

PISO



April 15, 2025

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	3
2.1 Waveform	3
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 Module Documentation	6
5.1 piso	7
5.2 tb_piso-v	9
5.3 tb_cocotb-py	11
5.4 tb_cocotb-v	14

1 Usage

1.1 Introduction

This core converts the parallel data into serial data. This is done on a positive clock edge in relation to the enable. The enable sets the rate and should only be held high for the same period as the main clock.

1.2 Dependencies

The following are the dependencies of the cores.

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

1.2.1 fusesoc_info Dependencies

- dep
 - AFRL:utility:helper:1.0.0

1.3 In a Project

This core is made to interface for parallel to serial data. Data is loaded into the core, and then enable is pulsed to push data out in a serial fashion. There is a count output that tells how many bits have been output. Data is always presented at the output so when count is at 31, the serial data output has bit 31 presented at the output. The series of steps to use it are as follows.

1. Set pdata to input data, and set load to 1 (data will be loaded on positive clock edge to internal registers).
2. Set load to 0.
3. Pulse enable to 1 that is synced to the main clock. Only hold high for one period of the master clock.
4. Repeat enable pulse until dcount is equal to 0.

2 Architecture

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

- **piso** Convert parallel data to serial data. (see core for documentation).

Please see 5 for more information.

2.1 Waveform



3 Building

The PISO 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.

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/piso.v
- tb
 - tb/tb_piso.v
- 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: Base simulation using icarus as default.
- 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
 - **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

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

```
$ pip install cocotb
```

- **sim_cocotb** Standard simulation PISO conversion using cocotb.

Then you must use the cocotb sim target. The targets above can be run with various parameters. This test will check the input/output against each other to validate core operation.

```
$ fusesoc run --target sim_cocotb AFRL:simple:piso  
  ↳ :1.0.0
```

5 Module Documentation

- **piso** PISO converter
- **tb_piso-v** Verilog test bench
- **tb_cocotb-py** Cocotb python test routines
- **tb_cocotb-v** Cocotb verilog test bench

The next sections document the module in detail.

pisov

AUTHORS

JAY CONVERTINO

DATES

2025/15/04

INFORMATION

Brief

PISO (parallel in serial out) The idea is to keep this core simple, and let the control logic be handled outside of this core.

License MIT

Copyright 2025 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.

piso

```
module piso #(
    parameter
    BUS_WIDTH
    =
    1
) ( input clk, input rstn, input ena, input load, input [BUS_WIDTH*8-1:0] {
```

parallel in serial out

Parametes

BUS_WIDTH width of the parallel data input in bytes.

Ports

clk	global clock for the core.
rstn	negative synchronus reset to clk.
ena	enable for core, use to change output rate. Enable serial shift output.
load	load parallel data into core. This can be done at any time.
pdata	parallel data input, registered at load only.
sdata	serialized data output.
dcount	Number of bits to shift out (what is left, first bit is always available).

tb_piso.v

AUTHORS

JAY CONVERTINO

DATES

2025/04/15

INFORMATION

Brief

Test bench for parallel to serial core.

License MIT

Copyright 2025 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_piso

```
module tb_piso ()
```

Test bench for piso. Pump in data to parallel port on each count == 0.

INSTANTIATED MODULES

piso

```
piso #(
```

```
BUS_WIDTH(1)
) dut ( .clk(tb_clk), .rstn(tb_rstn), .ena(tb_ena), .load(tb_load), .pdata(
```

Device under test, parallel to serial

tb_cocotb.py

AUTHORS

JAY CONVERTINO

DATES

2025/03/04

INFORMATION

Brief

Cocotb test bench

License MIT

Copyright 2025 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.

sipo

sipo

Convert serial data to parallel

VARIABLES

idle_read

self._idle_read

Event trigger for cocotb read

FUNCTIONS

`_restart`

```
def _restart(  
    self  
)
```

kill and restart `_run` thread.

`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.

`increment test`

Coroutine that is identified as a test routine. Write data, on one clock edge, read on the next.

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.

tb_cocotb.v

AUTHORS

JAY CONVERTINO

DATES

2025/04/15

INFORMATION

Brief

Test bench wrapper for cocotb

License MIT

Copyright 2025 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. BUS_WIDTH

tb_cocotb

```
module tb_cocotb #(
  parameter
  BUS_WIDTH
  =
  4
) ( input clk, input rstn, input ena, input load, input [BUS_WIDTH*8-1:0] t
```

PISO Wrapper

Parameters

BUS_WIDTH Width of the parallel input in bytes
parameter

Ports

clk	Clock
rstn	negative reset
ena	enable for core, use to change output rate. Enable serial shift output.
load	load parallel data into core. This can be done at any time.
pdata	parallel data input, registered at load only.
sdata	serialized data output.
dcount	Number of bits to shift out (what is left, first bit is always available).

INSTANTIATED MODULES

dut

```
piso #(
    BUS_WIDTH(BUS_WIDTH)
) dut ( .clk(clk), .rstn(rstn), .ena(ena), .load(load), .pdata(pdata), .sdata(sdata), .dcount(dcount) )
```

Device under test, piso