FT245_SYNC_TO_AXIS

November 14, 2024

Jay Convertino

Contents

1 Usage

1.1 Introduction

FT245 sync to AXIS core converts FT245 to the AXIS interface. This is done using asynchronous conversions with a few registers to sync signals in time.

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_tb
  - AFRL:simulation:axis_stimulator
  - AFRL:utility:sim_helper

1.3 In a Project

Connect the device to your AXIS bus. Connect the FT245 bus to the FTDI device.

2 Architecture

This core is made up of a single module.

- **ft245_sync_to_axis** Interface AXIS to F245 device (see core for documentation).

This core has 1 always blocks that are sensitive to the positive clock edge.

- **register signals** registers the ft245 data read fx signal and based upon its state the the state of tread outputs AXIS data.

Please see 5 for information on how the asynchronous assignments are done.

3 Building

The FT245 sync to AXIS is written in Verilog 2001. It 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

  Type: verilogSource

  - src/ft245_sync_to_axis.v

- tb

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

3.3 Targets

3.3.1 fusesoc_info Targets

- default

  Info: Default for IP intergration.

  - src

- sim

  Info: Default simulation using icarus.

  - src
  - tb
  - dep_tb

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

- **ft245_sync_to_axis** Interfaces AXIS to the FT245.

The next sections document the module in great detail.

ft245_sync_to_axis.v

AUTHORS

JAY CONVERTINO

DATES

2022/08/09

INFORMATION

Brief

Converter FT245 sync FIFO interface to AXIS.

License MIT

ft245_sync_to_axis

```
module ft245_sync_to_axis #(
parameter
BUS_WIDTH
 =
1
) ( input rstn, input ft245_dclk, inout [BUS_WIDTH-1:0] ft245_ben, inout [(
```

Converter FT245 sync FIFO interface to AXIS.

Parameters

BUS_WIDTH Width of the FT245 and AXIS bus.
parameter

Ports

rstn	Negative reset
ft245_dclk	Input clock from FIFO.
ft245_ben	Byte enable used in FT60x, similar to AXIS tkeep in 1 is a valid byte for each bit.
ft245_data	FIFO data bus
ft245_rdn	Enable read on active low
ft245_wrn	Enable write on active low
ft245_siwun	Send Immediate / Wakeup for USB suspend. Active low.
ft245_txen	When low, write data to the fifo.
ft245_rxfn	When low, read data from the fifo.
ft245_oen	Output enable active low
ft245_rstn	Negative Reset
ft245_wakeupn	Sleep ft245 active low
s_axis_tdata	Input axis data
s_axis_tkeep	Input axis data bytes that are valid. Each bit equals one byte.
s_axis_tvalid	Input axis data is valid when active high.
s_axis_tready	Input data bus is ready when signal is active high.
m_axis_tdata	Output axis data
m_axis_tkeep	Output what axis data bytes are valid. Each bit equals one byte.
m_axis_tvalid	Output is active high when axis data is valid.
m_axis_tready	Output data bus is told that the receive device is ready. The device is ready if it asserts this signal active high.

DATA STORE REGISTERS

Register data based upon ft245 clocks.

r_oen

```
reg r_oen
```

output enable registers

rr_oen

```
reg rr_oen
```

output enable registers registers

rrr_oen

```
reg rrr_oen
```

output enable registers registers registers

r_m_axis_tdata

```
reg [(
BUS_WIDTH*8
)-1:0] r_m_axis_tdata
```

master axis register to hold tdata for tready not ready at end condition

r_m_axis_tkeep

```
reg [BUS_WIDTH-1:0] r_m_axis_tkeep
```

master axis register to hold tkeep for tready not ready at end condition

r_m_axis_tvalid

```
reg r_m_axis_tvalid
```

master axis register to hold tvalid for tready not ready at end condition

ASSIGNMENTS

How various comibinations of logic are created and data dealt with.

ft245_data

```
assign ft245_data = (
                                                    rr_oen & r_oen ?

s_axis_tdata
: '
bz
)
```

combinartoral signals to convert registers and axis to and from ft245. tristate ft245 based on output enable state

ft245_ben

```
assign ft245_ben = (
                                                    rr_oen & r_oen ?

s_axis_tkeep
: '
bz
)
```

tristate ft245 based on output enable state

ft245_wrn

```
assign ft245_wrn = ft245_txen | ~ft245_rxfn | ~s_axis_tvalid | ~rr_oen
```

only allow write if there is space, nothing availbe to read, valid data available, and output enable is timed correctly.

ft245_oen

```
assign ft245_oen = rr_oen
```

output enable

ft245_rdn

```
assign ft245_rdn = ~m_axis_tready | rrr_oen | rr_oen & r_oen
```

only ready when output enable is correctly timed and we are ready for data ft245 will output data as soon as oen is applied (FWFT).

ft245_wakeupn

```
assign ft245_wakeupn = 1'b0
```

always keep it awake

ft245_siwun

```
assign ft245_siwun = 1'b0
```

always keep it awake

ft245_rstn

```
assign ft245_rstn = rstn
```

apply system reset to ft245

s_axis_tready

```
assign s_axis_tready = (

ft245_txen &
```

```
ft245_rxfn
) & rr_oen
```

convert ft245 to ready. only ready when write buffer is available, nothing is incoming, and output enable is set correctly.

m_axis_tdata

```
assign m_axis_tdata = (
                                                    rr_oen | r_oen ?

r_m_axis_tdata
:
ft245_data
)
```

output ft245 to master axis. at end, output registers incase next core was not ready.

m_axis_tkeep

```
assign m_axis_tkeep = (
                                                    rr_oen | r_oen ?

r_m_axis_tkeep
:
ft245_ben
)
```

output ft245 to master axis. at end, output registers incase next core was not ready.

m_axis_tvalid

```
assign m_axis_tvalid = (
                                                    rr_oen | r_oen ?

r_m_axis_tvalid
: ~
                                                                    (

rrr_oen | ft245_rxfn)
)
```

data is only valid in the correct output enable register state and is no longer valid if rxfn indicates the receive exhausted.