RISC-V RX CPU IP - Lattice Propel Builder 2.2

User Guide

FPGA-IPUG-02186-1.0

June 2022

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 associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. 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. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. 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.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
ABI	Application Binary Interface
AEE	Application Execution Environment
AXI	Advanced eXtensible Interface
CLINT	Core Local Interrupter
CPU	Central Processing Unit
CSR	Control and Status Register
DMIPS	Dhrystone MIPS (Million Instructions per Second)
DTCM	Data TCM
EIP	Interrupt Pending
FPGA	Field Programmable Gate Array
GDB	Gnu Debugger
HDL	Hardware Description Language
IE	Interrupt Enable
IRQ	Interrupt Request
ISA	Instruction Set Architecture
JTAG	Joint Test Action Group
ITCM	Instruction TCM
LUT	Lookup-Table
OpenOCD	Open On-Chip Debugger
OS	Operating System
PLIC	Platform-Level Interrupt Controller
PMP	Physical Memory Protection
RISC-V	Reduced instruction set computer-V (five)
RV32IMC	RISC-V Integer, M & Compressed Instruction Sets
RX	Real Time OS (RISC-V for RTOS applications)
SEE	Supervisor Execution Environment
TCM	Tightly Coupled Memory
UART	Universal Asynchronous Receiver Transmitter
WDT	Watchdog Timer Device

1. Introduction

The Lattice Semiconductor RISC-V RX soft IP contains a 32-bit RISC-V processor core and several submodules – Platform Level Interrupt Controller (PLIC), Core Local Interrupter (CLINT), watchdog and Tightly Coupled Memory (TCM). The CPU core supports RV32IMC instruction set and debug feature which is JTAG – IEEE 1149.1 compliant. The modules outside are accessed by the processor core using AXI interface.

The design is implemented in Verilog HDL. It can be configured and generated using the Lattice Propel™ Builder software. It is targeted to CertusPro-NX FPGA devices and implemented using Lattice Radiant™ software Place and Route tool integrated with the Synplify Pro® synthesis tool.

1.1. Features

The RISC-V RX soft IP has the following features:

- RV32IMC instruction set
- Five stage pipeline
- All three privilege modes supported: Machine Mode, Supervisor Mode and User Mode
- Instruction Cache and Data Cache
- Support for the AXI4 bus standard for data port
- Debug through GDB and OpenOCD
- PLIC module
- CLINT module
- Watchdog module
- Benchmark and Frequency:
  - Balanced mode: 1.01 DMIPS/MHz performance; 130 MHz(sp9)/110 MHz(sp7) on CertusPro-NX device

  Note:
  fmax is based on:
  - standalone processor core;
  - Radiant3.1 production build, with 9_High-Performance_1.0V (sp9) and 7_High[1]Performance_1.0V (sp7).

1.2. Conventions

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

2. Functional Descriptions

2.1. Overview

The RISC-V RX IP processes data and instructions while monitoring the external interrupts. As shown in Figure 2.1, the CPU IP has a 32-bit processor core and submodules, TCM, PLIC and CLINT/Watchdog of which are must have, Local UART of which is optional. The 32-bit processor uses local instruction port to get instructions from TCM and local data port for data access. The external interface for data port is AXI.

The CPU core, TCM, bridges, Mux, PLIC and UART run in fast system clock domain. CLINT and Watchdog run in both fast system clock domain and slow real time clock domain. The Debug module runs in both system clock domain and JTAG clock domain.

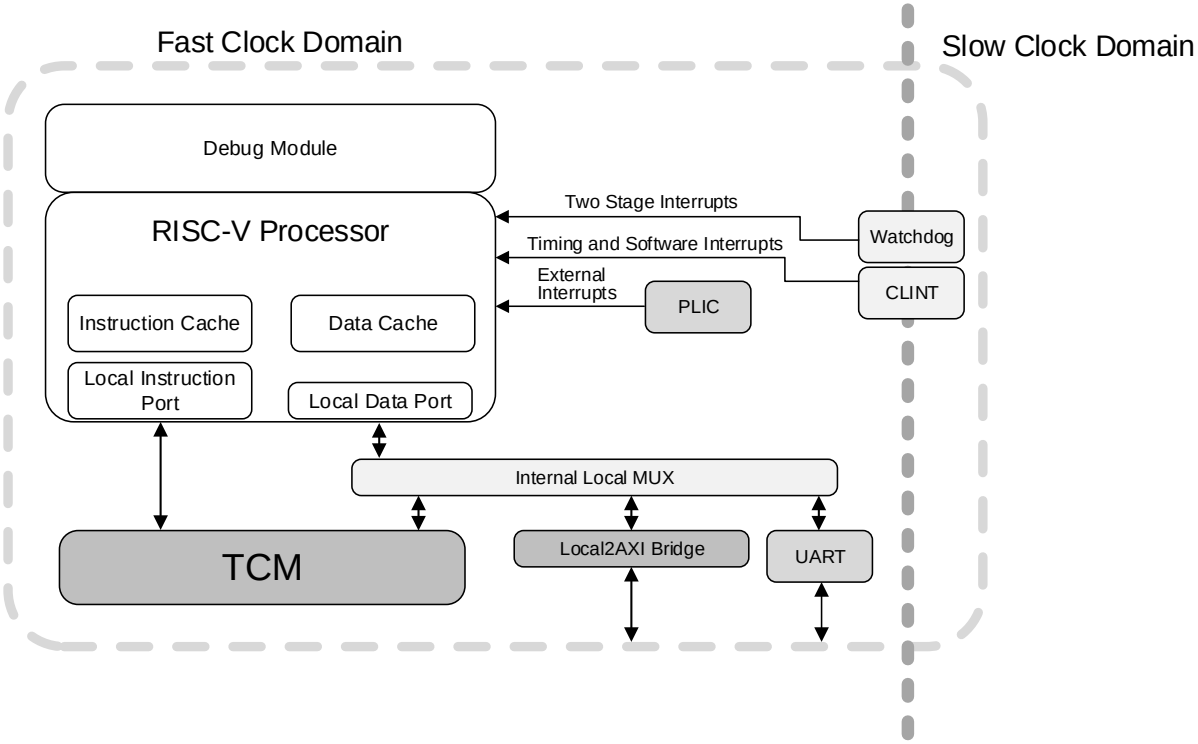

Figure 2.1. RISC-V RX Soft IP Diagram

2.2. Modules Description

2.2.1. RISC-V Processor Core

Figure 2.2 shows the processor core block diagram.

