

SPI_PROJECT

(SPI SLAVE WITH SINGLE PORT RAM)

Submitted By:

- Mahmoud Badry Mahmoud Ahmed
- Ahmed Khaled Marzouk
- Ismail Galal Mohamed

Submitted To:

Eng. Kareem Wessam

Contents

Submitted By:.....	1
o Mahmoud Badry Mahmoud Ahmed	1
o Ahmed Khaled Marzouk	1
o Ismail Galal Mohamed	1
Submitted To:	1
1. Design Flow	3
a. RAM Code	3
b. SPI Code	4
c. Wrapper Code	8
2. Verification Flow	8
a. Verification plan	8
o Reset functionality	9
o Write address state (11001100 location).....	9
o End communication (SS_n =1)	9
o Write data state (11001100 data written)	9
o End communication (SS_n =1)	9
o Read address state (11001100 location same address of write state)	9
o End communication (SS_n =1)	9
o Read data state (11001100 are data readen which was same data have written)	9
b. Test bench code	9
c. Waveforms	13
3. Synthesis and Implementation Flow	17
a. Constrain file.....	17
b. Synthesis Flow for encodings used	18
- Gray Encoding	18
- One_Hot Encoding	20
- Sequential Encoding.....	22
- Debugging core added	30
- Messages Tab showing no critical warnings or errors	30

1. Design Flow

a. RAM Code

```
1  module ram (din,dout,rx,tx,clk,rst_n);
2      parameter MEM_DEPTH = 256;
3      parameter ADDER_SIZE =8;
4      input [ADDER_SIZE+1:0] din;
5      output reg [ADDER_SIZE-1:0] dout;
6      input rx,clk,rst_n;
7      output reg tx;
8      reg [ADDER_SIZE-1:0] mem [MEM_DEPTH-1:0];
9      reg [ADDER_SIZE-1:0] internal_addr;
10
11     always @(posedge clk ) begin
12         if (~rst_n)
13             dout <=0;
14         else begin
15             if (din[9:8] == 2'b00 && rx== 1)begin
16                 internal_addr <= din[7:0];
17                 tx =0;
18             end
19             else if (din[9:8] == 2'b10 && rx== 1)begin
20                 internal_addr <= din[7:0];
21                 tx =0;
22             end
23             else if (din[9:8] == 2'b01 && rx== 1)begin
```

```

24         mem[internal_addr] <= din[7:0];
25         tx = 0;
26     end
27     else if (din[9:8] == 2'b11 && rx== 1)begin
28         dout <= mem[internal_addr];
29         tx =1;
30     end
31     else
32         tx=0;
33     end
34 end
35 endmodule

```

b.SPI Code

```

1 // SPI implemented with Finite State Machine connected on the other side with a RAM
2 module SPI_FSM_module(MOSI,MISO,SS_n,clk,rst_n,rx_data,rx_valid,tx_data,tx_valid);
3     //States
4     parameter IDLE          = 3'b000;
5     parameter CHK_CMD       = 3'b001;
6     parameter WRITE         = 3'b010;
7     parameter READ_ADD      = 3'b011;
8     parameter READ_DATA     = 3'b100;
9     //Input Ports
10    input  MOSI,SS_n,clk,rst_n,tx_valid;
11    input  [7:0]tx_data;
12    //Output Ports
13    output reg[9:0]rx_data;
14    output reg MISO;
15    output reg rx_valid;
16
17    (* fsm_encoding = "gray" *)
18    //Internal signals
19    reg [2:0] cs,ns; //cs-> current state, ns-> next state
20    integer READ_FLAG = 0;
21
22    integer counter_convert = 10;
23

```

```
25  /*****Combinational -> Next State*****/
26  always@(*)begin
27      case (cs)
28          IDLE:
29              if(SS_n)
30                  ns = IDLE;
31              else
32                  ns = CHK_CMD;
33          CHK_CMD:
34              if(SS_n)
35                  ns = IDLE;
36              else begin
37                  if(MOSI)begin
38                      if(READ_FLAG == 0)
39                          ns = READ_ADD;
40                      else
41                          ns = READ_DATA;
42                  end
43                  else
44                      ns = WRITE;
45              end
46          WRITE:
47              if(SS_n)
```

```

48         ns = IDLE;
49     else
50         ns = WRITE;
51     READ_ADD:
52         if(SS_n)
53             ns = IDLE;
54         else
55             ns = READ_ADD;
56     READ_DATA:
57         if(SS_n)
58             ns = IDLE;
59         else
60             ns = READ_DATA;
61     default : ns = IDLE;
62 endcase
63 end
64

```

```

65  /*****Sequential -> State Memory , rst_n is synch*****/
66  always@(posedge clk)begin
67      if(!rst_n)begin
68          cs <= IDLE;
69          READ_FLAG <= 0;
70      end
71      else
72          cs <= ns;
73      end
74
75  /*****Sequential -> Output*****/
76  always@(posedge clk )begin
77      case (cs)
78          IDLE:begin
79              counter_convert <= 9;
80          end
81          CHK_CMD:begin rx_valid <= 0; //data is unavailable
82                      rx_data[counter_convert] <= MOSI; //convert serial to parallel (MSB)
83          end
84          WRITE:begin
85              rx_data[counter_convert-1] <= MOSI; //convert serial to parallel
86              counter_convert <= counter_convert - 1;
87          end
88      endcase
89  end
90

```

```

88         if ( counter_convert == 0)
89             rx_valid <= 1; //data is available
90         end
91         READ_ADD:begin
92
93             rx_data[counter_convert-1] <= MOSI; //convert serial to parallel
94             counter_convert <= counter_convert - 1;
95
96             if ( counter_convert == 0) begin
97                 rx_valid <= 1; //data is available
98                 READ_FLAG <= 1; //go to read data state
99             end
100         end

