Low Power Multi-Clock Digital System using SPI

Elaborated By:

Eng. Youssef Nasser

Acknowledgments

I would like to express my sincere gratitude to **Engineer Osama Gamal** for giving me the opportunity to join the Si-Vision Academy in partnership with Synopsys, specifically in the field of Digital IC Design. His belief in my potential and unwavering support throughout the program have been truly invaluable. I also extend my heartfelt appreciation to **Engineer Rodina Samy** for her exceptional organizational skills in coordinating the academy, ensuring a smooth and enriching learning experience for all participants. I am deeply thankful to the **team of engineers at Si-Vision** for their expertise and willingness to share their knowledge, which has greatly contributed to my understanding and growth in Digital IC Design. Lastly, I would like to extend my appreciation to **Si-Vision and Synopsys** for their collaboration in organizing this academy, providing a remarkable learning platform. I am truly grateful for the support and guidance I have received from these individuals and organizations, which have played a pivotal role in my academic and professional development.

Abstract

This thesis focuses on the RTL implementation of a digital transmission system using the Verilog language. The system consists of several blocks, including an SPI slave that receives data from the master (Test-bench) and stores it in the register file during write operations, a register file that stores data and sends it along with a start signal to the CRC and Serializer blocks, a CRC block that generates check bits for the message, and a Serializer block that receives data and checks bytes, serializing them to the PHY using a divided clock. The design also includes a frequency divider for the system clock, clock gating for the SPI and system clocks, and a synchronizer for clock domain crossing. The thesis explores the integration of these blocks within a top module, aiming to achieve a functional and efficient digital transmission system

Table Of Contents

List Of Figures

List Of Tables

List Of Reports

Acronyms or Abbreviations

ATPG	Automatic Test Pattern Generation
ASYNC	Asynchronous
AWL	Atrenta Design Constraints
CDC	Clock Domain Crossing
CS	Chip Select
CRC	Cyclic Redundancy Check
DC	Design Compiler
ddc	Description And Design Constraints
DFT	Design For Testing
DRC	Design Rule Check
FCS	Frame Check Sequence
HVT	High Voltage Threshold
IP	Intellectual Property
lib	Library
LSB	Least Significant Bit
MISO	Master In Slave Out
MOSI	Master Out Slave In
MSB	Most Significant Bit
MUX	Multiplexer
PISO	Parallel In Serial Out
RDC	Reset Domain Crossing
RTL	Register Transfer Level
RX	Receiver
RST	Reset
SCLK	Serial Clock
SDC	Synopsys Design Constraints
SDGC	Spyglass Design Constraints
SDF	Standard Delay Format
SER	Serializer
Si	Silicon
SPI	Serial Peripheral Interface
Specs	Specification
Ss	Slow Slow
SPF	Stil Protocol File
STIL	Standard Test Interface Language
SYS	System
SVF	Standard Verification Format
SYNC	Synchronous
Tb	Test Bench
Tcl	Tool Command Language
tmax	Tetra Max
TX	Transmitter
UPF	Unified Power Format
VCS	Verilog Compiler and Simulator

Chapter 1 Introduction

History

The Serial Peripheral Interface (SPI) is a synchronous serial communication protocol that was developed by Motorola in the 1980s. It was initially introduced as a way to connect multiple peripheral devices to microcontrollers and microprocessors. SPI has since become a widely used communication standard in the electronics industry.

The original purpose of SPI was to enable efficient data transfer between a master device (usually a microcontroller or microprocessor) and one or more slave devices (such as sensors, displays, memory chips, or other peripheral devices). The protocol allows for full-duplex communication, meaning that data can be transmitted and received simultaneously.

SPI operates using a master-slave architecture. The master device controls communication by generating a clock signal and selecting the specific slave device with which it wants to communicate. Each slave device has a dedicated chip select (CS) line that is used to enable or disable communication with that particular device.

The SPI protocol uses four main signals:

1. SCLK (Serial Clock): The master device generates clock pulses to synchronize data transfer on this line.
2. MOSI (Master Output, Slave Input): The master device sends data to the slave device(s) on this line.
3. MISO (Master Input, Slave Output): The slave device(s) send data back to the master device on this line.
4. SS/CS (Slave Select/Chip Select): Each slave device has a dedicated chip select line that is used to select or deselect a particular device for communication.

SPI is known for its simplicity, flexibility, and high-speed data transfer capability. It is widely supported by various microcontrollers, microprocessors, and peripheral devices. Over time, SPI has evolved with additional features such as different clock modes, configurable data formats, and multiple slave device support.

Problem Statement

The project has a main objective and multiple aims need to be reached with success and with certain quality assurance. The objective of our project is to design and implement an RTL-based digital transmission system using Verilog, integrating multiple blocks such as an SPI slave, a register file, a CRC block, a Serializer block, a frequency divider, clock gating, and a synchronizer. The challenge is to ensure the proper functionality and efficient operation of each block within the top module, achieving a functional and optimized digital transmission system.

Specs

- the system is working with a 32MHz system clock , it includes CRC and Serializer
- SPI clock is 26 MHz , the spi domain include Register file and SPI slave
- The system is used to transmit packets with a total length of 72 bits for each.
- The Register file width is 8-bits (depth is determined according to the design requirements)
- CRC-16 following this Equation $G(X) = X^{16} + X^{12} + X^5 + 1$
- The Serializer is responsible for
    - Append the synchronization header (SHR) to the 32-bit PHY payload from the Register file, then the CRC 2 bytes calculated on the 32-bit payload bits.
    - Perform parallel to serial operation on the packet to send serial data to PHY according to the required PHY data rate.
    - SHR consists of 3 octets, each octet is 2 symbols of 4-bits for each; 2 octets of repeated 0101 for the preamble, followed by 7A access word.
- The PHY is working with 1Mpbs/2Mpbs rate, and the tx_mode bit from the Register file is controlling this.

- Register file contents are shown in the below table

Table 1 Register File Signals

Signal Name	Size	Type	Reset Value	Description
rf_tx_start	1	RW	0	Start TX signal
rf_tx_mode	1	RW	0	TX rate select : 1'b0 : 1MHz 1'b1 : 2MHz
rf_power_domain	1	RW	0	Power State: 1'b0 : Sleep Power State 1'b1 : Active Power state
rf_tx_data	32	RW	0	PHY payload bits
rf_tx_done	1	R	N/A	Done Signal after finishing transmission

- There are two clock gating units :
  - The 1st one gates the system clock when rf_tx_start is zero.
  - The 2nd one gates the SPI clock to regfile in case there is no write enable.
- There are 2 power domains in the design :
  - Sleep domain: Contains the SPI slave and Register file.
  - Active domain: Contains the Serializer and CRC.

Project Hierarchy

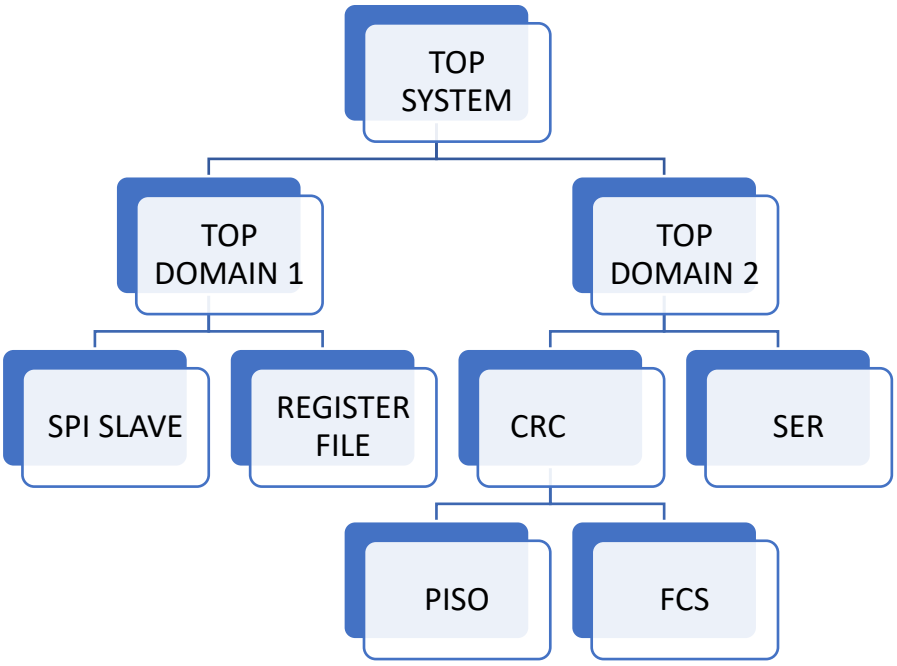


Figure 1 Project Hierarchy

Chapter 2 Design

Modules Functionality

1. SPI Master: In our design, the SPI master is the test bench who is controlling the whole design, for write operation 0x02 and for read operation 0x03 and both of them follow the addresses of the register file.

Table 2 PHY Command-List

Command	Operation	Argument A	Argument B
0X02	Register File Write	MOSI: Register Address MISO: Not used	MOSI: Register Data MISO: Not used
0X03	Register File Read	MOSI: Register Address MISO: Not used	MOSI: Not used MISO: Register Data

2. SPI Slave: Receive data sent from the master and stores it in the register file during a write operation. Send data from the register file to the master during a read operation.
3. Register File: Store data sent from SPI slave during a write operation, send data, and start signal to CRC and Serializer.

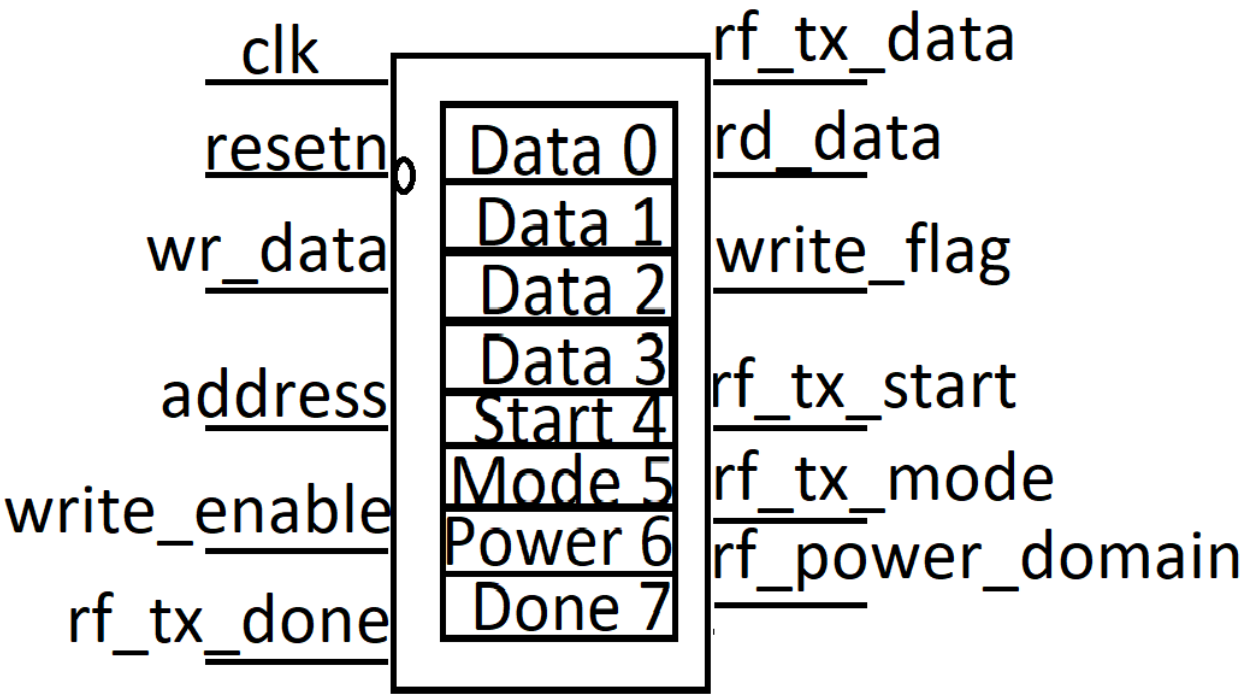


Figure 2 Register File Block

4. Frame Check Sequence (FCS): Performs logical operations to generate check bits that are sent with the message.

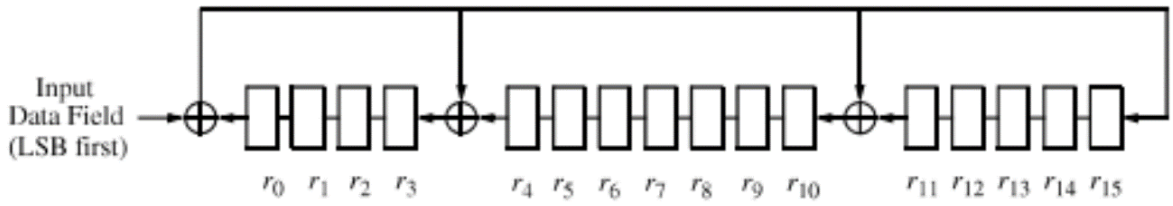


Figure 3 CRC-16

5. Parallel in Serial Out (PISO): The input of the FCS is just one bit, so the data is taken from the register file through the PISO then the output of the PISO is the input of the FCS.
6. Serializer: Receive data bytes from the register file and check bytes from CRC then serialize them to PHY with a divided clock of 1Mbps or 2 Mbps based on a selected mode
7. Reset Synchronizer: As we have two clock domains so we need two reset synchronizers to make a global reset that works with the Register File of Domain 1 and with all modules in Domain 2.
8. Bit Synchronizer: the number of Bit synchronizers is based on the number of signals that cross the domains, such as start, mode, and done
9. Mux 2X1: Used in DFT to choose between the scan signal and the real signal
10. Mux -negative 2X1: Used in DFT in the signals that needed to be inverted.

Note Beside: Refer to Appendix A to see all interfaces and signals of each module.

Chapter 3 RTL

SPI Slave

It's Synopsys IP and all associated documentation is proprietary to Synopsys Inc , so I can't write the RTL of this module.

Register File

```
`timescale 1ns/1ps
module register_file
#(parameter WIDTH = 8 ,DEPTH = 8 ,ADDRESS_WIDTH = 8)(
    input clk, // clock signal
    input   wire        resetn,
    input [WIDTH-1:0]   wr_data,            // data to be written
    input [ADDRESS_WIDTH-1:0] address,
    input   write_enable,        // write enable signal
    input    rf_tx_done,
    output reg write_flag,
    output reg  [WIDTH-1:0] rd_data ,       // data to be read
    output reg  [31:0] rf_tx_data     ,
    output reg  rf_tx_mode  ,       //TX rate select: 1'b0: 1 MHz 1'b1: 2 MHz
    output reg  rf_power_domain,  //Power State: 1'b0: Sleep Power State 1'b1: Active Power state
    output reg  rf_tx_start    );             //Start TX signal
reg [WIDTH-1:0] reg_file [0:DEPTH-1]; // register file with 5 entries
reg [2:0] counter;
integer i;
always @(posedge clk or negedge resetn) begin
if (!resetn) begin
counter <= 6'b0;
write_flag <=1'b0;
for(i=0;i<7;i=i+1) // updated today , input can not be 0 by reset
begin
reg_file[i]<=8'b0;
end
end
  else if (write_enable) // write operation
  begin
    reg_file[address] <= wr_data;
    reg_file[7] <= rf_tx_done;
    counter <= counter + 1;
    if(counter ==7)
      begin
     write_flag <= 1'b1;
     end
    else
     write_flag <= 1'b0;
  end end
always @(*)
begin
     rf_tx_data = {reg_file[0], reg_file[1], reg_file[2], reg_file[3]};
     rd_data = reg_file[address];
     rf_tx_start    = reg_file[4];
     rf_tx_mode = reg_file[5]   ;
     rf_power_domain = reg_file[6]  ;
end
```

```
endmodule
```

TOP Domain 1

```
`timescale 1ns/1ps
//TOP_Domain_1 without Sync
module TOP_Domain_1
#(parameter WIDTH = 8)
(
input csn,
input resetn,
input spi_clock,
input MOSI,
input rf_tx_done,
output rf_tx_start  ,
output  rf_tx_mode  ,
output  rf_power_domain,
output o_miso_ena,
output    [31:0]        rf_tx_data ,
output write_flag,
output MISO
);
wire WrEn;
wire [7:0] Address;
wire [7:0] slave_to_reg;
wire [7:0] reg_to_slave;

spi_slave block1(
.i_csn(csn),
.i_sck(spi_clock),
.i_sck_neg(spi_clock),
.i_mosi(MOSI),
.o_miso(MISO),
.i_rf_din(reg_to_slave), //wire
.o_rf_wre(WrEn),//wire
.o_rf_addr(Address),//WIRE
.o_dout(slave_to_reg),//wire
.o_miso_ena(o_miso_ena),
.i_scan_mode(1'b0),
.i_ioring_rst_n(1'b0),
.i_status(16'd0)
);
register_file block2(
.clk(spi_clock),
.resetn(resetn),
.write_enable(WrEn),
.address(Address),
.wr_data(slave_to_reg),
.rd_data(reg_to_slave),
.rf_tx_start(rf_tx_start),
.rf_tx_mode(rf_tx_mode),
.rf_power_domain(rf_power_domain),
.rf_tx_data(rf_tx_data),
.rf_tx_done(rf_tx_done),
.write_flag(write_flag)

);
```

```
endmodule
```

