

# AXIS\_FIFO



May 20, 2025

Jay Convertino

# Contents

<b>1 Usage</b>	<b>2</b>
1.1 Introduction . . . . .	2
1.2 Dependencies . . . . .	2
1.2.1 fusesoc_info Depenecies . . . . .	2
1.3 In a Project . . . . .	2
<b>2 Architecture</b>	<b>2</b>
<b>3 Building</b>	<b>3</b>
3.1 fusesoc . . . . .	3
3.2 Source Files . . . . .	3
3.2.1 fusesoc_info File List . . . . .	3
3.3 Targets . . . . .	4
3.3.1 fusesoc_info Targets . . . . .	4
3.4 Directory Guide . . . . .	4
<b>4 Simulation</b>	<b>5</b>
4.1 iverilog . . . . .	5
4.2 cocotb . . . . .	5
<b>5 Code Documentation</b>	<b>6</b>
5.1 axis_fifo . . . . .	7
5.2 axis_fifo_ctrl . . . . .	11
5.3 tb_axis . . . . .	13
5.4 tb_cocotb verilog . . . . .	16
5.5 tb_cocotb python . . . . .	19

# 1 Usage

## 1.1 Introduction

Standard AXIS FIFO with multiple options. The FIFO uses a AXIS interface for data in and out. It also emulates the Xilinx AXIS FIFO bugs and all. This is NOT dependent on Xilinx FPGA's and can be used on any FPGA supporting the Verilog block ram style primitive.

## 1.2 Dependencies

The following are the dependencies of the cores.

- fusesoc 2.X
- iverilog (simulation)
- cocotb (simulation)

### 1.2.1 fusesoc\_info Depenecies

- dep
  - AFRL:buffer:fifo
  - AFRL:utility:helper:1.0.0
- dep\_tb
  - AFRL:simulation:axis\_stimulator
  - AFRL:simulation:clock\_stimulator
  - AFRL:utility:sim\_helper

## 1.3 In a Project

Simply use this core between a sink and source devices. This buffer data from one bus to another. Check the code to see if others will work correctly.

# 2 Architecture

This AXIS FIFO is made for two modules. They are the FIFO, and AXIS FIFO control. The combination of these two provide the AXIS FIFO module. Having it made this way allows for future modules to be customized and brought in to change the FIFO's behavior. The current

modules emulate the Xilinx AXIS FIFO IP core available in Vivado 2018 and up.

AXIS FIFO control is the heart of the core when it comes to how it responds. The logic in the core is designed to emulate the Xilinx AXIS FIFO IP.

Please see 5 for more information.

## **3 Building**

The AXIS FIFO core is written in Verilog 2001. They should synthesize in any modern FPGA software. The core comes as a fusesoc packaged core and can be included in any other core. Be sure to make sure you have met the dependencies listed in the previous section. Linting for the lint target is performed using verible.

### **3.1 fusesoc**

Fusesoc is a system for building FPGA software without relying on the internal project management of the tool. Avoiding vendor lock in to Vivado or Quartus. These cores, when included in a project, can be easily integrated and targets created based upon the end developer needs. The core by itself is not a part of a system and should be integrated into a fusesoc based system. Simulations are setup to use fusesoc and are a part of its targets.

### **3.2 Source Files**

#### **3.2.1 fusesoc\_info File List**

- src
  - src/axis\_fifo.v
  - src/axis\_fifo\_ctrl.v
- tb
  - 'tb/tb\_axis.v': 'file\_type': 'verilogSource'
- tb\_cocotb
  - 'tb/tb\_cocotb.py': 'file\_type': 'user', 'copyto': '.'
  - 'tb/tb\_cocotb.v': 'file\_type': 'verilogSource'

## 3.3 Targets

### 3.3.1 fusesoc\_info Targets

- default  
Info: Default for IP intergration.
- lint  
Info: Lint with Verible
- sim  
Info: Constant data value with file check.
- sim\_rand\_data  
Info: Feed random data input with file check
- sim\_rand\_ready\_rand\_data  
Info: Feed random data input, and randomize the read ready on the output. Perform output file check.
- sim\_8bit\_count\_data  
Info: Feed a counter data as input, perform file check.
- sim\_rand\_ready\_8bit\_count\_data  
Info: Feed a counter data a input, and randomize the read ready on the output. Perform output file check.
- sim\_cocotb  
Info: Cocotb unit tests

## 3.4 Directory Guide

Below highlights important folders from the root of the directory.

