易 灵 思

Ruby RISC-V SoC Data Sheet

DS-RUBY-v1.6
December 2021
www.elitestek.com

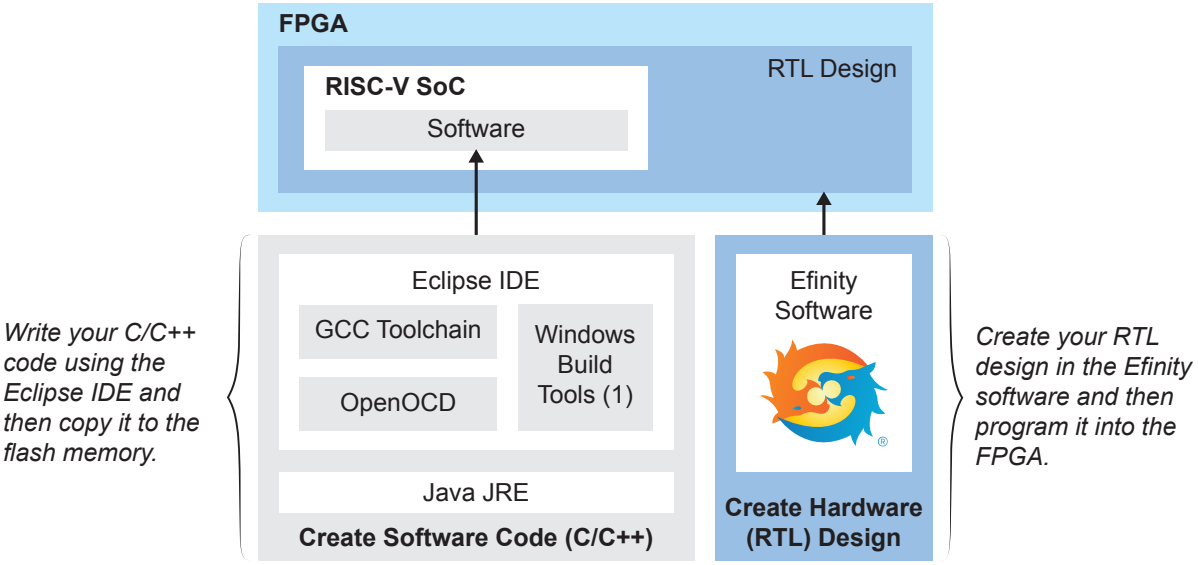
RISC-V®

Contents

Introduction

易灵思 provides the heavy-weight, cached soft RISC-V SoC, Ruby, which includes a DDR DRAM controller interface. This SoC is ideal for applications that use triple-speed Ethernet and communications protocols, provide real-time system controls, and perform image signal processing. Some example applications for the Ruby SoC are 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 钛金系列 and Trion FPGAs.

Figure 1: Ruby RISC-V SoC Design Flow



1. Windows build tools required on Windows platforms only.

Learn more: For details on developing RTL designs or creating software, refer to the Ruby RISC-V Hardware and Software User Guide.

VexRiscv RISC-V Core

The Ruby SoC is based on the VexRiscv core created by Charles Papon. The VexRiscv core is a 32-bit CPU using the ISA RISCV32I with M and C extensions and has five pipeline stages (fetch, decode, execute, memory, and writeback).

In the Ruby SoC, the VexRiscv core supports the AXI4 and APB3 bus interfaces, and has instruction and data caches.

The VexRiscv core won first place in the RISC-V SoftCPU contest in 2018.[1]

[1] 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
  - 4 KB data cache
  - 4 KB instruction cache
- 20 - 350 MHz system clock frequency
- 4 - 512 KB 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
- 1 AXI master channel to access the DDR memory
- 1 AXI slave channel to user logic
- APB3 peripherals:
  - 16 GPIOs
  - 3 I^2C masters and slaves
  - Machine timer
  - PLIC
  - 3 SPI flash masters with a maximum clock frequency of 25 MHz
  - 2 UARTs with 115,200 baud rate
  - 2 slave user peripherals

FPGA Support

The Ruby SoC supports T35, T55, T85, and T120 Trion® FPGAs.

钛金系列 Resource Utilization and Performance

FPGA	Logic/Adders	FlipFlops	Memory Blocks	DSP48 Blocks	f_{MAX} (MHz)	Efinity Version
Ti60 F225 C4	8,098	6,732	60	4	SoC: 292 Memory: 220	2021.2