```

```

101     READ_DATA:begin
102         if(tx_valid)begin
103
104             MISO <= tx_data[counter_convert -1]; //convert parallel to serial
105             counter_convert <= counter_convert - 1;
106         end
107         else begin
108
109             rx_data[counter_convert-1] <= MOSI;
110             counter_convert <= counter_convert - 1;
111
112             if ( counter_convert == 0) begin
113                 rx_valid <= 1;
114                 READ_FLAG <= 0; //to read address state
115                 counter_convert <= 9; //Ram will take one clk cycle to raise tx_valid to 1, so counter = 9 not 8
116             end
117         end
118     end
119 endcase
120 end
121 endmodule

```

c. Wrapper Code

```
1 module Wrapper_module(MOSI,MISO,SS_n,clk,rst_n);
2     //Input output ports
3     input MOSI,SS_n,clk,rst_n;
4     output MISO;
5
6     //Internal signals
7     wire [9:0]rx_data;
8     wire rx_valid,tx_valid;
9     wire [7:0]tx_data;
10
11
12     //Instantiations modules
13     ram M1(rx_data,tx_data,rx_valid,tx_valid,clk,rst_n);
14     SPI_FSM_module M2(MOSI,MISO,SS_n,clk,rst_n,rx_data,rx_valid,tx_data
15         ,tx_valid);
16 endmodule
```

2. Verification Flow

a. Verification plan

Our verification plan is directed and since we didn't initialize the ram, we will verify the following cases:

- Reset functionality
- Write address state (11001100 location)
- End communication (SS_n =1)
- Write data state (11001100 data written)
- End communication (SS_n =1)
- Read address state (11001100 location same address of write state)
- End communication (SS_n =1)
- Read data state (11001100 are data readen which was same data have written)

b. Test bench code

```

1  module Wrapper_tb();
2      //signal declairation
3      reg MOSI,SS_n,clk,rst_n;
4      wire MISO;
5      integer i;
6      //Module instantiations
7      Wrapper_module DUT(MOSI,MISO,SS_n,clk,rst_n);
8
9      //clock generation
10     initial begin
11         clk = 0;
12         repeat (150)
13             #1 clk = ~clk;
14     end
15
16     //test stimulus generation
17     initial begin
18         rst_n = 0; //enable reset
19         SS_n = 0; //start communication
20         @(negedge clk);
21         rst_n = 1;
22         for(i = 0;i<50;i = i+1)begin
23             MOSI = 0;
24             SS_n = 0; //start communication (CHK state 1 clk)
25             @(negedge clk);

```

```

26      MOSI = 0;    //write addr state (10 clk)
27      @(negedge clk);
28      MOSI = 0;
29      @(negedge clk);
30      MOSI = 1;
31      @(negedge clk);
32      MOSI = 1;
33      @(negedge clk);
34      MOSI = 0;
35      @(negedge clk);
36      MOSI = 0;
37      @(negedge clk);
38      MOSI = 1;
39      @(negedge clk);
40      MOSI = 1;
41      @(negedge clk);
42      MOSI = 0;
43      @(negedge clk);
44      MOSI = 0;
45      @(negedge clk);
46      SS_n = 1;    //end communication (IDEL state)
47      @(negedge clk); $stop;
48
49      SS_n = 0;    //start communication (CHK state 1 clk)
50      MOSI = 0;

```

```

51      @(negedge clk);
52      MOSI = 0;    //write data state (10 clk)
53      @(negedge clk);
54      MOSI = 1;
55      @(negedge clk);
56      MOSI = 1;
57      @(negedge clk);
58      MOSI = 1;
59      @(negedge clk);
60      MOSI = 0;
61      @(negedge clk);
62      MOSI = 0;
63      @(negedge clk);
64      MOSI = 1;
65      @(negedge clk);
66      MOSI = 1;
67      @(negedge clk);
68      MOSI = 0;
69      @(negedge clk);
70      MOSI = 0;
71      @(negedge clk);
72      SS_n = 1;    //end communication
73      @(negedge clk); $stop;
74
75      SS_n = 0;    //start communication (CHK state 1 clk)

```

```

76     MOSI = 1;
77     @(negedge clk);
78     MOSI = 1; //read addr state (same location @ which we
              //have write 10 clk )
79     @(negedge clk);
80     MOSI = 0;
81     @(negedge clk);
82     MOSI = 1;
83     @(negedge clk);
84     MOSI = 1;
85     @(negedge clk);
86     MOSI = 0;
87     @(negedge clk);
88     MOSI = 0;
89     @(negedge clk);
90     MOSI = 1;
91     @(negedge clk);
92     MOSI = 1;
93     @(negedge clk);
94     MOSI = 0;
95     @(negedge clk);
96     MOSI = 0;
97     @(negedge clk);
98     SS_n = 1; //end communication (IDEL state)
99     @(negedge clk); $stop;

