

# wishbone\_standard\_spi\_master.v

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2025/04/30

---

## INFORMATION

---

### Brief

---

Wishbone Standard SPI Master core.

### 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.

## wishbone\_standard\_spi\_master

---

```
module wishbone_standard_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
clk,
input
wire
rst,
input
wire
s_wb_cyc,
input
wire
s_wb_stb,
input
wire
s_wb_we,
input
wire
[ADDRESS_WIDTH-1:0]
s_wb_addr,
input
wire
[BUS_WIDTH*8-1:0]
s_wb_data_i,
input
wire
[BUS_WIDTH-1:0]
s_wb_sel,
output
wire
s_wb_ack,
output
wire
[BUS_WIDTH*8-1:0]
s_wb_data_o,
output
wire
s_wb_err,
output
wire
irq,
output

```

(

```

wire
sclk,
output
wire
mosi,
input
wire
miso,
output
wire
[SELECT_WIDTH-1:0]
ss_n
)

```

Wishbone Standard based SPI Master device.

## 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. All words put into the fifo together will keep chip select low.

## Ports

<b>clk</b> input wire	Clock for all devices in the core
<b>rst</b> input wire	Positive reset
<b>s_wb_cyc</b> input wire	Bus Cycle in process
<b>s_wb_stb</b> input wire	Valid data transfer cycle
<b>s_wb_we</b> input wire	Active High write, low read
<b>s_wb_addr</b> input wire [ADDRESS_WIDTH- 1:0]	Bus address
<b>s_wb_data_i</b> input wire [BUS_WIDTH* 8- 1:0]	Input data
<b>s_wb_sel</b> input wire [BUS_WIDTH- 1:0]	Device Select
<b>s_wb_ack</b> output wire	Bus transaction terminated

<b>s_wb_data_o</b> output wire [BUS_WIDTH* 8- 1:0]	Output data
<b>s_wb_err</b> output wire	Active high when a bus error is present
<b>irq</b> output wire	Interrupt when data is received
<b>sclk</b> output wire	spi clock, should only drive output pins to devices.
<b>mosi</b> output wire	transmit for master output
<b>miso</b> input wire	receive for master input
<b>ss_n</b> output wire [SELECT_WIDTH- 1:0]	slave select output

## 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\_wishbone\_standard

---

Module instance of up\_wishbone\_standard for the Wishbone Classic Standard 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.