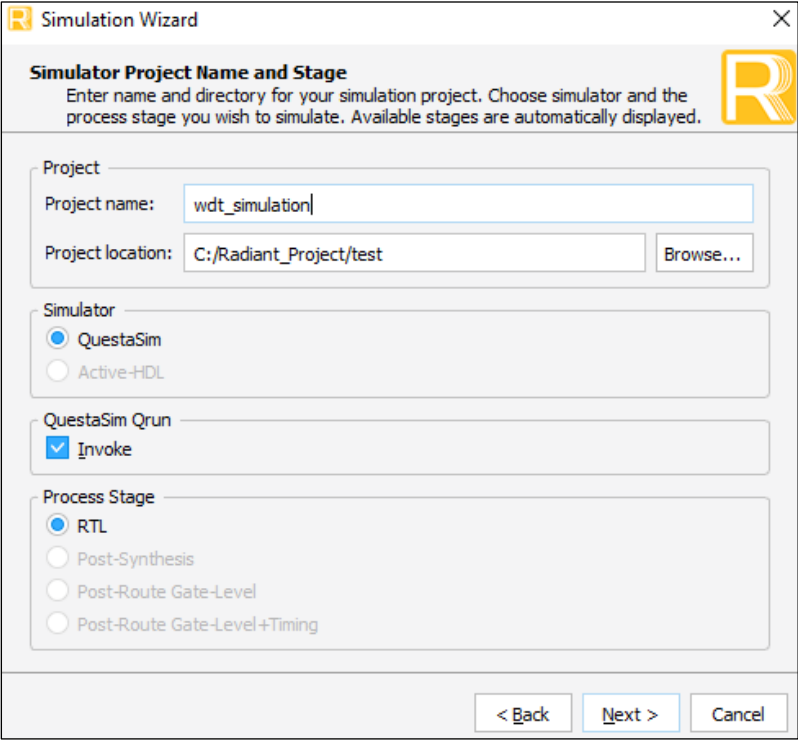
The Radiant software provides two predefined strategies: Area and Timing. It also enables you to create customized strategies. For details on how to create a new strategy, refer to the Strategies section of the Lattice Radiant Software user guide.

7.5. Running Functional Simulation

You can run functional simulation after the IP is generated.

To run functional simulation:

1. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard**, as shown in [Figure 7.4](#).



Simulation Wizard

Simulator Project Name and Stage
Enter name and directory for your simulation project. Choose simulator and the process stage you wish to simulate. Available stages are automatically displayed.

Project
Project name:
Project location:

Simulator
☒ QuestaSim
☐ Active-HDL

QuestaSim Qrun
☒ Invoke

Process Stage
☒ RTL
☐ Post-Synthesis
☐ Post-Route Gate-Level
☐ Post-Route Gate-Level+Timing

< Back Next > Cancel

Figure 7.4. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window, as shown in [Figure 7.5](#). Confirm that the **Source Files** are as shown in [Figure 7.5](#). Remove other files if there are any.

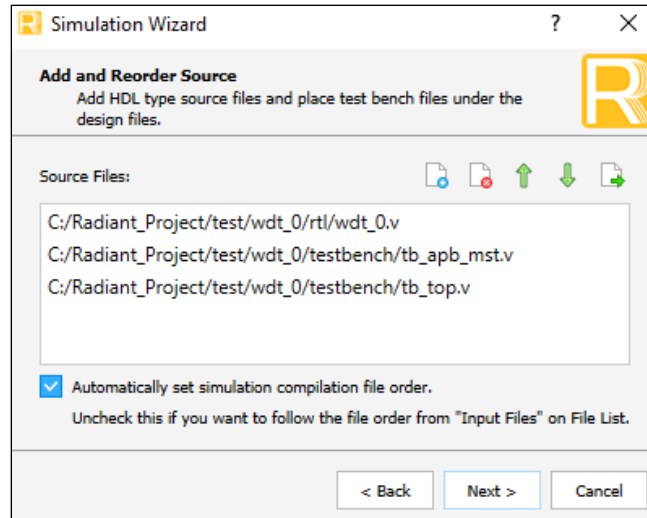


Figure 7.5. Add and Reorder Source

- Click **Next**. The **Parse HDL files for simulation** window is as shown. Confirm that the Simulation Top Module is *tb_top*.