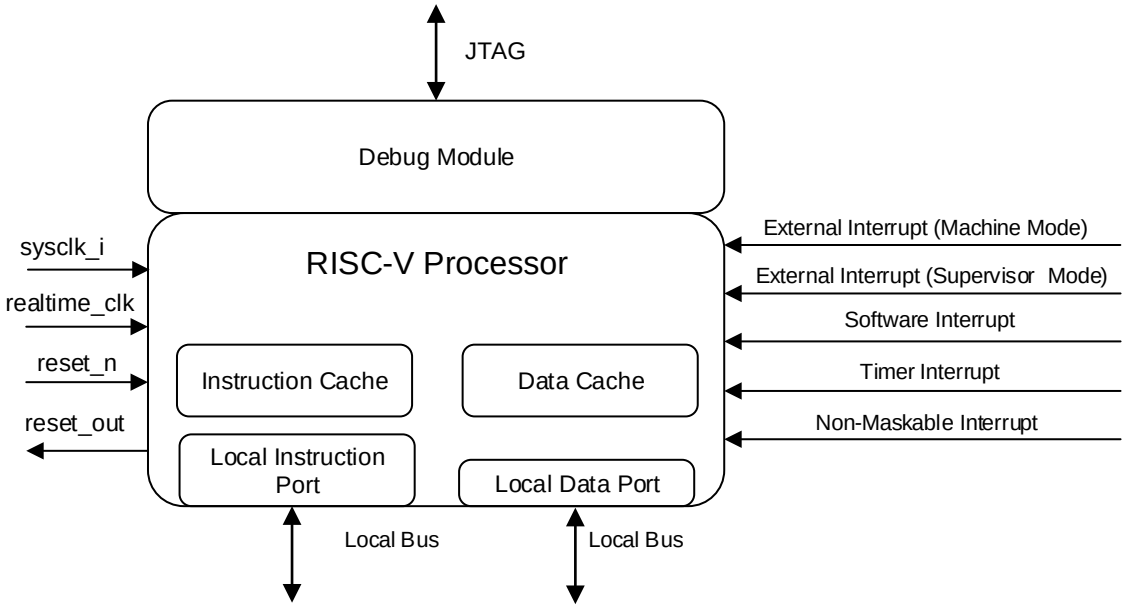

Figure 2.2. RISC-V RX Processor Core Block Diagram

2.2.1.1. Interrupt

There are four types of interrupts, Machine/Supervisor mode, External Interrupt from PLIC, software interrupt and timer interrupt from CLINT and Non-Maskable interrupt from outside.

By default, interrupts are handled in machine mode. Considering supervisor mode is supported, it is possible to delegate certain interrupts to supervisor mode.

2.2.1.2. Exception

If an exception occurs, the processor core stops the corresponding instruction, flushes all previous instructions, and waits until the terminated instruction reaches the writeback stage before jumping to the exception service routine.

2.2.1.3. Low Power Mode

Processor core enters low power mode when it executes the WFI instruction. PC halts during the low power mode. Processor wakes up, if there is external/timer interrupt.

2.2.1.4. Debug

The processor core supports the IEEE-1149.1 JTAG debug logic with two hardware breakpoints.

2.2.1.5. Cache

Both Instruction Cache and Data Cache have following configurations:

- cache size: 4096 bytes
- 32 bytes per cache line
- 2-way set associative

The cache strategy for data cache is write through. The cache eviction policy of both caches is round robin.

2.2.1.6. Privilege Mode

The processor supports User, Supervisor and Machine mode. Along with corresponding CSR registers, Figure 2.3 shows two typical software stacks:

- A simple system that supports only a single application running on an application execution environment (AEE). The application is coded to run with a particular application binary interface (ABI). ABI includes the supported user-level ISA plus a set of ABI calls to interact with the AEE. The ABI hides details of the AEE from the application to allow greater flexibility in implementing the AEE.
- Meanwhile, a conventional operating system (OS) can provide AEE and ABI. The OS interfaces with a supervisor execution environment (SEE) via a supervisor binary interface (SBI). An SBI comprises the user-level and supervisor-level ISA together with a set of SBI function calls.

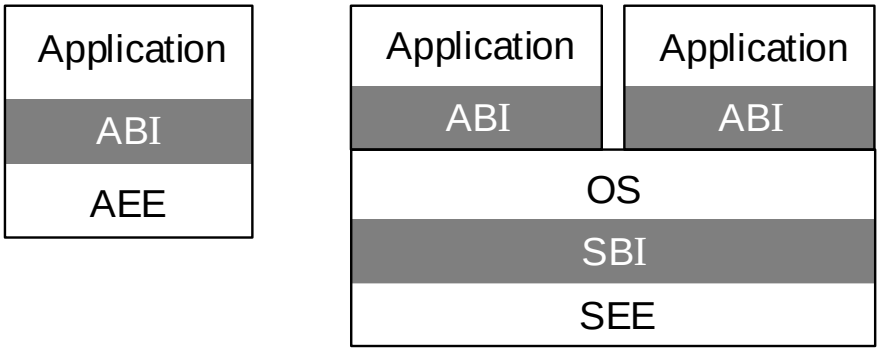

Figure 2.3. Various Forms of Privileged Execution

2.2.1.7. Control and status registers

The processor core supports three privilege modes. All supported Control and Status Registers (CSRs) are listed in Table 2.1.

Table 2.1. RISC-V Processor Core Control and Status Registers

Number	Privilege	Name	Description
Supervisor Trap Setup			
0x100	SRW	sstatus	Supervisor status register
0x104	SRW	sie	Supervisor interrupt enable register
0x105	SRW	stvec	Supervisor trap hander base address
Supervisor Trap Handling			
0x140	SRW	sscratch	Scratch register for supervisor trap handlers
0x141	SRW	sepc	Supervisor exception program counter
0x142	SRW	scause	Supervisor trap cause
0x143	SRW	stval	Supervisor bad address or instruction
0x144	SRW	sip	Supervisor interrupt pending
Machine Information Registers			
0xF11	MRO	mvendorid	Vendor ID
0xF12	MRO	marchid	Architecture ID
0xF13	MRO	mimpid	Implementation ID
0xF14	MRO	mhartid	Hardware thread ID
Machine Trap Setup			
0x300	MRW	mstatus	Machine status register
0x301	MRO	misa	ISA and extensions
0x302	MRW	medeleg	Machine exception delegation register
0x303	MRW	mideleg	Machine interrupt delegation register
0x304	MRW	mie	Machine interrupt enable register

Number	Privilege	Name	Description
0x305	MRW	mtvec	Machine trap hander base address
Machine Trap Handling			
0x340	MRW	mscratch	Scratch register for machine trap handlers
0x341	MRW	mepc	Machine exception program counter
0x342	MRO	mcause	Machine trap cause
0x343	MRO	mtval	Machine bad address or instruction
0x344	MRW	mip	Machine interrupt pending
Machine Counter/Timers			
0xB00	MRW	mcycle	Machine cycle counter
0xB02	MRW	minstret	Machine instructions-retired counter
0xB80	MRW	mcycleh	Upper 32 bits of mcycle
0xB82	MRW	minstreth	Upper 32 bits of minstret

2.2.2. Submodule

All the submodules are covered here. Every submodule has a fixed base address. See Table 2.10.