```

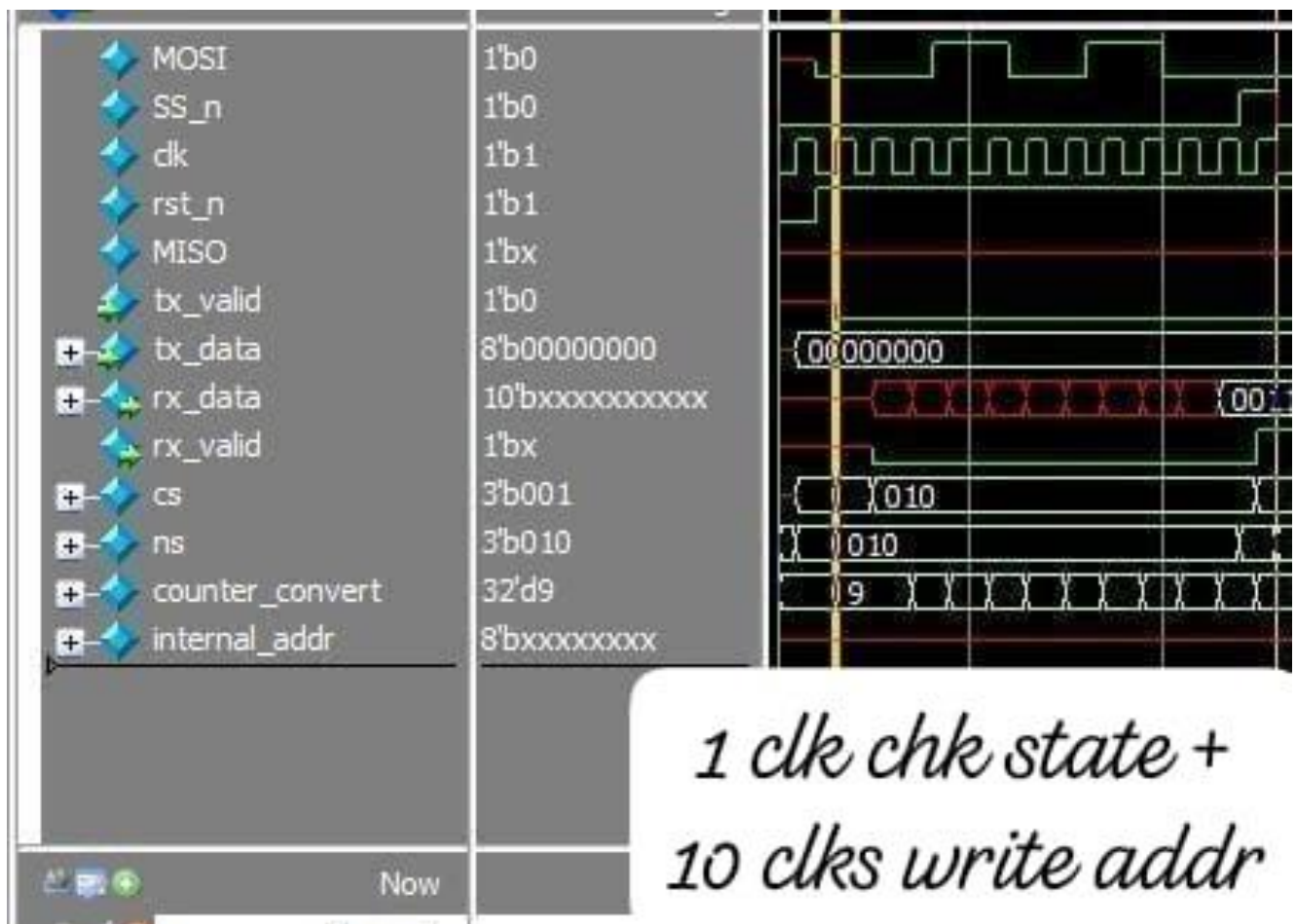
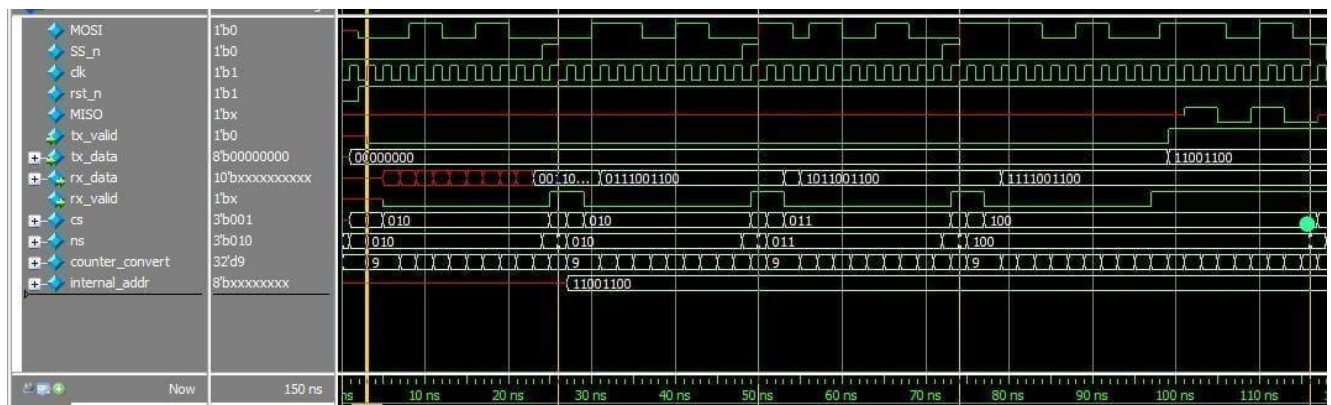
```

101
102     SS_n = 0; //start communication (CHK state 1 clk)
103     MOSI = 1;
104     @(negedge clk);
105     MOSI = 1; //read data state (same data which we have
              //write 10 clk write + 10 clk read)
106     @(negedge clk);
107     MOSI = 1;
108     @(negedge clk);
109     MOSI = 1;
110     @(negedge clk);
111     MOSI = 1;
112     @(negedge clk);
113     MOSI = 0;
114     @(negedge clk);
115     MOSI = 0;
116     @(negedge clk);
117     MOSI = 1;
118     @(negedge clk);
119     MOSI = 1;
120     @(negedge clk);
121     MOSI = 0;
122     @(negedge clk);
123     MOSI = 0;
124     @(negedge clk);
125     MOSI = 0;

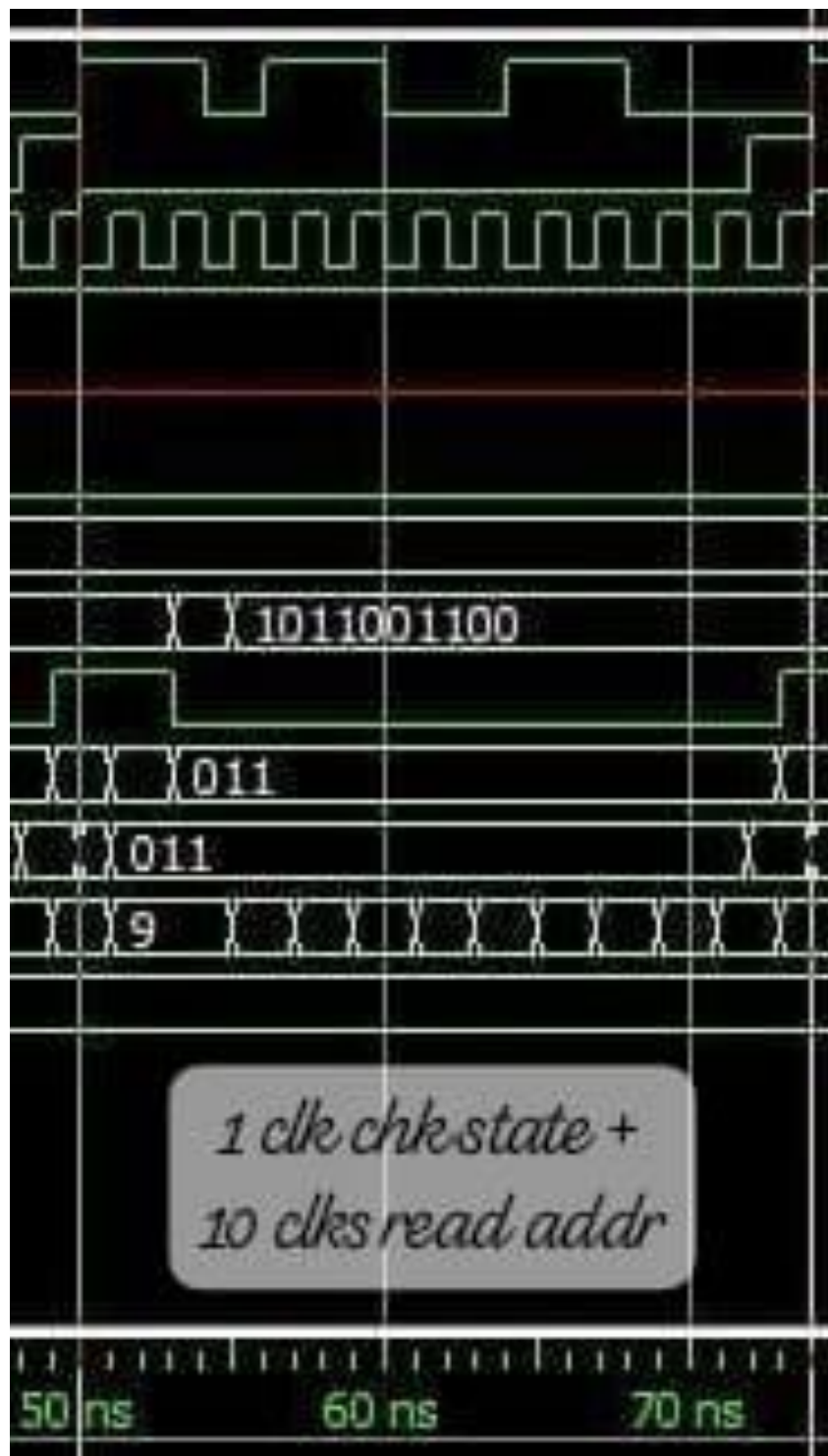
```