FCS

```
`timescale 1ns/1ps

module FCS(data_i,lfsr,clk,rstn);

output reg [0:15] lfsr;
input clk,rstn;
input data_i;

always @(posedge clk or negedge rstn)
begin
if (!rstn)
begin
lfsr <= 16'h 0000;  //initial value of crc =0
end
else
begin
    lfsr[0]  <= lfsr[1];
    lfsr[1]  <= lfsr[2];
    lfsr[2]  <= lfsr[3];
    lfsr[3]  <= lfsr[4] ^ lfsr[0] ^ data_i;
    lfsr[4]  <= lfsr[5];
    lfsr[5]  <= lfsr[6];
    lfsr[6]  <= lfsr[7];
    lfsr[7]  <= lfsr[8];
    lfsr[8]  <= lfsr[9];
    lfsr[9]  <= lfsr[10];
    lfsr[10] <= lfsr[11] ^ lfsr[0] ^ data_i;
    lfsr[11] <= lfsr[12];
    lfsr[12] <= lfsr[13];
    lfsr[13] <= lfsr[14];
    lfsr[14] <= lfsr[15];
    lfsr[15] <= lfsr[0] ^ data_i;
end


end

endmodule
```

PISO

```
`timescale 1ns/1ps
module PISO(
input clk,
input reset,
input [31:0] data_in,
input piso_load,
input enable,
output reg piso_serial_done_tick,
output reg piso_serial_out );
reg [5:0] counter; // PISO Block //log base 2 (DW)
reg [31:0] data_reg;
reg piso_serialize;
always@(posedge clk or negedge reset)
begin
    if(!reset)
    begin
    piso_serial_out<=0;
    data_reg<=0;
    counter <= 0;
    piso_serialize <= 0;
    end
    else begin
    if (counter == 32'd33)
    begin
    piso_serialize <= 0;
    end
    else if(piso_load && enable )
    begin
    data_reg <= data_in ;
    counter <= 0;
    piso_serialize <= 1'b1;
    end
    else if (piso_serialize)
    begin
    data_reg <= data_reg<<1;
    piso_serial_out <= data_reg[31];
    counter <= counter + 1 ;
    end end end
always@(posedge clk or negedge reset)
begin
    if(!reset)
    begin
    piso_serial_done_tick<=1'b0;
    end

    else if (counter == 32)
        begin
    piso_serial_done_tick <= 1;
    end
    else
    begin
    piso_serial_done_tick <= 1'b0;
    end
end
endmodule
```

TOP CRC

```
`timescale 1ns/1ps

module CRC
#(parameter DW = 32 )
(

input [31:0] crc_data_in,
input crc_load,

input clk,
input rstn,

input enable,

output crc_serial_done_tick,
output [0:15] LFSR

);


wire PISO_to_CRC;


PISO U0(

.clk(clk),
.reset(rstn),
.data_in(crc_data_in),
.piso_load(crc_load),

.enable(enable),
.piso_serial_done_tick(crc_serial_done_tick),
.piso_serial_out(PISO_to_CRC)

);


FCS U1(

.lfsr(LFSR),
.clk(clk),
.rstn(rstn),
.data_i(PISO_to_CRC)

);



endmodule
```

SERIALIZER

```
`timescale 1ns/1ps
module SER
(


input [31:0] data_payload ,
input sys_clk,reset,load,
input [0:15] data_crc,
input mode,
output reg serial_done_tick,
output reg serial_out

);

reg [4:0] counter;
reg [6:0] counter_tick;
wire [71:0] data_in;
reg [71:0] data_reg;

reg Div_clk_d1;

assign data_in = {24'b0101_0101_0101_0101_0111_1010,data_payload,data_crc};

always@(posedge sys_clk or negedge reset)
begin
                    if(reset == 0)
                    begin
                    Div_clk_d1 <= 0;
                    counter <= 0;
                    end
                    else if (mode == 0)
                    begin
                    if (counter == 31 )
                        begin
                    Div_clk_d1 <= 1'b1;
                    counter <= 0;
                        end
                            else begin
                            Div_clk_d1 <= 0 ;
                            counter <= counter +1;
                          end
                    end
                    else if (mode == 1)
                    begin
                    if (counter == 15 )
                        begin
                    Div_clk_d1 <= 1'b1;
                    counter <= 0;
                        end
                    else begin
                            Div_clk_d1 <= 0 ;
                            counter <= counter +1;
                    end
                    end
    end
```

```
always@(posedge sys_clk or negedge reset)
begin

 if(reset == 0) begin
          serial_out <=0;
        data_reg <=0;
        counter_tick <= 0;
        serial_done_tick <= 0;
 end

else if (load)
    data_reg <= data_in;
    else if (Div_clk_d1)
    begin

    data_reg <= data_reg<<1;
    serial_out <= data_reg[71];
    counter_tick <= counter_tick +1;

    end

else if (counter_tick == 75)
begin
serial_done_tick <= 1;
end
else
  begin
    serial_done_tick <= 0;
  end


 end

endmodule
```

TOP Domain 2

```
`timescale 1ns/1ps
//CRC with SER TOP
module TOP(

input crc_load_ext,

input clk,
input reset,
input [31:0] payload,

input mode,
input enable,

output packet,
output serial_done_tick

);

wire crc_load;
wire [0:15] crc_lfsr;


CRC CRC_Insta(

.enable(enable),
.crc_data_in(payload),
.crc_load(crc_load_ext),


.clk(clk),
.rstn(reset),

.crc_serial_done_tick(crc_load),
.LFSR(crc_lfsr)

);

SER SER_Insta(

.data_payload(payload),
.data_crc(crc_lfsr),
.sys_clk(clk),

.reset(reset),
.mode(mode),
.load(crc_load),



.serial_done_tick(serial_done_tick),
.serial_out(packet)

);

endmodule
```

Synchronizers

Reset Synchronizer

```
`timescale 1ns/1ps
module Reset_Sync #(parameter NUM_STAGES = 2)
(
    input   wire                    RST ,
    input   wire                    CLK ,
    output  reg                     Sync_RST
);

    reg          [NUM_STAGES-1:0]   register;

    always @(posedge CLK or negedge RST)
        begin

            if(!RST)
                begin

                    register <= 'b0                                         ;
                    Sync_RST <= 'b0                                         ;

                end
            else

                {Sync_RST,register} <= {register[NUM_STAGES-1:0],1'b1}   ;

        end

endmodule
```

Bit Synchronizer

```
`timescale 1ns/1ps

module BIT_SYNC #(parameter NUM_STAGES = 2, BUS_WIDTH = 1)
(
    input   wire [BUS_WIDTH-1:0] ASYNCH,
    input   wire    RST,
    input   wire    CLK,

    output  reg  [BUS_WIDTH-1:0]    SYNC
);
    reg          [NUM_STAGES-1:0]   register    [BUS_WIDTH-1:0] ;
    integer                         counter    ;


    always @(posedge CLK or negedge RST)
        begin

            if(!RST)

            for (counter = 0 ; counter < BUS_WIDTH ; counter = counter + 1 )
                    begin

                        register[counter] <= 'b0 ;
                        SYNC[counter]     <= 'b0 ;

                    end

            else
                begin

           for (counter = 0 ; counter < BUS_WIDTH ; counter = counter + 1 )

  {SYNC[counter],register[counter]} <={register[counter],ASYNCH[counter]}  ;

                end
        end

endmodule
```

TOP SYSTEM

```
`timescale 1ns/1ps

module TOP_SYSTEM
#(parameter WIDTH = 8 )
(
input csn,
input sys_reset,
input spi_clock,
input sys_clk,
input MOSI,

output o_miso_ena,
output packet,
output MISO

);


wire write_flag;
wire [31:0] Payload_data;
wire start_enable;
wire mode_to_mode;

wire domain2_enable;
wire domain2_mode;
wire domain2_done_async;
wire domain1_done_sync;

wire rst_d1;
wire rst_d2;


TOP_Domain_1 TOP_Domain_1_Insta (

.csn(csn),
.resetn(rst_d1),
.spi_clock(spi_clock),
.MOSI(MOSI),
.o_miso_ena(o_miso_ena),
.rf_tx_start(start_enable),
.rf_tx_mode(mode_to_mode),
.rf_power_domain(),
.rf_tx_data(Payload_data),
.rf_tx_done(domain1_done_sync),

.write_flag(write_flag),
.MISO(MISO)
);

TOP TOP_CRC_SER_Insta(

.crc_load_ext(write_flag),

.clk(sys_clk),
.reset(rst_d2),
```

```
.payload(Payload_data),
.mode(domain2_mode),

.enable(domain2_enable),

.packet(packet),
.serial_done_tick(domain2_done_async)

);

//output of start bit in domain 1 to enable in domain 2
BIT_SYNC BIT_SYNC_start(
.ASYNCH(start_enable),
.RST(rst_d2),
.CLK(sys_clk),

.SYNC(domain2_enable)
);

//output mode of domain 1 to mode of domain 2
BIT_SYNC BIT_SYNC_mode(
.ASYNCH(mode_to_mode),
.RST(rst_d2),
.CLK(sys_clk),
.SYNC(domain2_mode)
);

//output done tick of domain 2 to  domain 1
BIT_SYNC BIT_SYNC_done_tick(
.ASYNCH(domain2_done_async),
.RST(rst_d1),
.CLK(spi_clock),
.SYNC(domain1_done_sync)
);

//Reset domain 1
Reset_Sync Rst_D1
(
.RST(sys_reset),
.CLK(spi_clock),
.Sync_RST(rst_d1)
);

//Reset domain 2
Reset_Sync Rst_D2
(
.RST(sys_reset),
.CLK(sys_clk),
.Sync_RST(rst_d2)
);

endmodule
```

Chapter 4 Simulations Results

TOP Domain 1

Write Operation

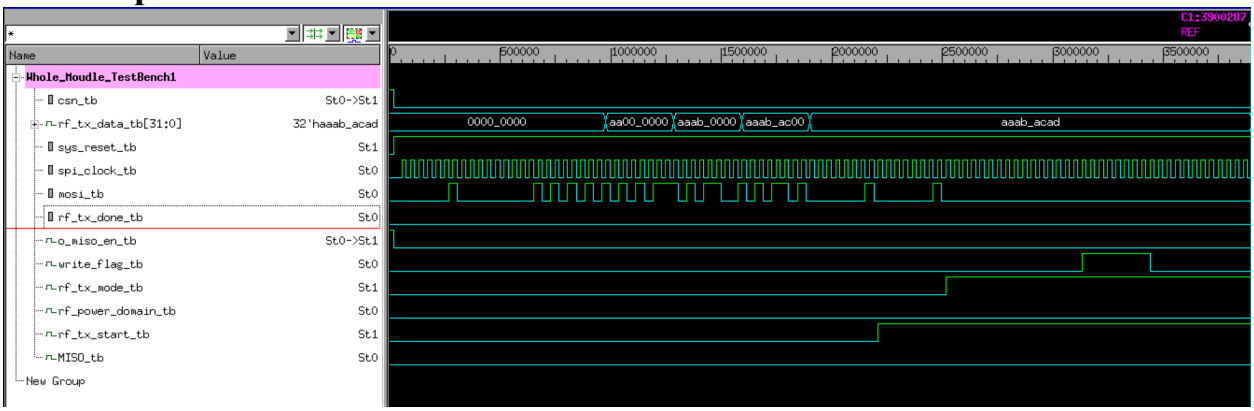


Figure 4 Testbench of Domain 1 | Write Operation

Write Command ‘h02 , Address h’00

As you see payload data is on MOSI, and when all data is written a write-flag pulse is on

Later on this write flag pulse will be used to load crc module

Read Operation

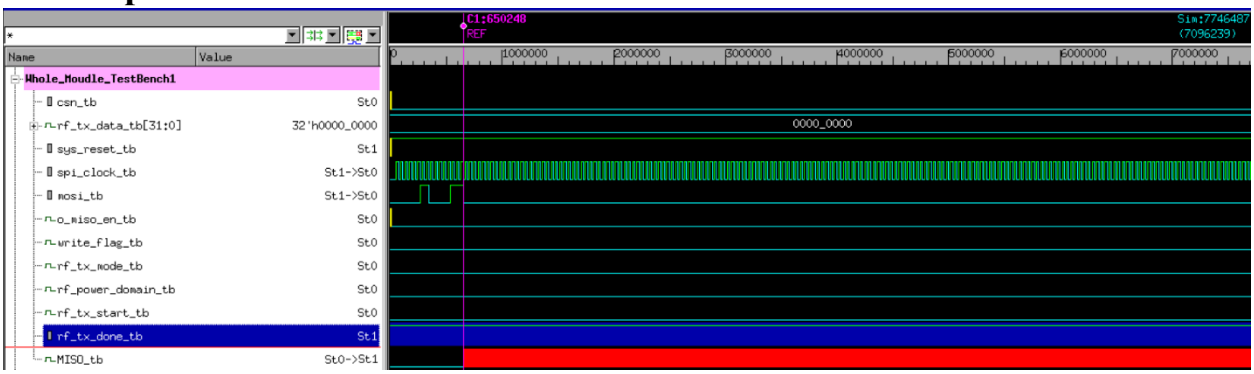
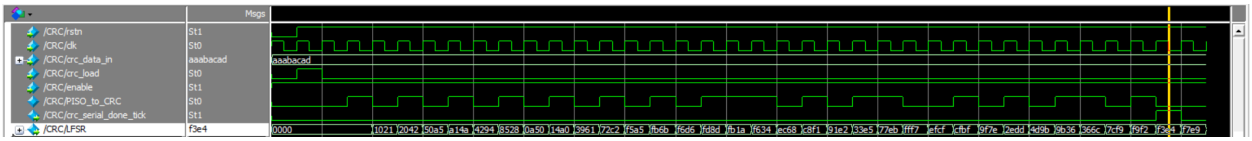


Figure 5 Testbench of Domain 1 | Read Operation

Assume rf_tx_done_tb is 1 , it’s in the 7th place in the register file .

Read Command h03, Address h’07 in binary 0000-0111 , as you see MISO read the data.

CRC Module



This module is the starting of the domain 2 , enable signal here is tx_start which SPI master send it to the register file through the spi slave , also crc_load is pulsed from write flag of register file .

When enable and crc_load are 1's data is loaded to CRC module , and when CRC check bytes are ready a done tick is pulsed which load the serializer module .

CRC check from online calculator

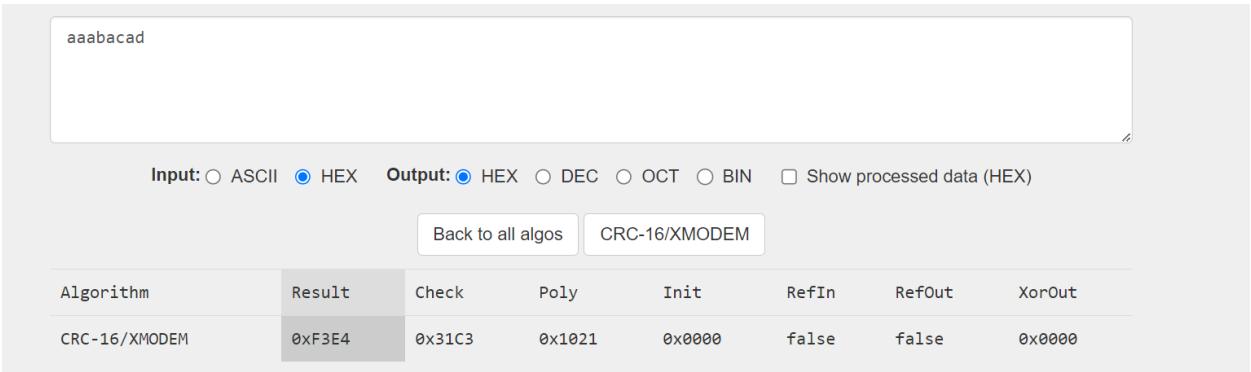


Figure 6 CRC-16 online calculator

Click **here** to see result from the website .

TOP Domain 2

2MHz

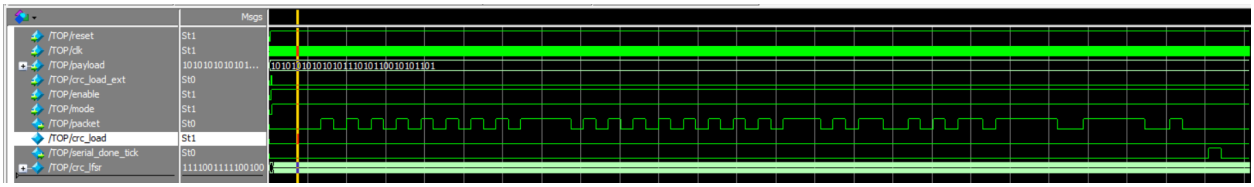


Figure 7 Testbench of Domain 2 - CLK 2MHz

This screenshot from ModelSim.

crc_load_ext is coming from CRC Module “crc load” , Enable signal is the tx_start which sent from SPI Master to spi slave to register file to TOP domain 2.
once crc_load_ext and enable signal are 1’s, that means data is loaded to CRC module . crc_load_ext is last just for 1 clock cycle .

when crc is ready , the done tick from CRC Module load the data to the serializer , and it last for one clock cycle.