2.2.2.1. PLIC

The Platform Level Interrupt Controller (PLIC) module is compliant with the RISC-V Platform-Level Interrupt Controller Specification (RISC-V Platform-Level Interrupt Controller Specification v1.0. 09.2021).

The PLIC multiplexes various device interrupts onto the external interrupt lines of Hart contexts (Note: The context is referred to the specific privilege mode in the specific Hart of specific RISC-V processor instance), with hardware support for interrupt priorities. PLIC supports up to 31 external interrupts (0 is reserved) with 7 priority levels, and each one has a corresponding interrupt ID, starting from 1. The first input interrupt (#1) is fixed to Watchdog Timer device.

The PLIC has two interrupt output signals connected to external interrupt inputs of the CPU – one for Machine mode, the other for Supervisor mode.

Figure 2.2 shows the block diagram of PLIC operation parameter. An example for how it works: interrupt input 1 gets asserted, it goes through the Gateway and sets Interrupt Pending (IP) bit of the Source. If its Interrupt Enable (IE) is set, the priority value can be passed and compared to other inputs all the way through the chain. The interrupt ID is similarly forwarded. So if the Max Priority is larger than the threshold, interrupt pending (EIP) can be asserted and sent to the processor. Meanwhile, the Gateway blocks subsequent interrupts from being forwarded until the current interrupt has been completed. Target 0 goes to machine mode external interrupt, and Target 1 goes to supervisor mode external interrupt.

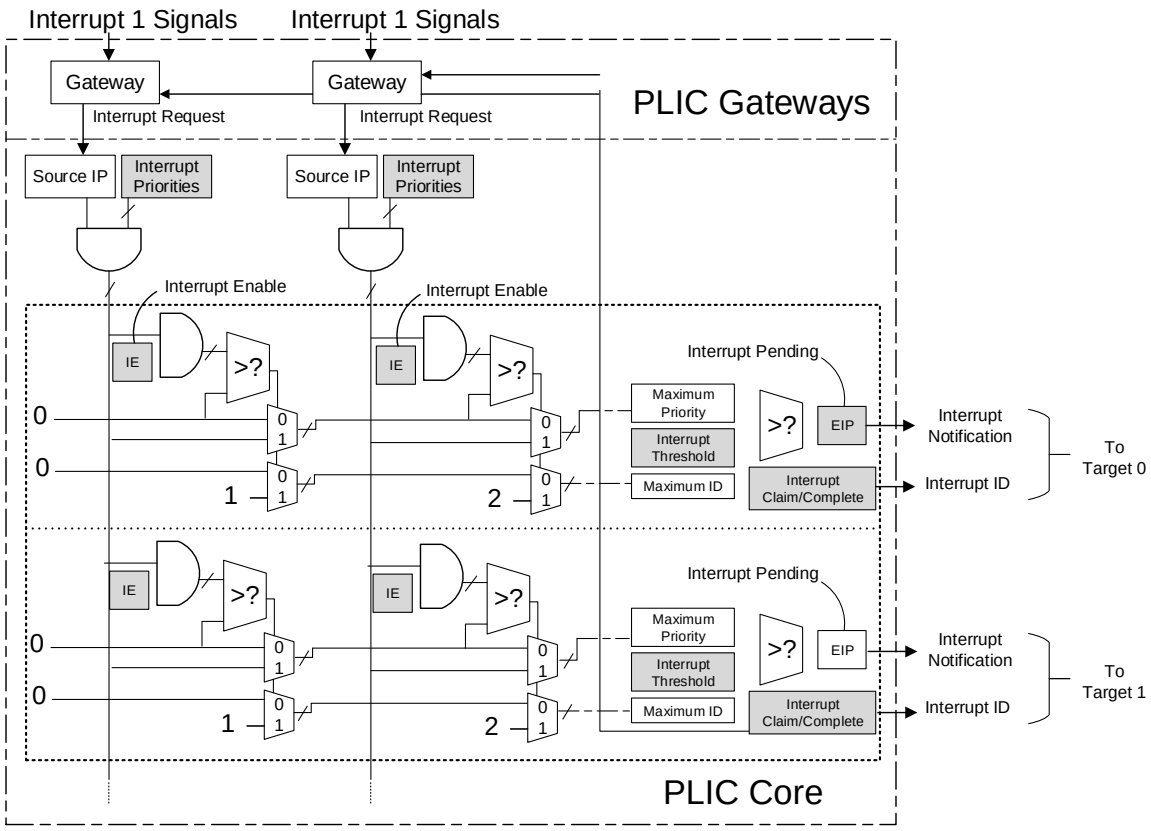

Figure 2.4. PLIC Operation Parameter Block Diagram

Following register blocks are defined in PLIC:

- Interrupt Priorities registers

Each PLIC interrupt source can be assigned a priority by writing to its 32-bit memory-mapped priority register. A priority value of 0 is reserved to mean ''never interrupt'' and effectively disables the interrupt. Priority 1 is the lowest active priority while the maximum level of priority depends on user settings. For example, the highest priority is 3 if width of PLIC Priority Register is set to 2. Ties between global interrupts of the same priority are broken by the Interrupt ID. Interrupts with the lowest ID have the highest effective priority.

The base address of Interrupt Source Priority block within PLIC Memory Map region is fixed at 0x000000.

- Interrupt Pending Bits registers

The current status of the interrupt source pending bits in the PLIC core can be read from the pending array, organized as 32-bit register. The pending bit for interrupt ID N is stored in bit (N mod 32) of word (N/32). Bit 0 of word 0, which represents the non-existent interrupt source 0, is hardwired to zero.

