



# Ruby Vision RISC-V SoC

## Data Sheet

---

DS-RUBYV-v1.1  
March 2021  
[www.elitestek.com](http://www.elitestek.com)



# Contents

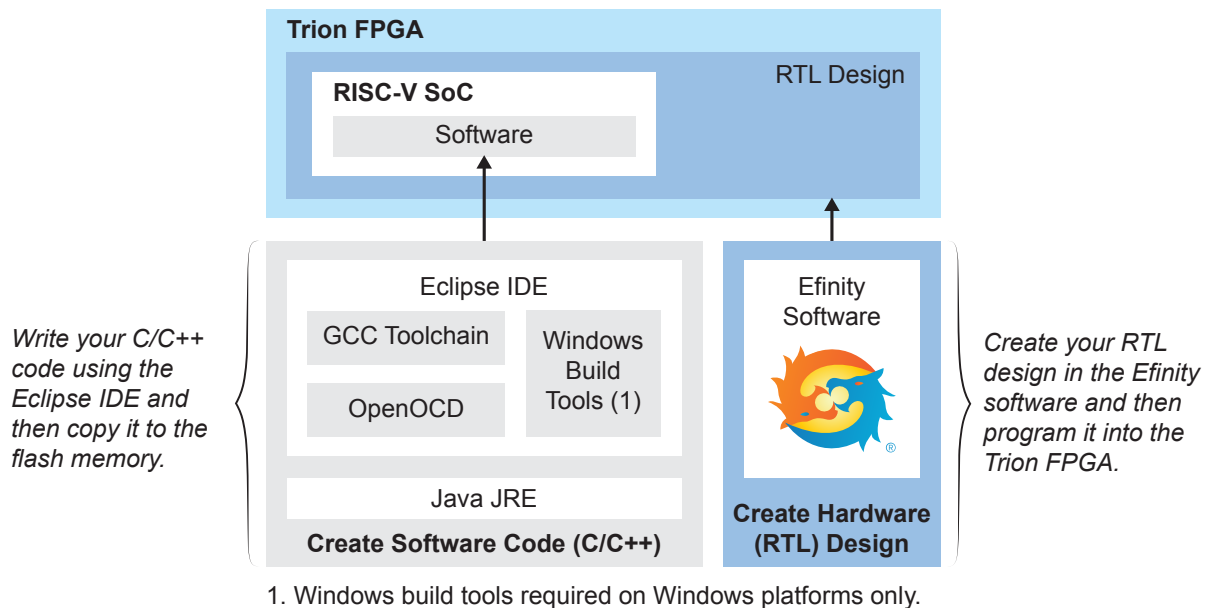
<b>Introduction.....</b>	<b>4</b>
VexRiscv RISC-V Core.....	4
<b>Features.....</b>	<b>5</b>
<b>Functional Description.....</b>	<b>6</b>
Ruby Vision SoC Address Map.....	7
Flash Address.....	7
Clocks.....	8
Interrupts.....	8
Resets.....	8
AXI Interface.....	9
APB3 Interface.....	13
JTAG Interface.....	14
GPIO Peripheral Interface.....	15
Input Register: 0x0000_0000.....	15
Output Register: 0x0000_0004.....	15
Output Enable Register: 0x0000_0008.....	15
Interrupt Rise Enable Register: 0x0000_0020.....	16
Interrupt Fall Enable Register: 0x0000_0024.....	16
Interrupt High Enable Register: 0x0000_0028.....	16
Interrupt Low Enable Register: 0x0000_002C.....	16
I <sup>2</sup> C Peripheral Interface.....	17
txData Register: 0x0000_0000.....	18
txAck Register: 0x0000_0004.....	18
rxData Register: 0x0000_0008.....	19
rxAck Register: 0x0000_000C.....	19
Interrupt Register: 0x0000_0020.....	20
Interrupt Clears Register: 0x0000_0024.....	21
Timeout Register: 0x0000_002C.....	21
Sampling Clock Divider Register: 0x0000_0028.....	21
tsuData Register: 0x0000_0030.....	21
Master Status Register: 0x0000_0040.....	22
tLow Register: 0x0000_0050.....	22
tHigh Register: 0x0000_0054.....	22
tBuf Register: 0x0000_0058.....	22
Filtering Status Register: 0x0000_0080.....	23
Hit Context Register: 0x0000_0084.....	23
PLIC Peripheral Interface.....	23
SPI Master Peripheral Interface.....	24
Cmd Register: 0x0000_0000.....	25
RSP Register: 0x0000_0004.....	25
Config Register: 0x0000_0008.....	25
Interrupt Register: 0x0000_000C.....	26
clockDivider Register: 0x0000_0020.....	26
ssSetup Register: 0x0000_0024.....	26
ssHold Register: 0x0000_0028.....	26
ssDisable Register: 0x0000_002C.....	27
ssActiveHigh Register: 0x0000_0030.....	27
UART Peripheral Interface.....	27
Control and Status Registers.....	28
Machine-Level ISA.....	28

Hart ID Register (mhartid): 0xF14.....	28
Machine Status Register (mstatus): 0x300.....	29
Machine Trap-Vector Base-Address Register (mtvec): 0x305.....	29
Machine Interrupt Enable Register (mie): 0x304.....	29
Machine Exception Program Counter (mepc): 0x341.....	30
Machine Cause Register (mcause): 0x342.....	30
Machine Trap Value Register (mtval): 0x343.....	31
Machine Interrupt Pending Register (mip): 0x344.....	31
<b>Revision History.....</b>	<b>32</b>

# Introduction

易灵思 provides the heavy-weight, cached soft RISC-V SoC, Ruby Vision, which includes a DDR DRAM controller interface. This SoC is ideal for applications that perform image signal processing such as embedded vision, industrial control, and multiple microphone voiceprint recognition, which require high performance with hardware acceleration. This core is similar to the open-source SaxonSOC, but it has been optimized for Trion® FPGAs. The Ruby Vision SoC is similar to the standard Ruby SoC, except it has additional APB3 user peripherals.

Figure 1: Ruby Vision RISC-V SoC Design Flow



**Learn more:** For details on developing RTL designs or creating software, refer to the Edge Vision Accelerator User Guide.

## VexRiscv RISC-V Core

The Ruby Vision SoC is based on the VexRiscv core created by Charles Papon. The VexRiscv core is a 32-bit CPU using the ISA RISC-V32I with M and C extensions, has five pipeline stages (fetch, decode, execute, memory, and writeback), and a configurable feature set. In the Ruby Vision SoC, the VexRiscv core supports the AXI4 and APB3 bus interfaces, has instruction and data caches, and can run at speeds up to 1.16 DMIPS/MHz.

