

# tb\_wishbone\_cocotb.py

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2024/12/09

---

## INFORMATION

---

### Brief

---

Cocotb test bench

### License MIT

---

Copyright 2024 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.

## FUNCTIONS

---

### random\_bool

---

```
def random_bool()
```

Return a infinte cycle of random bools

Returns: List

### start\_clock

---

```
def start_clock(  
    dut  
)
```

Start the simulation clock generator.

### Parameters

**dut**     Device under test passed from cocotb test function

## reset\_dut

---

```
async def reset_dut(  
    dut  
)
```

Cocotb coroutine for resets, used with await to make sure system is reset.

## single\_word

---

```
@cocotb.test()  
async def single_word(  
    dut  
)
```

Coroutine that is identified as a test routine. This routine tests for writing a single word, and then reading a single word.

### Parameters

**dut**     Device under test passed from cocotb.

## full\_empty

---

# Coroutine that is identified as a test routine. This routine tests for writing till the fifo is full, # Then reading from the full FIFO. ## Parameters: # dut - Device under test passed from cocotb. @cocotb.test() async def full\_empty(dut):

## random\_ready

---

# Coroutine that is identified as a test routine. This routine tests for randomized ready from the sink. ## Parameters: # dut - Device under test passed from cocotb. @cocotb.test() async def random\_ready(dut):

## in\_reset

---

```
@cocotb.test()  
async def in_reset(  
    dut  
)
```

Coroutine that is identified as a test routine. This routine tests if device stays in unready state when in reset.

### Parameters

**dut** Device under test passed from cocotb.

### no\_clock

---

```
@cocotb.test()
async def no_clock(
    dut
)
```

Coroutine that is identified as a test routine. This routine tests if no ready when clock is lost and device is left in reset.

### Parameters

**dut** Device under test passed from cocotb.