Mode is 1 , so the system clock is divided by 2 which means 32/2 = 16 , each bit will take 16 clock cycle as shown in Figure 7

There is internal counter which serialize the data depending on the mode .

After the transmission is done, the done signal is equal to 1 for one clock cycle , this signal will go to the Register file .

Payload data is ‘hAAABACAD ➔ ‘b 10101010101010111010110010101101 , lfsr is ‘b 1111001111100100

1MHz

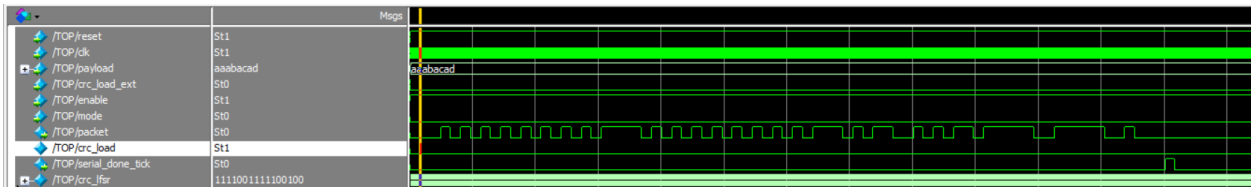


Figure 8 Testbench of Domain 2 - CLK 1MHz

Same as last simulation but the Mode is 0 , so the system clock is divided by 11 which means 32/1 =32, each bit will take 32 clock cycle.

TOP System

Write Operation

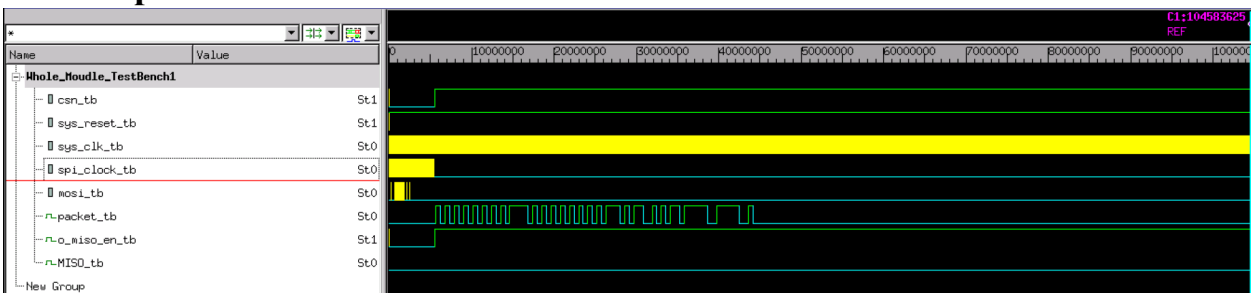


Figure 9 Testbench of TOP SYSTEM | Write Operation

Write Command ‘h02 , Address h’00 and Payload data on MOSI

Packet contains the preamble , Header , Payload , CRC

Read Operation

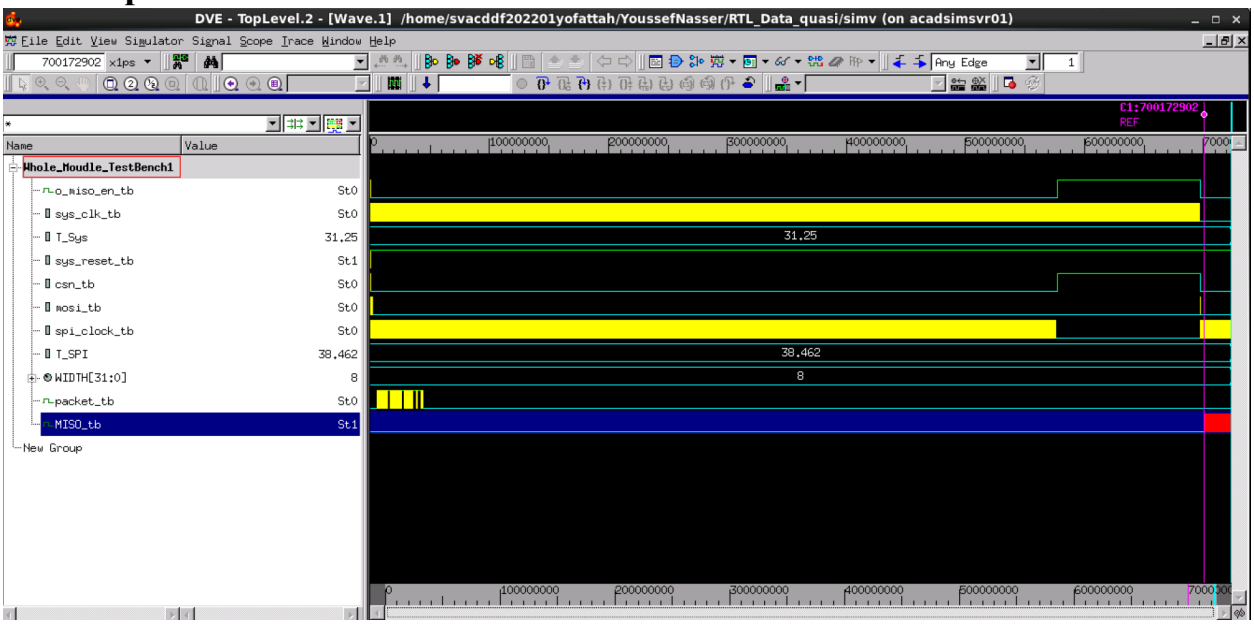


Figure 10 Testbench of TOP SYSTEM | Read Operation Part 1

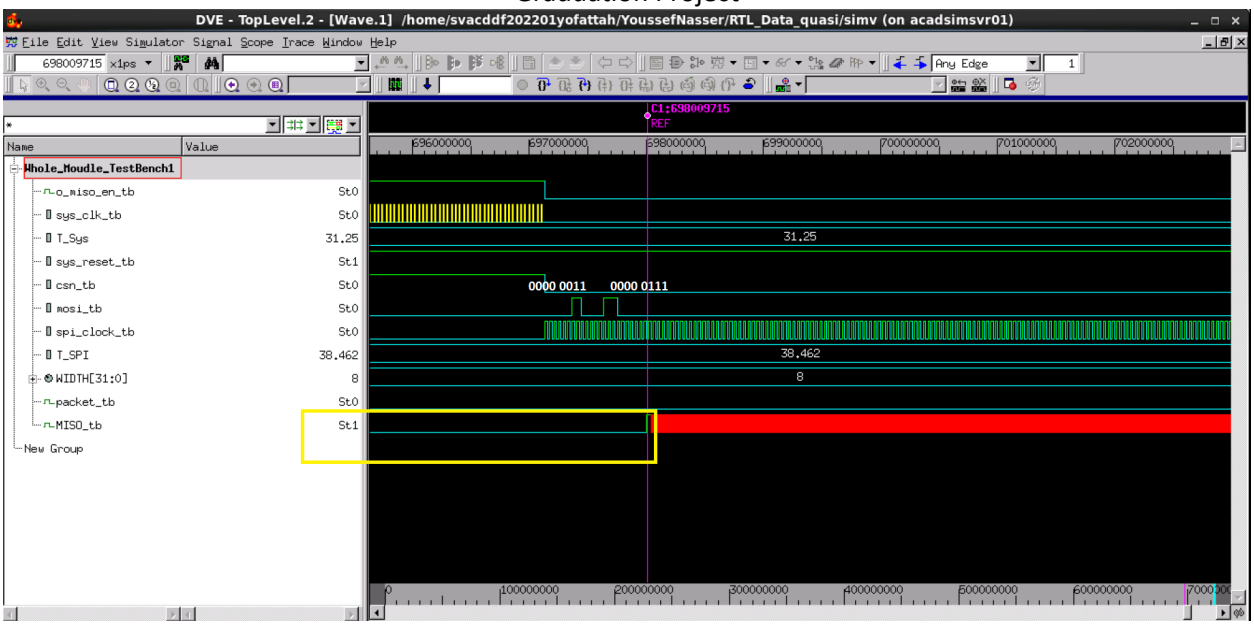


Figure 11 Testbench of TOP SYSTEM | Read Operation part 2

After finishing the transmission, rf_tx_done_tb is 1 ; it's in the 7th place in the register file .

Read Command h03, Address h'07 in binary 0000-0111 , as you see MISO read the data

Chapter 5 Spyglass

The Spyglass tool is a widely-used static code analysis and linting tool developed by Synopsys. It is primarily used for digital design verification and optimization in the field of electronic design automation (EDA). The tool helps designers improve the quality, reliability, and efficiency of their digital designs by performing comprehensive checks on the design code.

Note Beside: Lint and CDC goals are working, while RDC is not working.

Steps of the Spyglass

1. Write in the terminal spyglass
2. New Project
3. Upload Your Verilog files in HDL files
4. Upload the constraints.SDGC in constrained file
5. Upload the Waiver.AWL in waiver file
6. Read Design
7. Tick the synthesis box
8. Force Read Design
9. Select your goals and run them.

Spyglass Constraints

```
##System SDGC for Spyglass

//Define My Clocks
clock -name sys_clk domain clka -tag -period 31.25 -edge {0 15.625}
clock -name sys_clk domain clkb -tag -period 38.462 -edge {0 19.231}
//Define My Resets
reset -name TOP_SYSTEM.csn -value 1
reset -async -name sys_reset -value 0
//Define My Ports
abstract_port -ports sys_reset -clock sys_clk
abstract_port -ports csn -clock spi_clock
abstract_port -ports MOSI -clock spi_clock
abstract_port -ports MISO -clock spi_clock
abstract_port -ports o_miso_ena -clock spi_clock
abstract_port -ports packet -clock sys_clk
//Define Reset Synch
reset_synchronizer -name TOP_SYSTEM.Rst_D1.Sync_RST -clock spi_clock-reset
sys_reset value 0
reset_synchronizer -name TOP_SYSTEM.Rst_D2.Sync_RST -clock sys_clk -reset
sys_reset value 0
//Define BIT Synch
sync_cell -name BIT_SYNC -from_clk spi_clock -to_clk sys_clk
sync_cell -name BIT_SYNC -from_clk sys_clk -to_clk spi_clock
//Data quasi static
current_design "TOP_SYSTEM"
quasi_static -name "Payload_data" -override -disable_sva
```

Waivers

```
# waive -du {  {TOP_SYSTEM}  }  -msg {Instance output port 'rf_power_domain' is not connected}  -rule {  {W287b}  }  -comment {Created by svacddf202201yofattah on 11-
May-2023 01:03:45 for spi ' we don't have the control about it so i have waived it for the power pin , it will be introduced while UPF}
# waive -du {  {TOP_SYSTEM}  }  -msg {Instance output port 'o_miso_ena' is not connected}  -rule {  {W287b}  }  -comment {Created by svacddf202201yofattah on 11-
May-2023 01:03:45} ;# waive -du {  {TOP_SYSTEM}  }  -msg {Instance output port 'rf_power_domain' is not connected}  -rule {  {W287b}  }  -comment {Created by
svacddf202201yofattah on 11-May-2023 01:03:45 for spi ' we don't have the control about it so i have waived it for the power pin , it will be introduced while UPF} ;#
waive -du {  {spi_slave}  }  -msg {Variable 'tx_mode_s' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04}
# waive -du {  {spi_slave}  }  -msg {Variable 'idle_s' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04}
# waive -du {  {spi_slave}  }  -msg {Variable 'miso_ena_next' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04}
# waive -du {  {spi_slave}  }  -msg {Variable 'rx_mode_s' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04}
# waive -du {  {spi_slave}  }  -msg {Variable 'o_sleep' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04}
# waive -du {  {spi_slave}  }  -msg {Variable 'o_standby' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04}
# waive -du {  {spi_slave}  }  -msg {Variable 'o_deep_sleep' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:07:04 we don't have control on SPI pins ... so i have waived them}
# waive -msg {LHS: 'rf_power_domain' width 1 is less than RHS: 'reg_file[6]' width 8 in assignment [Hierarchy: 'TOP_SYSTEM.TOP_Domain_1_Insta.block2']}  -rule
{  {Av_width_mismatch_assign}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:18:30}
# waive -msg {LHS: 'rf_tx_mode' width 1 is less than RHS: 'reg_file[5]' width 8 in assignment [Hierarchy: 'TOP_SYSTEM.TOP_Domain_1_Insta.block2']}  -rule
{  {Av_width_mismatch_assign}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:18:30}
# waive -msg {LHS: 'rf_tx_start' width 1 is less than RHS: 'reg_file[4]' width 8 in assignment [Hierarchy: 'TOP_SYSTEM.TOP_Domain_1_Insta.block2']}  -rule
{  {Av_width_mismatch_assign}  }  -comment {Created by svacddf202201yofattah on 11-May-2023 01:18:30 the right hand side must be 1 bit as grad paper . we only need the
LSB of RegFile in order to write / read in these signals. }
# waive -msg {Recommended edge (positive) of clock 'TOP_SYSTEM.spi_clock' not used}  -rule {  {Clock_check04}  }  -comment {Created by svacddf202201yofattah on 11-
May-2023 01:24:21 it's from spi slave file , so we can not control it}
waive -du {  {TOP_SYSTEM}  }  -msg {Instance output port 'rf_power_domain' is not connected}  -rule {  {W287b}  }  -comment {Created by svacddf202201yofattah on 15-
May-2023 00:40:31.will be used in UPF }
waive -du {  {spi_slave}  }  -msg {Variable 'idle_s' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule {  {W528}  }
-comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19}
waive -du {  {spi_slave}  }  -msg {Variable 'o_sleep' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19}
waive -du {  {spi_slave}  }  -msg {Variable 'o_deep_sleep' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19}
waive -du {  {spi_slave}  }  -msg {Variable 'o_standby' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19}
waive -du {  {spi_slave}  }  -msg {Variable 'rx_mode_s' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19}
waive -du {  {spi_slave}  }  -msg {Variable 'tx_mode_s' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19}
waive -du {  {spi_slave}  }  -msg {Variable 'miso_ena_next' set but not read.[Hierarchy: ':TOP_SYSTEM:TOP_Domain_1_Insta@TOP_Domain_1:block1@spi_slave']}  -rule
{  {W528}  }  -comment {Created by svacddf202201yofattah on 15-May-2023 00:41:19 SPI PINS , we don't have the control to modify the code .}
waive -msg {Recommended edge (positive) of clock 'TOP_SYSTEM.spi_clock' not used}  -rule {  {Clock_check04}  }  -comment {Created by svacddf202201yofattah on 15-
May-2023 00:48:53 it's a bad  design because it works with both clk edges  butt sure it has a purpose }
waive -msg {LHS: 'rf_tx_start' width 1 is less than RHS: 'reg_file[4]' width 8 in assignment [Hierarchy: 'TOP_SYSTEM.TOP_Domain_1_Insta.block2']}  -rule
{  {Av_width_mismatch_assign}  }  -comment {the design specs need the signals to be 1 bit , but the word width is 8 bits , so it's okay to take the lsb from the word
to the singals.}
waive -msg {LHS: 'rf_tx_mode' width 1 is less than RHS: 'reg_file[5]' width 8 in assignment [Hierarchy: 'TOP_SYSTEM.TOP_Domain_1_Insta.block2']}  -rule
{  {Av_width_mismatch_assign}  }  -comment {the design specs need the signals to be 1 bit , but the word width is 8 bits , so it's okay to take the lsb from the word
to the singals.}
waive -msg {LHS: 'rf_power_domain' width 1 is less than RHS: 'reg_file[6]' width 8 in assignment [Hierarchy: 'TOP_SYSTEM.TOP_Domain_1_Insta.block2']}  -rule
{  {Av_width_mismatch_assign}  }  -comment {the design specs need the signals to be 1 bit , but the word width is 8 bits , so it's okay to take the lsb from the word
to the singals. }
waive -du {  {BIT_SYNC}  }  -msg {Asynchronous reset signal 'TOP_SYSTEM.sys_reset' is synchronized at least twice (at 'TOP_SYSTEM.BIT_SYNC_mode.register[0][0]' and
'TOP_SYSTEM.BIT_SYNC_start.register[0][0]') relative to clock signal 'TOP_SYSTEM.sys_clk'}  -rule {  {Reset_sync04}  }  -comment {i have global reset for the Domain2
and Register file in domain1 , so i have 2 Rest sync as i have 2 domains .}
waive -msg {Synchronized crossing: destination flop 'TOP_SYSTEM.TOP_CRC_SER_Insta.CRC_Insta.U0.piso_serial_out', clocked by 'TOP_SYSTEM.sys_clk', source flop
'TOP_SYSTEM.TOP_Domain_1_Insta.block2.write_flag', clocked by 'TOP_SYSTEM.spi_clock'. Data-enable sequencing check: Partially-Proved}  -rule {  {Ac_datahold01a}  }  -
comment {there are 3 bit sync for the signals that cross the domains}
waive -msg {Synchronized crossing: destination flop 'TOP_SYSTEM.TOP_CRC_SER_Insta.CRC_Insta.U0.piso_serialize', clocked by 'TOP_SYSTEM.sys_clk', source flop
'TOP_SYSTEM.TOP_Domain_1_Insta.block2.write_flag', clocked by 'TOP_SYSTEM.spi_clock'. Data-enable sequencing check: Partially-Proved}  -rule {  {Ac_datahold01a}  }  -
comment {there are 3 bit sync for the signals that cross the domains}
waive -msg {Synchronized crossing: destination flop 'TOP_SYSTEM.TOP_CRC_SER_Insta.CRC_Insta.U0.counter[5:0]', clocked by 'TOP_SYSTEM.sys_clk', source flop
'TOP_SYSTEM.TOP_Domain_1_Insta.block2.write_flag', clocked by 'TOP_SYSTEM.spi_clock'. Data-enable sequencing check: Partially-Proved}  -rule {  {Ac_datahold01a}  }  -
comment {there are 3 bit sync for the signals that cross the domains}
```