Trion Resource Utilization and Performance

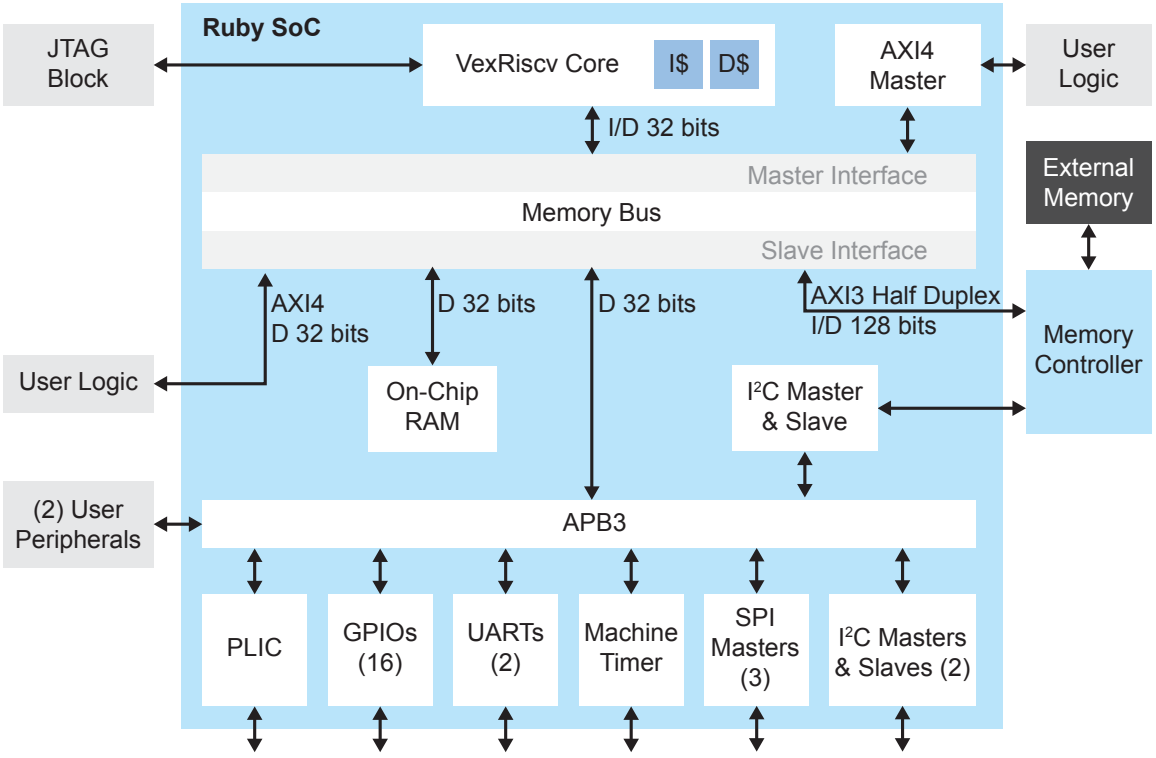
FPGA	Logic Utilization (LUTs)	Memory Blocks	f_{MAX} (MHz)	Efinity Version
T120 BGA324 C4	11,411	71	SoC: 113 Memory: 117	2021.2

Functional Description

The Ruby SoC incorporates a 32-bit RISC-V processor, 4 KB instruction cache, 4 KB data cache, 4 - 512 KB of on-chip RAM, and a variety of peripherals (including 2 APB3 slave peripherals and 1 AXI slave). You can configure the operating frequency from 20 - 350 MHz (the actual performance is limited by the design's f_{MAX}). The SoC includes 3 I^2C peripherals and 3 SPI masters. The default configuration has a 128-bit half-duplex AXI bus to communicate with the 易灵思 DDR controller core. This core uses the Trion FPGAs hard DDR DRAM interface to reset an external DRAM module (resets and re-initializes the Trion FPGA's DDR interface as well as the DDR module(s)).

You can customize the SoC using the IP Manager in the software v2020.2 and higher.

Figure 2: Ruby SoC Block Diagram



Address Map

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 and address mappings 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 GB	–	Cache
GPIO	SYSTEM_GPIO_0_IO_APB	4K	[0]: 12 [1]: 13	I/O
I^2C 0	SYSTEM_I2C_0_IO_APB	4K	8	I/O
I^2C 1	SYSTEM_I2C_1_IO_APB	4K	9	I/O
I^2C 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
UART 2	SYSTEM_UART_2_IO_APB	4K	3	I/O
User peripheral 0	IO_APB_SLAVE_0_APB	64K	–	I/O
User peripheral 1	IO_APB_SLAVE_1_APB	64K	–	I/O
On-chip BRAM	SYSTEM_RAM_A_BMB	4 - 512 KB	–	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 Ruby 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 0x0038_0000.

Clocks

Table 2: Clock Ports

Port	Direction	Description
io_systemClock	Input	Provides a 20 - 350 MHz clock for the SoC.
io_memoryClock	Input	Provides a 100 MHz clock for the external memory bus.

Interrupts

Table 3: Interrupt Ports

Port	Direction	Description
userInterruptA	Input	Provides an external interrupt.
io_axi4Interrupt	Input	User AXI slave channel interrupt.

Resets

The Ruby SoC has a 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^2C 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 io_ddrMasters_1_reset	Output	Responds to the reset for the DDR master.

AXI Interface

The Ruby SoC has a 128-bit half duplex AXI3 interface to communicate with the external memory.

Additionally it has an AXI4 interface to connect to user logic:

- There is one AXI4 slave interface, which is compatible with AXI-Lite (axlen is always 0.)
- There are two AXI4 master interfaces.

Learn more: Refer to the AMBA AXI and ACE Protocol Specification for AXI channel descriptions and handshake information.

AXI Interface to External Memory

Table 5: External Memory Slave Half-Duplex Address Channel for Read and Write

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

Table 6: External Memory Slave Write Data Channel

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

Table 7: External Memory Slave Write Respond Channel

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

Table 8: External Memory Slave Read Data Channel

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

Where *n* is the channel number.

Port	Direction	Description
io_ddrMasters_*n*_clk	Input	AXI master clock.
io_ddrMasters_*n*_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.
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 the 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.

Port	Direction	Description
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	External memory read ready.
io_ddrMasters_*n*_r_payload_data[127:0]	Output	External memory read data.
io_ddrMasters_*n*_r_payload_id[7:0]	Output	External memory read ID.
io_ddrMasters_*n*_r_payload_resp[1:0]	Output	External memory read respond.
io_ddrMasters_*n*_r_payload_last	Output	External memory 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 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.
jtagCtrl_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.

