

# **tb\_cocotb.v**

---

## **AUTHORS**

---

**JAY CONVERTINO**

---

## **DATES**

---

**2025/03/26**

---

## **INFORMATION**

---

### **Brief**

---

Test bench wrapper for cocotb

### **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.BUS\_WIDTH

## **tb\_cocotb**

---

```
module tb_cocotb #(
    parameter
        integer
        ADDRESS_WIDTH
        =
        32,
    parameter
        integer
        BUS_WIDTH
        =
        4,
    parameter
        [ADDRESS_WIDTH-1:0]
        SLAVE_ADDRESS
```

```
=  
32'h44A20000,  
parameter  
    [ADDRESS_WIDTH-1:0]  
SLAVE_REGION  
=  
32'h0000FFFF  
)  
  
output  
wire  
connected,  
input  
wire  
aclk,  
input  
wire  
arstn,  
input  
wire  
[ADDRESS_WIDTH-1:0]  
s_axi_awaddr,  
input  
wire  
[2:0]  
s_axi_awprot,  
input  
wire  
s_axi_awvalid,  
output  
wire  
s_axi_awready,  
input  
wire  
[BUS_WIDTH*8-1:0]  
s_axi_wdata,  
input  
wire  
[BUS_WIDTH-1:0]  
s_axi_wstrb,  
input  
wire  
s_axi_wvalid,  
output  
wire  
s_axi_wready,  
output  
wire  
[1:0]  
s_axi_bresp,  
output  
wire  
s_axi_bvalid,  
input  
wire  
s_axi_bready,  
output  
wire  
[ADDRESS_WIDTH-1:0]  
m_axi_awaddr,  
output  
wire  
[2:0]  
m_axi_awprot,  
output  
wire
```

```
m_axi_awvalid,
input
wire
m_axi_awready,
output
wire
[BUS_WIDTH*8-1:0]
m_axi_wdata,
output
wire
[BUS_WIDTH-1:0]
m_axi_wstrb,
output
wire
m_axi_wvalid,
input
wire
m_axi_wready,
input
wire
[BUS_WIDTH-1:0]
m_axi_bresp,
input
wire
m_axi_bvalid,
output
wire
m_axi_bready,
input
wire
[ADDRESS_WIDTH-1:0]
s_axi_araddr,
input
wire
[2:0]
s_axi_arprot,
input
wire
s_axi_arvalid,
output
wire
s_axi_arready,
output
wire
[BUS_WIDTH*8-1:0]
s_axi_rdata,
output
wire
[BUS_WIDTH-1:0]
s_axi_rresp,
output
wire
s_axi_rvalid,
input
wire
s_axi_rready,
output
wire
[ADDRESS_WIDTH-1:0]
m_axi_araddr,
output
wire
[2:0]
m_axi_arprot,
output
wire
```

```

    m_axi_arvalid,
    input
    wire
    m_axi_arready,
    input
    wire
        [BUS_WIDTH*8-1:0]
    m_axi_rdata,
    input
    wire
        [1:0]
    m_axi_rresp,
    input
    wire
    m_axi_rvalid,
    output
    wire
    m_axi_rready
)

```

## Parameters

<b>ADDRESS_WIDTH</b>	Width of the AXI LITE address port in bits.
<b>parameter integer</b>	
<b>BUS_WIDTH</b>	Width of the AXI LITE bus data port in bytes.
<b>parameter integer</b>	
<b>DATA_BUFFER</b>	Buffer data channel, 0 to disable.
<b>TIMEOUT_BEATS</b>	Number of clock cycles (beats) to count till timeout. 0 disables timeout.
<b>SLAVE_ADDRESS</b>	Array of Addresses for each slave (0 = slave 0 and so on).
<b>parameter [ADDRESS_WIDTH- 1:0]</b>	
<b>SLAVE_REGION</b>	Region for the address that is valid for the SLAVE ADDRESS.
<b>parameter [ADDRESS_WIDTH- 1:0]</b>	

## Ports

<b>connected</b>	Core has established channel connection
<b>output wire</b>	
<b>aclk</b>	Input clock
<b>input wire</b>	
<b>arstn</b>	Input negative reset
<b>input wire</b>	
<b>s_axi_awaddr</b>	Slave write input channel address
<b>input wire [ADDRESS_WIDTH- 1:0]</b>	
<b>s_axi_awprot</b>	Slave write input channel protection mode
<b>input wire [2:0]</b>	
<b>s_axi_awvalid</b>	Slave write input channel address is valid.
<b>input wire</b>	
<b>s_axi_awready</b>	Slave write input channel is ready.
<b>output wire</b>	
<b>s_axi_wdata</b>	Slave write input channel data
<b>input wire [BUS_WIDTH* 8- 1:0]</b>	
<b>s_axi_wstrb</b>	Slave write input channel valid bytes
<b>input wire [BUS_WIDTH- 1:0]</b>	
<b>s_axi_wvalid</b>	Slave write input channel data valid
<b>input wire</b>	
<b>s_axi_wready</b>	Slave write input channel is ready.
<b>output wire</b>	

<b>s_axi_bresp</b>	Slave write input channel response to write(s).
<b>s_axi_bvalid</b>	Slave write input channel response valid.
<b>s_axi_bready</b>	Slave write input channel response ready.
<b>m_axi_awaddr</b>	Master write output channel address.
<b>m_axi_awprot</b>	Master write output channel protection mode.
<b>m_axi_awvalid</b>	Master write output channel address is valid.
<b>m_axi_awready</b>	Master write output channel is ready.
<b>m_axi_wdata</b>	Master write output channel data.
<b>m_axi_wstrb</b>	Master write output channel data bytes valid.
<b>m_axi_wvalid</b>	Master write output channel data is valid.
<b>m_axi_wready</b>	Master write output channel data ready.
<b>m_axi_bresp</b>	Master write output channel response.
<b>m_axi_bvalid</b>	Master write output channel response valid.
<b>m_axi_bready</b>	Master write output channel response ready.

## INSTANTIATED MODULES

---

### dut

Device under test, axi\_lite\_wr\_addr