A pending bit in the PLIC core can be cleared by setting the associated enable bit then performing a claim.

The base address of Interrupt Pending Bits block within PLIC Memory Map region is fixed at 0x001000.

- Interrupt Enables registers

Each global interrupt can be enabled by setting the corresponding bit in the enables register. The enables registers are accessed as a contiguous array of 32-bit registers, packed the same way as the pending bits. Bit 0 of enable register 0 represents the non-existent interrupt ID 0 and is hardwired to 0. PLIC has 2 Interrupt Enable blocks, one for each context.

The context is referred to the specific privilege mode in the specific Hart of specific RISC-V processor instance.

For current IP, context 0 refers to hart 0 Machine mode and context 1 refers to hart 0 Supervisor mode.

The base address of Interrupt Enable Bits block within PLIC Memory Map region is fixed at 0x002000.

- Priority Thresholds registers

PLIC provides context based threshold register for the settings of an interrupt priority threshold of each context. The threshold register is a WARL field. The PLIC masks all PLIC interrupts of a priority less than or equal to threshold. For example, a threshold value of zero permits all interrupts with non-zero priority.

The base address of Priority Thresholds register block is located at 4K alignment starts from offset 0x200000.

- Interrupt Claim registers

The PLIC can perform an interrupt claim by reading the claim/complete register, which returns the ID of the highest priority pending interrupt or zero if there is no pending interrupt. A successful claim also can atomically clear the corresponding pending bit on the interrupt source.

The PLIC can perform a claim at any time and the claim operation is not affected by the setting of the priority threshold register.

The Interrupt Claim Process register is context based and is located at (4K alignment + 4) starts from offset 0x200000.

The register to acquire interrupt source ID of each context.

- Interrupt Completion registers

The PLIC signals the completion of executing an interrupt handler by host writing the interrupt ID received from the claim to the claim/complete register. The PLIC does not check whether or not the completion ID is the same as the last claim ID for that target. If the completion ID does not match an interrupt source that is currently enabled for the target, the completion is silently ignored.

The Interrupt Completion registers are context based and located at the same address with Interrupt Claim Process register, which is at (4K alignment + 4) starts from offset 0x200000.

Table 2.2 provides the descriptions of PLIC registers.

Table 2.2. PLIC Registers

<table>
<tr><th>Offset</th><th>Name</th><th>Description</th></tr>
<tr><td>0x00_0000</td><td></td><td>Reserved
(interrupt source 0 does not exist)</td></tr>
<tr><td>0x00_0004</td><td>PLIC_PRIORITY_SRC1</td><td>Interrupt source 1 priority

<table><tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr><tr><td>[31:3]</td><td>Reserved</td><td>RO</td><td>29</td><td>0x0</td></tr><tr><td>[2:0]</td><td>Priority</td><td>RW</td><td>3</td><td>0x0</td></tr></table>
Priority:
Sets the priority for a given global interrupt.</td></tr>
<tr><td>0x00_0008 ….
0x00_007C</td><td>PLIC_PRIORITY_SRC2
…
PLIC_PRIORITY_SRC31</td><td>Same as PLIC_PRIORITY_SRC1</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x00_1000</td><td>PLIC_PENDING1</td><td>PLIC Interrupt Pending Register 1 (pending1)<table><tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr><tr><td>[31:1]</td><td>PendingN</td><td>RO</td><td>31</td><td>0x0</td></tr><tr><td>[0]</td><td>Pending0</td><td>RO</td><td>1</td><td>0x0</td></tr></table>
Pending0:
Non-existent global interrupt 0 is hardwired to zero
PendingN:
Equal to PLIC_PENDING1[N], Pending bit for global interrupt N</td></tr>
</table>

<table>
<tr><th>Offset</th><th>Name</th><th>Description</th></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x00_2000</td><td>PLIC_ENABLE1_M</td><td>PLIC Interrupt Enable Register 1 (enable1) for Hart 0 M-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:1]</td><td>EnableN</td><td>RW</td><td>1</td><td>0x0</td></tr>
<tr><td>[0]</td><td>Enable0</td><td>RO</td><td>1</td><td>0x0</td></tr>
</table>
Enable0:
Non-existent global interrupt 0 is hardwired to zero
EnableN:
Equal to PLIC_ENABLE_M[N], enable bit for global interrupt N</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x00_2080</td><td>PLIC_ENABLE1_S</td><td>PLIC Interrupt Enable Register 1 (enable1) for Hart 0 S-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:1]</td><td>EnableN</td><td>RW</td><td>1</td><td>0x0</td></tr>
<tr><td>[0]</td><td>Enable0</td><td>RO</td><td>1</td><td>0x0</td></tr>
</table>
Enable0:
Non-existent global interrupt 0 is hardwired to zero
EnableN:
Equal to PLIC_ENABLE_M[N], enable bit for global interrupt N</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x20_0000</td><td>PLIC_THRESHOLD1_M</td><td>PLIC Interrupt Priority Threshold Register (threshold) for Hart 0 M-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:3]</td><td>Reserved</td><td>RO</td><td>29</td><td>0x0</td></tr>
<tr><td>[2:0]</td><td>Threshold</td><td>RW</td><td>3</td><td>0x0</td></tr>
</table>
Threshold:
Sets the priority threshold</td></tr>
<tr><td>0x20_0004</td><td>PLIC_CLAIM_1_M</td><td>PLIC Claim Register (claim) for Hart 0 M-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:0]</td><td>Claim</td><td>RO</td><td>32</td><td>0x0</td></tr>
</table>
Claim:
Read only field, which returns the ID of the highest priority pending interrupt or zero if there is no pending interrupt. A successful claim will also atomically clear the corresponding pending bit on the interrupt source.</td></tr>
<tr><td>0x20_0004</td><td>PLIC_COMPLETE_1_M</td><td>PLIC Complete Register (complete) for Hart 0 M-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:0]</td><td>Completion</td><td>WO</td><td>32</td><td>0x0</td></tr>
</table>
Completion:
Write only field, write to it to complete interrupt process</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x20_1000</td><td>PLIC_THRESHOLD1_S</td><td>PLIC Interrupt Priority Threshold Register (threshold) for Hart 0 S-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:3]</td><td>Reserved</td><td>RO</td><td>29</td><td>0x0</td></tr>
<tr><td>[2:0]</td><td>Threshold</td><td>RW</td><td>3</td><td>0x0</td></tr>
</table>
Threshold:</td></tr>
</table>