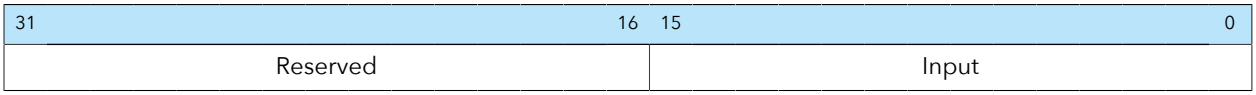
Table 22: GPIO Ports

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

Table 23: GPIO Register Map

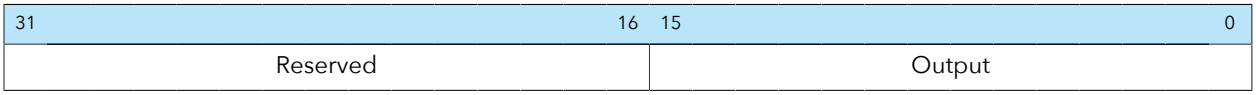
Address Offset	Register Name	Privilege	Width
0x0000_0000	Input	Read/Write	32
0x0000_0004	Output	Read/Write	32
0x0000_0008	Output Enable	Read/Write	32
0x0000_0020	Interrupt Rise Enable	Read/Write	32
0x0000_0024	Interrupt Fall Enable	Read/Write	32
0x0000_0028	Interrupt High Enable	Read/Write	32
0x0000_002C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

Input Register: 0x0000_0000



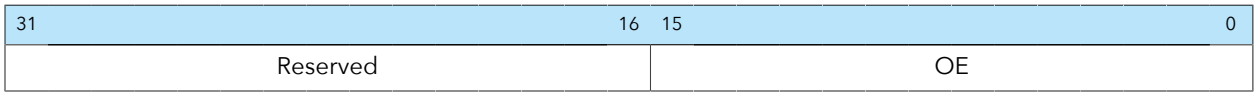
Bits	Field	Description	Privilege
0-15	Input	Set GPIO pin as an input (16 pins).	Read/Write
16-31	Reserved	Reserved.	N/A

Output Register: 0x0000_0004



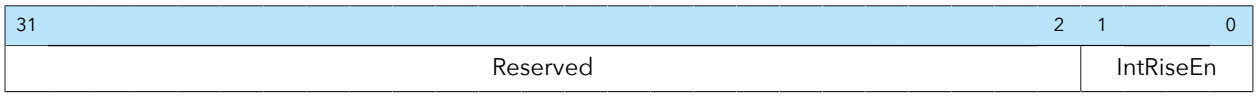
Bits	Field	Description	Privilege
0-15	Output	Set GPIO pin as an output (16 pins).	Read/Write
16-31	Reserved	Reserved.	N/A

Output Enable Register: 0x0000_0008



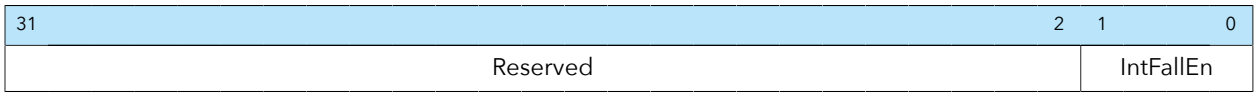
Bits	Field	Description	Privilege
0-15	OE	Enable GPIO output pin (16 pins).	Read/Write
16-31	Reserved	Reserved.	N/A

Interrupt Rise Enable Register: 0x0000_0020



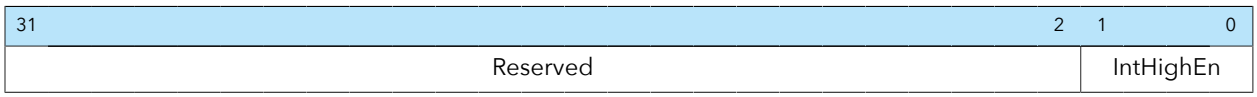
Bits	Field	Description	Privilege
0-1	IntRiseEn	Enable a rise interrupt on GPIO pins 0 and 1.	Read/Write
2-31	Reserved	Reserved.	N/A

Interrupt Fall Enable Register: 0x0000_0024



Bits	Field	Description	Privilege
0-1	IntFallEn	Enable a fall interrupt on GPIO pins 0 and 1.	Read/Write
2-31	Reserved	Reserved.	N/A

Interrupt High Enable Register: 0x0000_0028



Bits	Field	Description	Privilege
0-1	IntHighEn	Enable a high interrupt on GPIO pins 0 and 1.	Read/Write
2-31	Reserved	Reserved.	N/A

Interrupt Low Enable Register: 0x0000_002C

Bits	Field	Description	Privilege
0-1	IntLowEn	Enable a low interrupt on GPIO pins 0 and 1.	Read/Write
2-31	Reserved	Reserved.	N/A

I²C Peripheral Interface

The Ruby SoC has 3 I²C 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:

- I²C 0—`SYSTEM_I2C_0_IO_APB`
- I²C 1—`SYSTEM_I2C_1_IO_APB`
- I²C 2—`SYSTEM_I2C_2_IO_APB`

Table 24: I²C Peripheral Ports (User)

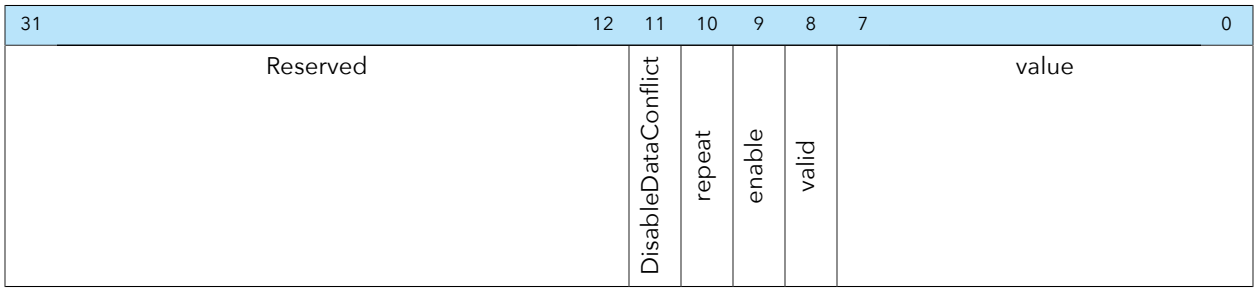
Where *n* is 0, 1, or 2.