The VexRiscv core won first place in the RISC-V SoftCPU contest in 2018.<sup>(1)</sup>

<sup>(1)</sup> <https://www.businesswire.com/news/home/20181206005747/en/RISC-V-SoftCPU-Contest-Winners-Demonstrate-Cutting-Edge-RISC-V>

# Features

- VexRiscv processor with 5 pipeline stages (fetch, decode, execute, memory, and write back), interrupts and exception handling with machine mode
  - data cache
  - instruction cache
- system clock frequency
- on-chip RAM with boot loader for SPI flash
- DDR controller
  - 100 MHz DDR memory bus frequency
  - 128-bit AXI data width
  - 3.5 GB memory module
  - 400 MHz DDR clock frequency, 800 Mbps
- 2 AXI master channels to access the DDR memory
- 1 AXI slave channel to user logic
- APB3 peripherals:
  - GPIOs
  - I<sup>2</sup>C masters and slaves
  - Machine timer
  - PLIC
  - 
  - 
  -

## FPGA Support

The Ruby Vision SoC supports T35, T55, T85, and T120 Trion<sup>®</sup> FPGAs.

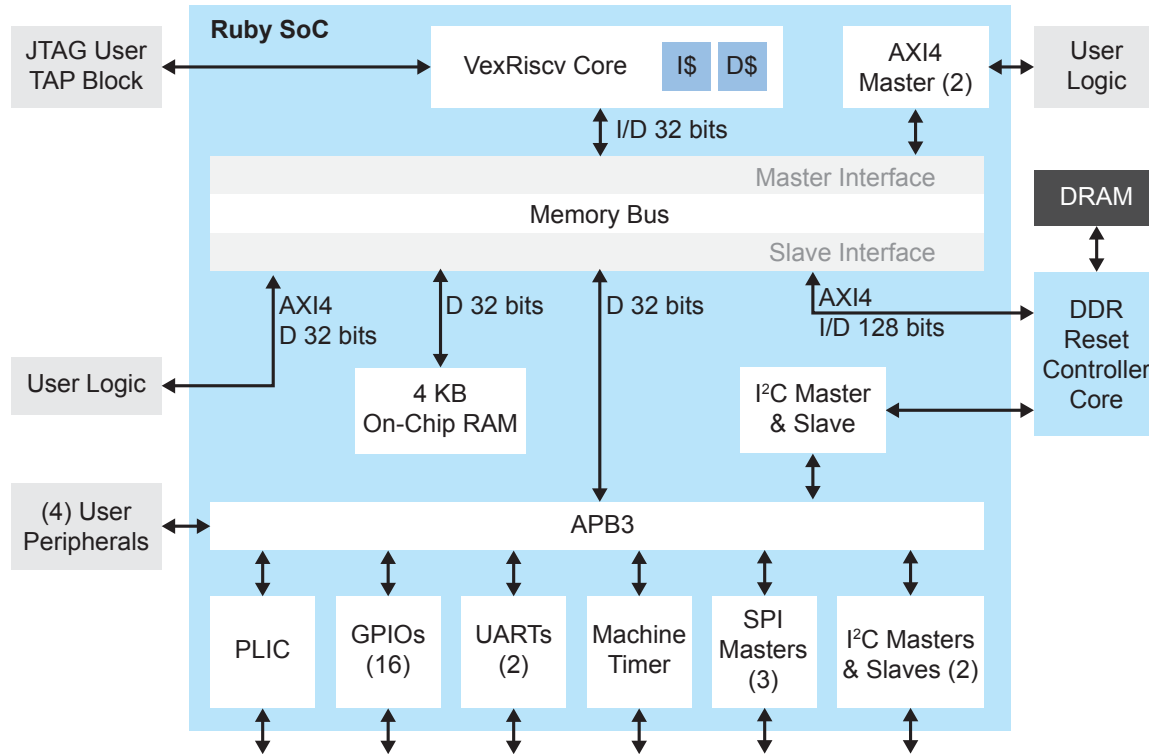
## Resource Utilization and Performance

FPGA	Logic Utilization (LUTs)	Memory Blocks	f <sub>MAX</sub> (MHz)	Language	Efinity Version
T120 BGA324 C4	5,987	77	SoC: 53.229 DDR: 100	Verilog HDL	2020.1

# Functional Description

The Ruby Vision SoC incorporates a 32-bit RISC-V processor, 4 KB instruction cache, 4 KB data cache, 4 KB of on-chip RAM, and a variety of peripherals (including 2 APB3 slave peripherals and 1 AXI slave). The SoC includes 3 I<sup>2</sup>C peripherals and a 128-bit half-duplex AXI bus to communicate with the 易灵思 DDR controller core. This core uses the Trion<sup>®</sup> FPGAs hard DDR DRAM interface to reset an external DRAM module (resets and re-initializes the Trion<sup>®</sup> FPGA's DDR interface as well as the DDR module(s)).

Figure 2: Ruby Vision SoC Block Diagram



## Ruby Vision SoC Address Map

The following table shows the address map for the RISC-V processor.



**Note:** Because the address range might be updated, 易灵思 recommends that you always refer to the parameter name when referencing an address in firmware, not by the actual address. The parameter names are defined in **soc.h**.

**Table 1: Default Address Map, Interrupt ID, and Cached Channels**

The AXI user slave channel is in a cacheless region (I/O) for compatibility with AXI-Lite.

Device	Parameter	Size	Interrupt ID	Region
Off-chip DRAM	SYSTEM_DDR_BMB	3.5 GBytes	-	Cache
GPIO	SYSTEM_GPIO_0_IO_APB	4K	[0]: 12 [1]: 13	I/O
I <sup>2</sup> C 0	SYSTEM_I2C_0_IO_APB	4K	8	I/O
I <sup>2</sup> C 1	SYSTEM_I2C_1_IO_APB	4K	9	I/O
I <sup>2</sup> C 2	SYSTEM_I2C_2_IO_APB	4K	10	I/O
Machine timer	SYSTEM_MACHINE_TIMER_APB	4K	31	I/O
PLIC	SYSTEM_PLIC_APB	4K	-	I/O
SPI master 0	SYSTEM_SPI_0_IO_APB	4K	4	I/O
SPI master 1	SYSTEM_SPI_1_IO_APB	4K	5	I/O
SPI master 2	SYSTEM_SPI_2_IO_APB	4K	6	I/O
UART 0	SYSTEM_UART_0_IO_APB	4K	1	I/O
UART 1	SYSTEM_UART_1_IO_APB	4K	2	I/O
User peripheral 0	IO_APB_SLAVE_0_APB	64K	-	I/O
User peripheral 1	IO_APB_SLAVE_1_APB	64K	-	I/O
User peripheral 2	IO_APB_SLAVE_2_APB	64K	-	I/O
User peripheral 3	IO_APB_SLAVE_3_APB	64K	-	I/O
On-chip BRAM	SYSTEM_RAM_A_BMB		-	Cache
AXI user slave	SYSTEM_AXI_A_BMB	16 MB	-	I/O
External interrupt	-	-	25	I/O



**Note:** The RISC-V GCC compiler does not support user address spaces starting at 0x0000\_0000.

## Flash Address

When the FPGA boots up, the SoC copies your binary application file from a SPI flash device to the DDR DRAM module, and then begins execution. The SPI flash binary address starts at .

