

tb_cocotb_wishbone_standard.v

AUTHORS

JAY CONVERTINO

DATES

2025/04/01

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
  ADDRESS_WIDTH
  =
  32,
  parameter
  BUS_WIDTH
  =
  4,
  parameter
  GPIO_WIDTH
  =
  32,
  parameter
```

```

    IRQ_ENABLE
    =
    0
) ( input clk, input rst, input s_wb_cyc, input s_wb_stb, input s_wb_we, input s_wb_ack, input s_wb_err, input irq, input gpio_io_i, output gpio_io_o, inout gpio_io_t )

```

Wishbone Stanard based GPIO communications device.

Parameters

ADDRESS_WIDTH <small>parameter</small>	Width of the uP address port, max 32 bit.
BUS_WIDTH <small>parameter</small>	Width of the uP bus data port.
GPIO_WIDTH <small>parameter</small>	Width of the GPIO for inputs and outputs
IRQ_ENABLE <small>parameter</small>	Enable interrupt

Ports

clk	Clock for all devices in the core
rst	Positive reset
s_wb_cyc	Bus Cycle in process
s_wb_stb	Valid data transfer cycle
s_wb_we	Active High write, low read
s_wb_addr	Bus address
s_wb_data_i	Input data
s_wb_sel	Device Select
s_wb_ack	Bus transaction terminated
s_wb_data_o	Output data
s_wb_err	Active high when a bus error is present
irq	Interrupt when data is received
gpio_io_i	Input for GPIO
gpio_io_o	Output for GPIO
gpio_io_t	Tristate for GPIO

INSTANTIATED MODULES

dut

```

wishbone_standard_gpio #(
    ADDRESS_WIDTH(ADDRESS_WIDTH),
    BUS_WIDTH(BUS_WIDTH),
    GPIO_WIDTH(GPIO_WIDTH),
    IRQ_ENABLE(IRQ_ENABLE)
) dut ( .clk(clk), .rst(rst), .s_wb_cyc(s_wb_cyc), .s_wb_stb(s_wb_stb), .s_wb_we(s_wb_we), .s_wb_ack(s_wb_ack), .s_wb_err(s_wb_err), .irq(irq), .gpio_io_i(gpio_io_i), .gpio_io_o(gpio_io_o), .gpio_io_t(gpio_io_t) )

```

Device under test, wishbone_standard_gpio