Port	Direction	Description
system_i2c_*n*_io_sda_write	Output	SDA output for user device.
system_i2c_*n*_io_sda_read	Input	SDA input for user device.
system_i2c_*n*_io_scl_write	Output	SCL output for user device.
system_i2c_*n*_io_scl_read	Input	SCL input for user device.

Table 25: I²C Register Map

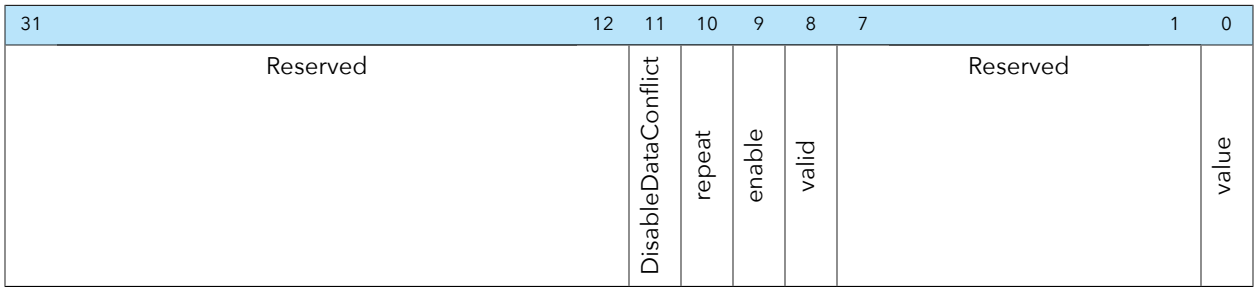
Address Offset	Register Name	Privilege	Width
0x0000_0000	txData	Read/Write	32
0x0000_0004	txAck	Read/Write	32
0x0000_0008	rxData	Read/Write	32
0x0000_000C	rxAck	Read/Write	32
0x0000_0020	Interrupt	Read/Write	32
0x0000_0024	Interrupt Clears	Read/Write	32
0x0000_0028	Sampling Clock Divider	Read/Write	32
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



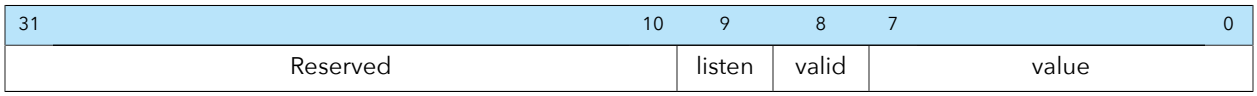
Bits	Field	Description	Privilege
0-7	value	Transmit data value.	Write
8	valid	Transmit data valid bit.	Read/Write
9	enable	Transmit data enable.	Read/Write
10	repeat	Transmit data repeat bit.	Write
11	DisableDataConflict	Disable transmit data conflict.	Write
12-31	Reserved	Reserved.	N/A

txAck Register: 0x0000_0004



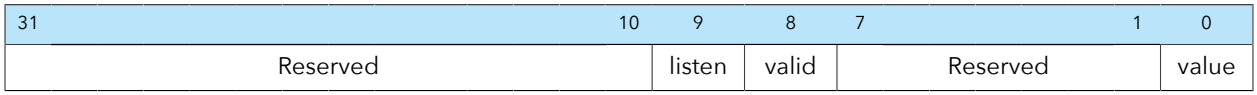
Bits	Field	Description	Privilege
0	value	Transmit acknowledge bit.	Write
1-7	Reserved	Reserved.	N/A
8	valid	Transmit acknowledge valid bit.	Read/Write
9	enable	Transmit acknowledge enable.	Read/Write
10	repeat	Transmit acknowledge repeat bit.	Write
11	DisableDataConflict	Disable transmit acknowledge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

rxData Register: 0x0000_0008



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



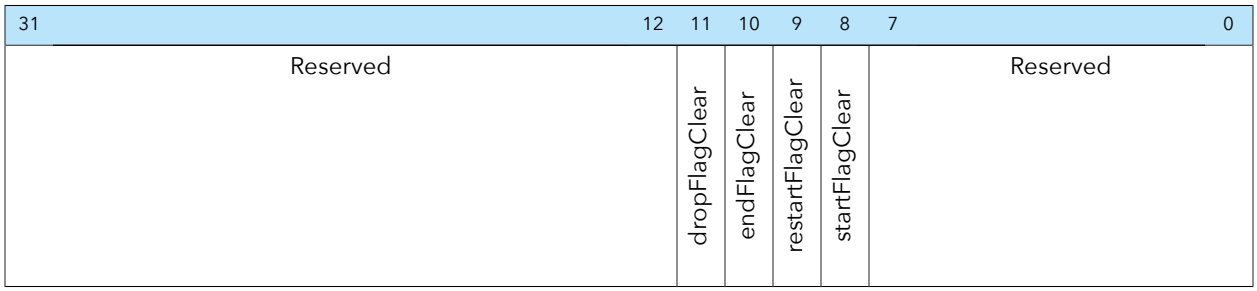
Bits	Field	Description	Privilege
0	value	Received acknowledge.	Read
1-7	Reserved	Reserved.	N/A
8	valid	Receive acknowledge valid.	Read
9	listen	Start listen acknowledge.	Write
10-31	Reserved	Reserved.	N/A