Figure 12 Waiver file

Table 3 Waiver Issues

Problem	Severity	Comment
spi slave interfaces	warning	I don't have the authority to modify the .v file
Sys reset is sync at least twice	warning	Because I use two reset sync as I have 2 domains
The power pin is not connected	warning	Because it will be used in UPF
Width mismatch of start/mode/done signals	warning	Because the word of the register file is 8 bits while these signals are just 1 bit each, it will take the LSB .. no worries

Spyglass Reports

Read Design

```
Date Created      : 21-05-2023 13:37:06
SpyGlass Version  : SpyGlass_vP-2019.06
Project Name      : spyglass-1
Goal Name         : Design_Read
Scenario Name     : default_scenario
Top               :

Message Summary
Severity    Non-Waived  Waived
FATAL              0         0
ERROR              0         0
WARNING            0         0
INFO               2         0
```

Report 1 Read Design | spyglass

Linting goals

```
Date Created      : 21-05-2023 13:39:19
SpyGlass Version  : SpyGlass_vP-2019.06
Project Name      : spyglass-1
Goal Name         : lint/lint_rtl
Scenario Name     : default_scenario
Top               :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         8
INFO              2         0
```

Report 2 lint rtl | spyglass

```
Date Created      : 21-05-2023 13:41:27
SpyGlass Version  : SpyGlass_vP-2019.06
Project Name      : spyglass-1
Goal Name         : lint/lint_turbo_rtl
Scenario Name     : default_scenario
Top               :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         8
INFO              3         0
```

Report 3 lint turbo rtl | spyglass

```
Date Created     : 21-05-2023 13:45:19
SpyGlass Version : SpyGlass_vP-2019.06
Project Name     : spyglass-1
Goal Name        : lint/lint_functional_rtl
Scenario Name    : default_scenario
Top              :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         3
INFO              5         0

-dashboard_data advanced_lint_partial_proven_properties 0
-dashboard_data advanced_lint_failed_properties 3
-dashboard_data advanced_lint_total_properties 3
-dashboard_data advanced_lint_average_depth
-dashboard_data advanced_lint_minimum_depth
-dashboard_waived_data advanced_lint_failed_properties 3
-dashboard_waived_data advanced_lint_total_properties 3
```

Report 4 lint functional rtl | spyglass

```
Date Created     : 21-05-2023 13:47:23
SpyGlass Version : SpyGlass_vP-2019.06
Project Name     : spyglass-1
Goal Name        : lint/lint_abstract
Scenario Name    : default_scenario
Top              :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         0
INFO              3         0

-dashboard_data sync_crossings
```

Report 5 lint abstract | spyglass

CDC goals

```
Date Created     : 21-05-2023 14:21:56
SpyGlass Version : SpyGlass_vP-2019.06
Project Name     : spyglass-2
Goal Name        : cdc/cdc_setup_check
Scenario Name    : default_scenario
Top              :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         0
INFO             20         0

-dashboard_data sync_crossings 0
```

Report 6 CDC setup check | spyglass

```
Date Created      : 21-05-2023 14:19:10
SpyGlass Version  : SpyGlass_vP-2019.06
Project Name      : spyglass-2
Goal Name         : cdc/cdc_verify
Scenario Name     : default_scenario
Top               :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         4
INFO             41         0

-dashboard_data cdc_partial_proven_properties 3
-dashboard_data cdc_failed_properties 0
-dashboard_data cdc_total_properties 4
-dashboard_data cdc_average_depth 200
-dashboard_data cdc_minimum_depth 55
-dashboard_data unsync_crossings 0
-dashboard_data sync_crossings 8
-dashboard_waived_data cdc_partial_proven_properties 3
-dashboard_waived_data cdc_total_properties 3
```

Report 7 DC verify | spyglass

```
Date Created      : 21-05-2023 14:14:22
SpyGlass Version  : SpyGlass_vP-2019.06
Project Name      : spyglass-2
Goal Name         : cdc/cdc_verify_struct
Scenario Name     : default_scenario
Top               :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         1
INFO             37         0

-dashboard_data unsync_crossings 0
-dashboard_data sync_crossings 8
```

Report 8 CDC verify struct | spyglass

```
Date Created     : 21-05-2023 14:20:59
SpyGlass Version : SpyGlass_vP-2019.06
Project Name     : spyglass-2
Goal Name        : cdc/clock_reset_integrity
Scenario Name    : default_scenario
Top              :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         1
INFO              8         0

-dashboard_data sync_crossings 0
```

Report 9 clock reset integrity | spyglass

```
Date Created     : 21-05-2023 14:19:59
SpyGlass Version : SpyGlass_vP-2019.06
Project Name     : spyglass-2
Goal Name        : cdc/cdc_abstract
Scenario Name    : default_scenario
Top              :

Message Summary
Severity   Non-Waived  Waived
FATAL             0         0
ERROR             0         0
WARNING           0         0
INFO             10         0

-dashboard_data sync_crossings 0
```

Report 10 CDC abstract | spyglass

Chapter 6 Synthesis

Synthesis refers to the process of converting a high-level hardware description of a digital circuit into a gate-level representation that can be implemented in hardware. It is a crucial step in the design flow of integrated circuits.

On the terminal write dc_shell -f My_Syn.tcl

Synthesis Flow

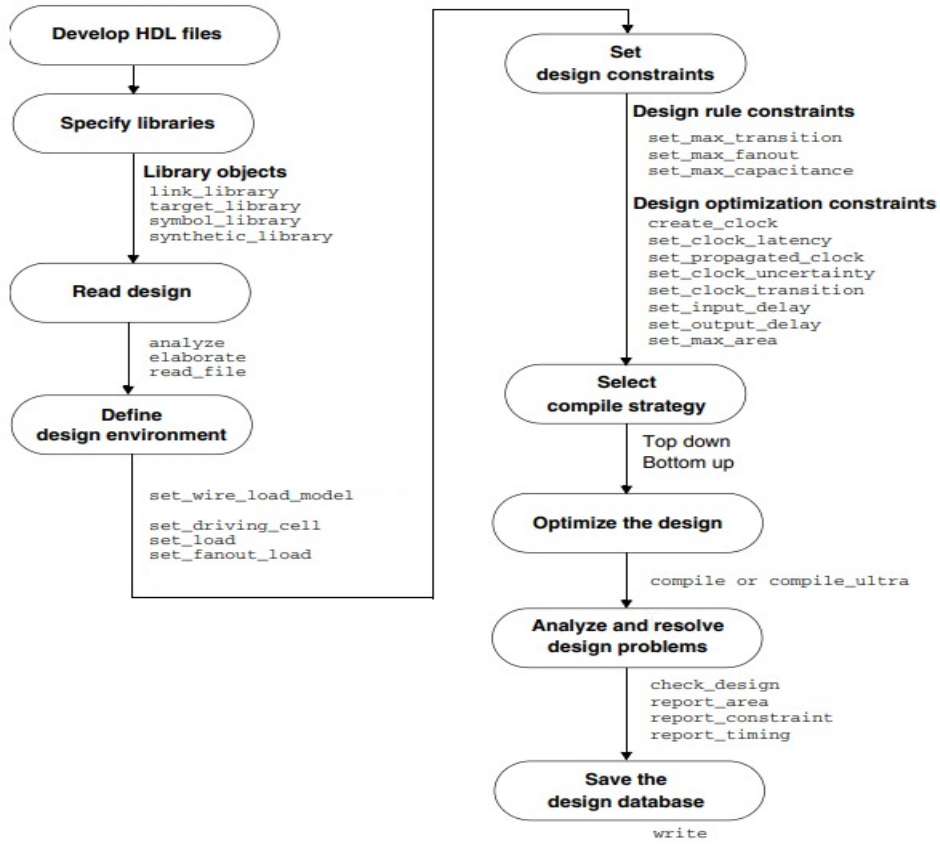


Figure 13 Basic Synthesis Flow

Library

Library name: saed32hvt_ss0p7vn40c.db

Library path : /tools/PDK/install/SAED_PDK32nm/memory/lib/stdcell_hvt/db_ccs/

Wire load model : 16000 , which has capacitance equal to 0.000625 and resistance equal to 1.41e-03

Buffer cell : IBUFFX4_HVT

Script

```
set path_lib /tools/PDK/install/SAED_PDK32nm/memory/lib/stdcell_hvt/db_ccs/
/tools/PDK/install/SAED_PDK32nm/memory/lib/stdcell_hvt/db_ccs/
lappend search_path $path_lib

set MLIB "saed32hvt_ss0p7vn40c.db"

set target_library [list $MLIB]
set link_library [list * $MLIB]

set file_format verilog
set path /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi

read_file -format $file_format $path/TOP_SYSTEM.v
read_file -format $file_format $path/TOP_Domain_1.v
read_file -format $file_format $path/register_file.v
read_file -format $file_format $path/spi_slave.v
read_file -format $file_format $path/TOP.v
read_file -format $file_format $path/SER4.v
read_file -format $file_format $path/CRC.v
read_file -format $file_format $path/PISO.v
read_file -format $file_format $path/FCS.v
read_file -format $file_format $path/BIT_SYNC.v
read_file -format $file_format $path/Reset_Sync.v

current_design TOP_SYSTEM
link

check_design

set_units -time ns
set_units -capacitance ff

set_wire_load_model -name "16000" -library saed32hvt_ss0p7vn40c

set_driving_cell -lib_cell IBUFFX4_HVT -library saed32hvt_ss0p7vn40c -pin Y
[get_ports MOSI]
set_driving_cell -lib_cell IBUFFX4_HVT -library saed32hvt_ss0p7vn40c -pin Y
[get_ports csn]

create_clock -name spi_clock -period 38.46 -waveform {0 19.23} [get_ports
spi_clock]
create_clock -name spi_clock -period 31.25 -waveform {0 15.625} [get_ports
sys_clk]

set_clock_uncertainty -setup 0.05 [get_clocks spi_clock]
set_clock_uncertainty -setup 0.05 [get_clocks sys_clk]

set_clock_transistion 0.5 -fall max {spi_clock sys_clk}
set_clock_transistion 0.5 -rise max {spi_clock sys_clk}

set_clock_groups -asynchronous -group [get_clocks {spi_clock}] \
-group [get_clocks {sys_clk}]

set_multicycle_path -setup 3 -from spi_clock -to sys_clk
set_multicycle_path -hold 2 -from spi_clock -to sys_clk
```

```
set_input_delay -clock spi_clock -max 7.69 [get_ports MOSI]
set_input_delay -clock spi_clock -max 7.69 [get_ports csn]

set_output_delay -clock spi_clock -max 7.69 [get_ports o_miso_ena]
set_output_delay -clock spi_clock -max 7.69 [get_ports MISO]
set_output_delay -clock sys_clk -max 6.25 [get_ports packet]

group_path -name INREG -from [all_inputs]
group_path -name REGOUT -to [all_outputs]
group_path -name INOUT -from [all_inputs] -to [all_outputs]

set_max_fanout 32 [all_inputs]
set_max_capacitance 0.15 [all_outputs]
set_input_transition 13 [all_inputs]

set_load -max 0.15 [get_ports MISO]
set_load -max 0.15 [get_ports packet]
set_load -max 0.15 [get_ports o_miso_ena]

set_max_area 0

compile_enable_register_merging false

compile_ultra -no_autoungroup -gate_clock


set_svf "My_System.svf"

set path2 /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/reports

write_file -format verilog -hierarchy -output $path2/GLS/TOP_SYSTEM_Netlist.v
write_file -format ddc -hierarchy -output $path2/TOP_SYSTEM_ddc.ddc
write_sdc -nosplit $path2/TOP_SYSTEM_sdf.sdf
write_sdc -nosplit $path2/TOP_SYSTEM_sdgc.sdc

report_area -hierarchy > $path2/area_final.rpt
report_power -hierarchy > $path2/power.rpt
report_timing -max_paths 100 -delay_type min > $path2/hold_final.rpt
report_timing -max_paths 99 -delay_type max > $path2/setup.rpt
report_timing > $path2/Timing_report_final.rpt
report_clock -attributes > $path2/clocks_final.rpt
report_constraint -all_violators > $path2/constraint_final.rpt
```

Note : The SVF file is very important for the formality , Clock gating is inserted automatically from Synthesis .

Synthesis Reports

Power Report

```
****************************************
Report : power
        -hier
        -analysis_effort low
Design : TOP_SYSTEM
Version: P-2019.03-SP3
Date   : Sun May 21 12:48:07 2023
****************************************


Library(s) Used:

    saed32hvt_ss0p7vn40c (File: /tools/PDK/install/SAED_PDK32nm/memory/lib/stdcell_hvt/db_ccs/saed32hvt_ss0p7vn40c.db)


Operating Conditions: ss0p7vn40c   Library: saed32hvt_ss0p7vn40c
Wire Load Model Mode: enclosed

Design        Wire Load Model            Library
------------------------------------------------
TOP_SYSTEM             16000             saed32hvt_ss0p7vn40c
TOP_Domain_1           8000              saed32hvt_ss0p7vn40c
TOP                    8000              saed32hvt_ss0p7vn40c
CRC                    8000              saed32hvt_ss0p7vn40c
BIT_SYNC_2             ForQA             saed32hvt_ss0p7vn40c
BIT_SYNC_1             ForQA             saed32hvt_ss0p7vn40c
BIT_SYNC_0             ForQA             saed32hvt_ss0p7vn40c
Reset_Sync_1           ForQA             saed32hvt_ss0p7vn40c
Reset_Sync_0           ForQA             saed32hvt_ss0p7vn40c
spi_slave              8000              saed32hvt_ss0p7vn40c
register_file          8000              saed32hvt_ss0p7vn40c
SER4                   8000              saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_LOW_spi_slave_0
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_spi_slave_0
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_spi_slave_1
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_0
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_7
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_6
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_5
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_4
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_3
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_2
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_register_file_1
                       ForQA             saed32hvt_ss0p7vn40c
PISO                   8000              saed32hvt_ss0p7vn40c
FCS                    ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_SER4_0
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_SER4_1
                       ForQA             saed32hvt_ss0p7vn40c
SNPS_CLOCK_GATE_HIGH_PISO
                       ForQA             saed32hvt_ss0p7vn40c

Global Operating Voltage = 0.7
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1pW


--------------------------------------------------------------------------------
                                   Switch   Int      Leak     Total
Hierarchy                          Power    Power    Power    Power    %
--------------------------------------------------------------------------------
TOP_SYSTEM                            0.280    0.724 6.81e+07   69.101 100.0
  Rst_D2 (Reset_Sync_0)            2.84e-03      N/A 4.94e+05    0.451   0.7
  Rst_D1 (Reset_Sync_1)            3.73e-03      N/A 4.94e+05    0.454   0.7
  BIT_SYNC_done_tick (BIT_SYNC_0)     0.000    0.000 4.94e+05    0.494   0.7
  BIT_SYNC_mode (BIT_SYNC_1)       2.20e-03 4.96e-04 5.47e+05    0.549   0.8
  BIT_SYNC_start (BIT_SYNC_2)      2.20e-03 4.96e-04 5.47e+05    0.549   0.8
  TOP_CRC_SER_Insta (TOP)              0.102    0.203 3.67e+07   36.980  53.5
    SER_Insta (SER4)               6.62e-02 4.15e-02 2.24e+07   22.459  32.5
    CRC_Insta (CRC)                3.59e-02    0.162 1.43e+07   14.521  21.0
      U1 (FCS)                     1.24e-02 1.68e-04 3.09e+06    3.098   4.5
      U0 (PISO)                    2.35e-02    0.162 1.12e+07   11.422  16.5
  TOP_Domain_1_Insta (TOP_Domain_1)    0.162    0.605 2.87e+07   29.468  42.6
    block2 (register_file)         3.18e-02    0.425 1.56e+07   16.069  23.3
    block1 (spi_slave)                 0.130    0.180 1.31e+07   13.400  19.4
```