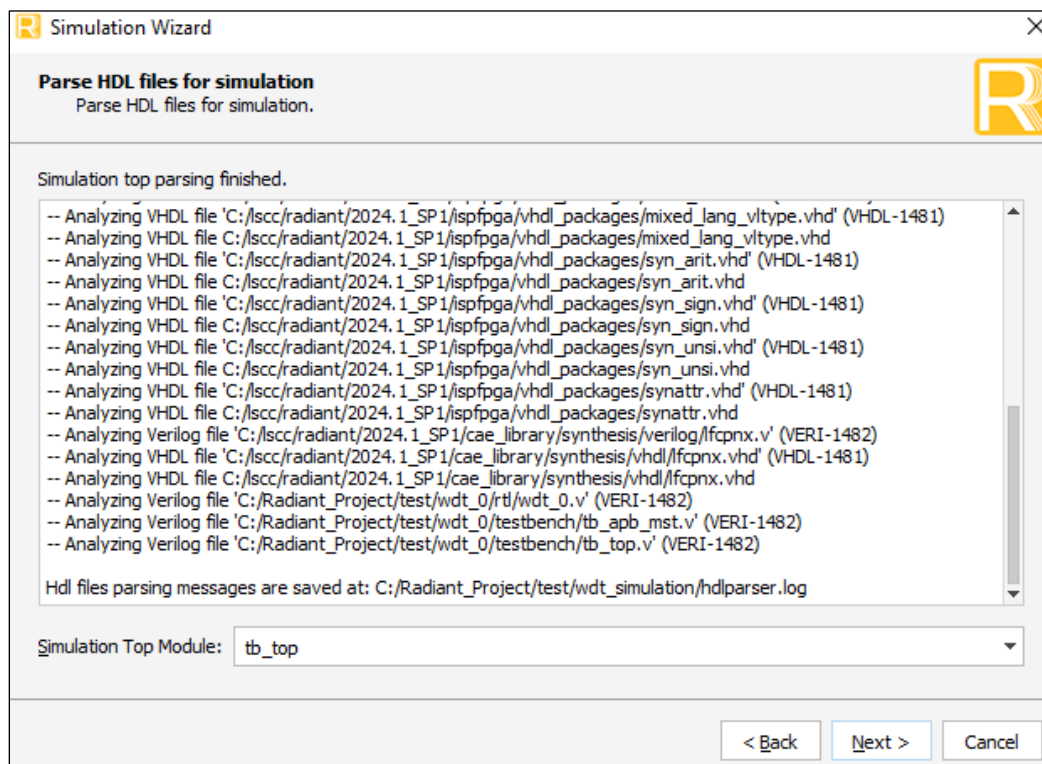


Figure 7.6. Parse HDL Files for Simulation

- Click **Next**. The Summary window is shown. Change the **Default Run** time to 0.