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

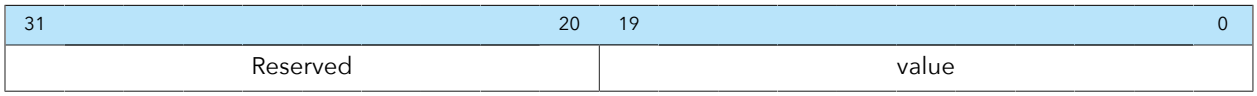
Bits	Field	Description	Privilege
0	rxDataEnable	Receive data interrupt enable	Read/Write
1	rxAckEnable	Receive acknowledge interrupt enable	Read/Write
2	txDataEnable	Transmit data interrupt enable	Read/Write
3	txAckEnable	Transmit acknowledge interrupt enable	Read/Write
4	startEnable	Start interrupt 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



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

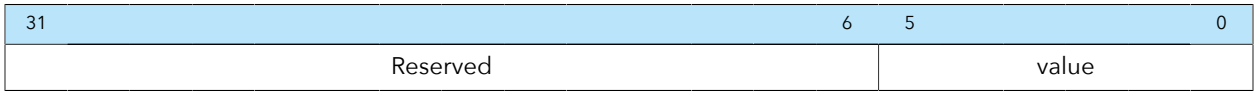


Bits	Field	Description	Privilege
0-19	value	Inactive timeout setting.	Write
20-31	Reserved	Reserved.	N/A

Sampling Clock Divider Register: 0x0000_0028

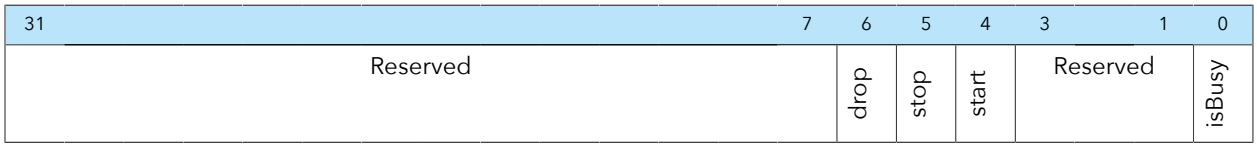
Bits	Field	Description	Privilege
0-9	samplingClockDividerWidth	Clock divider width. Controls the rate at which the I^2C controller reads SCL and SDA.	Read/Write
10-31	Reserved	Reserved.	N/A

tsuData Register: 0x0000_0030



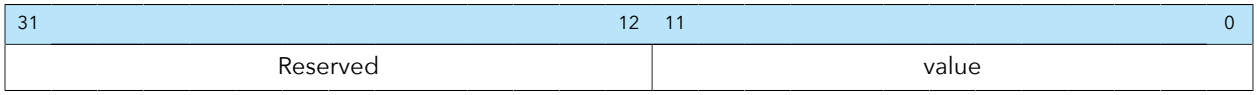
Bits	Field	Description	Privilege
0-5	value	Data setup time.	Write
6-31	Reserved	Reserved.	N/A

Master Status Register: 0x0000_0040



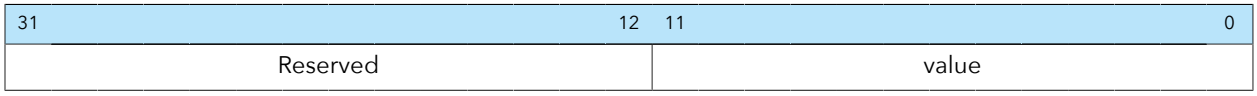
Bits	Field	Description	Privilege
0	isBusy	Master busy.	Read
1-3	Reserved	Reserved.	N/A
4	start	Master start.	Read/Write
5	stop	Master stop.	Read/Write
6	drop	Master drop.	Read/Write
6-31	Reserved	Reserved.	N/A

tLow Register: 0x0000_0050



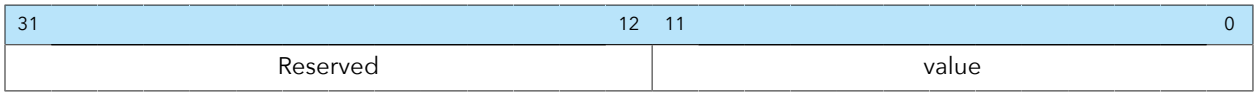
Bits	Field	Description	Privilege
0-11	value	SCL low period.	Write
12-31	Reserved	Reserved.	N/A

tHigh Register: 0x0000_0054



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

tBuf Register: 0x0000_0058



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.

Use the `SYSTEM_PLIC_CTRL` 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.

The **soc.h** file contains a number of PLIC parameters to specify the interrupt ID for the various peripherals.

Table 27: PLIC Interrupt ID Parameters

Where *n* is the peripheral number and *m* is the interrupt ID.

