

AXIS_STRING_TO_AXIS_DATA



December 12, 2024

Jay Convertino

Contents

1 Usage	2
1.1 Introduction	2
1.2 Dependencies	2
1.2.1 fusesoc_info Depenecies	2
1.3 In a Project	2
2 Architecture	2
3 Building	3
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
4 Simulation	5
4.1 iverilog	5
4.2 cocotb	5
5 Code Documentation	6
5.1 axis_string_to_axis_data	7
5.2 tb_axis	10
5.3 tb_cocotb verilog	13
5.4 tb_cocotb python	17

1 Usage

1.1 Introduction

This core takes a incoming string, removes the delimiters, terminators, and prefixes. After this all HEX value characters are converted into there binary values and output to the ports specified by the prefix.

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:utility:helper:1.0.0
- dep_tb
 - AFRL:simulation:axis_stimulator
 - AFRL:simulation:clock_stimulator

1.3 In a Project

Simply use this core between a sink and source AXIS devices. This will convert from input string into an output data one character at a time. Check the code to see if others will work correctly.

2 Architecture

The only module is the axis_string_to_axis_data module. It is listed below.

- **axis_string_to_axis_data** Implement an algorithm to convert input string to data (see core for documentation).

The only always process converts the input string to data.

1. If destination device is ready, clear oout registered output.
2. if we have valid data, insert it into the buffer and increment count.
 - (a) Counter down to last element? Clear and reset to full length.
 - (b) if we have the terminator and delimiter, process buffer.
 - i. Check for the type of prefix, based on that prefix look at each nibble and offset by its ASCII 0 to F to 0 to 15 binary.
 - ii. Check for set or clear keyword, if set output data. If clear, remove all data.

Please see ?? for more information.

3 Building

The AXIS string to AXIS data 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 meet the dependencies listed in the previous section.

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_string_to_axis_data.v': 'file_type': 'verilogSource'
- tb
 - 'tb/tb_axis.v': 'file_type': 'verilogSource'
 - 'tb/in.txt': 'file_type': 'user', 'copyto': ''

- 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.
- sim

Info: Test text input to core, and view its data output in binary.
- 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 data width converter. 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:string:
  ↳ axis_data_to_axis_string:1.0.0 --SBUS_WIDTH=4 --
  ↳ USER_WIDTH=8
```

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; fusesoc run --target
  ↳ sim_cocotb AFRL:string:axis_data_to_axis_string
  ↳ :1.0.0 --SBUS_WIDTH=$i ; echo "SLAVE_WIDTH:" $i;
  ↳ done
```

5 Code Documentation

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

- **axis_string_to_axis_data** AXIS string to AXIS data, convert input string to data.
- **tb_axis** Verilog test bench.
- **tb_cocotb verilog** Verilog test bench base for cocotb.
- **tb_cocotb python** cocotb unit test functions.

axis_string_to_axis_data.v

AUTHORS

JAY CONVERTINO

DATES

2022/09/19

INFORMATION

Brief

Take input string data and process it into tuser/tdata output.

License MIT

Copyright 2022 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_string_to_axis_data

```
module axis_string_to_axis_data #(
  parameter
  DELIMITER
  =
  " , "
  parameter
  TERMINATION
  =
  "\n"
  parameter
  STRING_LEN
```



```

=
4,
parameter
MBUS_WIDTH
=
1,
parameter
USER_WIDTH
=
4,
parameter
DEST_WIDTH
=
4,
parameter
PREFIX_LEN
=
1,
parameter
DATA_PREFIX
=
"#",
parameter
DEST_PREFIX
=
"&",
parameter
USER_PREFIX
=
" * ",
parameter
KEYWORD_LEN
=
3,
parameter
SET_KEYWORD
=
"set",
parameter
CLR_KEYWORD
=
"clr"
) ( input aclk, input arstn, output [(MBUS_WIDTH*8)-1:0] m_axis_tdata, output

```

Convert string data to raw binary data for axis bus.

Parameters

DELIMITER parameter	break value between multiple strings
TERMINATION parameter	termination value of full string from serial port, byte only. (\n = 0A \r = 0D).
STRING_LEN parameter	max lenh of string including delimiter
MBUS_WIDTH parameter	bus width of master (data) output
USER_WIDTH parameter	user width of master bus, only in 4 bit nibbles, and at least 4 bits.
DEST_WIDTH parameter	dest width of master bus, only in 4 bit nibbles, and at least 4 bits.
PREFIX_LEN	length of following prefix strings in bytes.

parameter

DATA_PREFIX prefix for data hex strings

parameter

DEST_PREFIX prefix for destination hex strings

parameter

USER_PREFIX prefix for user hex strings

parameter

KEYWORD_LEN length of the following keywords

parameter

SET_KEYWORD keyword to output data over tdata,tuser,tdest on master interface.

parameter

CLR_KEYWORD keyword to clear output data and buffers of master interface.

parameter

Ports

aclk Clock for AXIS

arstn Negative reset for AXIS

m_axis_tdata Output data

m_axis_tvalid When active high the output data is valid

m_axis_tuser Output user data

m_axis_tdest Output destination data

m_axis_tready When set active high the output device is ready for data.

s_axis_tdata Input string data

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.

tb_axis.v

AUTHORS

JAY CONVERTINO

DATES

2022/10/24

INFORMATION

Brief

Test bench for axis_string_to_axis_data using axis stim and clock stim.