## Clocks

Table 2: Clock Ports

Port	Direction	Description
io_systemClock	Input	Provides a 50 MHz clock for the SoC.
io_memoryClock	Input	Provides a 100 MHz clock for the DDR DRAM memory bus.

## Interrupts

Table 3: Interrupt Ports

Port	Direction	Description
userInterruptA	Input	Provides an external interrupt.
io_axiAInterrupt	Input	User AXI slave channel interrupt.

## Resets

The Ruby Vision SoC has as master reset signal, `io_asyncReset` that triggers a system reset. Your RTL design should hold `io_asyncReset` for 10 ns to reset the whole SoC system completely. When you assert `io_asyncReset`, the SoC asserts:

- `io_systemReset`, which resets the RISC-V processor, on-chip memory, and peripherals
- `io_memoryReset`, which resets the DDR controller, DRAM module, I<sup>2</sup>C master and slave connected to the DDR reset controller core, and any user logic
- `io_ddrMasters_0_reset`, which responds to the reset for each DDR master and is synchronized to the `io_ddrMasters_0_clk`

The SoC asserts the `io_memoryReset` and `io_ddrMaster_0_reset` signals at the same time to allow the AXI masters access to the AXI cross bar once the reset completes.

Once `io_systemReset` goes low, the user binary code is executed.

Table 4: Reset Ports

Port	Direction	Description
io_asyncReset	Input	Active-high asynchronous reset for the entire system.
io_systemReset	Output	Synchronous active-high reset for systemClk.
io_memoryReset	Output	DDR reset source from the RISC-V SoC.
io_ddrMasters_0_reset	Output	Responds to the reset for the DDR master.



## AXI Interface

The Ruby Vision SoC has a 128-bit slave interface to communicate with the DDR controller core. Additionally it has an AXI slave to connect to user logic. The AXI user slave channel is compatible with AXI-Lite. Refer to the AMBA AXI and ACE Protocol Specification for AXI channel descriptions and handshake information.

### AXI Interface to DDR DRAM

**Table 5: DDR Slave Half-Duplex Address Channel for Read and Write**

Port	Direction	Description
io_ddrA_arw_valid	Output	DDR address valid.
io_ddrA_arw_ready	Input	DDR address ready.
io_ddrA_arw_payload_addr[31:0]	Output	DDR address.
io_ddrA_arw_payload_id[7:0]	Output	DDR address ID.
io_ddrA_arw_payload_region[3:0]	Output	DDR region identifier.
io_ddrA_arw_payload_len[7:0]	Output	DDR burst length.
io_ddrA_arw_payload_size[2:0]	Output	DDR burst size.
io_ddrA_arw_payload_burst[1:0]	Output	DDR burst type, INCR only.
io_ddrA_arw_payload_lock	Output	DDR lock type.
io_ddrA_arw_payload_cache[3:0]	Output	DDR memory type.
io_ddrA_arw_payload_qos[3:0]	Output	DDR quality of service.
io_ddrA_arw_payload_prot[2:0]	Output	DDR protection type.
io_ddrA_arw_payload_write	Output	DDR address read/write selection: 0: Read 1: Write

**Table 6: DDR Slave Write Data Channel**

Port	Direction	Description
io_ddrA_w_valid	Output	DDR write valid.
io_ddrA_w_ready	Input	DDR write ready.
io_ddrA_w_payload_data[127:0]	Output	DDR write data.
io_ddrA_w_payload_strb[15:0]	Output	DDR write strobe.
io_ddrA_w_payload_last	Output	DDR write last.

**Table 7: DDR Slave Write Respond Channel**

Port	Direction	Description
io_ddrA_b_valid	Input	DDR write respond valid.
io_ddrA_b_ready	Output	DDR respond ready.
io_ddrA_b_payload_id[7:0]	Input	DDR respond ID.
io_ddrA_b_payload_resp[1:0]	Input	DDR write respond.

**Table 8: DDR Slave Read Data Channel**

Port	Direction	Description
io_ddrA_r_valid	Input	DDR read valid.
io_ddrA_r_ready	Output	DDR read ready.
io_ddrA_r_payload_data[127:0]	Input	DDR read data.
io_ddrA_r_payload_id[7:0]	Input	DDR read ID.
io_ddrA_r_payload_resp[1:0]	Input	DDR read respond.
io_ddrA_r_payload_last	Input	DDR read last.

## AXI Interface to User Logic

**Table 9: User Slave Write Address Channel**

Port	Direction	Description
axiA_awvalid	Output	User write address valid.
axiA_awready	Input	User write address ready.
axiA_awaddr[31:0]	Output	User write address.
axiA_awid[7:0]	Output	User write address ID.
axiA_awregion[3:0]	Output	User region identifier.
axiA_awlen[7:0]	Output	User burst length.
axiA_awsz[2:0]	Output	User burst size.
axiA_awburst[1:0]	Output	User burst type, INCR only.
axiA_awlock	Output	User lock type.
axiA_awcache[3:0]	Output	User memory type.
axiA_awqos[3:0]	Output	User quality of service.
axiA_awprot[2:0]	Output	User protection type.

**Table 10: User Slave Write Data Channel**

Port	Direction	Description
axiA_wvalid	Output	User write valid.
axiA_wready	Input	User write ready.
axiA_wdata[31:0]	Output	User write data.
axiA_wstrb[3:0]	Output	User write strobe.
axiA_wlast	Output	User write last.

**Table 11: User Slave Write Respond Channel**

Port	Direction	Description
axiA_bvalid	Input	User write respond valid.
axiA_bready	Output	User respond ready.
axiA_bid[7:0]	Input	User respond ID.
axiA_bresp[1:0]	Input	User write respond.