<table>
<tr><th>Offset</th><th>Name</th><th>Description</th></tr>
<tr><td></td><td></td><td>Sets the priority threshold</td></tr>
<tr><td>0x20_1004</td><td>PLIC_CLAIM_1_S</td><td>PLIC Claim Register (claim) for Hart 0 S-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:0]</td><td>Claim</td><td>RO</td><td>32</td><td>0x0</td></tr>
</table>
Claim:
Read only field, which returns the ID of the highest priority pending interrupt or zero if there is no pending interrupt. A successful claim will also atomically clear the corresponding pending bit on the interrupt source.</td></tr>
<tr><td>0x20_1004</td><td>PLIC_COMPLETE_1_S</td><td>PLIC Complete Register (complete) for Hart 0 S-Mode
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:0]</td><td>Completion</td><td>WO</td><td>32</td><td>0x0</td></tr>
</table>
Completion:
Write only field, write to it to complete interrupt process</td></tr>
</table>

2.2.2.2. CLINT

The CLINT module implements mtime, mtimecmp and some other memory-mapped CSR registers that associated with timer and software interrupts.

There are two clocks for CLINT. The msip register is clocked by the system clock, while the mtimecmp and mtime are clocked by real time clock which is typically 32 KHz for Lattice FPGA.

Table 2.3 provides the descriptions of CLINT registers.

Table 2.3. CLINT Registers

<table>
<tr><th>Offset</th><th>Name</th><th>Description</th></tr>
<tr><td>0x00_0000</td><td>CLINT_MSIP</td><td>MSIP register for hart 0
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:1]</td><td>Reserved</td><td>RO</td><td>31</td><td>0x0</td></tr>
<tr><td>[0]</td><td>msip</td><td>RW</td><td>1</td><td>0x0</td></tr>
</table>
msip:
reflect the memory-mapped MSIP bit of the mip CSR register.</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x00_4000</td><td>CLINT_MTIMECMP_L</td><td>Machine Timer register – mtimecmp
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:0]</td><td>mtimecmp_l</td><td>RW</td><td>32</td><td>Unchanged</td></tr>
</table>
mtimecmp_l:
lower 32 bits of mtimecmp CSR register. The first reset value is 0xFFFF_FFFF, after first write, the reset does not change the value of this field.</td></tr>
<tr><td>0x00_4004</td><td>CLINT_MTIMECMP_H</td><td>Machine Timer register – mtimecmp
<table>
<tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr>
<tr><td>[31:0]</td><td>mtimecmp_h</td><td>RW</td><td>32</td><td>Unchanged</td></tr>
</table>
mtimecmp_h:
higher 32 bits of mtimecmp CSR register. The first reset value is 0xFFFF_FFFF, after first write, the reset does not change the value of this field.</td></tr>
</table>

<table>
<tr><th>Offset</th><th>Name</th><th>Description</th></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>0x00_BFF8</td><td>CLINT_MTIME_L</td><td>Machine Timer register - mtime
<table><tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr><tr><td>[31:0]</td><td>mtime_l</td><td>RW</td><td>32</td><td>0x0</td></tr></table>
mtime_l:
lower 32 bits of mtime CSR register</td></tr>
<tr><td>0x00_BFFC</td><td>CLINT_MTIME_H</td><td>Machine Timer register - mtime
<table><tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr><tr><td>[31:0]</td><td>mtime_h</td><td>RW</td><td>32</td><td>0x0</td></tr></table>
mtime_h:
higher 32 bits of mtime CSR register</td></tr>
</table>

2.2.2.3. Watchdog Timer

The watchdog timer device (WDT) provides a simple two-stage timer controlled through one memory-mapped CSR register, WDCSR.

WDT waits a software-configured period of time with the expectation that system software re-initializes the watchdog state within this period of time. If this time period elapses without software re-init occurring, then a first-stage timeout register bit S1WTO is set within WDCSR that asserts an interrupt request output signal to notify the system of a stage 1 watchdog timeout. If a second period of time elapses without software re-init of the watchdog, then a second-stage timeout register bit (S2WTO) is set within WDCSR that generates a separate interrupt request output signal to notify the system of a stage 2 watchdog timeout.

For current IP, the stage 1 watchdog timeout is connected to PLIC input channel 1 (fixed) and stage 2 watchdog timeout is connected to system reset.

The mtime CSR register provides the time base for the watchdog timeout period. The timeout period itself – in units of watchdog clock tick – is specified by the WTOCNT field of the WDCSR CSR register. When WDCSR is written, the WTOCNT value initializes a down counter that decrements with each watchdog tick.

The watchdog tick occurs when bit 14 of mtime transitions from 0 to 1. So the watchdog timeout period is 0.512 second (based on real time clock of 32 KHz), and maximum timeout period (WTOCNT = 0x3FF) is about 524 seconds.

WDT is included in CLINT module. WDT shares the same base address (0xF200_0000) as that of the CLINT. Table 2.4 provides the descriptions of WDT registers.

Table 2.4. WDT Registers

<table>
<tr><th>Offset</th><th>Name</th><th>Description</th></tr>
<tr><td>0x00_D000</td><td>WDT_WDCSR</td><td>Watchdog register
<table><tr><th>Field</th><th>Name</th><th>Access</th><th>Width</th><th>Reset</th></tr><tr><td>[31:14]</td><td>Reserved</td><td>RO</td><td>18</td><td>0x0</td></tr><tr><td>[13:4]</td><td>WTOCNT</td><td>RW</td><td>10</td><td>0x0</td></tr><tr><td>[3]</td><td>S2WTO</td><td>RW</td><td>1</td><td>0x0</td></tr><tr><td>[2]</td><td>S1WTO</td><td>RW</td><td>1</td><td>0x0</td></tr><tr><td>[1]</td><td>Reserved</td><td>RW</td><td>1</td><td>0x0</td></tr><tr><td>[0]</td><td>WDEN</td><td>RW</td><td>1</td><td>0x0</td></tr></table></td></tr>
</table>

Offset	Name	Description
		WDEN: When set, enables the WDT. When clear, the WDT is disabled and S1WTO and S2WTO output signals are forced to be zero (de-asserted). When system reset is asserted, WDT is disabled accordingly by setting WDEN to 0. S1WTO: stage 1 watchdog timeout, active high. S2WTO: stage 2 watchdog timeout, active high. WTOCNT: 10 bit timeout counter. If it is non-zero and WDEN is set, it decrements every timeout period.