1. **docs** Contains all documentation related to this project.
  - **manual** Contains user manual and github page that are generated from the latex sources.
2. **src** Contains source files for the core
3. **tb** Contains test bench files for iverilog and cocotb

## 4 Simulation

There are a few different simulations that can be run for this core. All currently use iVerilog (icarus) to run. The first is iverilog, which uses verilog only for the simulations. The other is cocotb. This does a unit test approach to the testing and gives a list of tests that pass or fail.

### 4.1 iverilog

All simulation targets that do NOT have cocotb in the name use a verilog test bench with verilog stimulus components. These all read in a file and then write a file that has been processed by the FIFO. Then the input and output file are compared with a MD5 sum to check that they match. If they do not match then the test has failed. All of these tests provide fst output files for viewing the waveform in the there target build folder.

### 4.2 cocotb

To use the cocotb tests you must install the following python libraries.

```
$ pip install cocotb
$ pip install cocotbext-axi
```

Then you must use the cocotb sim target. In this case it is sim\_cocotb. This target can be run with various bus and fifo parameters.

```
$ fusesoc run --target sim_cocotb AFRL:buffer:axis_fifo
  ↳ :1.0.0 --BUS_WIDTH=8 --FIFO_DEPTH=32
```

The following is an example command to run through various parameters without typing them one by one.

```
$ for i in {1..32}; do sleep 5; export RY=$((($RANDOM
  ↳ %32+1)); fusesoc run --target sim_cocotb AFRL:
  ↳ buffer:axis_fifo:1.0.0 --BUS_WIDTH=$i --
  ↳ FIFO_DEPTH=$RY; echo "BUS_WIDTH:" $i "FIFO_DEPTH:
  ↳ " $RY; done
```

## 5 Code Documentation

Natural docs is used to generate documentation for this project. The next lists the following sections.

- **AXIS\_FIFO** will buffer data from input to output.
- **AXIS\_FIFO\_CONTROL** emulates the Xilinx FIFO IP interface and its behavior.
- **tb\_axis** Verilog test bench.
- **tb\_cocotb verilog** Verilog test bench base for cocotb.
- **tb\_cocotb python** cocotb unit test functions.

# axis\_fifo.v

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2021/06/29

---

## INFORMATION

---

### Brief

---

Wraps the standard FIFO with an axi streaming interface.

### License MIT

---

Copyright 2021 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.

## axis\_fifo

---

```
module axis_fifo #(
    parameter
    FIFO_DEPTH
    =
    256,
    parameter
    COUNT_WIDTH
    =
    8,
    parameter
    BUS_WIDTH
    =
    1,
    parameter
```



```

USER_WIDTH
=
1,
parameter
DEST_WIDTH
=
1,
parameter
RAM_TYPE
=
"block",
parameter
PACKET_MODE
=
0,
parameter
COUNT_DELAY
=
1,
parameter
COUNT_ENA
=
1
) ( input m_axis_aclk, input m_axis_arstn, output m_axis_tvalid, input m_ax:

```

AXIS fifo

## Parameters

<b>FIFO_DEPTH</b> parameter	Depth of the fifo, must be a power of two number(divisable aka $256 = 2^8$ ). Any non-power of two will be rounded up to the next closest.
<b>COUNT_WIDTH</b> parameter	Data count output width in bits. Should be the same power of two as fifo depth(256 for fifo depth... this should be 8).
<b>BUS_WIDTH</b> parameter	Width of the axis data bus input/output in bytes.
<b>USER_WIDTH</b> parameter	Width of the axis user bus input/output in bits.
<b>DEST_WIDTH</b> parameter	Width of the axis dest bus input/output in bits.
<b>RAM_TYPE</b> parameter	RAM type setting.
<b>PACKET_MODE</b> parameter	Set axis fifo to wait for tlast before allowing a read on master port output.
<b>COUNT_DELAY</b> parameter	Delay count by one clock cycle of the data count clock.
<b>COUNT_ENA</b> parameter	Enable count, set this to 0 to disable (only disable if read/write/data_count are on the same clock domain!).