**Table 12: User Slave Read Address Channel**

Port	Direction	Description
axiA_arvalid	Output	User read address valid.
axiA_arready	Input	User read address ready.
axiA_araddr[31:0]	Output	User read address.
axiA_arid[7:0]	Output	User read address ID.
axiA_arregion[3:0]	Output	User region identifier.
axiA_arlen[7:0]	Output	User burst length.
axiA_arsize[2:0]	Output	User burst size.
axiA_arburst[1:0]	Output	User burst type, INCR only.
axiA_arlock	Output	User lock type.
axiA_arcache[3:0]	Output	User memory type.
axiA_arqos[3:0]	Output	User quality of service.
axiA_arprot[2:0]	Output	User protection type.

**Table 13: User Slave Read Data Channel**

Port	Direction	Description
axiA_rvalid	Input	User read valid.
axiA_rready	Output	User read ready.
axiA_rdata[31:0]	Input	User read data.
axiA_rid[7:0]	Input	User read ID.
axiA_rresp[1:0]	Input	User read respond.
axiA_rlast	Input	User read last.

**Table 14: User Master Clock and Reset**

Port	Direction	Description
io_ddrMasters_0_clk	Input	AXI master clock.
io_ddrMasters_0_reset	Output	AXI master active high reset.

## AXI Master Interface

**Table 15: User Master Write Address Channel**

Where  $n$  is the channel number.

Port	Direction	Description
io_ddrMasters_n_aw_valid	Input	User write address valid.
io_ddrMasters_n_aw_ready	Output	User write address ready.
io_ddrMasters_n_aw_payload_addr[31:0]	Input	User write address.
io_ddrMasters_n_aw_payload_id[7:0]	Input	User write address ID.
io_ddrMasters_n_aw_payload_region[3:0]	Input	User region identifier.
io_ddrMasters_n_aw_payload_len[7:0]	Input	User burst length.

Port	Direction	Description
io_ddrMasters_n_aw_payload_size[2:0]	Input	User burst size.
io_ddrMasters_n_aw_payload_burst[1:0]	Input	User burst type, INCR only.
io_ddrMasters_n_aw_payload_lock	Input	User lock type.
io_ddrMasters_n_aw_payload_cache[3:0]	Input	User memory type.
io_ddrMasters_n_aw_payload_qos[3:0]	Input	User quality of service.
io_ddrMasters_n_aw_payload_prot[2:0]	Input	User protection type.

**Table 16: User Master Write Data Channel**

Where  $n$  is the channel number.

Port	Direction	Description
io_ddrMasters_n_w_valid	Input	User write valid.
io_ddrMasters_n_w_ready	Output	User write ready.
io_ddrMasters_n_w_payload_data[127:0]	Input	User write data.
io_ddrMasters_n_w_payload_strb[15:0]	Input	User write strobe.
io_ddrMasters_n_w_payload_last	Input	User write last.

**Table 17: User Master Write Respond Channel**

Where  $n$  is channel number.

Port	Direction	Description
io_ddrMasters_n_b_valid	Output	User write respond valid.
io_ddrMasters_n_b_ready	Input	User respond ready.
io_ddrMasters_n_b_payload_id[7:0]	Output	User respond ID.
io_ddrMasters_n_b_payload_resp[1:0]	Output	User write respond.

**Table 18: User Master Read Address Channel**

Where  $n$  is the channel number.

Port	Direction	Description
io_ddrMasters_n_ar_valid	Input	User read address valid.
io_ddrMasters_n_ar_ready	Output	User read address ready.
io_ddrMasters_n_ar_payload_addr[31:0]	Input	User read address.
io_ddrMasters_n_ar_payload_id[7:0]	Input	User read address ID.
io_ddrMasters_n_ar_payload_region[3:0]	Input	User region identifier.
io_ddrMasters_n_ar_payload_len[7:0]	Input	User burst length.
io_ddrMasters_n_ar_payload_size[2:0]	Input	User burst size.
io_ddrMasters_n_ar_payload_burst[1:0]	Input	User burst type, INCR only.
io_ddrMasters_n_ar_payload_lock	Input	User lock type.
io_ddrMasters_n_ar_payload_cache[3:0]	Input	User memory type.
io_ddrMasters_n_ar_payload_qos[3:0]	Input	User quality of service.
io_ddrMasters_n_ar_payload_prot[2:0]	Input	User protection type.

**Table 19: User Master Read Data Channel**

Where  $n$  is the channel number.

Port	Direction	Description
io_ddrMasters_n_r_valid	Output	User read valid.
io_ddrMasters_n_r_ready	Input	DDR read ready.
io_ddrMasters_n_r_payload_data[127:0]	Output	DDR read data.
io_ddrMasters_n_r_payload_id[7:0]	Output	DDR read ID.
io_ddrMasters_n_r_payload_resp[1:0]	Output	DDR read respond.
io_ddrMasters_n_r_payload_last	Output	DDR read last.

## APB3 Interface

The following table shows the ports for the APB3 user slave peripherals. Refer to the AMBA APB Protocol Specification for APB port descriptions and handshake information.

**Table 20: APB3 Ports**

Where  $n$  is 0 or 1

Port	Direction	Description
io_apbSlave_n_PADDR[15:0]	Output	User address.
io_apbSlave_n_PSEL	Output	User select.
io_apbSlave_n_PENABLE	Output	User enable.
io_apbSlave_n_PREADY	Input	User ready.
io_apbSlave_n_PWRITE	Output	User direction.
io_apbSlave_n_PWDATA[31:0]	Output	User write data.
io_apbSlave_n_PRDATA[31:0]	Input	User read data.
io_apbSlave_n_PSLVERROR	Input	User transfer failure.

## JTAG Interface

The Ruby Vision SoC uses the JTAG User TAP interface block to communicate with the OpenOCD debugger.

*Table 21: JTAG Ports*

Port	Direction	Description
jtagCtrl_enable	Input	Indicates that the user instruction is active for the interface.
tagCtrl_capture	Input	TAP controller is in the capture state.
jtagCtrl_shift	Input	TAP controller is in the shift state.
jtagCtrl_update	Input	TAP controller in the update state.
jtagCtrl_reset	Input	TAP controller is in the reset state.
jtagCtrl_tdi	Input	JTAG TDI for debugging.
jtagCtrl_tdo	Output	JTAG TDO for debugging.
jtagCtrl_tck	Input	JTAG TCK for debugging.

# GPIO Peripheral Interface

Use the `SYSTEM_GPIO_0_IO_APB` parameter to reference the GPIO interface.

Table 22: GPIO Ports

Port	Direction	Description
<code>system_gpio_0_io_read[15:0]</code>	Input	GPIO input.
<code>system_gpio_0_io_write[15:0]</code>	Output	GPIO output.
<code>system_gpio_0_io_writeEnable[15:0]</code>	Output	GPIO output enable.