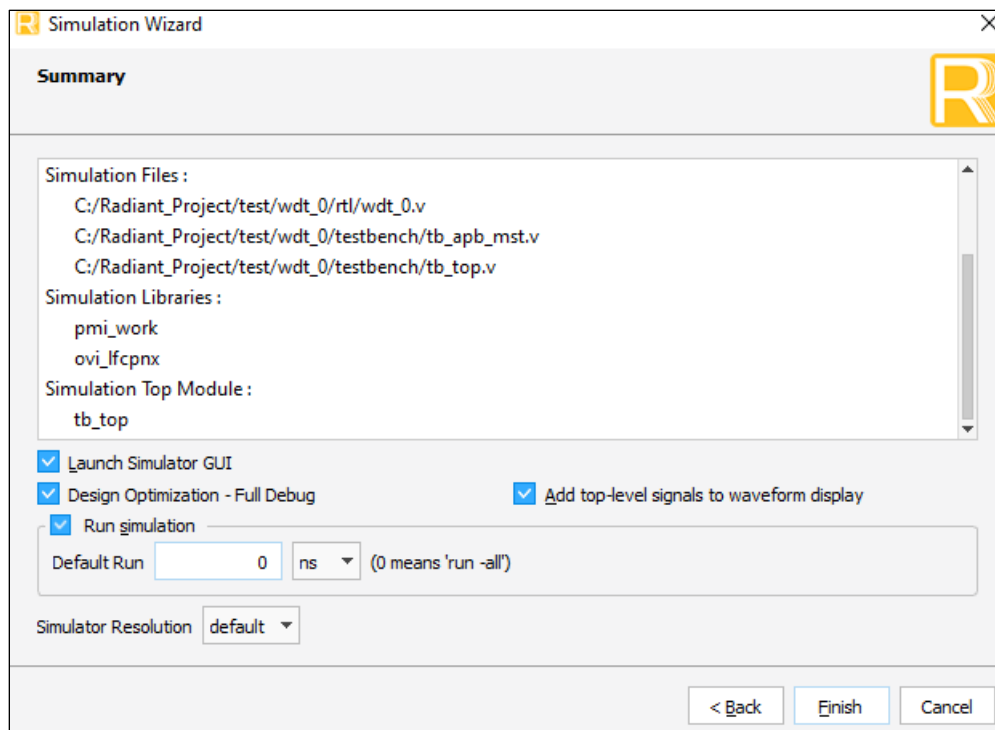


Figure 7.7. Summary

- Click **Finish** to run the simulation. The waveforms in [Figure 7.8](#) shows an example simulation result.

Note: It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant Software Suite.