Report 11 Power Report | DC

Area Report

```
****************************************
Report : area
Design : TOP_SYSTEM
Version: P-2019.03-SP3
Date   : Sun May 21 12:48:07 2023
****************************************

Library(s) Used:

    saed32hvt_ss0p7vn40c (File: /tools/PDK/install/SAED_PDK32nm/memory/lib/stdcell_hvt/db_ccs/saed32hvt_ss0p7vn40c.db)

Number of ports:                          443
Number of nets:                          1069
Number of cells:                          651
Number of combinational cells:            346
Number of sequential cells:               268
Number of macros/black boxes:               0
Number of buf/inv:                         60
Number of references:                       8

Combinational area:               801.316043
Buf/Inv area:                      91.745986
Noncombinational area:           1890.068961
Macro/Black Box area:               0.000000
Net Interconnect area:            532.792591

Total cell area:                 2691.385004
Total area:                      3224.177595

Hierarchical area distribution
------------------------------

                                 Global cell area          Local cell area
                                 ------------------  --------------------------
Hierarchical cell                Absolute   Percent  Combi-    Noncombi-  Black-
                                 Total      Total    national  national   boxes   Design
-------------------------------  ---------  -------  --------  ---------  ------  -----------------------------------
TOP_SYSTEM                       2691.3850    100.0    4.0663     0.0000  0.0000  TOP_SYSTEM
BIT_SYNC_done_tick                 21.3481      0.8    0.0000    21.3481  0.0000  BIT_SYNC_0
BIT_SYNC_mode                      22.6188      0.8    1.2707    21.3481  0.0000  BIT_SYNC_1
BIT_SYNC_start                     22.6188      0.8    1.2707    21.3481  0.0000  BIT_SYNC_2
Rst_D1                             21.3481      0.8    0.0000    21.3481  0.0000  Reset_Sync_1
Rst_D2                             21.3481      0.8    0.0000    21.3481  0.0000  Reset_Sync_0
TOP_CRC_SER_Insta                1463.3612     54.4    0.0000     0.0000  0.0000  TOP
TOP_CRC_SER_Insta/CRC_Insta       569.7909     21.2    0.0000     0.0000  0.0000  CRC
TOP_CRC_SER_Insta/CRC_Insta/U0    443.4813     16.5  145.8787   291.7573  0.0000  PISO
TOP_CRC_SER_Insta/CRC_Insta/U0/clk_gate_counter_reg
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_PISO
TOP_CRC_SER_Insta/CRC_Insta/U1    126.3096      4.7   12.4531   113.8565  0.0000  FCS
TOP_CRC_SER_Insta/SER_Insta       893.5703     33.2  262.7849   619.0948  0.0000  SER4
TOP_CRC_SER_Insta/SER_Insta/clk_gate_data_reg_reg
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_SER4_1
TOP_CRC_SER_Insta/SER_Insta/clk_gate_serial_out_reg
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_SER4_0
TOP_Domain_1_Insta               1114.6756     41.4    0.0000     0.0000  0.0000  TOP_Domain_1
TOP_Domain_1_Insta/block1         477.7907     17.8  217.5473   242.1992  0.0000  spi_slave
TOP_Domain_1_Insta/block1/clk_gate_miso_reg_reg
                                    6.3536      0.2    0.0000     6.3536  0.0000  SNPS_CLOCK_GATE_LOW_spi_slave_0
TOP_Domain_1_Insta/block1/clk_gate_mosi_reg_reg
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_spi_slave_0
TOP_Domain_1_Insta/block1/clk_gate_rf_add_reg_reg
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_spi_slave_1
TOP_Domain_1_Insta/block2         636.8849     23.7  156.0444   434.0780  0.0000  register_file
TOP_Domain_1_Insta/block2/clk_gate_counter_reg
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_0
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[0]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_7
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[1]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_6
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[2]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_5
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[3]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_4
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[4]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_3
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[5]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_2
TOP_Domain_1_Insta/block2/clk_gate_reg_file_reg[6]
                                    5.8453      0.2    0.0000     5.8453  0.0000  SNPS_CLOCK_GATE_HIGH_register_file_1
-------------------------------  ---------  -------  --------  ---------  ------  -----------------------------------
Total                                                801.3160  1890.0690  0.0000
```

Report 12 Area Report | DC

Clock Report

```
****************************************
Report : clocks
Design : TOP_SYSTEM
Version: P-2019.03-SP3
Date   : Sun May 21 12:48:07 2023
****************************************

Attributes:
    d - dont_touch_network
    f - fix_hold
    p - propagated_clock
    G - generated_clock
    g - lib_generated_clock

Clock          Period   Waveform            Attrs     Sources
--------------------------------------------------------------------------------
spi_clock       38.46   {0 19.23}                     {spi_clock}
sys_clk         31.25   {0 15.625}                    {sys_clk}
--------------------------------------------------------------------------------
```

Report 13 Clock Report | DC

Constraints Report

```
****************************************
Report : constraint
         -all_violators
Design : TOP_SYSTEM
Version: P-2019.03-SP3
Date   : Sun May 21 12:48:07 2023
****************************************


   max_area

                            Required        Actual
   Design                     Area            Area           Slack
   -----------------------------------------------------------------
   TOP_SYSTEM                   0.00        3224.18       -3224.18  (VIOLATED)
```

Report 14 Constraints all violators report | DC

Chapter 7 Formality

A logical equivalence check is very important, it compares the RTL and the Netlist

Script

```
#formality -f My_Formality.tcl

##formality setup##

set synopsys_auto_setup true

##formality Guidence ##

set_svf -append
{/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/default.svf}


## formality REF ##
read_verilog -container r -libname WORK -05
{ /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/TOP_SYSTEM.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/TOP_Domain_1.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/register_file.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/spi_slave.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/TOP.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/SER4.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/CRC.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/PISO.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/FCS.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/BIT_SYNC.v
/home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/Reset_Sync.v }

read_db -container
{ /tools/PDK/install/SAED_PDK32nm/memory/lib_stdcell_hvt/db_ccs/saed32hvt_ss0
p7vn40c.db }

set_top r:/WORK/TOP_SYSTEM

## formality Imp ##

read_verilog -container i -libname WORK -05
{ /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/reports/GLS/TOP_SY
STEM_Netlist.v }

read_db
{ /tools/PDK/install/SAED_PDK32nm/memory/lib_stdcell_hvt/db_ccs/saed32hvt_ss0
p7vn40c.db }

set_top i:/WORK/TOP_SYSTEM

##formality matching points##
matching
##formality verifying##
verify
```

Result of Formality

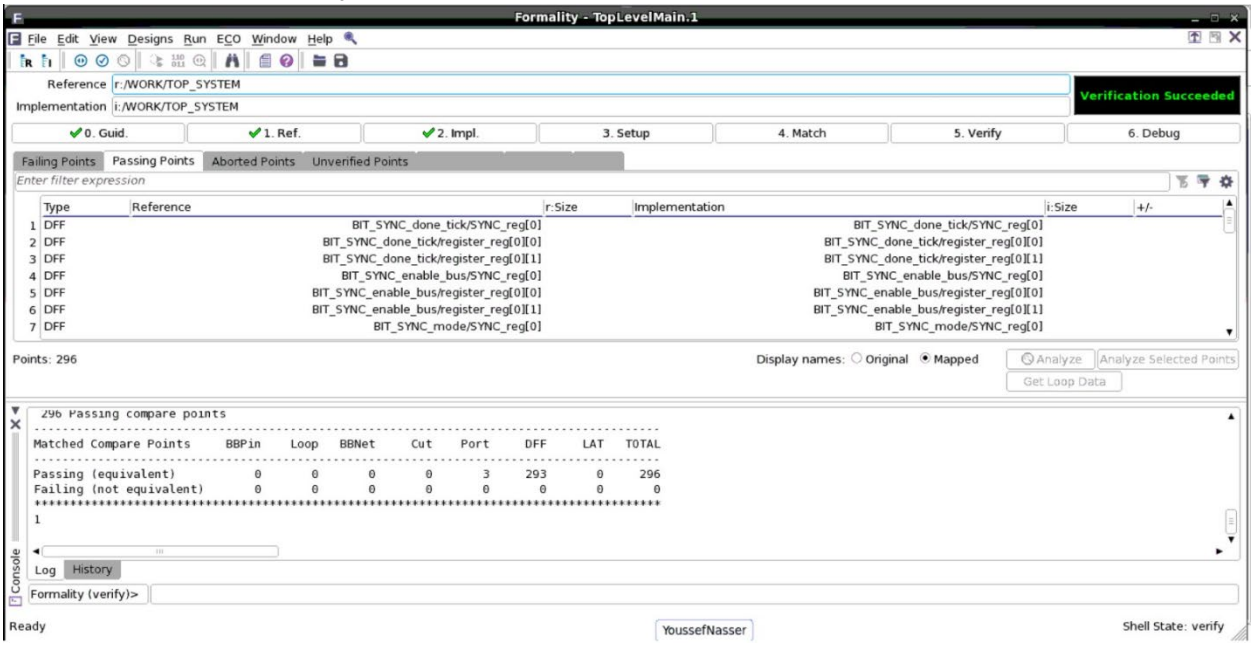


Figure 14 Formality Result

Chapter 8 Gate Level Simulation

Steps

1. Get the Verilog file of the used library
2. Get the test bench of the system
3. Get the Netlist that was generated from Synthesis
4. Open the terminal and run this command
5. vcs -v2005 -files TOP_SYSTEM_Netlist.v saed32nm_hvt.v TOP_SYSTEM_tb.v -R -gui
6. Scroll till you reach to the test bench module name

Note : The test bench is here Appendix B

Results

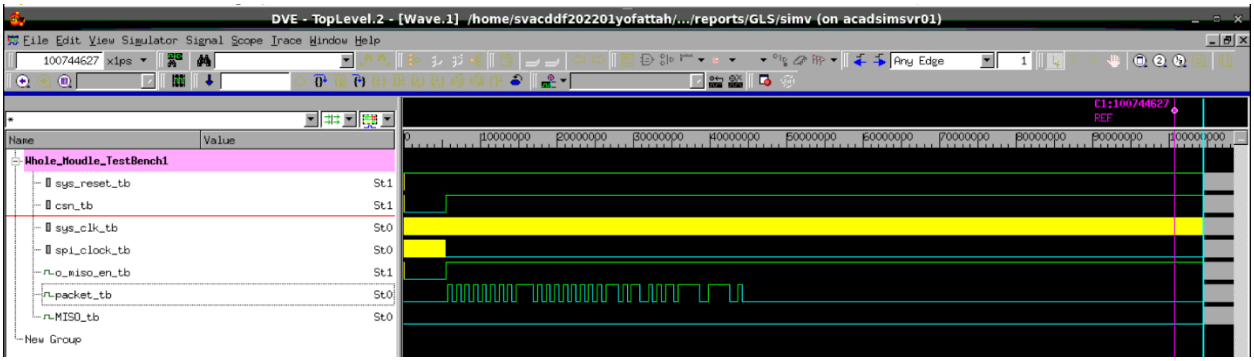


Figure 15 Post-Synthesis Simulation

Chapter 9 DFT

DFT stands for Design for Testability. It is a set of techniques and methodologies used to design integrated circuits (ICs) that can be easily tested for manufacturing defects and ensure high-quality production.

The main goal of DFT is to enhance the testability of ICs and facilitate the detection and diagnosis of faults or defects during the manufacturing process or in the field. By incorporating DFT features into the design, it becomes easier to verify the correctness of the manufactured chips and identify any potential failures or errors.

DFT FLOW

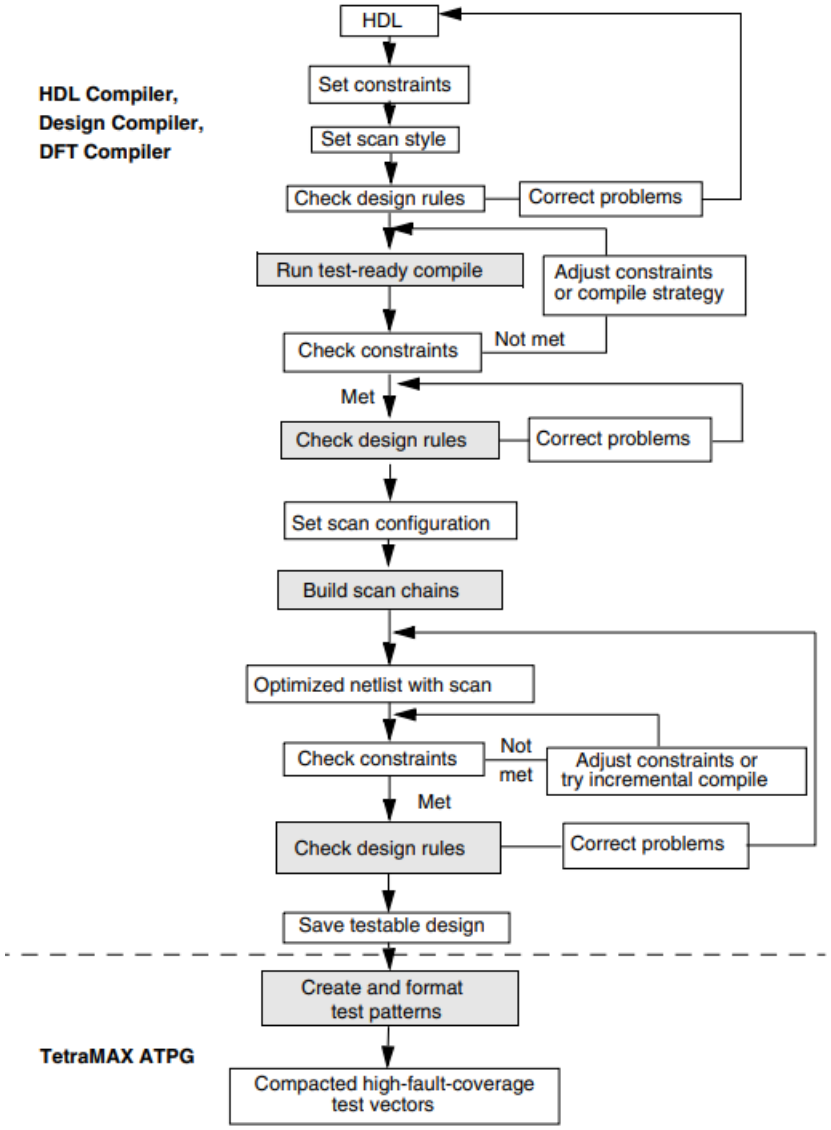


Figure 16 DFT FLOW

RTL PREPARATION

- Scan ports
  - scan_mode
  - scan_shift_enable
  - scan_shift_cg
  - scan_reset_domain1
  - scan_reset_domain2
  - scan_<spi>_clk
  - scan_<sys>_clk
  - scan_input1
  - scan_input2
  - scan_output1
  - scan_output2
- Scan Considerations
  - Install Mux to choose between the functional mode and test mode
  - Clock gating should be done using latches
  - If your design is working with both edge clock, put an **inverter** in the Mux test mode pin in order to make the design work with one clock.

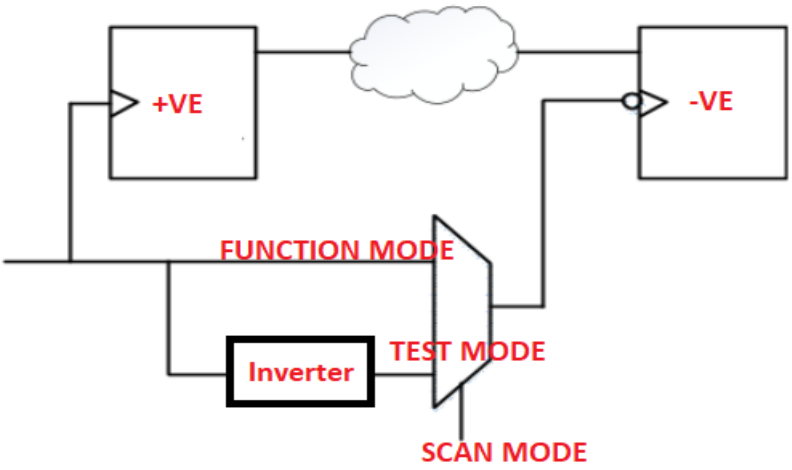


Figure 17 Double edge clock | DFT

If there are negative edge flops in the design and functional paths exist between positive and negative edge flops, then negative edge flops clocks are multiplexed using “scan_mode” to be the negation of their original clocks during scan mode so that during capture, no violation occurs.