Table 23: GPIO Register Map

Address Offset	Register Name	Privilege	Width
<code>0x0000_0000</code>	Input	Read/Write	32
<code>0x0000_0004</code>	Output	Read/Write	32
<code>0x0000_0008</code>	Output Enable	Read/Write	32
<code>0x0000_0020</code>	Interrupt Rise Enable	Read/Write	32
<code>0x0000_0024</code>	Interrupt Fall Enable	Read/Write	32
<code>0x0000_0028</code>	Interrupt High Enable	Read/Write	32
<code>0x0000_002C</code>	Interrupt Low Enable	Read/Write	32

## Input Register: `0x0000_0000`

31	0
Input	

Bits	Field	Description	Privilege
0-31	Input	Set GPIO pin as an input (32 pins).	Read/Write

## Output Register: `0x0000_0004`

31	0
Output	

Bits	Field	Description	Privilege
0-31	Output	Set GPIO pin as an output (32 pins).	Read/Write

## Output Enable Register: `0x0000_0008`

31	0
OE	

Bits	Field	Description	Privilege
0-31	OE	Enable GPIO output pin (32 pins).	Read/Write

*Interrupt Rise Enable Register: 0x0000\_0020*

31	0
IntRiseEn	

Bits	Field	Description	Privilege
0-31	IntRiseEn	Enable a rise interrupt on the GPIO pin (32 pins).	Read/Write

*Interrupt Fall Enable Register: 0x0000\_0024*

31	0
IntFallEn	

Bits	Field	Description	Privilege
0-31	IntFallEn	Enable a fall interrupt on the GPIO pin (32 pins).	Read/Write

*Interrupt High Enable Register: 0x0000\_0028*

31	0
IntHighEn	

Bits	Field	Description	Privilege
0-31	IntHighEn	Enable a high interrupt on the GPIO pin (32 pins).	Read/Write

*Interrupt Low Enable Register: 0x0000\_002C*

31	0
IntLowEn	

Bits	Field	Description	Privilege
0-31	IntLowEn	Enable a low interrupt on the GPIO pin (32 pins).	Read/Write



## I<sup>2</sup>C Peripheral Interface

The Ruby Vision SoC has 3 I<sup>2</sup>C master/slave peripherals. You use the `system_i2c_2*` ports to calibrate the DDR DRAM memory; if you do not want to perform calibration, you can use this peripheral for your own purposes. Use these parameters to reference the interface:

- I<sup>2</sup>C 0—SYSTEM\_I2C\_0\_IO\_APB
- I<sup>2</sup>C 1—SYSTEM\_I2C\_1\_IO\_APB
- I<sup>2</sup>C 2—SYSTEM\_I2C\_2\_IO\_APB

**Table 24: I<sup>2</sup>C Peripheral Ports (User)**

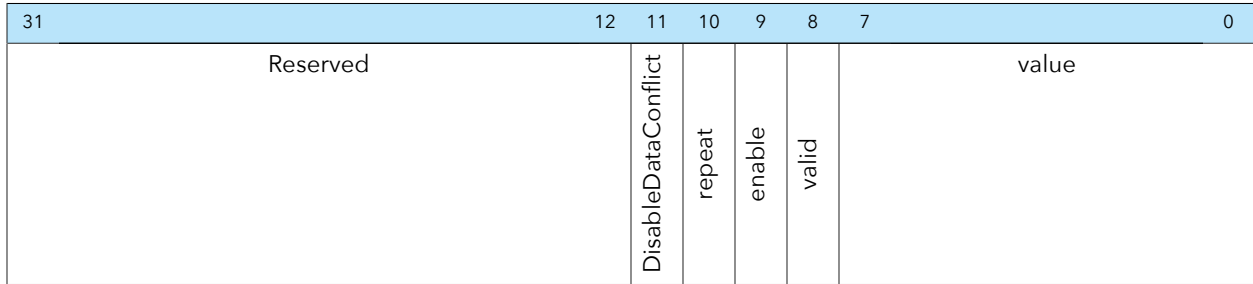
Where *n* is 0, 1, or 2.

Port	Direction	Description
<code>system_i2c_n_io_sda_write</code>	Output	SDA output for user device.
<code>system_i2c_n_io_sda_read</code>	Input	SDA input for user device.
<code>system_i2c_n_io_scl_write</code>	Output	SCL output for user device.
<code>system_i2c_n_io_scl_read</code>	Input	SCL input for user device.

**Table 25: I<sup>2</sup>C Register Map**

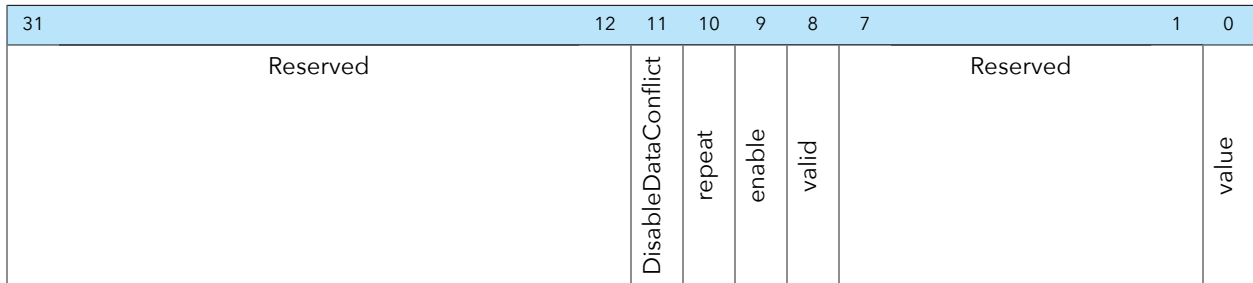
Address Offset	Register Name	Privilege	Width
0x0000_0000	txData	Read/Write	32
0x0000_0004	txAck	Read/Write	32
0x0000_0008	rxData	Read/Write	32
0x0000_000C	rxAck	Read/Write	32
0x0000_0020	Interrupt	Read/Write	32
0x0000_0024	Interrupt Clears	Read/Write	32
0x0000_0028	Sampling Clock Divider	Read/Write	32
0x0000_002C	Timeout	Write	32
0x0000_0030	tsuData	Write	32
0x0000_0040	Master Status	Read/Write	32
0x0000_0050	tlow	Read/Write	32
0x0000_0054	tHigh	Read/Write	32
0x0000_0058	tBuf	Read/Write	32
0x0000_0080	Filtering Status	Read/Write	32
0x0000_0084	Hit Context	Read/Write	32
0x0000_0088	Filtering Configuration	Read/Write	32

