

# axi\_lite\_spi\_master.v

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2025/04/30

---

## INFORMATION

---

### Brief

---

AXI Lite SPI Master is a core for interfacing with SPI Slave devices.

### License MIT

---

Copyright 2025 Jay Convertino

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## axi\_lite\_spi\_master

---

```
module axi_lite_spi_master #(
  parameter
    ADDRESS_WIDTH
    =
    32,
  parameter
    BUS_WIDTH
    =
    4,
  parameter
    WORD_WIDTH
    =
    4,
  parameter
```

```

CLOCK_SPEED
=
100000000,
parameter
SELECT_WIDTH
=
16,
parameter
DEFAULT_RATE_DIV
=
0,
parameter
DEFAULT_CPOL
=
0,
parameter
DEFAULT_CPHA
=
0,
parameter
FIFO_ENABLE
=
0
)

input
wire
aclk,
input
wire
arstn,
input
wire
s_axi_awvalid,
input
wire
[ADDRESS_WIDTH-1:0]
s_axi_awaddr,
input
wire
[ 2:0]
s_axi_awprot,
output
wire
s_axi_awready,
input
wire
s_axi_wvalid,
input
wire
[(BUS_WIDTH*8)-1:0]
s_axi_wdata,
input
wire
[ 3:0]
s_axi_wstrb,
output
wire
s_axi_wready,
output
wire
s_axi_bvalid,
output
wire
[ 1:0]
s_axi_bresp,

```

(

```

input
wire
s_axi_bready,
input
wire
s_axi_arvalid,
input
wire
[ADDRESS_WIDTH-1:0]
s_axi_araddr,
input
wire
[ 2:0]
s_axi_arprot,
output
wire
s_axi_arready,
output
wire
s_axi_rvalid,
output
wire
[(BUS_WIDTH*8)-1:0]
s_axi_rdata,
output
wire
[ 1:0]
s_axi_rresp,
input
wire
s_axi_rready,
output
wire
irq,
output
wire
sclk,
output
wire
mosi,
input
wire
miso,
output
wire
[SELECT_WIDTH-1:0]
ss_n
)

```

AXI Lite based SPI Master device. BUS\_WIDTH is 4 bytes.

## Parameters

<b>ADDRESS_WIDTH</b> parameter	Width of the uP address port, max 32 bit.
<b>BUS_WIDTH</b> parameter	Width of the uP bus data port, only valid values are 2 or 4.
<b>WORD_WIDTH</b> parameter	Width of each SPI Master word. This will also set the bits used in the TX/RX data registers. Must be less than or equal to BUS_WIDTH. VALID: 1 to 4.
<b>CLOCK_SPEED</b> parameter	This is the aclk frequency in Hz, this is the the frequency used for the bus and is divided by the rate.
<b>SELECT_WIDTH</b> parameter	Bit width of the slave select, defaults to 16 to match altera spi ip.

<b>DEFAULT_RATE_DIV</b> parameter	Default divider value of the main clock to use for the spi data output clock rate. 0 is 2 (2^(X+1) X is the DEFAULT_RATE_DIV)
<b>DEFAULT_CPOL</b> parameter	Default clock polarity for the core (0 or 1).
<b>DEFAULT_CPHA</b> parameter	Default clock phase for the core (0 or 1).
<b>FIFO_ENABLE</b> parameter	Enable a 16 word fifo for RX and TX. The chip select will stay asserted between words.

## Ports

<b>aclk</b> input wire	Clock for all devices in the core
<b>arstn</b> input wire	Negative reset
<b>s_axi_awvalid</b> input wire	Axi Lite aw valid
<b>s_axi_awaddr</b> input wire [ADDRESS_WIDTH- 1:0]	Axi Lite aw addr
<b>s_axi_awprot</b> input wire [2:0]	Axi Lite aw prot
<b>s_axi_awready</b> output wire	Axi Lite aw ready
<b>s_axi_wvalid</b> input wire	Axi Lite w valid
<b>s_axi_wdata</b> input wire [(BUS_WIDTH* 8)- 1:0]	Axi Lite w data
<b>s_axi_wstrb</b> input wire [3:0]	Axi Lite w strb
<b>s_axi_wready</b> output wire	Axi Lite w ready
<b>s_axi_bvalid</b> output wire	Axi Lite b valid
<b>s_axi_bresp</b> output wire [1:0]	Axi Lite b resp
<b>s_axi_bready</b> input wire	Axi Lite b ready
<b>s_axi_arvalid</b> input wire	Axi Lite ar valid
<b>s_axi_araddr</b> input wire [ADDRESS_WIDTH- 1:0]	Axi Lite ar addr
<b>s_axi_arprot</b> input wire [2:0]	Axi Lite ar prot
<b>s_axi_arready</b> output wire	Axi Lite ar ready
<b>s_axi_rvalid</b> output wire	Axi Lite r valid
<b>s_axi_rdata</b> output wire [(BUS_WIDTH* 8)- 1:0]	Axi Lite r data
<b>s_axi_rresp</b> output wire [1:0]	Axi Lite r resp
<b>s_axi_rready</b> input wire	Axi Lite r ready
<b>irq</b>	Interrupt when data is received

```

output wire
sclk                                spi clock, should only drive output pins to devices.
output wire
mosi                                transmit for master output
output wire
miso                                receive for master input
input wire
ss_n                                slave select output
output wire [SELECT_WIDTH- 1:0]

```

## up\_rreq

---

```
wire up_rreq
```

uP read bus request

## up\_rack

---

```
wire up_rack
```

uP read bus acknowledge

## up\_raddr

---

```

wire [ADDRESS_WIDTH-(
BUS_WIDTH
2
)-1:0] up_raddr
/

```

uP read bus address

## up\_rdata

---

```
wire [31:0] up_rdata
```

uP read bus request

## up\_wreq

---

```
wire up_wreq
```

uP write bus request

## up\_wack

---

```
wire up_wack
```

uP write bus acknowledge

## up\_waddr

---

```
wire [ADDRESS_WIDTH-(  
BUS_WIDTH  
2  
)-1:0] up_waddr
```

/

uP write bus address

## up\_wdata

---

```
wire [31:0] up_wdata
```

uP write bus data

## INSTANTIATED MODULES

---

### inst\_up\_axi

---

Module instance of up\_axi for the AXI Lite bus to the uP bus.

### inst\_up\_spi\_master

---

Module instance of up\_spi\_master creating a Logic wrapper for spi master axis bus cores to interface with uP bus.