2.2.2.4. TCM

Tightly Coupled Memory (TCM) is a good candidate if on chip memory is sufficient enough for the application. TCM can offer better performance, timing result and some other benefits. For this IP, TCM is the only supported memory for instruction memory and default memory for data memory. Data memory can be further extended with external main memory such as on-chip DRAM.

Both Instruction TCM (ITCM) and Data TCM (DTCM) are implemented based on EBR block and can support up to 128 KB per configuration.

2.2.2.5. UART

There is an optional UART as local slave. This UART has fixed memory assignment.

2.3. Signal Description

Table 2.5 to Table 2.7 list the ports of the CPU soft IP in different categories.

2.3.1. sysClock and Reset

Table 2.5. Clock and Reset Ports

Name	Direction	Width	Description
clk_system_i	In	1	High speed system clock input.
clk_realtime_i	In	1	Low speed real time clock input.
rstn_i	In	1	System reset (active low).
system_resetn_o	Out	1	Combined system reset and Debug Reset from JTAG.
uart_rxd_i	In	1	Input rxd for local UART. Only available when UART_EN enabled.
uart_txd_o	Out	1	Output txd for local UART. Only available when UART_EN enabled.

2.3.2. Data Interface

Table 2.6. Data Ports (AXI)

Name	Direction	Width	Group	Description
AXI_MST_DATA_AWREADY	In	1	AXI4 Master Write Address Channel	—
AXI_MST_DATA_AWVALID	Out	1		—
AXI_MST_DATA_AWADDR	Out	32		—
AXI_MST_DATA_AWLEN	Out	8		—
AXI_MST_DATA_AWSIZE	Out	3		—

Name	Direction	Width	Group	Description
AXI_MST_DATA_AWBURST	Out	2		Fixed 2’b01
AXI_MST_DATA_AWLOCK	Out	1		Fixed 1’b0
AXI_MST_DATA_AWCACHE	Out	4		Fixed 4’b1111
AXI_MST_DATA_AWPROT	Out	3		Fixed 3'b010
AXI_MST_DATA_AWQOS	Out	4		Fixed 4'b0000
AXI_MST_DATA_AWREGION	Out	4		Fixed 4'b0000
AXI_MST_DATA_AWID	Out	1		Fixed 1’b0
AXI_MST_DATA_WREADY	In	1	AXI4 Master Write Data Channel	—
AXI_MST_DATA_WVALID	Out	1		—
AXI_MST_DATA_WDATA	Out	32		—
AXI_MST_DATA_WLAST	Out	1		—
AXI_MST_DATA_WSTRB	Out	4		—
AXI_MST_DATA_BVALID	In	1	AXI4 Master Write Response Channel	—
AXI_MST_DATA_BRESP	In	2		Not used
AXI_MST_DATA_BID	In	1		Not used
AXI_MST_DATA_BREADY	Out	1		—
AXI_MST_DATA_ARVALID	In	1	AXI4 master Read Address Channel	—
AXI_MST_DATA_ARREADY	Out	1		—
AXI_MST_DATA_ARCACHE	Out	4		Fixed 4'b1111
AXI_MST_DATA_ARPROT	Out	3		Fixed 3'b010
AXI_MST_DATA_ARQOS	Out	4		Fixed 4'b0000
AXI_MST_DATA_ARREGION	Out	4		Fixed 4'b0000
AXI_MST_DATA_ARID	Out	1		Fixed 1'b0
AXI_MST_DATA_ARADDR	Out	32		—
AXI_MST_DATA_ARLEN	Out	8		—
AXI_MST_DATA_ARSIZE	Out	3		—
AXI_MST_DATA_ARBURST	Out	2		Fixed 2'b01
AXI_MST_DATA_ARLOCK	Out	1		Fixed 1’b0
AXI_MST_DATA_RID	In	1	AXI4 Master Read Data Channel	Not used
AXI_MST_DATA_RDATA	In	32		—
AXI_MST_DATA_RRESP	In	2		—
AXI_MST_DATA_RLAST	In	1		Not used
AXI_MST_DATA_RVALID	In	1		—
AXI_MST_DATA_RREADY	Out	1		—

2.3.3. Interrupt interface

Table 2.7. Interrupt Ports

Name	Type	Width	Description
EXT_IRQ_Sx	In	2 ~ 14	Peripheral interrupts.

2.4. Attribute Summary

The configurable attributes are shown in Table 2.8 and are described in Table 2.9.

The attributes can be configured through the Propel Builder software.

Table 2.8. Configurable Attributes

Attribute	Selectable Values	Default	Dependency on Other Attributes
General			
Processor Mode	Balanced	Balanced	—
Debug Enable	Disabled, Enabled	Enabled	—
JTAG Channel Selection	14, 15, 16	14	—
Number of User Interrupt Requests	2 ~ 14	8	—
Interrupt for Supervisor Mode	Disabled, Enabled	Disabled	—
Width of PLIC priority register	2, 3	3	—
TCM Size (KB)	16, 32, 64,128K	16	—
Reset Assertion	sync, async	sync	—
Read Port: Enable Output Register(A)	Disabled, Enabled	Disabled	—
Read Port: Enable Output Register(B)	Disabled, Enabled	Disabled	—
Memory Initialization	None, Initialize to all 0s, Initialize to all 1s, Memory File	None	—
Memory File	None, file path	None	—
Memory File Format	Hex, binary	Hex	—
Enable UART instance	Disabled, Enabled	Disabled	—
System Clock Frequency (Mhz)	2 - 200	100	—
Serial Data Width	5, 6, 7, 8	8	—
Stop Bits	0, 1	1	—
Parity Enable	Disabled, Enabled	Disabled	—
UART Standard Baud Rate	2400, 4800, 9600, 14400, 19200, 28800, 38400, 56000, 57600, 115200	115200	—