*txData Register: 0x0000\_0000*



Bits	Field	Description	Privilege
0-7	value	Transmit data value.	Write
8	valid	Transmit data valid bit.	Read/Write
9	enable	Transmit data enable.	Read/Write
10	repeat	Transmit data repeat bit.	Write
11	DisableDataConflict	Disable transmit data conflict.	Write
12-31	Reserved	Reserved.	N/A

*txAck Register: 0x0000\_0004*



Bits	Field	Description	Privilege
0	value	Transmit acknowledge bit.	Write
1-7	Reserved	Reserved.	N/A
8	valid	Transmit acknowledge valid bit.	Read/Write
9	enable	Transmit acknowledge enable.	Read/Write
10	repeat	Transmit acknowledge repeat bit.	Write
11	DisableDataConflict	Disable transmit acknowledge conflict.	Write
12-31	Reserved	Reserved.	N/A

*rxData Register: 0x0000\_0008*

31	10	9	8	7	0
Reserved			listen	valid	value

Bits	Field	Description	Privilege
0-7	value	Received data.	Read
8	valid	Receive data valid.	Read
9	listen	Start listen data.	Write
10-31	Reserved	Reserved.	N/A

*rxAck Register: 0x0000\_000C*

31	10	9	8	7	1	0
Reserved			listen	valid	Reserved	value

Bits	Field	Description	Privilege
0	value	Received acknowledge.	Read
1-7	Reserved	Reserved.	N/A
8	valid	Receive acknowledge valid.	Read
9	listen	Start listen acknowledge.	Write
10-31	Reserved	Reserved.	N/A

## Interrupt Register: 0x0000\_0020

31	22	21	20	19	18	17	16	15	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved			filterFlag	clockGenBusyFlag	Reserved		filterEnable	clockGenBusyEnable	Reserved			dropFlag	endFlag	restartFlag	startFlag	dropEnable	endEnable	restartEnable	startEnable	txAckEnable	txDataEnable	rxAckEnable	rxDataEnable

Bits	Field	Description	Privilege
0	rxDataEnable	Receive data interrupt enable	Read/Write
1	rxAckEnable	Receive acknowledge interrupt enable	Read/Write
2	txDataEnable	Transmit data interrupt enable	Read/Write
3	txAckEnable	Transmit acknowledge interrupt enable	Read/Write
4	startEnable	Start interrupt enable	Read/Write
5	restartEnable	Restart interrupt enable	Read/Write
6	endEnable	End interrupt enable	Read/Write
7	dropEnable	Drop interrupt enable	Read/Write
8	startFlag	Start interrupt flag	Read
9	restartFlag	Restart interrupt flag	Read
10	endFlag	End interrupt flag	Read
11	dropFlag	Drop interrupt flag	Read
12-15	Reserved	Reserved.	N/A
16	clockGenBusyEnable	Master clock generation interrupt enable.	Read/Write
17	filterEnable	Slave address filtering hit interrupt enable	Read/Write
18-19	Reserved	Reserved.	N/A
20	clockGenBusyFlag	Master clock generation interrupt flag.	Read
21	filterFlag	Slave address filtering hit interrupt flag.	Read
22-31	Reserved	Reserved.	N/A

*Interrupt Clears Register: 0x0000\_0024*

31	12	11	10	9	8	7	0		
Reserved				dropFlagClear	endFlagClear	restartFlagClear	startFlagClear	Reserved	

Bits	Field	Description	Privilege
0-7	Reserved	Reserved.	N/A
8	startFlagClear	Clear start flag.	Write
9	restartFlagClear	Clear restart flag.	Write
10	endFlagClear	Clear end flag.	Write
10	dropFlagClear	Clear drop flag.	Write
12-31	Reserved	Reserved.	N/A

*Timeout Register: 0x0000\_002C*

31	20	19	0
Reserved		value	

Bits	Field	Description	Privilege
0-19	value	Inactive timeout setting.	Write
20-31	Reserved	Reserved.	N/A

*Sampling Clock Divider Register: 0x0000\_0028*

31	10	9	0
Reserved		samplingClockDividerWidth	

Bits	Field	Description	Privilege
0-9	samplingClockDividerWidth	Clock divider width. Controls the rate at which the I <sup>2</sup> C controller reads SCL and SDA.	Read/Write
10-31	Reserved	Reserved.	N/A

*tsuData Register: 0x0000\_0030*

31	6	5	0
Reserved		value	

Bits	Field	Description	Privilege
0-5	value	Data setup time.	Write
6-31	Reserved	Reserved.	N/A

*Master Status Register: 0x0000\_0040*

31	7	6	5	4	3	1	0
Reserved				drop	stop	start	Reserved
							isBusy

Bits	Field	Description	Privilege
0	isBusy	Master busy.	Read
1-3	Reserved	Reserved.	N/A
4	start	Master start.	Read/Write
5	stop	Master stop.	Read/Write
6	drop	Master drop.	Read/Write
6-31	Reserved	Reserved.	N/A

*tLow Register: 0x0000\_0050*

31	12	11	0
Reserved		value	

Bits	Field	Description	Privilege
0-11	value	SCL low period.	Write
12-31	Reserved	Reserved.	N/A

*tHigh Register: 0x0000\_0054*

31	12	11	0
Reserved		value	

Bits	Field	Description	Privilege
0-11	value	SCL high period.	Write
12-31	Reserved	Reserved.	N/A

*tBuf Register: 0x0000\_0058*

31	12	11	0
Reserved		value	

Bits	Field	Description	Privilege
0-11	value	Start and stop bus free time.	Write
12-31	Reserved	Reserved.	N/A

*Filtering Status Register: 0x0000\_0080*

31	8	7	6	5	4	3	2	1	0						
Reserved								hit_7	hit_6	hit_5	hit_4	hit_3	hit_2	hit_1	hit_0

Bits	Field	Description	Privilege
0	hit_0	Filtering hit bit 0.	Read
1	hit_1	Filtering hit bit 1.	Read
2	hit_2	Filtering hit bit 2.	Read
3	hit_3	Filtering hit bit 3.	Read
4	hit_4	Filtering hit bit 4.	Read
5	hit_5	Filtering hit bit 5.	Read
6	hit_6	Filtering hit bit 6.	Read
7	hit_7	Filtering hit bit 7.	Read
8-32	Reserved	Reserved.	N/A

*Hit Context Register: 0x0000\_0084*

31	1	0
Reserved		rw

Bits	Field	Description	Privilege
0	rw	Hit context read.	Read
1-31	Reserved	Reserved.	N/A

## PLIC Peripheral Interface

Use the `SYSTEM_PLIC_APB` parameter to reference the interface PLIC interface.