Parameter	Refer to
SYSTEM_PLIC_SYSTEM_I2C_*n*_IO_INTERRUPT *m*	Interrupt Register: 0x0000_0020 on page 20 Interrupt Clears Register: 0x0000_0024 on page 21
SYSTEM_PLIC_SYSTEM_GPIO_*n*_IO_INTERRUPTS_0 *m*	Interrupt Low Enable Register: 0x0000_002C on page 16 Interrupt High Enable Register: 0x0000_0028 on page 16 Interrupt Fall Enable Register: 0x0000_0024 on page 16 Interrupt Rise Enable Register: 0x0000_0020 on page 16
SYSTEM_PLIC_SYSTEM_AXI_A_INTERRUPT	Interrupts on page 8
SYSTEM_PLIC_SYSTEM_SPI_*n*_IO_INTERRUPT *m*	Interrupt Register: 0x0000_000C on page 27
SYSTEM_PLIC_SYSTEM_UART_*n*_IO_INTERRUPT *m*	Status Register: 0x0000_0004 on page 29
SYSTEM_PLIC_USER_INTERRUPT_A_INTERRUPT SYSTEM_PLIC_USER_INTERRUPT_B_INTERRUPT	Interrupts on page 8

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 28: SPI Master Ports

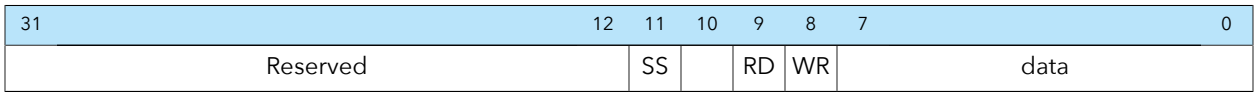
Where *n* is 0, 1, or 2

Port	Direction	Description
system_spi_*n*_io_sclk_write	Output	SPI SCK.
system_spi_*n*_io_data_0_writeEnable	Output	SPI output enable for data 0.
system_spi_*n*_io_data_0_read	Input	SPI input for data 0.
system_spi_*n*_io_data_0_write	Output	SPI output for data 0.
system_spi_*n*_io_data_1_writeEnable	Output	SPI output enable for data 1.
system_spi_*n*_io_data_1_read	Input	SPI input for data 1.
system_spi_*n*_io_data_1_write	Output	SPI output for data 1.
system_spi_*n*_io_data_2_writeEnable	Output	SPI output enable for data 2.
system_spi_*n*_io_data_2_read	Input	SPI input for data 2.
system_spi_*n*_io_data_2_write	Output	SPI output for data 2.
system_spi_*n*_io_data_3_read	Input	SPI input for data 3.
system_spi_*n*_io_data_3_write	Output	SPI output for data 3.
system_spi_*n*_io_data_3_writeEnable	Output	SPI output enable for data 3.
system_spi_*n*_io_ss	Output	SPI SS.

Table 29: SPI Master Register Map

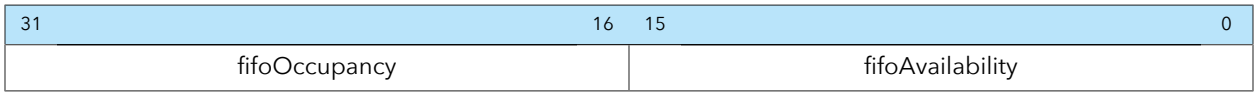
Address Offset	Register Name	Privilege	Width
0x0000_0000	Cmd	Read/Write	32
0x0000_0004	RSP	Read	32
0x0000_0008	Config	Write	32
0x0000_000C	Interrupt	Read/Write	32
0x0000_0020	ClockDivider	Write	32
0x0000_0024	ssSetup	Write	32
0x0000_0028	ssHold	Write	32
0x0000_002C	ssDisable	Write	32
0x0000_0030	ssActiveHigh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

Cmd Register: 0x0000_0000



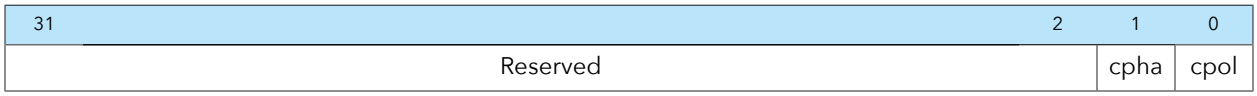
Bits	Field	Description	Privilege
0-7	data	FIFO data value transmit/receive.	Read/Write
8	WR	Write trigger.	Write
9	RD	Read trigger.	Write
10	Reserved	Reserved.	N/A
11	SS	SPI chip select.	Read/Write
12-31	Reserved	Reserved.	N/A

RSP Register: 0x0000_0004



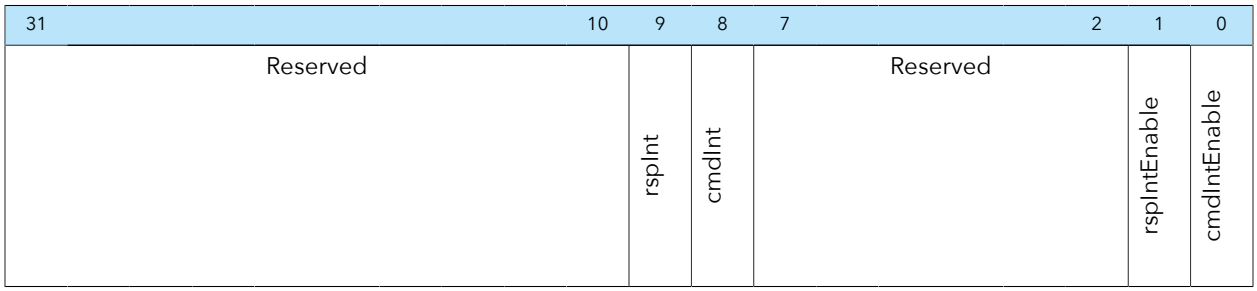
Bits	Field	Description	Privilege
0-15	fifoAvailability	FIFO Availability.	Read
16-32	fifoOccupancy	FIFO Occupancy.	Read