Table 2.9. Attributes Description

Attribute	Description
Processor Mode	Balanced Mode offers RV32IMC with some other feature sets (not user accessible) to get most comprehensive performance.
Debug Enable	Whether to enable Debug module or not.
JTAG Channel Selection	Select the channel of harden JTAG block.
Number of User Interrupt Requests	Number of Interrupt for peripherals.
Interrupt for Supervisor Mode	Whether or not to enable interrupt for supervisor mode. If not, then all external interrupts go to machine mode only.
Width of PLIC priority register	Data width of PLIC priority register. Default to 3 bits, so 8 priority levels total.
TCM Size (KB)	Total size of TCM.
Reset Assertion	Configure the reset mode to be either sync or async.
Read Port: Enable Output Register(A)	Enable output register for Instruction port.
Read Port: Enable Output Register(B)	Enable output register for Data port.
Memory Initialization	Initialize the content of TCM to be 0, 1, or from memory file.
Memory File	File path of memory file.
Memory File Format	Format of memory file.
Enable UART instance	Disabled, Enabled.
System Clock Frequency (Mhz)	Specifies the target frequency of system clock. This is used for baud rate calculation.
Serial Data Width	Specifies the default data bit width of UART transactions.
Stop Bits	Specifies the default number of stop bits to be transmitted and received.
Parity Enable	Specifies the absence/presence of parity.
UART Standard Baud Rate	Selects between Standard Baud Rate and Custom Baud Rate for the reset value of Divisor Latch Register. The selected Baud rate is used to set the reset value of Divisor Latch Register as follows: {DLR_MSB, DLR_LSB} = System Clock Frequency (MHz) x 1000000 / Selected Baud Rate

2.5. Memory Map

To achieve better overall performance, this IP separates the whole 4G memory range into several sections with some suggestions, refer to Table 2.10.

Table 2.10. SOC Memory Map

Base Address	Range	End Address	Description
Region #0 (0x0000_0000 - 0x0FFF_FFFF) - RISC-V RX IP			
0x0000_0000	128KB	0x0001_FFFF	TCM
0x0002_0000	—*	0x0FFF_FFFF	User Memory extension
Region #15 (0xF000_0000 - 0xFFFF_FFFF) - RISC-V RX IP			
0xF000_0000	1KB	0xF000_03FF	Local UART when UART_EN asserted; Otherwise, Reserved
0xF000_0400	32767KB	0xF1FF_FFFF	Reserved
0xF200_0000	1024KB	0xF20F_FFFF	CLINT & Watchdog timer
0xF210_0000	NA	0xFBFF_FFFF	Reserved
0xFC00_0000	4096KB	0xFC3F_FFFF	PLIC
0xFC40_0000	NA	0xFFFF_FFFF	Reserved
Region #1 (0x1000_0000 - 0x1FFF_FFFF)			
0x1000_0000	1KB	0x1000_03FF	UART
0x1000_1000	1KB	0x1000_13FF	GPIO
…	—*	—*	—*
Region #2 (0x2000_0000 - 0x2FFF_FFFF)			
…	—*	—*	—*

***Note**: The actual valid base address/range/end address is determined by the user SOC design.

The total 4G memory space is divided into 16 256-MB regions to ease potential (future) PMP settings.

Processor cache range is 0x0000_0000 to 0x0FFF_FFFF, so the region #0 is the only region for cacheable components. The first 128 KB (0x0000_0000 to 0x0001_FFFF) are reserved for TCM. The remaining spaces are for user external memory extension – either on chip EBR based memory or off chip memory like flash and sdram.

The region #15 is reserved for RISC-V RX IP – local UART, CLINT, Watchdog timer and PLIC are assigned to this region.

All the other regions are for user extension.

3. RISC-V RX CPU IP Generation

This section provides information on how to generate the CPU IP Core module using Propel Builder.

To generate the CPU IP Core module:

1. In Propel Builder, create a new design. Select the CPU package.
2. Enter the component name as shown in Figure 3.1. Click **Next**.

