AXIS_STRING_TO_AXIS_DATA

AFRL

AIR FORCE RESEARCH LABORATORY

November 21, 2024

Jay Convertino

Contents

1 Usage

1.1 Introduction

This core takes a incoming string, removes the delimiters, terminators, and prefixs. 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 keywork, if set output data. If clear, remove all data.

Please see 5 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': '.'

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.

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
   - **cocotb** testbench files

4 Simulation

There are a few different simulations that can be run for this core.

4.1 iverilog

iverilog is used for simple test benches for quick verification, visually, of the core.

4.2 cocotb

Future simulations will use cocotb. This feature is not yet implemented.

5 Module Documentation

There is a single async module for this core.

- **axis_string_to_axis_data** AXIS string to AXIS data, convert input string to data.

The next sections document the module in great detail.

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

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, outpu
```

Parse raw binary data into ASCII string output.

Parameters

DELIMITER
parameter
break value between multple strings

TERMINATION
parameter
termination value of full string from serial port, byte only. (\n = 0A \r = 0D).

STRING_LEN
parameter
max lenth 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.