Config Register: 0x0000_0008



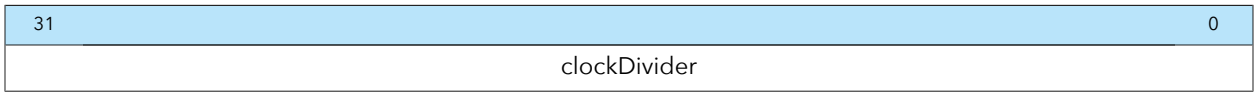
Bits	Field	Description	Privilege
0	cpol	Clock polarity setting.	Write
1	cpha	Clock phase setting.	Write
2-31	Reserved	Reserved.	N/A

Interrupt Register: 0x0000_000C



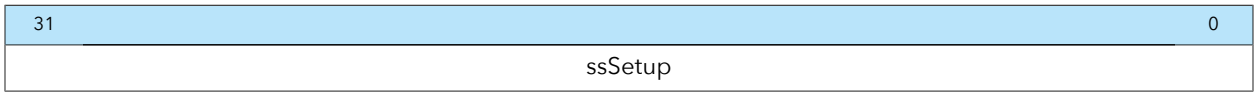
Bits	Field	Description	Privilege
0	cmdIntEnable	Command FIFO empty interrupt enable.	Read/Write
1	rspIntEnable	Read FIFO not empty interrupt enable.	Read/Write
2-7	Reserved	Reserved.	N/A
8	cmdInt	Command FIFO empty interrupt pending.	Read/Write
9	rspInt	Read FIFO not empty interrupt pending.	Read/Write
10-31	Reserved	Reserved.	N/A

clockDivider Register: 0x0000_0020



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

ssSetup Register: 0x0000_0024



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

ssHold Register: 0x0000_0028

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`

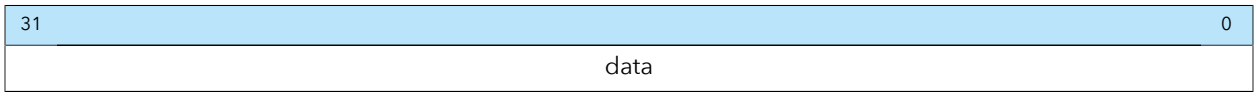
Table 30: UART Ports

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 31: SPI Master Register Map

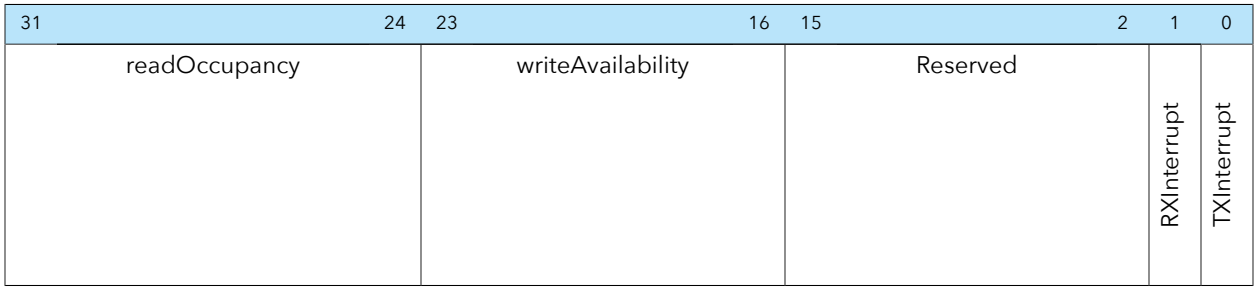
Address Offset	Register Name	Privilege	Width
0x0000_0000	Data	Read/Write	32
0x0000_0004	Status	Read/Write	32
0x0000_0008	Clock divider	Read/Write	32
0x0000_000C	Config register	Read/Write	32

Data Register: 0x0000_0000



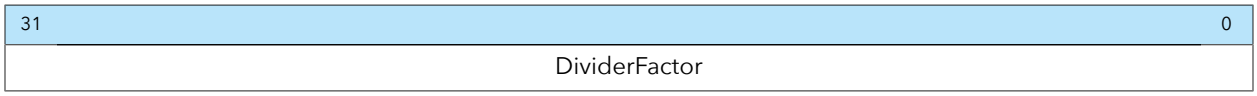
Bits	Field	Description	Privilege
0-31	data	Stores data that is transmitted or received.	Read/Write

Status Register: 0x0000_0004



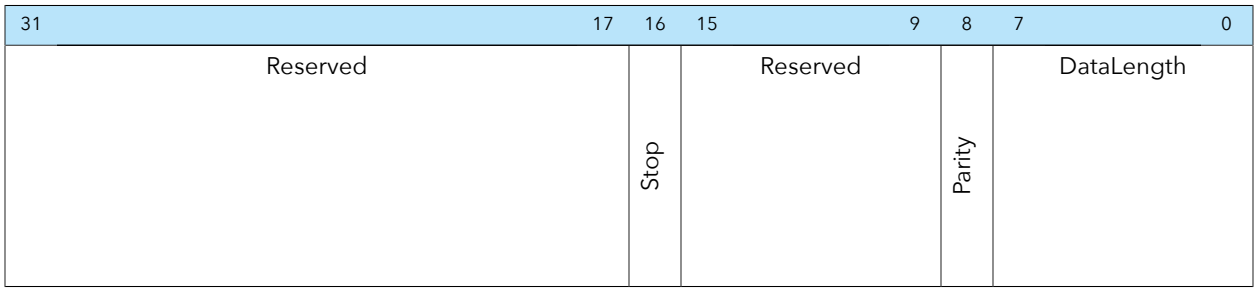
Bits	Field	Description	Privilege
0	TXInterrupt	Interrupt when TX FIFO is empty.	Read/Write
1	RXInterrupt	Interrupt when RX FIFO is not empty.	Read/Write
2-15	Reserved	Reserved.	N/A
16-23	writeAvailability	UART FIFO availablity.	Read/Write
24-31	readOccupancy	UART FIFO occupancy.	Read/Write

