

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
    BAUD_CLOCK_SPEED
    =
    20000000,
  parameter
    BAUD_RATE
    =
    20000000,
  parameter
    PARITY_ENA
    =
    0,
  parameter
```

```

    PARITY_TYPE
    =
    0,
    parameter
    STOP_BITS
    =
    1,
    parameter
    DATA_BITS
    =
    8,
    parameter
    RX_DELAY
    =
    0,
    parameter
    RX_BAUD_DELAY
    =
    0,
    parameter
    TX_DELAY
    =
    0,
    parameter
    TX_BAUD_DELAY
    =
    0,
    parameter
    BUS_WIDTH
    =
    1
) ( input aclk, input arstn, output parity_err, output frame_err, input [BUS

```

Test bench for axis uart.

Parameters

BAUD_CLOCK_SPEED parameter	This is the aclk frequency in Hz
BAUD_RATE parameter	Serial Baud, this can be any value including non-standard.
PARITY_ENA parameter	Enable Parity for the data in and out.
PARITY_TYPE parameter	Set the parity type, 0 = even, 1 = odd, 2 = mark, 3 = space.
STOP_BITS parameter	Number of stop bits, 0 to crazy non-standard amounts.
DATA_BITS parameter	Number of data bits, 1 to crazy non-standard amounts.
RX_DELAY parameter	Delay in rx data input.
RX_BAUD_DELAY parameter	Delay in rx baud enable. This will delay when we sample a bit (default is midpoint when rx delay is 0).
TX_DELAY parameter	Delay in tx data output. Delays the time to output of the data.
TX_BAUD_DELAY parameter	Delay in tx baud enable. This will delay the time the bit output starts.
BUS_WIDTH parameter	AXIS data bus width in bytes.

Ports

aclk	Clock for AXIS
arstn	Negative reset for AXIS
parity_err	Indicates error with parity check (active high)
frame_err	Indicates error with frame (active high)
s_axis_tdata	Input data for UART TX.
s_axis_tvalid	When set active high the input data is valid
s_axis_tready	When active high the device is ready for input data.
m_axis_tdata	Output data from UART RX
m_axis_tvalid	When active high the output data is valid
m_axis_tready	When set active high the output device is ready for data.
uart_clk	Clock used for BAUD rate generation
uart_rstn	Negative reset for UART, for anything clocked on uart_clk
tx	transmit for UART (output to RX)
rx	receive for UART (input from TX)
rts	request to send is a loop with CTS
cts	clear to send is a loop with RTS

INSTANTIATED MODULES

dut

```
axis_uart #(
    BAUD_CLOCK_SPEED(BAUD_CLOCK_SPEED),
    BAUD_RATE(BAUD_RATE),
    PARITY_ENA(PARITY_ENA),
    PARITY_TYPE(PARITY_TYPE),
    STOP_BITS(STOP_BITS),
    DATA_BITS(DATA_BITS),
    RX_DELAY(RX_DELAY),
    RX_BAUD_DELAY(RX_BAUD_DELAY),
    TX_DELAY(TX_DELAY),
    TX_BAUD_DELAY(TX_BAUD_DELAY),
    BUS_WIDTH(BUS_WIDTH)
) dut ( .acclk(acclk), .arstn(arstn), .parity_err(parity_err), .frame_err(frame_err), .tx(tx), .rx(rx), .rts(rts), .cts(cts) );
```

Device under test, axis_uart