## Ports

<b>m_axis_aclk</b>	Clock for AXIS
<b>m_axis_arstn</b>	Negative reset for AXIS
<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>m_axis_tdata</b>	Output data
<b>m_axis_tkeep</b>	Output valid byte indicator
<b>m_axis_tlast</b>	Indicates last word in stream.
<b>m_axis_tuser</b>	Output user bus

<b>m_axis_tdest</b>	Output destination
<b>s_axis_aclk</b>	Clock for AXIS
<b>s_axis_arstn</b>	Negative reset for AXIS
<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>s_axis_tdata</b>	Input data
<b>s_axis_tkeep</b>	Input valid byte indicator
<b>s_axis_tlast</b>	Is this the last word in the stream (active high).
<b>s_axis_tuser</b>	Input user bus
<b>s_axis_tdest</b>	Input desitination
<b>data_count_aclk</b>	Clock for data count
<b>data_count_arstn</b>	Negative edge reset for data count.
<b>data_count</b>	Output that indicates the amount of data in the FIFO.

## INSTANTIATED MODULES

## axis\_fifo

```
fifo #(  
FIFO_DEPTH  
c_FIFO_DEPTH),  
BYTE_WIDTH  
c_FIFO_WIDTH),  
COUNT_WIDTH  
COUNT_WIDTH),  
FWFT  
1),  
RD_SYNC_DEPTH  
0),  
WR_SYNC_DEPTH  
0),  
DC_SYNC_DEPTH  
0),  
COUNT_DELAY  
COUNT_DELAY),  
COUNT_ENA  
COUNT_ENA),
```

```

DATA_ZERO
1),
ACK_ENA
0),
RAM_TYPE
RAM_TYPE)
) axis_fifo ( .rd_clk (m_axis_aclk), .rd_rstn (m_axis_arstn), .rd_en (s_rd_en

```

Generic FIFO that acts like a Xilinx FIFO.

## axis\_control

```

axis_fifo_ctrl #(
BUS_WIDTH
BUS_WIDTH),
FIFO_WIDTH
c_FIFO_WIDTH),
USER_WIDTH
USER_WIDTH),
DEST_WIDTH
DEST_WIDTH),
PACKET_MODE(PACKET_MODE)
) axis_control ( .m_axis_aclk (m_axis_aclk), .m_axis_arstn (m_axis_arstn),

```

Create signals to control FIFO and provide AXIS interace.

## axis\_ctrl\_fifo.v

---

### AUTHORS

---

JAY CONVERTINO

---

### DATES

---

2021/06/29

---

### INFORMATION

---

#### Brief

---

Wraps the standard FIFO with an axi streaming interface.

#### License MIT

---

Copyright 2021 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.

## axis\_fifo\_ctrl

---

```
module axis_fifo_ctrl #(
    parameter
    BUS_WIDTH
    =
    1,
    parameter
    FIFO_WIDTH
    =
    8,
    parameter
    FIFO_POWER
    =
    8,
    parameter
```

```

USER_WIDTH
=
1,
parameter
DEST_WIDTH
=
1,
parameter
PACKET_MODE
=
0
) ( input m_axis_aclk, input m_axis_arstn, output m_axis_tvalid, input m_ax:

```

AXIS fifo control

## Parameters

<b>BUS_WIDTH</b> parameter	Width of the axis data bus input/output in bytes. FIFO_WIDTH - FIFO_POWER -
<b>USER_WIDTH</b> parameter	Width of the axis user bus input/output in bits.
<b>DEST_WIDTH</b> parameter	Width of the axis dest bus input/output in bits.
<b>PACKET_MODE</b> parameter	Set axis fifo to wait for tlast before allowing a read on master port output.