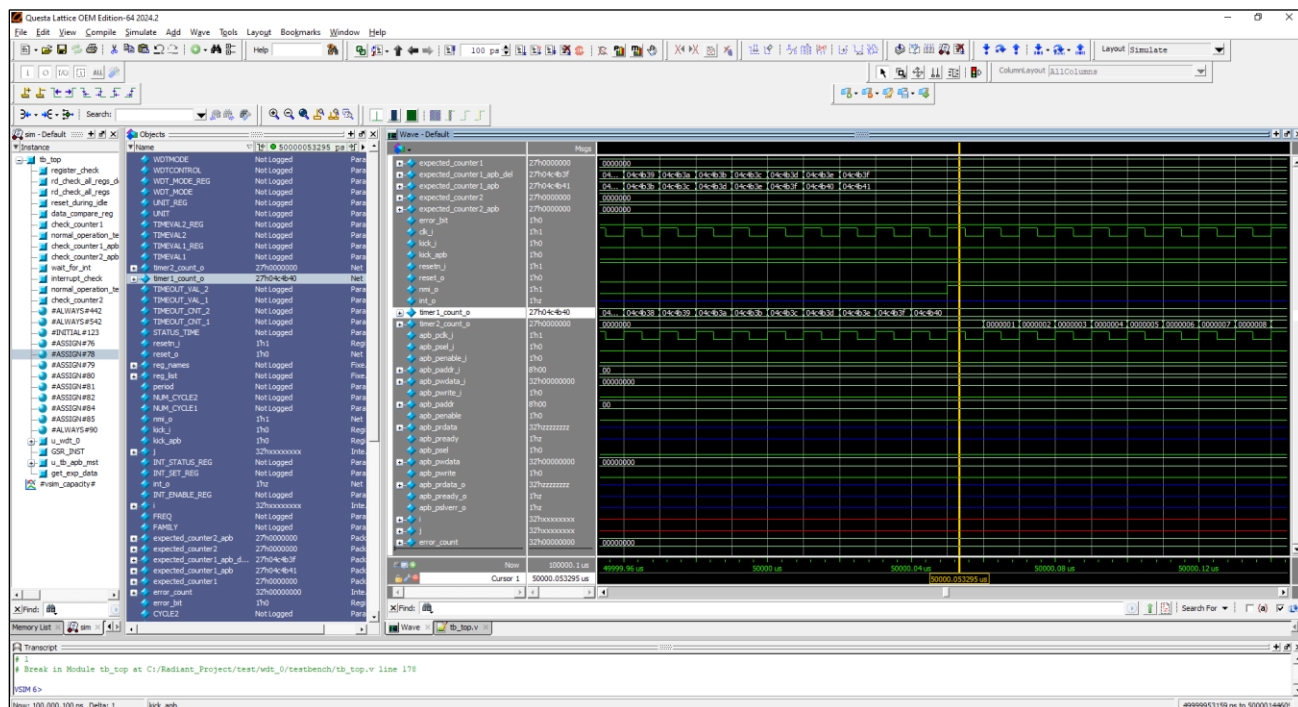


Figure 7.8. Simulation Waveform

Appendix A. Resource Utilization

Table A.1 shows the resource utilization of the Watchdog Timer IP for the LFMXO5-25-9BBG400I device using Synplify Pro of Lattice Radiant software 2022.1. Default configuration is used, and some attributes are changed from the default value to show the effect on resource utilization.

Table A.1. LFMXO5-25-9BBG400I Device Resource Utilization

Configuration	Clk Fmax (MHz) ¹	Registers	LUTs	EBRs
Default	169.66	51	100	0
<i>Enable APB</i> = Checked Others = Default	200	109	181	0
Timer Stage 1 <i>Number of Cycles</i> : 500, Timer Stage 2 <i>Number of Cycles</i> : 500, Others = Default	200	37	73	0
Timer Stage 1 <i>Number of Cycles</i> : 75000000, Timer Stage 2 <i>Number of Cycles</i> : 75000000, Others = Default	200	51	98	0
<i>Mode</i> : Time, <i>System Clock Frequency</i> : 150, Others = Default	200	51	106	0
<i>Mode</i> : Time, Timer Stage 1 <i>Timeout</i> : 500, Timer Stage 2 <i>Timeout</i> : 500, Others = Default	189.97	52	107	0

Note:

1. Fmax is generated when the FPGA design only contains Watchdog Timer IP, and the target frequencies are 150 MHz (for configured *System Clock Frequency*) and 100MHz (for other configurations). These values may be reduced when user logic is added to the FPGA design.

Table A.2 shows the resource utilization of the Watchdog Timer IP for the LFMXO5-25-7BBG400I device using Synplify Pro of Lattice Radiant software 2022.1. Default configuration is used, and some attributes are changed from the default value to show the effect on resource utilization.

Table A.2. LFMXO5-25-7BBG400I Device Resource Utilization

Configuration	Clk Fmax (MHz) ¹	Registers	LUTs	EBRs
Default	132.14	51	100	0
<i>Enable APB</i> = Checked Others = Default	152.02	109	181	0
Timer Stage 1 <i>Number of Cycles</i> : 500, Timer Stage 2 <i>Number of Cycles</i> : 500, Others = Default	170.45	36	73	0
Timer Stage 1 <i>Number of Cycles</i> : 75000000, Timer Stage 2 <i>Number of Cycles</i> : 75000000, Others = Default	143.74	51	98	0
<i>Mode</i> : Time, <i>System Clock Frequency</i> : 150, Others = Default	172.56	51	107	0
<i>Mode</i> : Time, Timer Stage 1 <i>Timeout</i> : 500, Timer Stage 2 <i>Timeout</i> : 500, Others = Default	163.99	52	107	0

Note:

1. Fmax is generated when the FPGA design only contains Watchdog Timer IP, and the target frequencies are 150 MHz (for configured *System Clock Frequency*) and 100MHz (for other configurations). These values may be reduced when user logic is added to the FPGA design.

Table A.3 shows the resource utilization of the Watchdog Timer IP for the LAV-AT-E70-3LFG1156I device using Synplify Pro of Lattice Radiant software 2022.1. Default configuration is used, and some attributes are changed from the default value to show the effect on resource utilization.