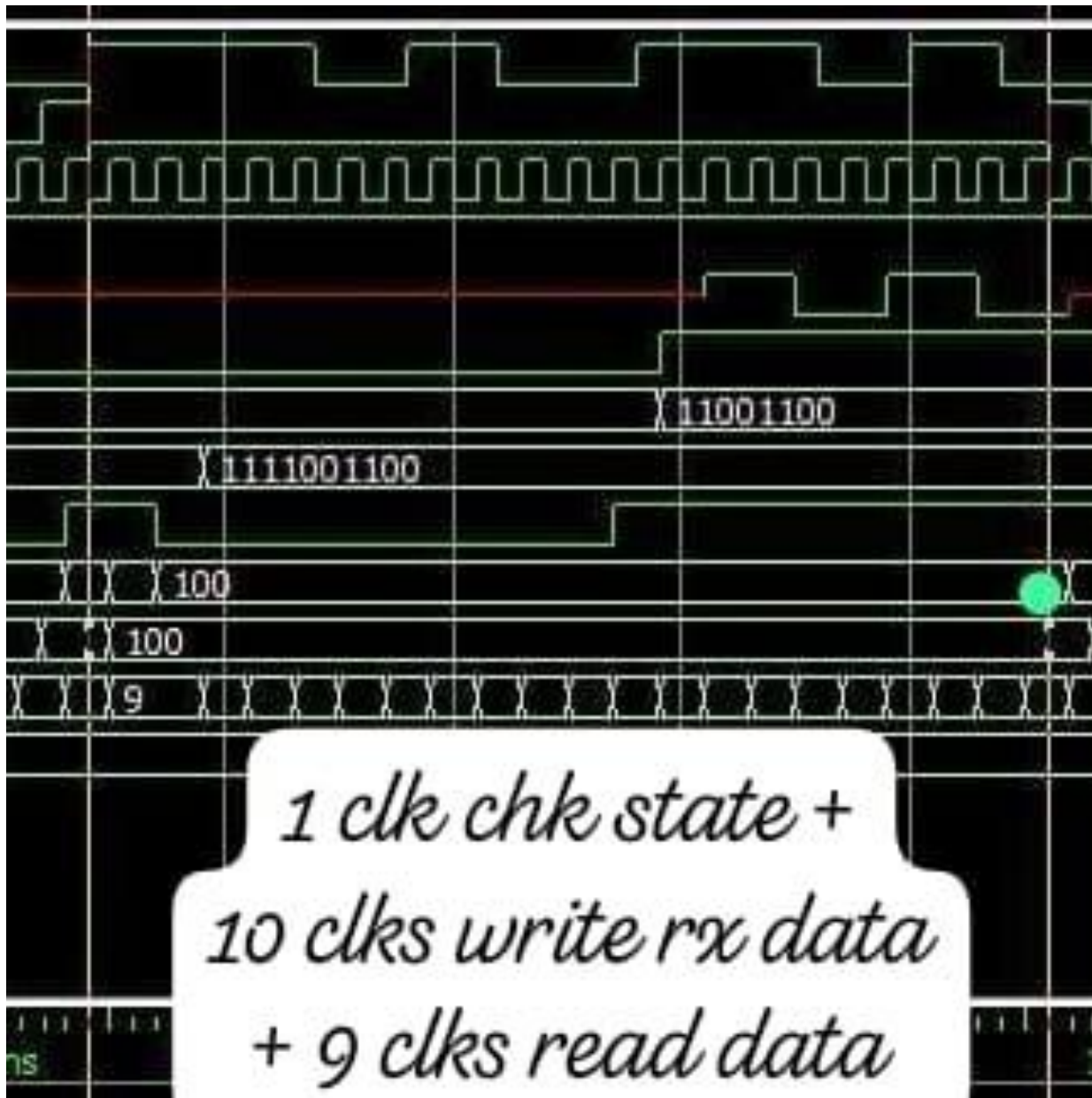
```
126         @(negedge clk);
127
128         MOSI = 1;
129         @(negedge clk);
130         MOSI = 1;
131         @(negedge clk);
132         MOSI = 1;
133         @(negedge clk);
134         MOSI = 1;
135         @(negedge clk);
136         MOSI = 0;
137         @(negedge clk);
138         MOSI = 0;
139         @(negedge clk);
140         MOSI = 1;
141         @(negedge clk);
142         MOSI = 1;
143         @(negedge clk);
144         MOSI = 0;
145         @(negedge clk);
146         SS_n = 1;           //end communication
147         @(negedge clk);
148     end
149 end
150 endmodule
```