TOP SYSTEM & DFT

```
`timescale 1ns/1ps

module TOP_SYSTEM_DFT
#(parameter WIDTH = 8 )
(
input csn,
input sys_reset,
input spi_clock,
input sys_clk,

input MOSI,

//DFT
input scan_spi_clock,
input scan_sys_clock,
input i_ioring_rst_n,
input Scan_set_reset_2,
input Scan_mode,
input scan_shift_enable,
input scan_shift_cg,
input scan_input_1,
input scan_input_2,
output scan_output_1,
output scan_output_2,

//DFT


output o_miso_ena,
output packet,
output MISO

);


wire write_flag;
wire [31:0] Payload_data;
wire start_enable;
wire mode_to_mode;

wire domain2_enable;
wire domain2_mode;
wire domain2_done_async;
wire domain1_done_sync;

wire rst_d1;
wire rst_d2;
```

```
TOP_Domain_1 TOP_Domain_1_Insta (

.csn(csn),
.resetn(rst_d1),
.spi_clock(spi_clock),
.MOSI(MOSI),
.o_miso_ena(o_miso_ena),
.rf_tx_start(start_enable),
.rf_tx_mode(mode_to_mode),
.rf_power_domain(),
.Scan_mode(Scan_mode),
.scan_shift_enable(scan_shift_enable),
.scan_shift_cg(scan_shift_cg),
.scan_spi_clock(scan_spi_clock),
.scan_input_1(scan_input_1),
.scan_output_1(scan_output_1),
.i_ioring_rst_n(i_ioring_rst_n),
.rf_tx_data(Payload_data),
.rf_tx_done(domain1_done_sync),

.write_flag(write_flag),
.MISO(MISO)
);
```

```
TOP TOP_CRC_SER_Insta(

.crc_load_ext(write_flag),

.clk(sys_clk),
.reset(rst_d2),
.payload(Payload_data),
.mode(domain2_mode),

.enable(domain2_enable),

.packet(packet),
.serial_done_tick(domain2_done_async),

.Scan_set_reset_2(Scan_set_reset_2),
.Scan_mode(Scan_mode),
.scan_shift_enable(scan_shift_enable),
.scan_shift_cg(scan_shift_cg),
.scan_sys_clock(scan_sys_clock),
.scan_input_1(scan_input_2),
.scan_output_1(scan_output_2)

);
```

```
//output of start bit in domain 1 to enable in domain 2
BIT_SYNC BIT_SYNC_start(
.ASYNCH(start_enable),
.RST(rst_d2),
.CLK(sys_clk),
.Scan_mode(Scan_mode),
.Scan_CLK(scan_sys_clock),
.Scan_Reg_reset(Scan_set_reset_2),
.SYNC(domain2_enable)
);

//output mode of domain 1 to mode of domain 2
BIT_SYNC BIT_SYNC_mode(
.ASYNCH(mode_to_mode),
.RST(rst_d2),
.CLK(sys_clk),
.Scan_mode(Scan_mode),
.Scan_CLK(scan_sys_clock),
.Scan_Reg_reset(Scan_set_reset_2),
.SYNC(domain2_mode)
);

//output done tick of domain 2 to  domain 1
BIT_SYNC BIT_SYNC_done_tick(
.ASYNCH(domain2_done_async),
.RST(rst_d1),
.CLK(spi_clock),
.Scan_mode(Scan_mode),
.Scan_CLK(scan_spi_clock),
.Scan_Reg_reset(i_ioring_rst_n),
.SYNC(domain1_done_sync)
);

//Reset domain 1
Reset_Sync Rst_D1
(
.RST(sys_reset),
.CLK(spi_clock),
.Scan_Reg_reset(i_ioring_rst_n),
.Scan_mode(Scan_mode),
.Scan_CLK(scan_spi_clock),
.Sync_RST(rst_d1)
);

//Reset domain 2
Reset_Sync Rst_D2
(
.RST(sys_reset),
.CLK(sys_clk),
.Scan_Reg_reset(Scan_set_reset_2),
.Scan_mode(Scan_mode),
.Scan_CLK(scan_sys_clock),
.Sync_RST(rst_d2)
);
```

```
endmodule
```

TOP DOMAIN 1 & DFT

```
`timescale 1ns/1ps
//TOP_Domain_1 without Sync
module TOP_Domain_1
#(parameter WIDTH = 8)
(
input csn,
input resetn,
input spi_clock,
input MOSI,
input rf_tx_done,

output rf_tx_start  ,
output  rf_tx_mode  ,
output  rf_power_domain,
output o_miso_ena,
output [31:0] rf_tx_data ,

//DFT//

input i_ioring_rst_n,
input Scan_mode,
input scan_shift_enable,
input scan_shift_cg,
input scan_spi_clock,
input scan_input_1,
output scan_output_1,

//DFT//

output write_flag,
output MISO

);
wire ve_clock;
wire domain_1_clock;
wire CSN_reset;
wire domain_1_reset_2;
wire WrEn;
wire [7:0] Address;
wire [7:0] slave_to_reg;
wire [7:0] reg_to_slave;
wire Reg_reset ;

spi_slave block1(

.i_csn(CSN_reset),
.i_sck(domain_1_clock),
.i_sck_neg(ve_clock),
.i_mosi(MOSI),
.o_miso(MISO),
.i_rf_din(reg_to_slave), //wire
.o_rf_wre(WrEn),//wire
```

```
.o_rf_addr(Address),//WIRE
.o_dout(slave_to_reg),//wire

.o_miso_ena(o_miso_ena),
.i_scan_mode(Scan_mode),
.i_ioring_rst_n(i_ioring_rst_n),
.i_status(16'd0)

);

register_file block2(

.clk(domain_1_clock),
.resetn(Reg_reset),
.write_enable(WrEn),
.address(Address),
.wr_data(slave_to_reg),
.rd_data(reg_to_slave),
.rf_tx_start(rf_tx_start),
.rf_tx_mode(rf_tx_mode),
.rf_power_domain(rf_power_domain),
.rf_tx_data(rf_tx_data),
.rf_tx_done(rf_tx_done),
.write_flag(write_flag)

);

Mux2X1 Domain_1_csn(
.Function_mode(spi_clock),
.Test_mode(scan_spi_clock),
.sel(Scan_mode),
.Mux_Out(domain_1_clock)
);

Mux2X1_ve_2x1 Domain_ve_spi_clock(
.Function_mode(spi_clock),
.Test_mode(scan_spi_clock),
.sel(Scan_mode),
.Mux_Out(ve_clock)
);


Mux2X1 Domain_1_csn(
.Function_mode(csn),
.Test_mode(i_ioring_rst_n),
.sel(Scan_mode),
.Mux_Out(CSN_reset)
);

Mux2X1 Domain_1_csn(
.Function_mode(reset),
.Test_mode(i_ioring_rst_n),
.sel(Scan_mode),
.Mux_Out(Reg_reset)
);
```

```
endmodule
```

TOP DOMAIN 2 & DFT

```
`timescale 1ns/1ps
//CRC with SER TOP
module TOP_DFT(

input crc_load_ext,
input clk,
input reset,
input [31:0] payload,
input mode,
input enable,

//DFT//

input Scan_set_reset_2,
input Scan_mode,
input scan_shift_enable,
input scan_shift_cg,
input scan_sys_clock,
input scan_input_2,
output scan_output_2,

//DFT//
output packet,
output serial_done_tick

);
wire domain_2_reset;
wire domain_2_clock;
wire crc_load;
wire [0:15] crc_lfsr;


CRC CRC_Insta(

.enable(enable),
.crc_data_in(payload),
.crc_load(crc_load_ext),
.clk(domain_2_clock),
.rstn(domain_2_reset),

.crc_serial_done_tick(crc_load),
.LFSR(crc_lfsr)

);
```

```
SER SER_Insta(

.data_payload(payload),
.data_crc(crc_lfsr),
.sys_clk(domain_2_clock),
.reset(domain_2_reset),
.mode(mode),
.load(crc_load),
.serial_done_tick(serial_done_tick),
.serial_out(packet)

);


Mux2X1 Domain_1_system_clock(
.Function_mode(clk),
.Test_mode(scan_sys_clock),
.sel(Scan_mode),
.Mux_Out(domain_2_clock)
);

Mux2X1 Domain_2_reset_sync(
.Function_mode(reset),
.Test_mode(Scan_set_reset_2),
.sel(Scan_mode),
.Mux_Out(domain_2_reset)
);

endmodule
```

BIT SYNC & DFT

```
`timescale 1ns/1ps

module BIT_SYNC #(parameter NUM_STAGES = 2, BUS_WIDTH = 1)
(
    input   wire [BUS_WIDTH-1:0]    ASYNCH  ,
    input   wire                    RST     ,
    input   wire                    CLK     ,
    input   wire Scan_mode , Scan_CLK , Scan_Reg_reset,
    output  reg  [BUS_WIDTH-1:0]    SYNC
);

wire Reg_reset;
wire Sync_clk;

Mux2X1 laaac3(
.Function_mode(RST),
.Test_mode(Scan_Reg_reset),
.sel(Scan_mode),
.Mux_Out(Reg_reset)
);

Mux2X1 laaac4(
.Function_mode(CLK),
.Test_mode(Scan_CLK),
.sel(Scan_mode),
.Mux_Out(Sync_clk)
);

    reg          [NUM_STAGES-1:0]   register    [BUS_WIDTH-1:0] ;
    integer                         counter ;


    always @(posedge Sync_clk or negedge Reg_reset)
        begin

            if(!Reg_reset)

           for (counter = 0 ; counter < BUS_WIDTH ; counter = counter + 1 )
                    begin
                        register[counter] <= 'b0;
                        SYNC[counter]     <= 'b0;
                                        end

            else
                begin

                    for (counter = 0 ; counter < BUS_WIDTH ; counter =
counter + 1 )
{SYNC[counter],register[counter]} <= {register[counter],ASYNCH[counter]};
                end
        end

endmodule
```

Reset SYNC & DFT

```
`timescale 1ns/1ps
module Reset_Sync #(parameter NUM_STAGES = 2)
(
    input   wire                    RST , Scan_Reg_reset ,
    input   wire                    CLK ,Scan_mode ,Scan_CLK

    output  reg                     Sync_RST
);
    wire Reg_reset;
    wire Sync_clk;
    reg          [NUM_STAGES-1:0]   register                              ;

Mux2X1 laaac1(
.Function_mode(RST),
.Test_mode(Scan_Reg_reset),
.sel(Scan_mode),
.Mux_Out(Reg_reset)
);

Mux2X1 laaac2(
.Function_mode(CLK),
.Test_mode(Scan_CLK),
.sel(Scan_mode),
.Mux_Out(Sync_clk)
);

    always @(posedge Sync_clk or negedge Reg_reset)
        begin

            if(!Reg_reset)
                begin
                    register <= 'b0  ;
                    Sync_RST <= 'b0  ;
                end
            else
                {Sync_RST,register} <= {register[NUM_STAGES-1:0],1'b1}  ;
        end

endmodule
```

Mux 2X1

Mux 2x1 without inverter

```
`timescale 1ns/1ps

module mux2X1  (

 input   wire                  Function_mode,
 input   wire                  Test_mode,
 input   wire                  sel,
 output  reg                   Mux_Out

     );

     always @(*)
     begin
        if (sel)
           begin      Mux_Out = Test_mode;
            end
         else
         begin        Mux_Out = Function_mode;
         end
      end
     endmodule
```

Mux 2x1 with inverter

```
`timescale 1ns/1ps

module mux2X1  (

 input   wire                  Function_mode,
 input   wire                  Test_mode,
 input   wire                  sel,
 output  reg                   Mux_Out

     );

     always @(*)
     begin
        if (sel)
           begin      Mux_Out = !Test_mode;
            end
         else
         begin        Mux_Out = Function_mode;
         end
      end
     endmodule
```

Synthesis Script

```
set path_lib /tools/PDK/install/SAED_PDK32nm/memory/lib/stdcell_hvt/db_ccs/
set RTL_path /home/svacddf202201yofattah/YoussefNasser/DFT_test_1
lappend search_path $path_lib

set MLIB "saed32hvt_ss0p7vn40c.db"

set target_library [list $MLIB]
set link_library [list * $MLIB]

set file_format verilog

read_file -format $file_format $RTL_path/TOP_SYSTEM_DFT.v
read_file -format $file_format $RTL_path/TOP_Domain_1_DFT.v
read_file -format $file_format $RTL_path/register_file.v
read_file -format $file_format $RTL_path/spi_slave.v
read_file -format $file_format $RTL_path/TOP_DFT.v
read_file -format $file_format $RTL_path/SER4.v
read_file -format $file_format $RTL_path/CRC.v
read_file -format $file_format $RTL_path/PISO.v
read_file -format $file_format $RTL_path/FCS.v
read_file -format $file_format $RTL_path/BIT_SYNC.v
read_file -format $file_format $RTL_path/Reset_Sync1.v
read_file -format $file_format $RTL_path/Mux2x1.v
read_file -format $file_format $RTL_path/Mux_ve_2x1.v

current_design TOP_SYSTEM_DFT
link

check_design

set_units -time ns
set_units -capacitance ff

set_wire_load_model -name "16000" -library saed32hvt_ss0p7vn40c

set_driving_cell -lib_cell IBUFFX4_HVT -library saed32hvt_ss0p7vn40c -pin Y
[get_ports MOSI]
set_driving_cell -lib_cell IBUFFX4_HVT -library saed32hvt_ss0p7vn40c -pin Y
[get_ports csn]

create_clock -name spi_clock -period 38.46 -waveform {0 19.23} [get_ports
spi_clock]
create_clock -name spi_clock -period 31.25 -waveform {0 15.625} [get_ports
sys_clk]

set_clock_uncertainty -setup 0.05 [get_clocks spi_clock]
set_clock_uncertainty -setup 0.05 [get_clocks sys_clk]

set_clock_transistion 0.5 -fall max {spi_clock sys_clk}
set_clock_transistion 0.5 -rise max {spi_clock sys_clk}
set_clock_groups -asynchronous -group [get_clocks {spi_clock}] \
-group [get_clocks {sys_clk}]

set_multicycle_path -setup 3 -from spi_clock -to sys_clk
set_multicycle_path -hold 2 -from spi_clock -to sys_clk
```

```
create_clock -name scan_spi_clock -period 100000 -waveform {0 50000}
[get_ports scan_spi_clock]
create_clock -name scan_sys_clock -period 84000 -waveform {0 42000}
[get_ports scan_sys_clock]

set_clock_uncertainty -setup 0.05 [get_clocks scan_spi_clock]
set_clock_uncertainty -setup 0.05 [get_clocks scan_sys_clock]

set_input_delay -clock scan_spi_clock -max 250 [get_ports Scan_mode]


set_input_delay -clock spi_clock -max 7.69 [get_ports MOSI]
set_input_delay -clock spi_clock -max 7.69 [get_ports csn]

set_output_delay -clock spi_clock -max 7.69 [get_ports o_miso_ena]
set_output_delay -clock spi_clock -max 7.69 [get_ports MISO]
set_output_delay -clock sys_clk -max 6.25 [get_ports packet]

group_path -name INREG -from [all_inputs]
group_path -name REGOUT -to [all_outputs]
group_path -name INOUT -from [all_inputs] -to [all_outputs]

set_max_fanout 32 [all_inputs]
set_max_capacitance 0.15 [all_outputs]

set_load -max 0.15 [get_ports MISO]
set_load -max 0.15 [get_ports packet]
set_load -max 0.15 [get_ports o_miso_ena]

set_max_area 0

compile_enable_register_merging false

set_case_analysis 1 [get_ports Scan_mode]

compile_ultra -no_autoungroup -gate_clock


set_svf "My_System.svf"

set path2 /home/svacddf202201yofattah/YoussefNasser/DFT_test_1/reports

write_file -format verilog -hierarchy -output $path2/GLS/TOP_SYSTEM_Netlist.v
write_file -format ddc -hierarchy -output $path2/TOP_SYSTEM_ddc.ddc
write_sdc -nosplit $path2/TOP_SYSTEM_sdf.sdf
write_sdc -nosplit $path2/TOP_SYSTEM_sdgc.sdc
write_sdc -nosplit $path2/TOP_SYSTEM_sdgc.sdgc

report_area -hierarchy > $path2/area_final.rpt
report_power -hierarchy > $path2/power.rpt
report_timing -max_paths 100 -delay_type min > $path2/hold_final.rpt
report_timing -max_paths 99 -delay_type max > $path2/setup.rpt
report_timing > $path2/Timing_report_final.rpt
report_clock -attributes > $path2/clocks_final.rpt
report_constraint -all_violators > $path2/constraint_final.rpt
```

