

# 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> <i>parameter</i>	File name for input.
<b>OUT_FILE_NAME</b> <i>parameter</i>	File name for output.
<b>FIFO_DEPTH</b> <i>parameter</i>	Number of transactions to buffer.
<b>RAND_READY</b> <i>parameter</i>	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.