c. Waveforms









Notice that in the upper snippets we receive on MISO port the same pattern we have sent before (11001100).

3. Synthesis and Implementation Flow

a. Constrain file

```
## Clock signal
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33}
[get_ports clk]
create_clock -period 10.000 -name sys_clk_pin -waveform {0.000
5.000} -add [get_ports clk]
```

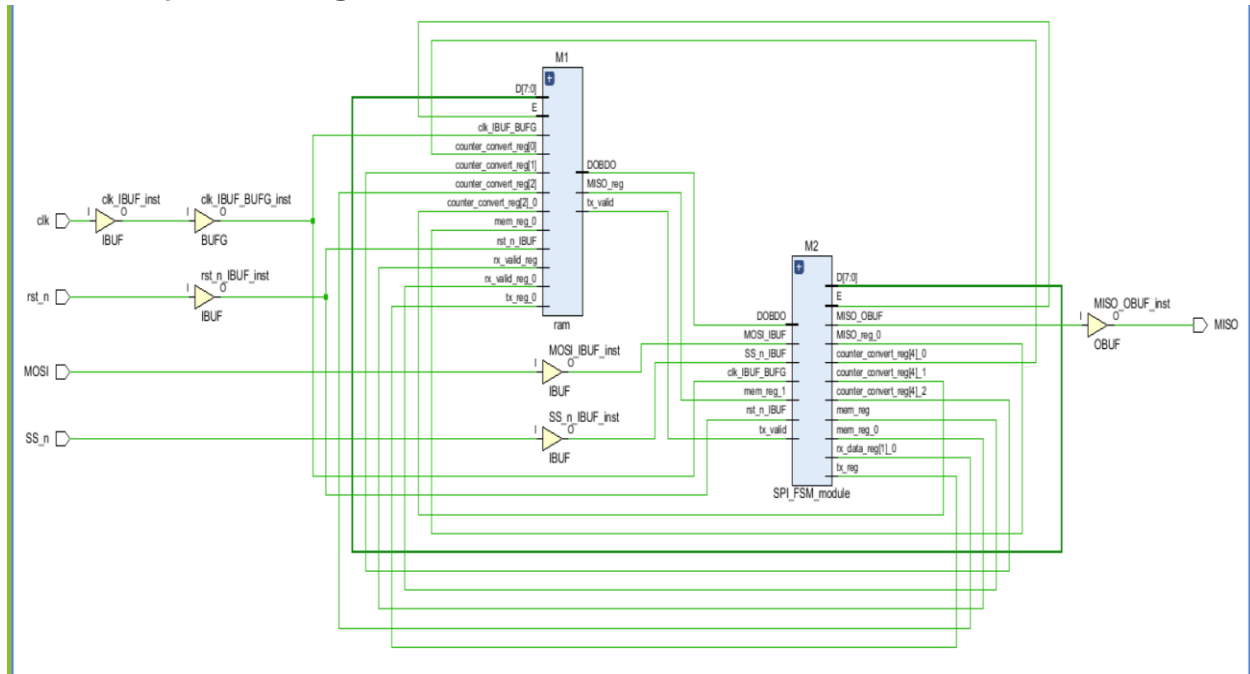
```
## Switches
set_property -dict {PACKAGE_PIN V17 IOSTANDARD LVCMOS33}
[get_ports rst_n]
set_property -dict {PACKAGE_PIN V16 IOSTANDARD LVCMOS33}
[get_ports SS_n]
set_property -dict {PACKAGE_PIN W16 IOSTANDARD LVCMOS33}
[get_ports MOSI]
```

```
## LEDs
set_property -dict {PACKAGE_PIN U16 IOSTANDARD LVCMOS33}
[get_ports MISO]
set_property -dict {PACKAGE_PIN F16 IOSTANDARD LVCMOS33}
```