License MIT

Copyright 2022 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
```

Test bench for axis_string_to_axis_data. 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.

INSTANTIATED MODULES

clk_stim

```
clk_stimulus #(
    CLOCKS(1),
    CLOCK_BASE(1000000),
    CLOCK_INC(1000),
    RESETS(1),
    RESET_BASE(2000),
    RESET_INC(100)
) clk_stim ( .clkv(tb_dut_clk), .rstnv(tb_dut_rstn), .rstv() )
```

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.txt")
) slave_axis_stim ( .m_axis_aclk(tb_dut_clk), .m_axis_arstn(tb_dut_rstn), .r
```

Device under test SLAVE stimulus module.

dut

```
axis_string_to_axis_data #(
    DELIMITER(";",
    TERMINATION("\n"),
    STRING_LEN(4),
    MBUS_WIDTH(BUS_WIDTH),
    USER_WIDTH(USER_WIDTH),
    DEST_WIDTH(DEST_WIDTH),
    PREFIX_LEN(1),
    DATA_PREFIX("#"),
    DEST_PREFIX("&"),
    USER_PREFIX("*"),
    KEYWORD_LEN(3),
```

```

SET_KEYWORD("set"),
CLR_KEYWORD("clr")
) dut ( .aclk(tb_dut_clk), .arstn(tb_dut_rstn), .m_axis_tdata(tb_dut_data),

```

Device under test, axis_string_to_axis_data

master_axis_stim

```

master_axis_stimulus #(
BUS_WIDTH(BUS_WIDTH),
USER_WIDTH(USER_WIDTH),
DEST_WIDTH(DEST_WIDTH),
FILE("out.bin")
) 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

2024/12/12

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
  DELIMITER
  =
  " , "
  parameter
  TERMINATION
  =
  "\n"
  parameter
  STRING_LEN
```

```

    =
    4,
    parameter
    MBUS_WIDTH
    =
    1,
    parameter
    USER_WIDTH
    =
    4,
    parameter
    DEST_WIDTH
    =
    4,
    parameter
    PREFIX_LEN
    =
    1,
    parameter
    DATA_PREFIX
    =
    "#",
    parameter
    DEST_PREFIX
    =
    "&",
    parameter
    USER_PREFIX
    =
    " * ",
    parameter
    KEYWORD_LEN
    =
    3,
    parameter
    SET_KEYWORD
    =
    "set",
    parameter
    CLR_KEYWORD
    =
    "clr"
) ( input aclk, input arstn, output [(MBUS_WIDTH*8)-1:0] m_axis_tdata, output

```

Test bench for string to data converter. 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

DELIMITER parameter	break value between multiple strings
TERMINATION parameter	termination value of full string from serial port, byte only. (\n = 0A \r = 0D).
STRING_LEN parameter	max length of string including delimiter
MBUS_WIDTH parameter	bus width of master (data) output
USER_WIDTH parameter	user width of master bus, only in 4 bit nibbles, and at least 4 bits.
DEST_WIDTH	dest width of master bus, only in 4 bit nibbles, and at least 4 bits.

parameter

PREFIX_LEN length of following prefix strings in bytes.

parameter

DATA_PREFIX prefix for data hex strings

parameter

DEST_PREFIX prefix for destination hex strings

parameter

USER_PREFIX prefix for user hex strings

parameter

KEYWORD_LEN length of the following keywords

parameter

SET_KEYWORD keyword to output data over tdata,tuser,tdest on master interface.

parameter

CLR_KEYWORD keyword to clear output data and buffers of master interface.

parameter

Ports

aclk Clock for AXIS

arstn Negative reset for AXIS

m_axis_tdata Output data

m_axis_tvalid When active high the output data is valid

m_axis_tuser Output user data

m_axis_tdest Output destination data

m_axis_tready When set active high the output device is ready for data.

s_axis_tdata Input string data

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.

INSTANTIATED MODULES

dut

```
axis_string_to_axis_data #(
    DELIMITER(DELIMITER),
    TERMINATION(TERMINATION),
    STRING_LEN(STRING_LEN),
    MBUS_WIDTH(MBUS_WIDTH),
    USER_WIDTH(USER_WIDTH),
    DEST_WIDTH(DEST_WIDTH),
    PREFIX_LEN(PREFIX_LEN),
    DATA_PREFIX(DATA_PREFIX),
    DEST_PREFIX(DEST_PREFIX),
```


USER_PREFIX(USER_PREFIX),	.
KEYWORD_LEN(KEYWORD_LEN),	.
SET_KEYWORD(SET_KEYWORD),	.
CLR_KEYWORD(CLR_KEYWORD)	.
) dut (.aclk(tb_dut_clk), .arstn(tb_dut_rstn), .m_axis_tdata(tb_dut_data),	

Device under test, axis_string_to_axis_data

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

create_string

```
def create_string(
    dut,
    tx_frame
)
```

Return a string equal to the core output

Parameters

dut device under test passed from cocotb test function
tx_frame transmitted frame data

Returns: String

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.

Parameters

dut Device under test passed from cocotb.

conversion_test

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

Coroutine that is identified as a test routine. This routine tests for correct output of strings by comparing to a python generated version.

Parameters

dut Device under test passed from cocotb.

conversion_test_random_ready

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

Coroutine that is identified as a test routine. This routine tests for correct output of strings by comparing to a python generated version, with ready randomized.

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.