Clock Divider Register: 0x0000_0008



Bits	Field	Description	Privilege
0-31	DividerFactor	Divider factor for the UART baud rate. Baudrate = SystemClk/(Data Length * DividerFactor)	Read/Write

Config Register: 0x0000_000C



Bits	Field	Description	Privilege
0-7	DataLength	Data length.	Read/Write
8	Parity	Parity bit number.	Read/Write
9-15	Reserved	Reserved.	
16	Stop	Stop bit number.	Read/Write
17-32	Reserved	Reserved.	N/A

Control and Status Registers

The following tables show the machine-level CSR implementation.

Table 32: Machine Information Register

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

Table 33: 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

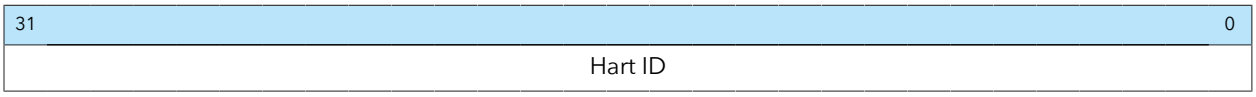
Table 34: Machine Trap Handling Registers

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.



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

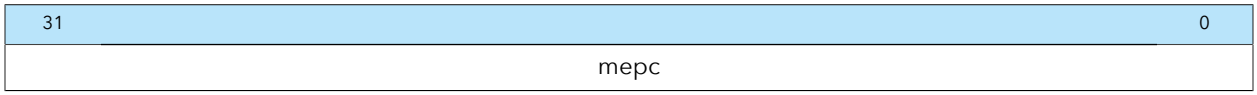
The `mie` register is a 12-bit read/write register containing interrupt enable bits.

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

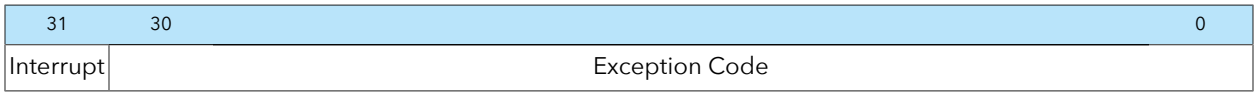
`mepc` is a 32-bit read/write register. The low bit of `mepc` (`mepc[0]`) is always zero. On implementations that support only IALIGN=32, the two low bits (`mepc[1:0]`) are always zero.



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.



Bits	Field	Description	Privilege
0-30	Exception code	See Table 35: Machine Cause Register (mcause) Values after Trap on page 33.	Read
31	Interrupt	mcause interrupt bit.	Read

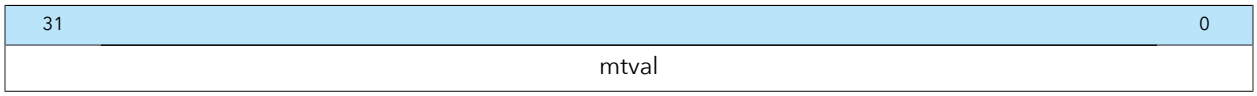
Table 35: Machine Cause Register (mcause) Values after Trap

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

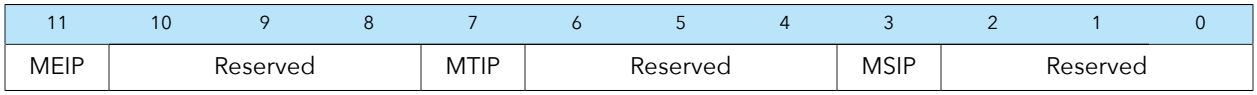
The `mtval` register is a 32-bit register. When a trap is taken into M-mode, `mtval` is either set to zero or written with exception-specific information to assist software in handling the trap. Otherwise, `mtval` is never written by the implementation, though it may be explicitly written by software.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



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.



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 36: Revision History

Date	Version	Description
December 2021	1.6	Updated resource numbers. Clarified AXI interface description. (DOC-633)
September 2021	1.5	The SoC minimum frequency changed to 20 MHz. (DOC-544)
July 2021	1.4	Updated for the Efinity v2021.1 release. Updated the SoC f_{MAX} range. Updated the GPIO register descriptions. (DOC-475) Corrected the SoC block diagram. The AXI interface to the external memory is AXI3 half duplex. Updated the diagram to show support for different types of external mamory.
March 2021	1.3	Corrected widths for axiA_wdata, axiA_wstrb, and axiA_rdata. (DOC-409) Updated SoC operating frequency. Updated on-chip RAM size.
November 2020	1.2	Added UART register descriptions. Updated the address map to show parameters instead of address ranges.
August 2020	1.1	User peripheral address size changed to 64K. io_apbSlave_PADDR size changed to 15:0. AXI user slave peripheral address size changed to 16 MB. Corrected typos.
June 2020	1.0	Initial release.