The snippets above shows the constrain file modifications in order to set input ,outputs and clks.

b.Synthesis Flow for encodings used

- Gray Encoding



Synthesis Schematic

```
32 | Parameter READ_DATA bound to: 3'b100
33 | INFO: [Synth 8-5534] Detected attribute (* fsm_encoding = "gray" *) [D:/KW Diploma/SPI Project/SPI_FSM_module.v:20]
34 | INFO: [Synth 8-155] case statement is not full and has no default [D:/KW Diploma/SPI Project/SPI_FSM_module.v:77]
35 | INFO: [Synth 8-6155] done synthesizing module 'SPI_FSM_module' (2#1) [D:/KW Diploma/SPI Project/SPI_FSM_module.v:3]
36 | INFO: [Synth 8-6155] done synthesizing module 'Wrapper_module' (3#1) [D:/KW Diploma/SPI Project/Wrapper_module.v:3]
37 | -----
38 | Finished RTL Elaboration : Time (s): cpu = 00:00:07 ; elapsed = 00:00:07 . Memory (MB): peak = 410.633 ; gain = 153.902
39 | -----
40 |
```

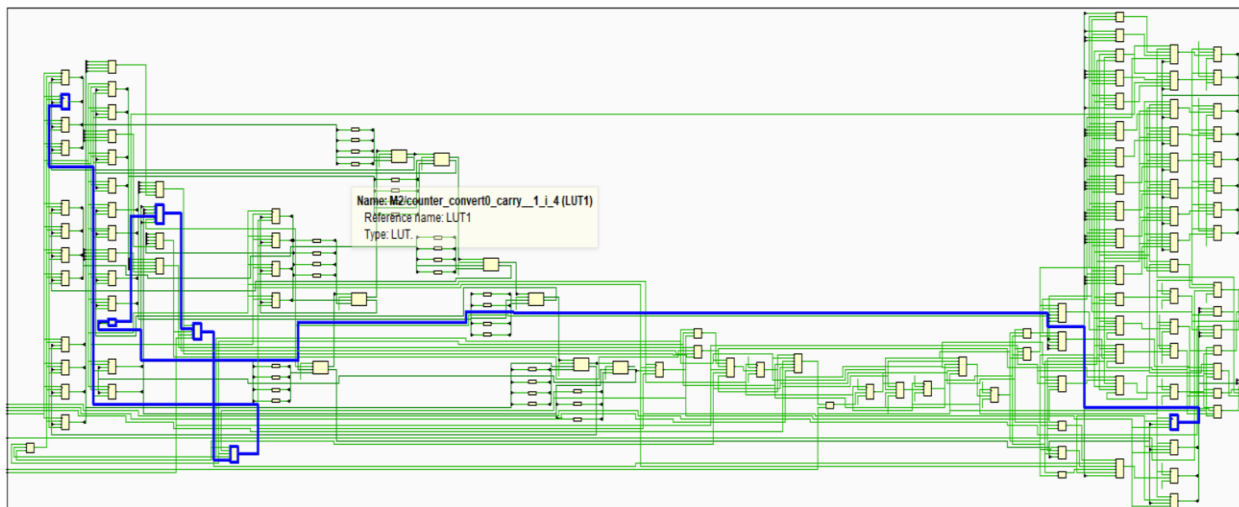
Synthesis report showing encoding used

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 5.059 ns	Worst Hold Slack (WHS): 0.074 ns	Worst Pulse Width Slack (WPWS): 4.500 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 152	Total Number of Endpoints: 152	Total Number of Endpoints: 60

