

# tb\_cocotb.v

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2025/01/21

---

## 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.

## tb\_cocotb

---

```
module tb_cocotb #(
  parameter
    CLOCK_SPEED
    =
    20000000,
  parameter
    BUS_WIDTH
    =
    1,
  parameter
    RX_BAUD_DELAY
    =
    0,
  parameter
```

```

TX_BAUD_DELAY
=
0
) ( input wire aclk, input wire arstn, input wire [ 2:0] reg_parity, input v

```

Test bench for axis uart.  
 AXIS UART DTE, a UART with AXI Streaming interface.

### Parameters

<b>CLOCK_SPEED</b> <i>parameter</i>	This is the aclk frequency in Hz
<b>BUS_WIDTH</b> <i>parameter</i>	AXIS data bus width in bytes.
<b>RX_BAUD_DELAY</b> <i>parameter</i>	DELAY RX internal baud rate by CLOCK_SPEED number of cycles.
<b>TX_BAUD_DELAY</b> <i>parameter</i>	DELAY TX internal baud rate by CLOCK_SPEED number of cycles.

### Ports

<b>aclk</b>	Clock for AXIS
<b>arstn</b>	Negative reset for AXIS
<b>reg_parity</b>	Set the parity type, 0 = none, 1 = odd, 2 = even, 3 = mark , 4 = space
<b>reg_stop_bits</b>	Set the number of stop bits (0 to 3, 0=0, 1=1, 2=2, 3=??).
<b>reg_data_bits</b>	Set the number of data bits up to the BUS_WIDTH*8 (1 to 16, all values are biased by 1, 0+1=1).
<b>reg_baud_rate</b>	Frequency in Hz for the output/input data rate. This can be up to half of AXIS clock (any 32 bit unsigned value in Hz).
<b>reg_istatus_bits</b>	Collection of input status bits for dtr,cts,dts,dcd.
<b>reg_ostatus_bits</b>	Collection of output status bits for rx/tx frame, rx parity.
<b>s_axis_tdata</b>	Input data for UART TX.
<b>s_axis_tvalid</b>	When set active high the input data is valid
<b>s_axis_tready</b>	When active high the device is ready for input data.
<b>m_axis_tdata</b>	Output data from UART RX
<b>m_axis_tvalid</b>	When active high the output data is valid
<b>m_axis_tready</b>	When set active high the output device is ready for data.
<b>uart_clk</b>	Clock used for BAUD rate generation
<b>uart_rstn</b>	Negative reset for UART, for anything clocked on uart_clk
<b>tx</b>	transmit for UART (output to RX)
<b>rx</b>	receive for UART (input from TX)
<b>rts</b>	request to send is a loop with CTS
<b>dtr</b>	data terminal ready
<b>cts</b>	clear to send is a loop with RTS
<b>ri</b>	ring indicator

### INSTANTIATED MODULES

dut

```

axis_uart #(
    CLOCK_SPEED(CLOCK_SPEED),
    BUS_WIDTH(BUS_WIDTH),
    RX_BAUD_DELAY(0),
    TX_BAUD_DELAY(0)
) dut ( .aclk(aclk), .arstn(arstn), .reg_parity(reg_parity), .reg_stop_bits(

```

Device under test, axis\_uart