DFT Script

```
set top_module TOP_SYSTEM_DFT

set path_lib /tools/PDK/install/SAED_PDK32nm/memory/lib_stdcell_hvt/db_ccs/
set RTL_path /home/svacddf202201yofattah/YoussefNasser/DFT_test_1

set MLIB "saed32hvt_ss0p7vn40c.db"

set target_library [list $MLIB]
set link_library [list * $MLIB]

set file_format verilog

read_file -format $file_format $RTL_path/TOP_SYSTEM_DFT.v
read_file -format $file_format $RTL_path/TOP_Domain_1_DFT.v
read_file -format $file_format $RTL_path/register_file.v
read_file -format $file_format $RTL_path/spi_slave.v
read_file -format $file_format $RTL_path/TOP_DFT.v
read_file -format $file_format $RTL_path/SER4.v
read_file -format $file_format $RTL_path/CRC.v
read_file -format $file_format $RTL_path/PISO.v
read_file -format $file_format $RTL_path/FCS.v
read_file -format $file_format $RTL_path/BIT_SYNC.v
read_file -format $file_format $RTL_path/Reset_Sync1.v
read_file -format $file_format $RTL_path/Mux2x1.v
read_file -format $file_format $RTL_path/Mux_ve_2x1.v

current_design TOP_SYSTEM_DFT
link

check_design

source $RTL_path/My_Syn.tcl

compile -scan

set_scan_configuration -clock_mixing no_mix -style multiplexed_flip_flop -
replace ture \
-add_lockup true -reuse_mv_cells true -count_per_domain 1

set compile_fix_multiple_port_nets "tue"

set_clock_gating_style -control_point before -control_signal scan_enable

set test_default_period 100
set test_default_delay 0
set test_default_bidir_delay 0
set test_default_strobe 20
set test_default_strobe_width 0
```

```
set_dft_signal -view existing_dft -type ScanMasterClock -timing {30 60} -port
[get_ports scan_sys_clock]
set_dft_signal -view existing_dft -type Reset -active_state 0 -port
[get_ports Scan_set_reset_2]
set_dft_signal -view existing_dft -type ScanMasterClock -timing {30 60} -port
[get_ports scan_spi_clock]
set_dft_signal -view existing_dft -type Reset -active_state 0 -port
[get_ports i_ioring_rst_n]
set_dft_signal -view existing_dft -type constant -active_state 1 -port
[get_ports Scan_mode]

set_dft_signal -view spec -type TestMode -active_state 1 -port [get_ports
Scan_mode]
set_dft_signal -view spec -type Scan_Enable -active_state 1 -port [get_ports
scan_shift_enable]
set_dft_signal -view spec -type Scan_DataIn   -port [get_ports scan_input_1]
set_dft_signal -view spec -type Scan_DataOut  -port [get_ports scan_output_1]
set_dft_signal -view spec -type Scan_DataIn   -port [get_ports scan_input_2]
set_dft_signal -view spec -type Scan_DataOut  -port [get_ports scan_output_2]
set_dft_signal -view spec -type Scan_Enable -usage clock_gating -port
[get_ports scan_shift_cg]

gui_start

create_test_protocol

dft_drc -verbose

preview_dft -show scan_summary

insert_dft

compile  -scan -incremental

dft_drc -verbose -coverage_estimate

write_test_protocol -output System_Tetra_Max_final_isa.spf
write_test_protocol -output System_Tetra_Max_final_isa.stil

write_file -format verilog -hierarchy -output sys_dft_final.v
```

Reports of DFT

```
    Uncollapsed Stuck Fault Summary Report
-------------------------------------------------
fault class                     code   #faults
--------------------------------  ----  ---------
Detected                         DT       6429
Possibly detected                PT          0
Undetectable                     UD          9
ATPG untestable                  AU         75
Not detected                     ND          3
-------------------------------------------------
total faults                                6516
test coverage                              98.80%
-------------------------------------------------
```

Report 15 DFT report | DC

```
Report : clocks
Design : TOP_SYSTEM_DFT
Version: P-2019.03-SP3
Date   : Tue May 23 14:40:44 2023
****************************************

Attributes:
    d - dont_touch_network
    f - fix_hold
    p - propagated_clock
    G - generated_clock
    g - lib_generated_clock

Clock          Period   Waveform            Attrs     Sources
------------------------------------------------------------------------------
scan_spi_clock
              100000.00 {0 50000}                     {scan_spi_clock}
scan_sys_clock
              84000.00  {0 42000}                     {scan_sys_clock}
spi_clock       38.46   {0 19.23}                     {spi_clock}
sys_clk         31.25   {0 15.625}                    {sys_clk}
------------------------------------------------------------------------------
1
```

Report 16 DFT Clocks Report | DC

```
****************************************
Report : constraint
         -all_violators
Design : TOP_SYSTEM_DFT
Version: P-2019.03-SP3
Date   : Tue May 23 14:40:44 2023
****************************************


   max_area

                            Required        Actual
   Design                     Area            Area           Slack
   -------------------------------------------------------------------
   TOP_SYSTEM_DFT              0.00         3385.61       -3385.61  (VIOLATED)


1
```

Report 17 DFT All Constraints Violators Report | DC

We need `System_Tetra_Max_final_isa.spf and System_Tetra_Max_final_isa.stil to start the Tetra Max`

```
TBH: there is a violation C26 for a signal in my Register file
but I have searched it does not affect the process.
```

Chapter 10 Tetra-Max +BONUS

TetraMax is a leading automatic test pattern generation (ATPG) tool developed by Synopsys, a well-known electronic design automation (EDA) company. ATPG is a crucial step in the semiconductor manufacturing process that involves generating test patterns to ensure the functionality and reliability of integrated circuits (ICs) or chips.

TetraMax is designed to address the challenges of testing modern complex ICs, including the ever-increasing design size, advanced process technologies, and power-related issues. It helps semiconductor companies and chip designers achieve high test quality and maximize the manufacturing yield by detecting and diagnosing faults or defects in ICs.

Flow

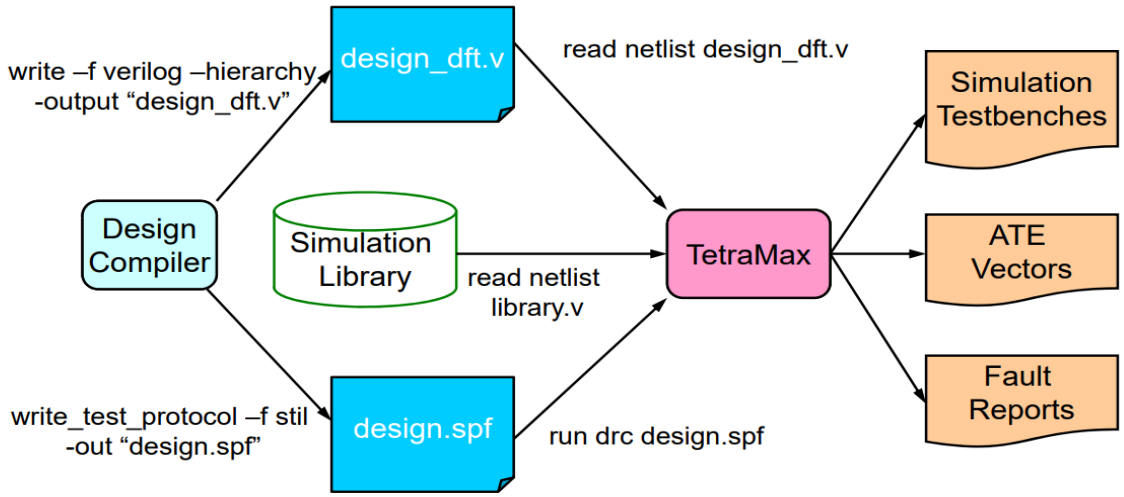


Figure 18 TetraMax flow

Script

```
##ATPG##
#tmax -tcl ATPG_script.tcl
##path
set RTL_PATH /home/svacddf202201yofattah/YoussefNasser/DFT_test_1/TetraMax
##read_Netlist files##
read_netlist $RTL_PATH/sys_dft_final.v
read_netlist $RTL_PATH/saed32nm_hvt.v
##build_model###
run_build_model TOP_SYSTEM_DFT
##DRC_Setup##
set_drc $RTL_PATH/System_Tetra_Max_final_isa.spf
run_drc
##Test##
add_faults -module TOP_SYSTEM_DFT
run_atpg -auto_compression basic_scan_only

report_faults -sum -uncoll
report_faults -sum -coll
```

Tetra-Max Reports

```
report_faults -sum -coll
      Collapsed Stuck Fault Summary Report
 ------------------------------------------------
 fault class                    code   #faults
 -----------------------------  ----  ---------
 Detected                       DT        4451
 Possibly detected              PT           0
 Undetectable                   UD           6
 ATPG untestable                AU          56
 Not detected                   ND           0
 ------------------------------------------------
 total faults                              4513
 test coverage                            98.76%
 ------------------------------------------------
 #internal patterns                          72
     #basic_scan patterns                    72
 ------------------------------------------------
            CPU Usage Summary Report
 ------------------------------------------------
 Total CPU time                           0.05
 ------------------------------------------------
report_faults -sum -uncoll
     Uncollapsed Stuck Fault Summary Report
 ------------------------------------------------
 fault class                    code   #faults
 -----------------------------  ----  ---------
 Detected                       DT        6374
 Possibly detected              PT           0
 Undetectable                   UD           9
 ATPG untestable                AU          63
 Not detected                   ND           0
 ------------------------------------------------
 total faults                              6446
 test coverage                            99.02%
 ------------------------------------------------
 ATPG performed for stuck fault model using internal pattern source.
 ------------------------------------------------------------
 #patterns      #faults      #ATPG faults  test       process
 stored      detect/active  red/au/abort  coverage   CPU time
 ---------   -------------  ------------  ---------  --------
 Begin deterministic ATPG: #uncollapsed_faults=4161, abort_limit=10...
 32            3978    183          0/0/0     96.18%       0.04
 64             166     17          0/0/0     98.76%       0.05
 72              17      0          0/0/0     99.02%       0.05

     Uncollapsed Stuck Fault Summary Report
 ------------------------------------------------
 fault class                    code   #faults
 -----------------------------  ----  ---------
 Detected                       DT        6374
 Possibly detected              PT           0
 Undetectable                   UD           9
 ATPG untestable                AU          63
 Not detected                   ND           0
 ------------------------------------------------
 total faults                              6446
 test coverage                            99.02%
 ------------------------------------------------
            Pattern Summary Report
 ------------------------------------------------
```

Report 18 TetraMax Report | tmax

Chapter 11 UPF

UPF stands for Unified Power Format. It is a widely used industry-standard format for specifying and managing power-related intent in electronic designs, specifically in digital integrated circuits (ICs). UPF is primarily used during the design and verification stages of IC development to ensure efficient power management and control.

The main purpose of UPF is to describe the power-saving techniques and strategies that are implemented in a design, including power domains, power states, power control signals, and power supply networks. It provides a standardized representation of power intent, allowing designers to specify how different parts of the design should be powered and controlled to optimize power consumption.

Flow

1. RTL + UPF constraints, you can run it on icc-shell in order to make sure of the functionality
2. Synthesis on dc-shell
3. Netlist + UPF
4. PnR
5. Netlist + UPF
6. Analysis

As we are front-end digital IC design engineers we care only about the first 2 points, but actually the icc-shell does not work on the academy server so I have done the second point only.

Task

Implement UPF taking into consideration that there are 2 power domains, sleep and active, and that you have to isolate any signals going from the active domain to sleep one through your top level.

The only signal from the active domain to the sleep domain is the done tick.

Pre-UPF steps

First in synthesis don’t forget before compiling to run these commands

- set upf_levshi_on_constraint_only true
  #“in order not to define voltage level shifters automatically”
- load_upf MY_UPFF.upf
- set_voltage 0.7 -object_list vdd_dsleep_top
- set_volatage 0.7 -object_list vdd_active_top
- set_voltage 0 -object_list vss_top

Also don’t forget to define the isolation control signal in the TOP_SYSTEM_DFT and connect it to the power pin of the Register File.

Script

```
#UPF#
set_design_attributes -elements {.} -attribute
conservative_diff_supply_only_isolation true
set_design_attributes -elements {.} -attribute
enable_state_propagation_in_add_power_state TRUE
set_design_attributes -elements {.} -attribute correlated_supply_group "*"

set upf_create_implicit_supply_sets true

#deep sleep domain
create_power_domain PD_dpsleep -include_scope

#active domain
create_power_domain PD_active -elements {TOP_CRC_SER_Insta}

#.................................create supply ports and
nets ..............................#
#ports
create_supply_port vdd_dsleep_top
create_supply_port vdd_active_top
create_supply_port vss_top

#nets
create_supply_net vdd_dsleep_top
create_supply_net vdd_active_top
create_supply_net vss_top

#to connect top level power domain supply ports to supply nets
connect_supply_net vdd_dsleep_top  -ports  vdd_dsleep_top
connect_supply_net vdd_active_top  -ports  vdd_active_top
connect_supply_net vss_top  -ports  vss_top


#set primary supplies for each power domain
create_supply_set PD_dpsleep.primary \
-function {power vdd_dsleep_top} \
-function {ground vss_top}
-update

create_supply_set PD_dpsleep.default_isolation \
-function {power vdd_dsleep_top} \
-function {ground vss_top}
-update

create_supply_set PD_active.primary \
-function {power vdd_active_top} \
-function {ground vss_top}
-update


associate_supply_set PD_dpsleep.primary -handle PD_active.default_isolation
```

```
#..............isolation strategies ……….....#

set_isolation active_iso_0 -isolation_supply_set PD_active.primary -
domainPD_active - -clamp_value 0 -applies_to outputs -diff_supply_only true
set_isolation_control active_iso_0 -domain PD_active  -isolation_sigal
o_pm_sleep_iso_and  -isolation_sense low -location parent

set_isolation active_iso_1  -isolation_supply_set PD_active.primary -domain
PD_active -clamp_value 1  -diff_supply_only true -applies_to_outputs  -
elements {BIT_SYNC_done_tick/SYNC}


set_isolation_control active_iso_1 -domain PD_active  -isolation_sigal
o_pm_sleep_iso_and  -isolation_sense low -location parent


#.................create power state table and add power states ..........#

add_power_state PD_dpsleep.primary  -state VDD_ALWAYS_ON_STATE {-supply_expr
{power == `{FULL_ON, 0.4, 0.5, 0.7}}}
add_power_state PD_dpsleep.primary  -state VSS_ON_STATE        {-supply_expr
{ground == `{FULL_ON, 0}}}

add_power_state PD_active.primary  -state ACTIVE_ON_STATE      {-supply_expr
{power == `{FULL_ON, 0.4, 0.7, 0.7}}}
add_power_state PD_active.primary  -state ACTIVE_OFF_STATE     {-supply_expr
{power == `{OFF}}}


create_pst top_pst -supplies [list PD_dpsleep.primary.power
PD_active.primary.power  PD_dpsleep.primary.ground]
add_pst_state ALWAYS_ON  -pst top_pst -state {VDD_ALWAYS_ON_STATE
ACTIVE_OFF_STATE   VSS_ON_STATE}
add_pst_state ACTIVE  -pst top_pst -state {VDD_ALWAYS_ON_STATE
ACTIVE_ON_STATE   VSS_ON_STATE}
```

UPF diagram

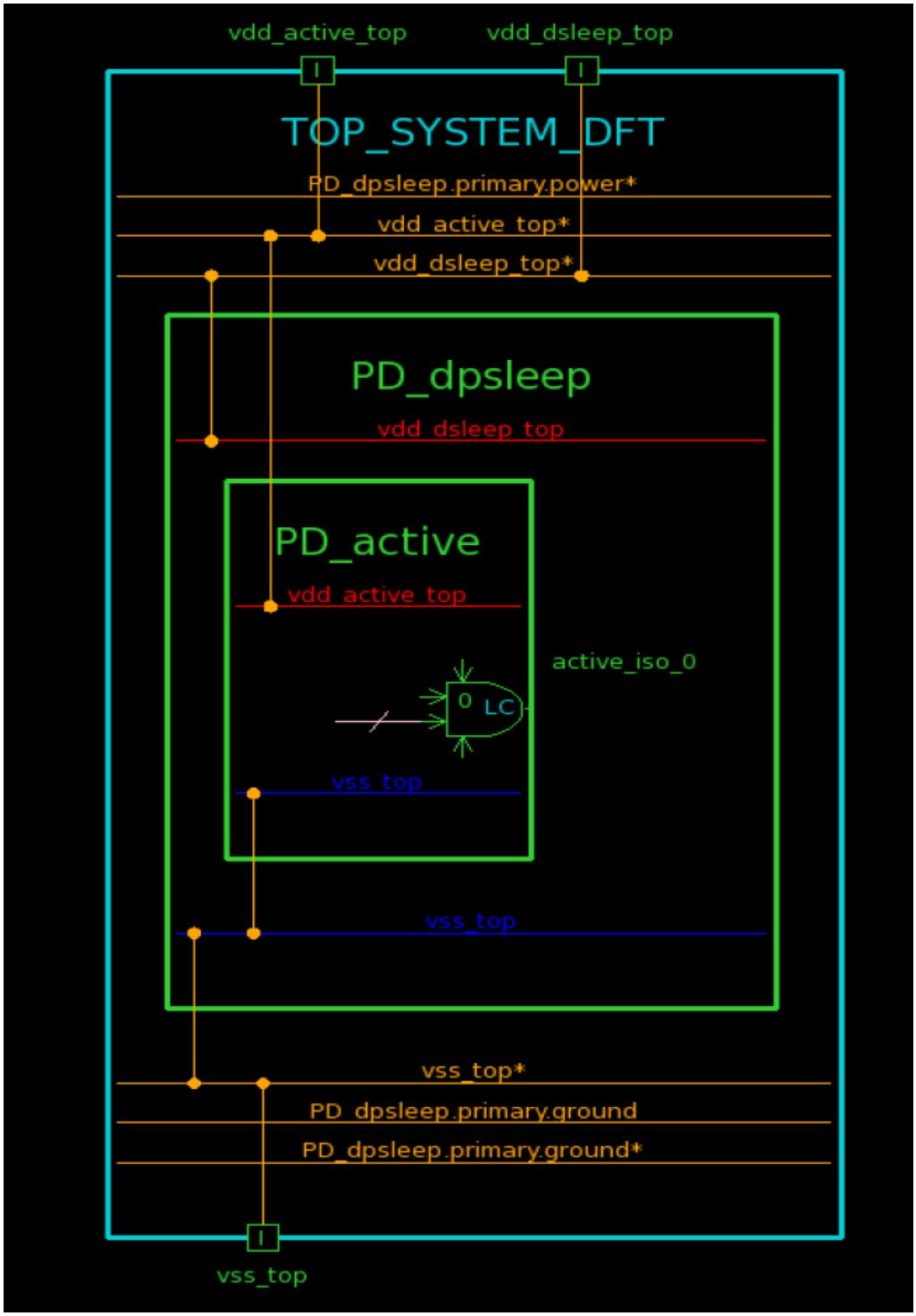


Figure 19 UPF diagram

Chapter 13 Problems That I Faced

- How to write a clean RTL
  - It was solved by Spyglass
- How to deal with two clock domains, it was the first time to face CDC and RDC
  - CDC was solved by putting double FFs for the signals that cross the domains
  - RDC was solved by putting two reset sync because we have two clock domains
- How to implement a clock divider without using the clock divider module
  - It was solved by making a counter and the shift register is just checking on it, not on the clock edge
- How to choose a cell in the library
  - It was solved by reading the data sheet of the library
- How to deal with Design Compiler
  - It was solved by reading the Design Compiler manual
- How to write Constrains of Spyglass without having its help
  - It was solved by using the add-constrains option in Spyglass
- How to deal with a design which is working with a double clock edge in DFT
  - It was solved by using an inverter to the negative edge clock
- How to run Formality without errors
  - It was solved by getting the SVF file from Design Compiler
- How to run Gate Level Simulation
  - It was solved by searching for the library Verilog file
- How to use Tetra-Max
  - It was solved by reading Tetra-Max user guide
- How to write Test bench
  - To be honest, am not good at writing test bench, am a beginner
- How to validate the UPF
  - I can’t make sure of my work because there is no way to run it without ICC shell and it was not available on the server

Chapter 12 Conclusion

In conclusion, the project consists of two domains. The first domain deals with a system clock of 26 MHz and comprises an SPI slave, register file, and other related components. The second domain operates on a system clock of 32 MHz and involves CRC calculations and serialization. The serializer operates in two modes: the first mode serializes with a clock frequency of 2 MHz, derived from the system clock, and the second mode operates at 1MHz frequency.