Table A.3. LAV-AT- E70-3LFG1156I Device Resource Utilization

Configuration	Clk Fmax (MHz) ¹	Registers	LUTs	EBRs
Default	245.58	51	77	0
Enable APB = Checked Others = Default	231.70	109	164	0
Timer Stage 1 Number of Cycles: 500, Timer Stage 2 Number of Cycles: 500, Others = Default	251.76	37	57	0
Timer Stage 1 Number of Cycles: 75000000, Timer Stage 2 Number of Cycles: 75000000, Others = Default	251.76	51	77	0
Mode: Time, System Clock Frequency: 150, Others = Default	251.76	51	77	0
Mode: Time, Timer Stage 1 Timeout: 500, Timer Stage 2 Timeout: 500, Others = Default	251.76	52	77	0

Note:

1. Fmax is generated when the FPGA design only contains Watchdog Timer IP, and the target frequencies are 150 MHz (for configured System Clock Frequency) and 100MHz (for other configurations). These values may be reduced when user logic is added to the FPGA design.

Table A.4 shows the resource utilization of the Watchdog Timer IP for the LN2-CT-20-1ASG410I device using Synplify Pro of Lattice Radiant software 2024.2. Default configuration is used, and some attributes are changed from the default value to show the effect on resource utilization.

Table A.4. LN2-CT-20-1ASG410I Device Resource Utilization

Configuration	Clk Fmax (MHz) ¹	Registers	LUTs	EBRs
Default	250	52	80	0
Enable APB = Checked Others = Default	250	129	151	0
Timer Stage 1 Number of Cycles: 500, Timer Stage 2 Number of Cycles: 500, Others = Default	250	38	57	0
Timer Stage 1 Number of Cycles: 75000000, Timer Stage 2 Number of Cycles: 75000000, Others = Default	250	56	85	0
Mode: Time, System Clock Frequency: 150, Others = Default	250	52	76	0
Mode: Time, Timer Stage 1 Timeout: 500, Timer Stage 2 Timeout: 500, Others = Default	250	55	63	0

Note:

1. Fmax is generated when the FPGA design only contains Watchdog Timer IP, and the target frequency is 100 MHz. These values may be reduced when user logic is added to the FPGA design.

References

- [Watchdog Timer IP Release Notes \(FPGA-RN-02024\)](#)
- [AMBA 3 APB Protocol v1.0 Specification](#)
- [Lattice Radiant Timing Constraints Methodology \(FPGA-AN-02059\)](#)
- [Avant Evaluation Board User Guide \(FPGA-EB-02057\)](#)
- [Avant-E web page](#)
- [Avant-G web page](#)
- [Avant-X web page](#)
- [Certus-N2 web page](#)
- [Certus-NX web page](#)
- [CertusPro-NX web page](#)
- [CrossLink web page](#)
- [CrossLink-NX web page](#)
- [Mach-NX web page](#)
- [MachXO5-NX web page](#)
- [Watchdog Timer IP Core web page](#)
- [Lattice Propel Design Environment web page](#)
- [Lattice Propel SDK User Guide](#)
- [Lattice Radiant Software web page](#)
- [Lattice Solutions IP Cores web page](#)
- [Lattice Solutions Reference Designs web page](#)
- [Lattice Insights web page for Lattice Semiconductor training courses and learning plans](#)

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Note: In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