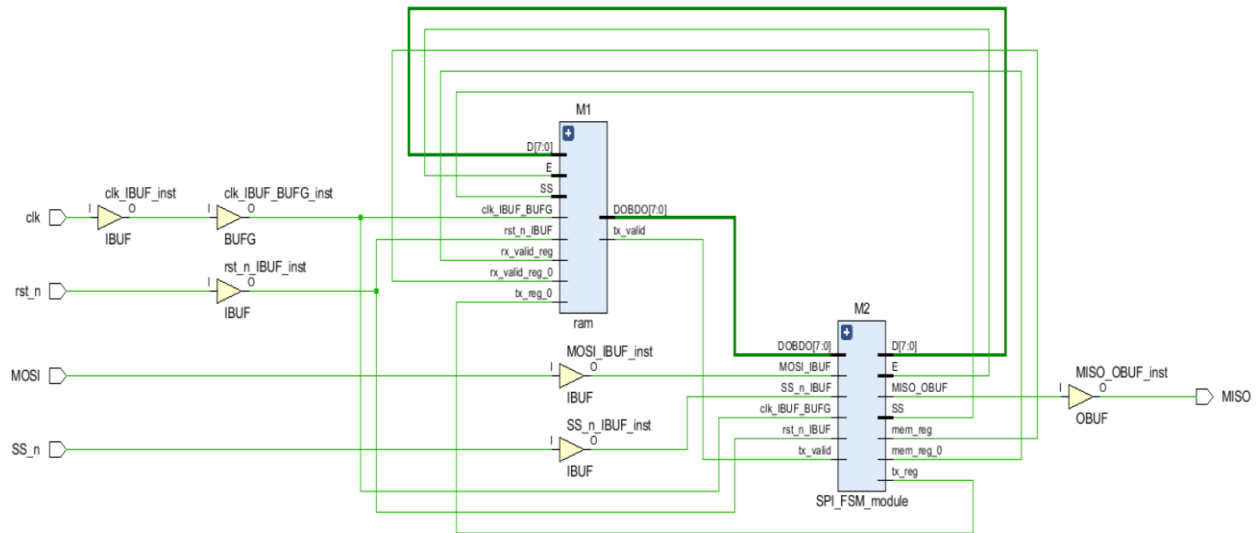
All user specified timing constraints are met.

Timing report



Schematic Showing Critical Path

- One_Hot Encoding



Synthesis Schematic

```

encoding
Next Previous Highlight Match Case Whole Words 4 Match(es)
32 Parameter READ_DATA bound to: 3'b100
33 INFO: [Synth 8-5534] Detected attribute (* fsm_encoding = "one_hot" *) [D:/KW Diploma/SPI Project/SPI_FSM_module.v:20]
34 INFO: [Synth 8-155] case statement is not full and has no default [D:/KW Diploma/SPI Project/SPI_FSM_module.v:77]
35 INFO: [Synth 8-6155] done synthesizing module 'SPI_FSM_module' (2#1) [D:/KW Diploma/SPI Project/SPI_FSM_module.v:3]
36 INFO: [Synth 8-6155] done synthesizing module 'Wrapper_module' (3#1) [D:/KW Diploma/SPI Project/Wrapper_module.v:3]
37 -----
38 Finished RTL Elaboration : Time (s): cpu = 00:00:07 ; elapsed = 00:00:09 . Memory (MB): peak = 409.496 ; gain = 152.586
39 -----
40

```

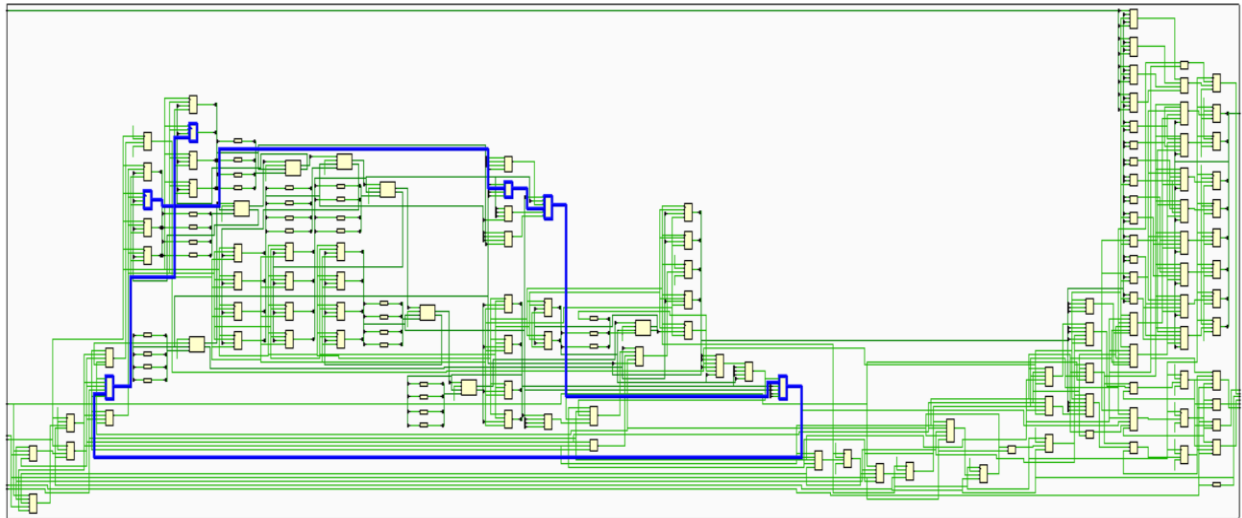
Synthesis report showing encoding used

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 4.830 ns	Worst Hold Slack (WHS): 0.074 ns	Worst Pulse Width Slack (WPWS): 4.500 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 154	Total Number of Endpoints: 154	Total Number of Endpoints: 62