*Table 26: RISC-V PLIC Operation Parameters*

Defines	Description
Interrupt priorities registers	The interrupt priority for each interrupt source.
Interrupt pending bits registers	The interrupt pending status of each interrupt source.
Interrupt enables registers	Enables the interrupt source of each context.
Priority thresholds registers	The interrupt priority threshold of each context.
Interrupt claim registers	The register to acquire interrupt source ID of each context.
Interrupt completion registers	The register to send interrupt completion message to the associated gateway.

## SPI Master Peripheral Interface

The SPI master peripheral interface supports traditional 4-wire SPI as well as quad-SPI mode, which sends 4 data bits per clock cycle. When implementing the SPI peripheral in traditional dual-line mode, use the `data_0` ports as MOSI and and the `data_1` ports as MISO.

Use these parameters to reference the interface:

- SPI master 0—`SYSTEM_SPI_0_IO_APB`
- SPI master 1—`SYSTEM_SPI_1_IO_APB`
- SPI master 2—`SYSTEM_SPI_2_IO_APB`

**Table 27: SPI Master Ports**

Where *n* is 0, 1, or 2

Port	Direction	Description
<code>system_spi_n_io_sclk_write</code>	Output	SPI SCK.
<code>system_spi_n_io_data_0_writeEnable</code>	Output	SPI output enable for data 0.
<code>system_spi_n_io_data_0_read</code>	Input	SPI input for data 0.
<code>system_spi_n_io_data_0_write</code>	Output	SPI output for data 0.
<code>system_spi_n_io_data_1_writeEnable</code>	Output	SPI output enable for data 1.
<code>system_spi_n_io_data_1_read</code>	Input	SPI input for data 1.
<code>system_spi_n_io_data_1_write</code>	Output	SPI output for data 1.
<code>system_spi_n_io_data_2_writeEnable</code>	Output	SPI output enable for data 2.
<code>system_spi_n_io_data_2_read</code>	Input	SPI input for data 2.
<code>system_spi_n_io_data_2_write</code>	Output	SPI output for data 2.
<code>system_spi_n_io_data_3_read</code>	Input	SPI input for data 3.
<code>system_spi_n_io_data_3_write</code>	Output	SPI output for data 3.
<code>system_spi_n_io_data_3_writeEnable</code>	Output	SPI output enable for data 3.
<code>system_spi_n_io_ss</code>	Output	SPI SS.

**Table 28: SPI Master Register Map**

Address Offset	Register Name	Privilege	Width
<code>0x0000_0000</code>	Cmd	Read/Write	32
<code>0x0000_0004</code>	RSP	Read	32
<code>0x0000_0008</code>	Config	Write	32
<code>0x0000_000C</code>	Interrupt	Read/Write	32
<code>0x0000_0020</code>	ClockDivider	Write	32
<code>0x0000_0024</code>	ssSetup	Write	32
<code>0x0000_0028</code>	ssHold	Write	32
<code>0x0000_002C</code>	ssDisable	Write	32
<code>0x0000_0030</code>	ssActiveHigh	Write	32



*Cmd Register: 0x0000\_0000*

31	12	11	10	9	8	7	0
Reserved			SS	RD	WR	data	

Bits	Field	Description	Privilege
0-7	data	FIFO data value transmit/receive.	Read/Write
8	WR	Write trigger.	Write
9	RD	Read trigger.	Write
10	Reserved	Reserved.	N/A
11	SS	SPI chip select.	Read/Write
12-31	Reserved	Reserved.	N/A

*RSP Register: 0x0000\_0004*

31	16	15	0
fifoOccupancy		fifoAvailability	

Bits	Field	Description	Privilege
0-15	fifoAvailability	FIFO Availability.	Read
16-32	fifoOccupancy	FIFO Occupancy.	Read

*Config Register: 0x0000\_0008*

31	2	1	0
Reserved		cpha	cpol

Bits	Field	Description	Privilege
0	cpol	Clock polarity setting.	Write
1	cpha	Clock phase setting.	Write
2-31	Reserved	Reserved.	N/A

*Interrupt Register: 0x0000\_000C*

31	10	9	8	7	2	1	0
Reserved				rsplnt	cmdlnt	Reserved	
						rsplntEnable	cmdlntEnable

Bits	Field	Description	Privilege
0	cmdlntEnable	Command FIFO empty interrupt enable.	Read/Write
1	rsplntEnable	Read FIFO not empty interrupt enable.	Read/Write
2-7	Reserved	Reserved.	N/A
8	cmdlnt	Command FIFO empty interrupt pending.	Read/Write
9	rsplnt	Read FIFO not empty interrupt pending.	Read/Write
10-31	Reserved	Reserved.	N/A

*clockDivider Register: 0x0000\_0020*

31	clockDivider	0
----	--------------	---

Bits	Field	Description	Privilege
0-31	clockDivider	SPI frequency = FCLK / (2 * clockDivider)	Write

*ssSetup Register: 0x0000\_0024*

31	ssSetup	0
----	---------	---

Bits	Field	Description	Privilege
0-31	ssSetup	Time between the chip select enable and the next byte.	Write

*ssHold Register: 0x0000\_0028*

31	ssHold	0
----	--------	---

Bits	Field	Description	Privilege
0-31	ssHold	Time between the last byte transmission and the chip select disable.	Write

*ssDisable Register: 0x0000\_002C*

31	0
ssDisable	

Bits	Field	Description	Privilege
0-31	ssDisable	Time between the chip select disable and the chip select enable.	Write

*ssActiveHigh Register: 0x0000\_0030*

31	0
ssActiveHigh	

Bits	Field	Description	Privilege
0-31	ssActiveHigh	These bits correspond to the hardware SPI chip select. 0: Chip select is active low. 1: Chip select is active high.	Write

## UART Peripheral Interface

The UART peripheral runs at 115200 baud and supports 8 data bits, no parity, and 1 stop bit. Use these parameters to reference the interface:

- UART 0—SYSTEM\_UART\_0\_IO\_APB
- UART 1—SYSTEM\_UART\_1\_IO\_APB

*Table 29: UART Ports*

Port	Direction	Description
system_uart_0_io_txd	Output	UART 0 transmit.
system_uart_0_io_rxd	Input	UART 0 receive.
system_uart_1_io_txd	Output	UART 1 transmit.
system_uart_1_io_rxd	Input	UART 1 receive.

*Table 30: SPI Master Register Map*

Address Offset	Register Name	Privilege	Width
0x0000_0000	Data	Read/Write	32
0x0000_0004	Status	Read/Write	32
0x0000_0008	Clock divider	Read/Write	32
0x0000_000C	Config register	Read/Write	32

## Control and Status Registers

The following tables show the machine-level CSR implementation.