Revision 1.9, IP v1.6.1, June 2026

Section	Change Summary
All	- Updated the IP version information on the cover page. - Added a note on the IP version in the <i>Quick Facts</i> and <i>Revision History</i> sections. - Made editorial fixes.
Introduction	- Updated Table 1.1. Summary of the Watchdog Timer IP. - In the IP Support Summary section: - Removed the <i>Hardware Validated</i> column from Table 1.2. Watchdog Timer IP Support Readiness. - Added a reference to the <i>Example Design</i> section. - Removed the <i>Hardware Support</i> section. - Added the Attribute Names section.
Functional Descriptions	Updated <i>kick</i> to <i>kick_i</i> in the Watchdog Timer Non-APB Mode Operation section.
Register Description	- Updated the width values in Table 5.3. WDT_MODE. - Updated the <i>cycle_timeoutval_st2</i> description in Table 5.6. STATUS_CYCLE_2.
Designing with the IP	- Added a note on screenshots in this section. - Updated the C code implementation step in the Generating the Example Design section. - Updated Figure 6.6. Sample C Code for Watchdog Timer IP.
Designing with the IP	- Updated the <i>Add and Reorder Source</i> step in the Running Functional Simulation section. - Updated the Figure 7.5. Add and Reorder Source caption.
References	Added the <i>Watchdog Timer IP Release Notes (FPGA-RN-02024)</i> , <i>Watchdog Timer IP Core</i> web page, and <i>Lattice Solutions Reference Designs</i> web page.

Revision 1.8, IP v1.6.0, January 2025

Section	Change Summary
All	- Added the IP version information on the cover page. - Removed the previous <i>3.4. Hardware Evaluation</i> section. - Made editorial fixes.
Abbreviations in This Document	Replaced acronyms with abbreviations and updated this section.
Introduction	- Moved introductory paragraph in the 1. Introduction section to the 1.1. Overview of the IP section and updated the heading numbers of remaining sections accordingly. - Updated Table 1.1. Quick Facts. - Added the 1.3. IP Support Summary, 1.5. Licensing and Ordering Information, 1.6. Hardware Support, and 1.7. Minimum Device Requirements sections and updated the heading numbers of remaining sections accordingly. - Renamed the previous <i>1.3. Conventions</i> section to 1.8. Naming Conventions and removed the <i>Attribute</i> information.
Functional Descriptions	- Renamed the previous <i>2.1 Overview</i> section to 2.1. IP Architecture Overview and updated its content. - Added <i>non-APB Mode</i> and <i>APB Mode</i> to the 2.1.2. Reset Signals section. - Added the 2.2. Clocking, 2.3. Reset, and 2.4. User Interfaces sections. - Moved the 2.6. <i>Programming Flow</i> section to the 2.5. Programming Flow section and updated its content. - Moved the previous 2.5. Timing Diagrams section to the 2.6. Operation Details section and updated its content.
IP Parameter Description	Moved the content from previous 2.3. <i>Attribute Summary</i> section to this section and updated its content.

Section	Change Summary
Signal Description	Moved the content from previous 2.2. <i>Signal Description</i> section to this section and updated its content.
Register Description	Moved the content from previous 2.4. <i>Register Description</i> section to this section and updated its content.
Example Design	Added this section.
Designing with the IP	- Renamed the previous 3. <i>IP Generation and Evaluation</i> section to the 7. Designing with the IP and updated its content. - Moved the content from previous 3.1. <i>Generation and Synthesis</i> section to the 7.1. Generating and Instantiating the IP section and updated its content, including all the figures. - Removed the <i>eval/constraint.pdc</i> attribute from Table 7.1. Generated File List. - Moved the content from previous 3.3. <i>Constraining the IP</i> section to the 7.3. Timing Constraints section and updated its content. - Added the 7.2. Design Implementation and 7.4. Specifying the Strategy sections. - Moved the content from previous 3.2. <i>Running Functional Simulation</i> section to the 7.5. Running Functional Simulation section and updated its content.
Appendix A. Resource Utilization	Added resource utilizations for the Lattice Radiant software version 2024.2.
References	Updated this section.

Revision 1.7, April 2024

Section	Change Summary
All	- Renamed document from <i>Watchdog Timer IP Core - Lattice Radiant Software</i> to <i>Watchdog Timer IP</i>. - Changed <i>black box</i> to <i>closed-box</i>.
Inclusive Language	Added the inclusive language boilerplate.
Functional Description	- Added information on kick signal in the Overview subsection. - Moved the content for reset ports to the Reset Ports subsection. - Added the Kick Signal subsection. - Updated the description for <i>apb_psel_i</i> and <i>apb_pready_o</i> in Table 2.1. Watchdog Timer IP Core Signal Description.
References	Updated references.