## Ports

<b>m_axis_aclk</b>	Clock for AXIS
<b>m_axis_arstn</b>	Negative reset for AXIS
<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>m_axis_tdata</b>	Output data
<b>m_axis_tkeep</b>	Output valid byte indicator
<b>m_axis_tlast</b>	Indicates last word in stream.
<b>m_axis_tuser</b>	Output user bus
<b>m_axis_tdest</b>	Output destination
<b>s_axis_tlast</b>	Is this the last word in the stream (active high).
<b>rd_en</b>	Active high enable of read interface.
<b>rd_valid</b>	Active high output that the data is valid.
<b>rd_data</b>	Output data
<b>rd_empty</b>	Active high output when read is empty.
<b>wr_full</b>	Active high output that the FIFO is full.

## tb\_axis.v

---

### AUTHORS

---

JAY CONVERTINO

---

### DATES

---

2024/12/09

---

### INFORMATION

---

#### Brief

---

Test bench for axis\_fifo using axis stim and clock stim.

#### 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.

## tb\_axis

---

```
module tb_axis #(
  parameter
  IN_FILE_NAME
  =
  "in.bin",
  parameter
  OUT_FILE_NAME
  =
  "out.bin",
  parameter
  FIFO_DEPTH
  =
  128,
  parameter
```

```

    RAND_READY
    =
    0
)

```

Test bench for axis\_fifo. This will run a file through the system and write its output. These can then be compared to check for errors. If the files are identical, no errors. A FST file will be written.

### Parameters

<b>IN_FILE_NAME</b> parameter	File name for input.
<b>OUT_FILE_NAME</b> parameter	File name for output.
<b>FIFO_DEPTH</b> parameter	Number of transactions to buffer.
<b>RAND_READY</b> parameter	0 = no random ready. 1 = randomize ready.

## INSTANTIATED MODULES

---

### clk\_stim

---

```

clk_stimulus #(
    CLOCKS(2),
    CLOCK_BASE(1000000),
    CLOCK_INC(10),
    RESETS(2),
    RESET_BASE(2000),
    RESET_INC(100)
) clk_stim ( .clkv({tb_dut_clk, tb_stim_clk}), .rstnv({tb_dut_rstn, tb_stim_rstn})

```

Generate a 50/50 duty cycle set of clocks and reset.

### slave\_axis\_stim

---

```

slave_axis_stimulus #(
    BUS_WIDTH(BUS_WIDTH),
    USER_WIDTH(USER_WIDTH),
    DEST_WIDTH(DEST_WIDTH),
    FILE(IN_FILE_NAME)
) slave_axis_stim ( .m_axis_aclk(tb_stim_clk), .m_axis_arstn(tb_stim_rstn),

```

Device under test SLAVE stimulus module.

## dut

---

```
axis_fifo #(
    FIFO_DEPTH(FIFO_DEPTH),
    COUNT_WIDTH(8),
    BUS_WIDTH(BUS_WIDTH),
    USER_WIDTH(USER_WIDTH),
    DEST_WIDTH(DEST_WIDTH),
    RAM_TYPE("block"),
    PACKET_MODE(0),
    COUNT_DELAY(1),
    COUNT_ENA(1)
) dut ( .s_axis_aclk(tb_stim_clk), .s_axis_arstn(tb_stim_rstn), .s_axis_tva
```

Device under test, axis\_fifo

## master\_axis\_stim

---

```
master_axis_stimulus #(
    BUS_WIDTH(BUS_WIDTH),
    USER_WIDTH(USER_WIDTH),
    DEST_WIDTH(DEST_WIDTH),
    RAND_READY(RAND_READY),
    FILE(OUT_FILE_NAME)
) master_axis_stim ( .s_axis_aclk(tb_dut_clk), .s_axis_arstn(tb_dut_rstn),
```

Devie under test MASTER stimulus module.