**Table 31: Machine Information Register**

Address	Register Name	Privilege	Description	Width
0xF14	mhartid	Read	Hardware thread ID.	32

**Table 32: Machine Trap Registers**

Address	Register Name	Privilege	Description	Width
0x300	mstatus	Read/Write	Machine status register.	13
0x304	mie	Read/Write	Machine interrupt enable register.	12
0x305	mtvec	Read/Write	Machine trap handler base address.	32

**Table 33: Machine Trap Handling Registers**

Address	Register Name	Privilege	Description	Width
0x341	mpec	Read/Write	Machine exception program counter.	32
0x342	mcause	Read	Machine trap cause.	32
0x343	mtval	Read	Machine bad address or instruction.	32
0x344	mip	Read/Write	Machine interrupt pending.	12

## Machine-Level ISA

### Hart ID Register (mhartid): 0xF14

The `mhartid` CSR is a 32-bit read-only register containing the integer ID of the hardware thread running the code. This register must be readable in any implementation. Hart IDs might not necessarily be numbered contiguously in a multiprocessor system, but at least one hart must have a hart ID of zero. Hart IDs must be unique.

31	0
Hart ID	

Bits	Field	Description	Privilege
0-31	Hart ID	Hardware thread ID.	Read

### Machine Status Register (*mstatus*): 0x300

The *mstatus* register is a 13-bits read/write register formatted. The *mstatus* register keeps track of and controls the hart's current operating state. Restricted views of the *mstatus* register appear as the *sstatus* and *ustatus* registers in the S-level and U-level ISAs, respectively.

12	11	10	9	8	7	6	5	4	3	2	1	0
MPP	Reserved			MPIE	Reserved			MIE	Reserved			

Bits	Field	Description	Privilege
0-2	Reserved	Reserved.	N/A
3	MIE	Machine interrupt enable register.	Read/Write
4-6	Reserved	Reserved.	N/A
7	MPIE	Machine previous interrupt enable.	Read/Write
8-10	Reserved	Reserved.	N/A
11-12	MPP	Machine Previous privilege mode.	Read/Write

### Machine Trap-Vector Base-Address Register (*mtvec*): 0x305

The *mtvec* register is a 32-bit read/write register that holds trap vector configuration, consisting of a vector base address (*base*) and a vector mode (*mode*).

31											2	1	0
base											mode		

Bits	Field	Description	Privilege
0-1	mode	Vector mode. 0: Direct. All exceptions set pc to BASE 1: Vectored. Asynchronous interrupts set pc to BASE + 4xcause ≥ 2: Reserved	Read/Write
2-31	base	Vector base address.	Read/Write

### Machine Interrupt Enable Register (*mie*): 0x304

The *mie* register is a 12-bit read/write register containing interrupt enable bits.

11	10	9	8	7	6	5	4	3	2	1	0
MEIE	Reserved			MTIE	Reserved			MSIE	Reserved		

Bits	Field	Description	Privilege
0-2	Reserved	Reserved.	N/A
3	MSIE	Machine software interrupt enable.	Read/Write
4-6	Reserved	Reserved.	N/A
7	MTIE	Machine timer interrupt enable.	Read
8-10	Reserved	Reserved.	N/A
11	MEIE	Machine external interrupt enable.	Read

### Machine Exception Program Counter (mepc): 0x341

mepc is a 32-bit read/write register. The low bit of mepc (mepc[0]) is always zero. On implementations that support only IALIGN=32, the two low bits (mepc[1:0]) are always zero.

31	0
mepc	

Bits	Field	Description	Privilege
0-31	mepc	Machine exception program counter.	Read/Write

### Machine Cause Register (mcause): 0x342

The mcause register is a 32-bit read-write register. When a trap is taken into M-mode, mcause is written with a code indicating the event that caused the trap. Otherwise, mcause is never written by the implementation, though it may be explicitly written by software.

31	30	0
Interrupt	Exception Code	

Bits	Field	Description	Privilege
0-30	Exception code	See <a href="#">Table 34: Machine Cause Register (mcause) Values after Trap</a> on page 30.	Read
31	Interrupt	mcause interrupt bit.	Read

**Table 34: Machine Cause Register (mcause) Values after Trap**

Interrupt	Exception Code	Description
1	0	Reserved.
1	1	Supervisor software interrupt.
1	2	Reserved.
1	3	Machine software interrupt.
1	4	User timer interrupt.
1	5	Supervisor timer interrupt.
1	6	Reserved.
1	7	Machine timer interrupt.
1	8	User external interrupt.
1	9	Supervisor external interrupt.
1	10	Reserved.
1	11	Machine external interrupt.
1	≥12	Reserved.
0	0	Instruction address misaligned.
0	1	Instruction access fault.
0	2	Illegal instruction.
0	3	Breakpoint.

Interrupt	Exception Code	Description
0	4	Load address misaligned.
0	5	Load access fault.
0	6	Store/AMO address misaligned.
0	7	Store/AMO access fault.
0	8	Reserved.
0	9	Reserved.
0	10	Reserved.
0	11	Environment call from M-mode.
0	12	Instruction page fault.
0	13	Load page fault.
0	14	Reserved.
0	15	Store/AMO page fault.
0	≥16	Reserved.

### Machine Trap Value Register (mtval): 0x343

The `mtval` register is a 32-bit register. When a trap is taken into M-mode, `mtval` is either set to zero or written with exception-specific information to assist software in handling the trap. Otherwise, `mtval` is never written by the implementation, though it may be explicitly written by software. The hardware platform will specify which exceptions must set `mtval` informatively and which may unconditionally set it to zero.

31	0
mtval	

Bits	Field	Description	Privilege
0-31	mtval	Machine trap value register bit.	Read/Write

### Machine Interrupt Pending Register (mip): 0x344

The `mip` register is a 12-bit read/write register containing information on pending interrupts.

11	10	9	8	7	6	5	4	3	2	1	0
MEIP	Reserved			MTIP	Reserved			MSIP	Reserved		

Bits	Field	Description	Privilege
0-2	Reserved	Reserved.	N/A
3	MSIP	Machine software interrupt pending.	Read/Write
4-6	Reserved	Reserved.	N/A
7	MTIP	Machine timer interrupt pending.	Read
8-10	Reserved	Reserved.	N/A
11	MEIP	Machine external interrupt pending.	Read

# Revision History

*Table 35: Revision History*

<b>Date</b>	<b>Version</b>	<b>Description</b>
March 2021	1.1	Corrected widths for axiA_wdata, axiA_wstrb, and axiA_rdata.
December 2020	1.0	Initial release.