Revision 1.6, November 2023

Section	Change Summary
Disclaimers	Updated this section.
Functional Description	- Replaced 'computer' with 'processor' throughout this section. - Updated Table 2.1. Watchdog Timer IP Core Signal Description. - Updated Table 2.4. Register Address Map to replace the Access Type for WDT_MODE from RW to RO. - Updated the Register Description section. - Updated the Programming Flow section.
IP Generation and Evaluation	- Updated Figure 3.2. Configure the User Interface of Watchdog Timer IP Core. - Updated Table 3.1. Generated File List to add <i>testbench/dut_inst.v</i>, <i>testbench/dut_params.v</i>, and <i>eval/constraint.pdc</i>. - Added the Constraining the IP section
Technical Support Assistance	Updated to add the link for the Lattice Answer Database.
References	Updated to add the link for Lattice Insights.
Appendix A. Resource Utilization	Updated to replace 500E with E70.

Revision 1.5, October 2022

Section	Change Summary
Introduction	Added the feature Supports APB Interface to Features.
Functional Description	- Updated Figure 2.1. Functional Block Diagram and Figure 2.2. Timing Diagram. - Newly added the Register Description and Programming Flow sections. - In Table 2.1. Watchdog Timer IP Core Signal Description: - removed WDT Ports; - added APB Ports. - Table 2.2. Attributes Table: added the Enable APB attribute. - Table 2.3. Attributes Descriptions: added the Enable APB attribute.
IP Generation and Evaluation	Updated Figure 3.1. Module/IP Block Wizard, Figure 3.2. Configure the User Interface of Watchdog Timer IP Core, Figure 3.3. Check Generating Result, Figure 3.4. Simulation Wizard, Figure 3.5. Adding and Reordering Source, and Figure 3.6. Simulation Waveform.
Appendix A. Resource Utilization	- Updated Table A.1. Resource Utilization (LFMX05-25-9BBG400I) and Table A.2. Resource Utilization (LFMX05-25-7BBG400I). - Newly added Table A.3. Resource Utilization (LAV-AT- E70-3LFG1156I).

Revision 1.4, May 2022

Section	Change Summary
Introduction	Updated Table 1.1. Quick Facts. Added Table A.2. Resource Utilization (LFMX05-25-7BBG400I) to the Resource Utilization row.
IP Generation and Evaluation	Updated Figure 3.1. Module/IP Block Wizard, Figure 3.2. Configure the User Interface of Watchdog Timer IP Core, and Figure 3.3. Check Generating Result.
Appendix A. Resource Utilization	Updated Table A.1. Resource Utilization (LFMX05-25-9BBG400I) and Table A.2. Resource Utilization (LFMX05-25-7BBG400I).

Revision 1.3, June 2021

Section	Change Summary
Introduction	- Remove second paragraph. - Updated Table 1.1. Quick Facts. - Revised Supported FPGA Families - Revised Targeted Devices - Revised Lattice Implementation. - Revised reference to Lattice Radiant Software User Guide
References	Revised reference to Lattice Radiant Software User Guide and removed link.

Revision 1.2, June 2020

Section	Change Summary
Introduction	- Added Certus-NX support. - Updated Table 1.1 to add LFD2NX-40 as targeted device. - Updated Synopsis Synplify Pro version. - Updated Lattice Implementation to Lattice Radiant 2.1.
Attribute Summary	Removed Calculated group. The calculated <i>Timeout (number of cycles)</i> are now reflected in Timer Stage N group. Default value of <i>Timeout (number of cycles)</i> is changed due to IP Engine limitation.
IP Generation and Evaluation	Updated Figure 3.2.
Appendix A. Resource Utilization	- Updated device to LIFCL-40-9BG400I. - Added column for Fmax in Table A1.
All	Updated references to Lattice Radiant Software 2.1 User Guide.

Revision 1.1, February 2020

Section	Change Summary
Introduction	Updated Table 1.1 to add LIFCL-17 as targeted device.

Revision 1.0, December 2019

Section	Change Summary
All	Initial release.



www.latticesemi.com