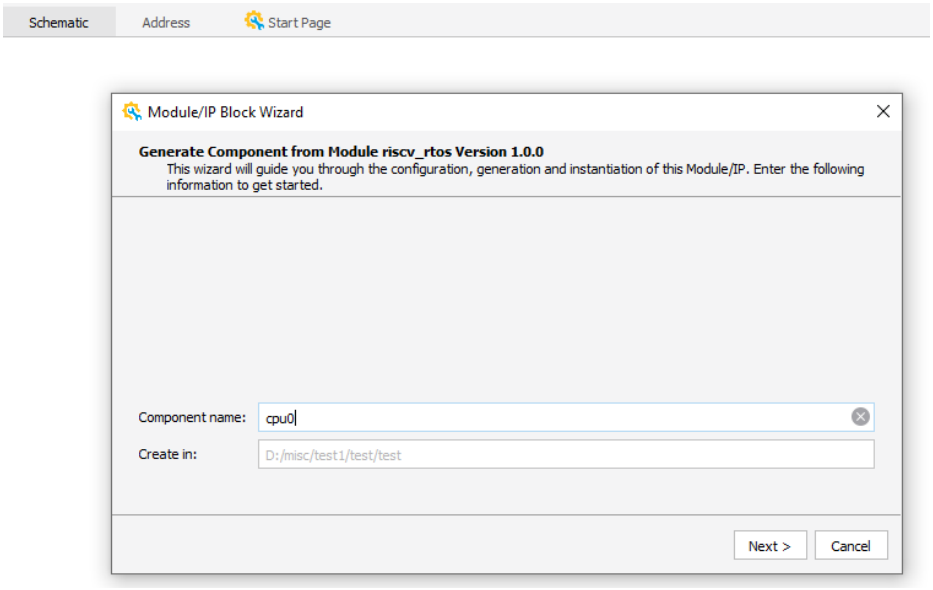

Figure 3.1. Entering Component Name

3. Configure the parameters as shown in Figure 3.2. Click **Generate**.

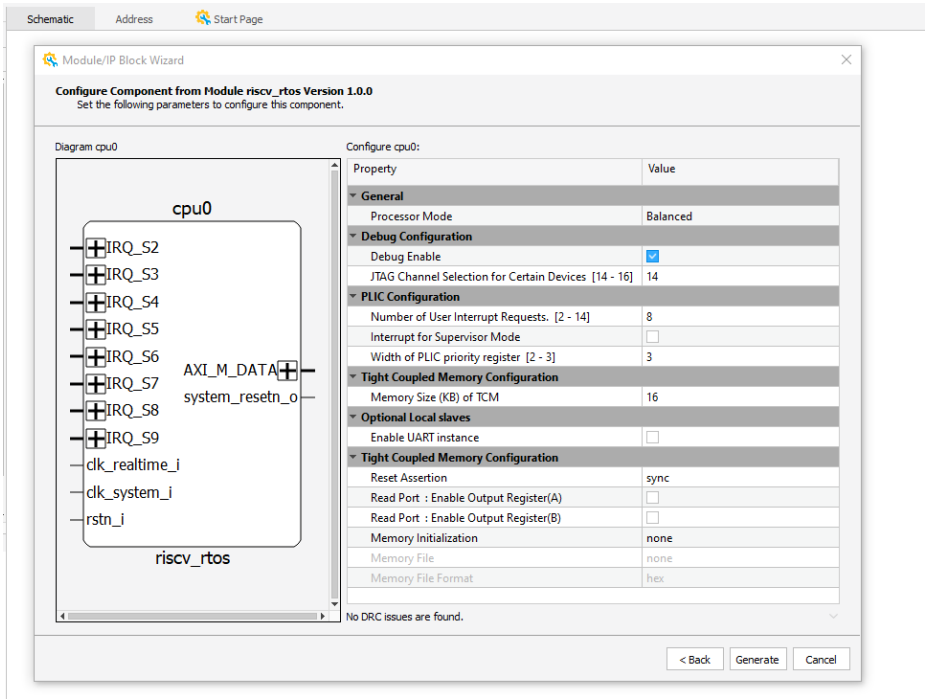

Figure 3.2. Configuring Parameters

4. Verify the information. Click **Finish**.

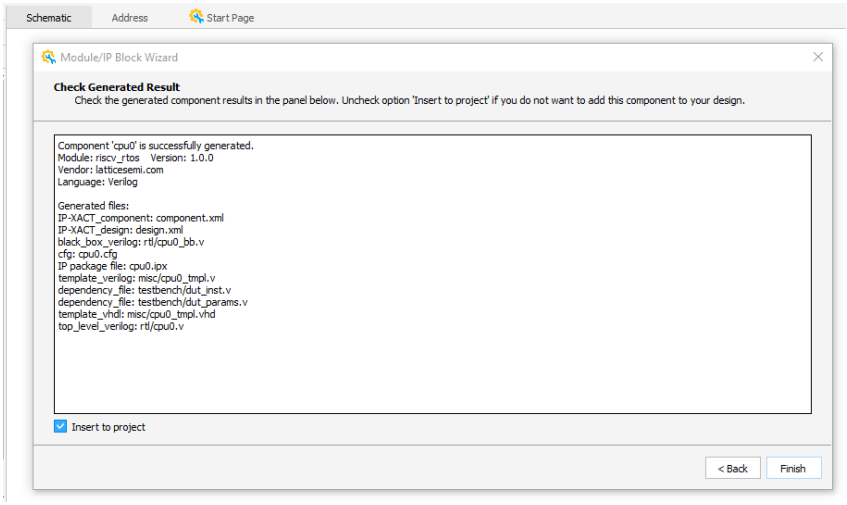

Figure 3.3. Verifying Results

5. Confirm or modify the module instance name. Click **OK**.

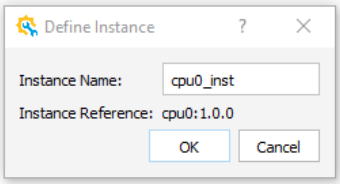

Figure 3.4. Specifying Instance Name

The CPU IP instance is successfully generated as shown in Figure 3.5.

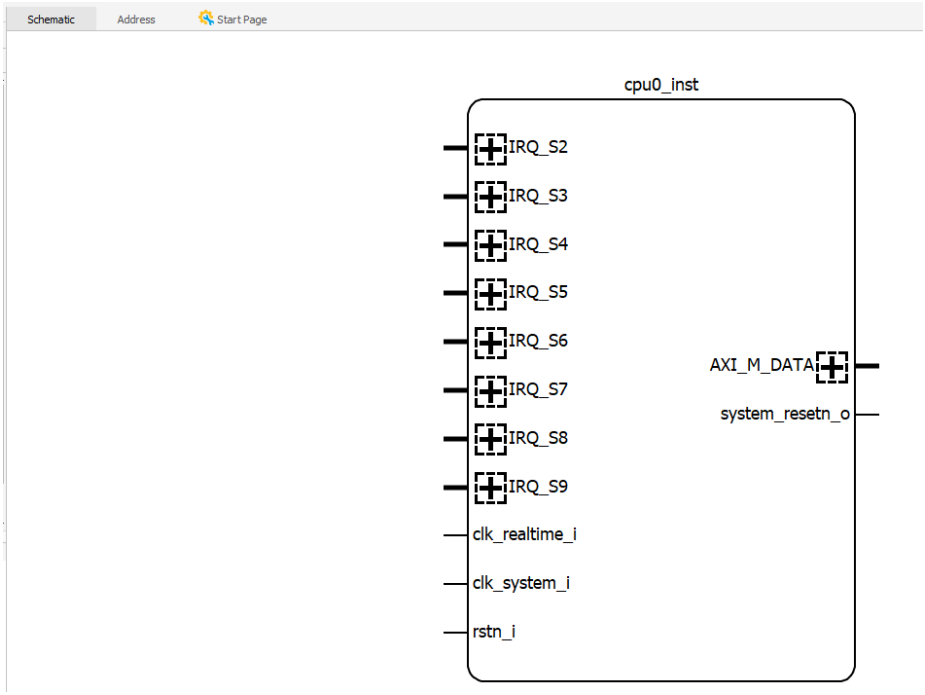

Figure 3.5. Generated Instance

Appendix A. Resource Utilization

Table A.1. Resource Utilization in CertusPro-NX Device

Configuration	LUTs	Registers	sysMEM EBRs
Processor core	4055	1969	16
Processor core + PLIC + CLINT + Watchdog + TCM (TCM = 128K, default configurations for other options)	5025	2666	80

Note: Resource utilization characteristics are generated using Lattice Radiant software.

References

- The RISC-V Instruction Set Manual Volume I: Unprivileged ISA (20191213)
- The RISC-V Instruction Set Manual Volume II: Privileged Architecture (20190608)
- RISC-V Platform Specification Version 0.2
- RISC-V Platform-Level Interrupt Controller Specification v1.0. 09.2021
- SiFive Interrupt Cookbook v1.2
- RISC-V Watchdog Timer Specification Version 1.0-draft-0.5
- AMBA 3 AHB-Lite Protocol V1.0
- AMBA AXI and ACE Protocol Specification vF.b
- Lattice Propel Builder 2.2 User Guide (FPGA-UG-02156)

Technical Support Assistance

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

Revision History

Revision 1.0, June 2022

Section	Change Summary
All	Production release.

www.latticesemi.com