This low-power SPI project holds significant importance in real-life applications such as IoT and Edge Devices. Lastly, I am proud to have successfully navigated the digital IC design flow, from RTL (Register Transfer Level) to UPF (Unified Power Format).

Chapter 14 Future Work

In the future, I plan to develop a controller to oversee the entire system and make it more efficient. Additionally, I aim to modify the serializer to transmit data in 8-bit segments instead of adhering strictly to the current design requirements. This modification is necessary as the current approach requires a substantial number of registers to transmit a single frame.

It is worth noting that I also intend to create a receiver component. However, in order to properly detect the frequency (1MHz or 2MHz), the receiver should receive the preamble, header, payload, and perform CRC checks in the correct order, the transmission bits should be sent from the least significant bit rather than the most significant bit.

It is important to mention that the mentor who participated in our initial session suggested sending data from the most significant bit to the least significant bit, but this approach does not align with the logical requirements of the Rx.

Chapter 15 References

Si-vision Lectuers

Design Compiler Manual

DFT user guide

tmax rules

Appendix A

Appendix A contains the Blocks / Schematics for TOP SYSTEM .

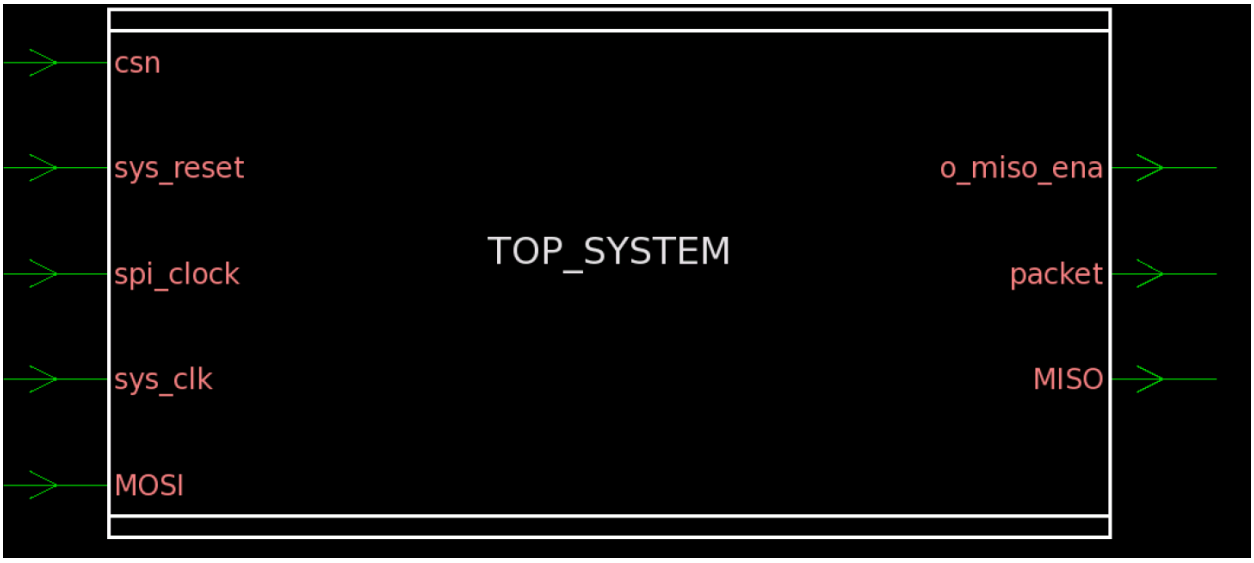


Figure 20 TOP SYSTEM Block

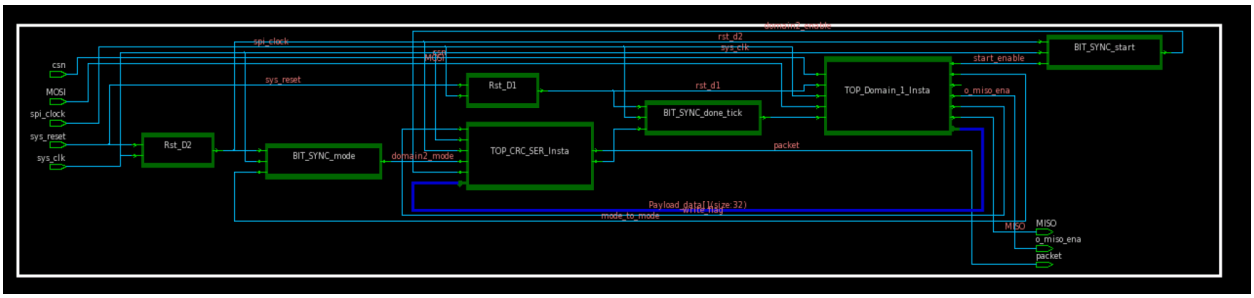


Figure 21 TOP SYSTEM Schematic

Figure 23 TOP SYSTEM WITH DFT Block

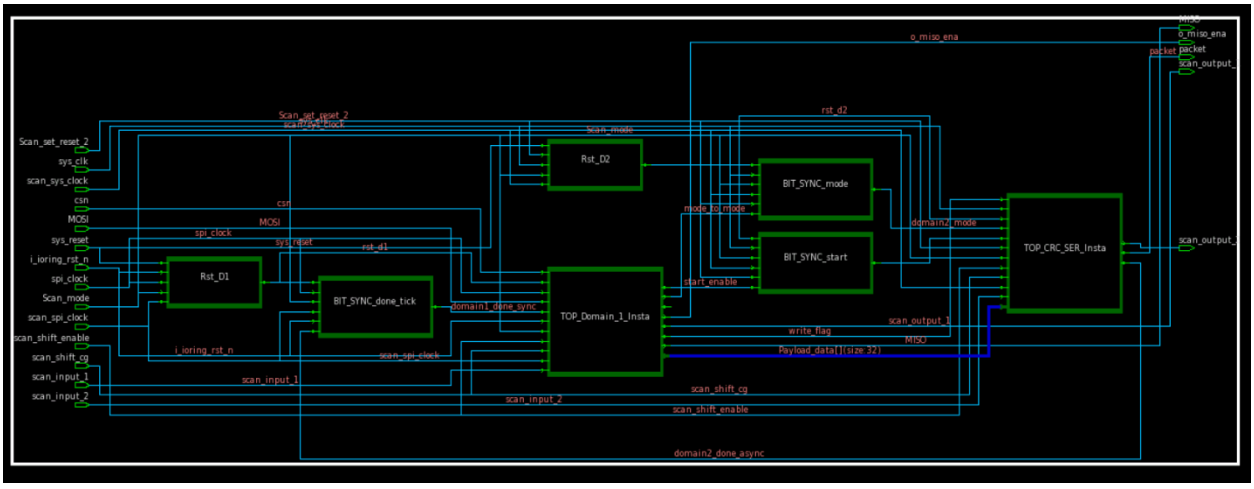


Figure 22 TOP SYSTEM WITH DFT Schematic

Appendix B

```
`timescale 1ns/1ps


module Whole_Moudle_TestBench1
#(parameter WIDTH = 8 )
 ();
reg csn_tb;
reg sys_reset_tb;
reg spi_clock_tb; // 26 MHz of the first Domain
reg sys_clk_tb; // 32 MHz Of the Second Domain
//reg sck_neg_clock_tb;
reg mosi_tb;
wire        packet_tb;
wire MISO_tb;
wire o_miso_en_tb;
parameter T_SPI = 38.462; // SPI CLK 26 MHz 1000/26
parameter T_Sys = 31.25; // Sys CLK 32 MHz 1000/32

//TOP BLOCK module
TOP_SYSTEM Slave_Reg_Sync(

.csn(csn_tb),
.spi_clock(spi_clock_tb),
.MOSI(mosi_tb),
.sys_reset(sys_reset_tb),
.sys_clk(sys_clk_tb),


. o_miso_en(o_miso_en_tb),
.packet(packet_tb),
.MISO(MISO_tb)
);
initial begin
//Initialization
csn_tb=1;
spi_clock_tb=0;
//sck_neg_clock_tb=0;
sys_clk_tb=0;
mosi_tb=0;

sys_reset_tb = 1'b0;
#(T_Sys/2);
sys_reset_tb = 1'b1;

csn_tb=0;
mosi_tb=0;


fork

begin // MOSI data from SPI Master which is our Test Bench
#(T_SPI/2);

    //Command = 8'b0000_0010;
```

```
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;

        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 1;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;


//Address=8'h00;

        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;

        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;

        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;

       //Data = 8'hAA;
        mosi_tb = 1;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
        mosi_tb = 1;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;

        mosi_tb = 1;
        #T_SPI;
        mosi_tb = 0;
        #T_SPI;
```

```
mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;

   //Data = 8'hAB;

mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 1;
#T_SPI;
mosi_tb = 1;
#T_SPI;
//Data = 8'hAC;

mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 1;
#T_SPI;
mosi_tb = 1;
#T_SPI;

mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;

//Data = 8'hAD;

mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 1;
#T_SPI;
mosi_tb = 0;
#T_SPI;
```

```
mosi_tb = 1;
#T_SPI;
mosi_tb = 1;
#T_SPI;

mosi_tb = 0;
#T_SPI;
mosi_tb = 1;
#T_SPI;

// start [4]


mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 0;
#T_SPI;
mosi_tb = 1;
#T_SPI;

//mode[5]

 mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;

mosi_tb = 0;
#T_SPI;
mosi_tb = 0;
#T_SPI;

// power [6]
```

```
      mosi_tb = 0;
     #T_SPI;
     mosi_tb = 0;
     #T_SPI;
     mosi_tb = 0;
     #T_SPI;
     mosi_tb = 0;
     #T_SPI;

     mosi_tb = 0;
     #T_SPI;
     mosi_tb = 0;
     #T_SPI;

     mosi_tb = 0;
     #T_SPI;
     mosi_tb = 0;
     #T_SPI;
end
begin // SPI_CLOCK Generator
csn_tb = 0;
spi_clock_tb=1'b0;
#T_SPI;
repeat(142)
 begin
spi_clock_tb=1'b1;
#(T_SPI*0.5);
spi_clock_tb=1'b0;
#(T_SPI*0.5);
end
csn_tb = 1;
spi_clock_tb=1'b0;
end
begin // System_Clock Generator
sys_clk_tb=1'b0;
repeat(3000)
 begin
sys_clk_tb=1'b1;
#(T_Sys*0.5);
sys_clk_tb=1'b0;
#(T_SPI*0.5);
end
end
join
$stop;
end
endmodule
```

Appendix C

In this appendix, I will include all the commands I use to execute scripts. Additionally, I will provide all the necessary paths and a Google Drive link that contains all the RTL and scripts.

Go to this path /home/svacddf202201yofattah/YoussefNasser

Folder	Description	Process
RTL_Data_quasi	• Contains RTL of design without DFT • Contains SPYGLASS (Prj1 for Linting , Prj 2 For CDC) • Formality Script • Contains Synthesis Script • Contains reports of Spyglass and DC reports • Contains items of Gate level simulation in reports/GLS • Contains test bench for the system R/W also contains Tb for TOP domain 1	READ AND RUN
DFT_test_1	• Contains DFT script • Contains Synthesis Script Test bench for DFT • Contains Tetra Max script in TetraMax folder	READ AND RUN
UPF	• Contains UPF script • Contains Synthesis Script	READ AND RUN
GP	Contains all RTL , Scripts and Reports	Read Only!!!

To run test bench of the TOP system, write operation please go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi

Command:

vcs -v2005 -files CRC.v FCS.v BIT_SYNC.v Reset_Sync.v PISO.v register_file.v SER4.v spi_slave.v TOP.v TOP_Domain_1.v TOP_SYSTEM.v TOP_SYSTEM_tb.v -R -gui

To run test bench of the TOP system, write + read operation please go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi

Command :

vcs -v2005 -files CRC.v FCS.v BIT_SYNC.v Reset_Sync.v PISO.v register_file.v SER4.v spi_slave.v TOP.v TOP_Domain_1.v TOP_SYSTEM.v TOP_SYSTEM_tb1.v -R -gui

To run test bench of the top domain 1 , please go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/domain_1_under_test

Command :

vcs -v2005 -files register_file.v spi_slave.v TOP.v TOP_Domain_1.v TOP_SYSTEM_tb.v -R -gui

To Run gate level simulation , please go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/reports/GLS

Command :

vcs -v2005 -files saed32nm_hvt.v TOP_SYSTEM_Netlist.v TOP_SYSTEM_tb.v -R -gui

To Run Test bench of DFT go to this path , run this command

Path : /home/svacddf202201yofattah/YoussefNasser/DFT_test_1

Command :

Vcs -v2005 -files CRC.v FCS.v BIT_SYNC.v Reset_Sync1.v PISO.v register_file.v SER4.v spi_slave.v TOP_DFT.v TOP_Domain_1_DFT.v TOP_SYSTEM_DFT.v TOP_SYSTEM_tb.v -R -gui

To Find Spyglass projects , project 1 for Linting only , Project 2 for CDC only , go to this path

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi

To find the needed spyglass reports , constrains and .AWL file go to this path

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/SpyGlass_Reports

Also to read the summarized AWL , you can refer to Spyglass section and find table 3

To run Synthesis for the Design without DFT , go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi

Command :

dc_shell -f My_Syn.tcl

to find the needed Synthesis reports Area/Power go to this path

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/DC_reports

Or refer to Synthesis section

To run Formality , go this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/RTL_Data_quasi/

Graduation Project

Command :

formality -f My_Formality.tcl

To Run DFT go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/DFT_test_1

Command :

dc_shell -f My_DFT.tcl

to find Synthesis Reports of DFT , go to this path

Path : /home/svacddf202201yofattah/YoussefNasser/DFT_test_1/reports

To Run TetraMax go to this path and run this command

Path : /home/svacddf202201yofattah/YoussefNasser/DFT_test_1/TetraMax

Command :

tmax -tcl ATPG_script.tcl

to find Tetra Max reports , refer to TetraMax section

To Run UPF , go to this path and follow steps

Path : /home/svacddf202201yofattah/YoussefNasser/UPF

dc_shell -f My_Syn.tcl

steps , go to power , show power diagram , press ok . or refer to UPF section

To Read RTL and Scripts .. read only not running them go to this path

Path : /home/svacddf202201yofattah/YoussefNasser/GP

To Read All RTL and Scripts on Windows OS

Go to this google drive link
https://drive.google.com/drive/folders/1nsgtEb73WgugMRRlEZxgx15KiBESA9IQ?usp=share_link

In case that you can not understand anything or can not reach to any file please contact me on my email or whatsapp

Email : youssefnasserabdelaal@gmail.com

Whatsapp : +201279444247

Thanks for reading.