# tb\_cocotb.v

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2024/12/10

---

## INFORMATION

---

### Brief

---

Test bench wrapper for cocotb

### 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.

## tb\_cocotb

---

```
module tb_cocotb #(
  parameter
    FIFO_DEPTH
    =
    256,
  parameter
    COUNT_WIDTH
    =
    8,
  parameter
    BUS_WIDTH
    =
    1,
  parameter
```

```

USER_WIDTH
=
1,
parameter
DEST_WIDTH
=
1,
parameter
RAM_TYPE
=
"block",
parameter
PACKET_MODE
=
0,
parameter
COUNT_DELAY
=
1,
parameter
COUNT_ENA
=
1
) ( input m_axis_aclk, input m_axis_arstn, output m_axis_tvalid, input m_ax:

```

Test bench for axis\_fifo. This will run a file through the system and write its output. These can then be compared to check for errors. If the files are identical, no errors. A FST file will be written.

## Parameters

<b>FIFO_DEPTH</b> parameter	Depth of the fifo, must be a power of two number(divisable aka $256 = 2^8$ ). Any non-power of two will be rounded up to the next closest.
<b>COUNT_WIDTH</b> parameter	Data count output width in bits. Should be the same power of two as fifo depth(256 for fifo depth... this should be 8).
<b>BUS_WIDTH</b> parameter	Width of the axis data bus input/output in bytes.
<b>USER_WIDTH</b> parameter	Width of the axis user bus input/output in bits.
<b>DEST_WIDTH</b> parameter	Width of the axis dest bus input/output in bits.
<b>RAM_TYPE</b> parameter	RAM type setting.
<b>PACKET_MODE</b> parameter	Set axis fifo to wait for tlast before allowing a read on master port output.
<b>COUNT_DELAY</b> parameter	Delay count by one clock cycle of the data count clock.
<b>COUNT_ENA</b> parameter	Enable count, set this to 0 to disable (only disable if read/write/data_count are on the same clock domain!).

## Ports

<b>m_axis_aclk</b>	Clock for AXIS
<b>m_axis_arstn</b>	Negative reset for AXIS
<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>m_axis_tdata</b>	Output data
<b>m_axis_tkeep</b>	Output valid byte indicator
<b>m_axis_tlast</b>	Indicates last word in stream.

<b>m_axis_tuser</b>	Output user bus
<b>m_axis_tdest</b>	Output destination
<b>s_axis_aclk</b>	Clock for AXIS
<b>s_axis_arstn</b>	Negative reset for AXIS
<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>s_axis_tdata</b>	Input data
<b>s_axis_tkeep</b>	Input valid byte indicator
<b>s_axis_tlast</b>	Is this the last word in the stream (active high).
<b>s_axis_tuser</b>	Input user bus
<b>s_axis_tdest</b>	Input destination
<b>data_count_aclk</b>	Clock for data count
<b>data_count_arstn</b>	Negative edge reset for data count.
<b>data_count</b>	Output that indicates the amount of data in the FIFO.

## INSTANTIATED MODULES

---

### dut

---

```
axis_fifo #(
    FIFO_DEPTH(FIFO_DEPTH),
    COUNT_WIDTH(COUNT_WIDTH),
    BUS_WIDTH(BUS_WIDTH),
    USER_WIDTH(USER_WIDTH),
    DEST_WIDTH(DEST_WIDTH),
    RAM_TYPE(RAM_TYPE),
    PACKET_MODE(PACKET_MODE),
    COUNT_DELAY(COUNT_DELAY),
    COUNT_ENA(COUNT_ENA)
) dut ( .s_axis_aclk(s_axis_aclk), .s_axis_arstn(s_axis_arstn), .s_axis_tva
```

Device under test, axis\_fifo

# tb\_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 infinite 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

---

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

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.

## random\_ready

---

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

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.

## 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.