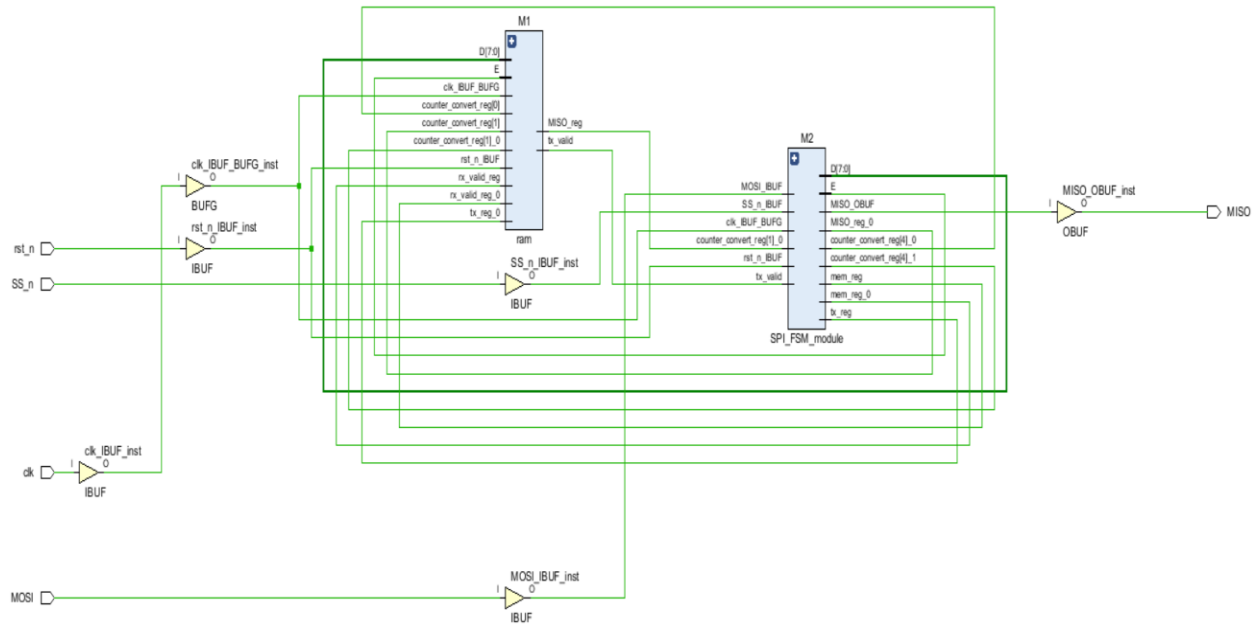
All user specified timing constraints are met.

Timing report



Schematic Showing Critical Path

- Sequential Encoding



Synthesis Schematic

```

32 |     Parameter READ_DATA bound to: 3'b100
33 | INFO: [Synth 0-5534] Detected attribute (* fsm_encoding = "sequential" *) [D:/KW Diploma/SPI Project/SPI_FSM_module.v:20]
34 | INFO: [Synth 0-155] case statement is not full and has no default [D:/KW Diploma/SPI Project/SPI_FSM_module.v:77]
35 | INFO: [Synth 0-6155] done synthesizing module 'SPI_FSM_module' (2#1) [D:/KW Diploma/SPI Project/SPI_FSM_module.v:3]
36 | INFO: [Synth 0-6155] done synthesizing module 'Wrapper_module' (3#1) [D:/KW Diploma/SPI Project/Wrapper_module.v:3]
37 | -----
38 | Finished RTL Elaboration : Time (s): cpu = 00:00:04 ; elapsed = 00:00:04 . Memory (MB): peak = 410.824 ; gain = 153.348
39 | -----

```

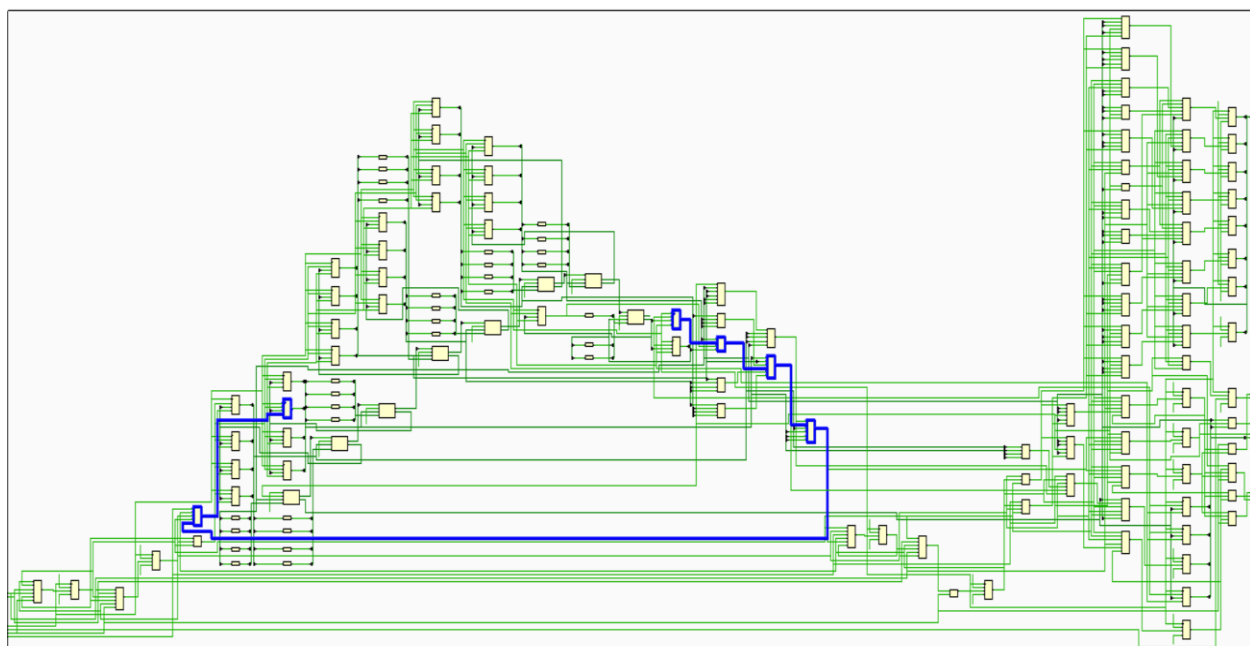
Synthesis report showing encoding used

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 4.824 ns	Worst Hold Slack (WHS): 0.074 ns	Worst Pulse Width Slack (WPWS): 4.500 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 152	Total Number of Endpoints: 152	Total Number of Endpoints: 60

All user specified timing constraints are met.

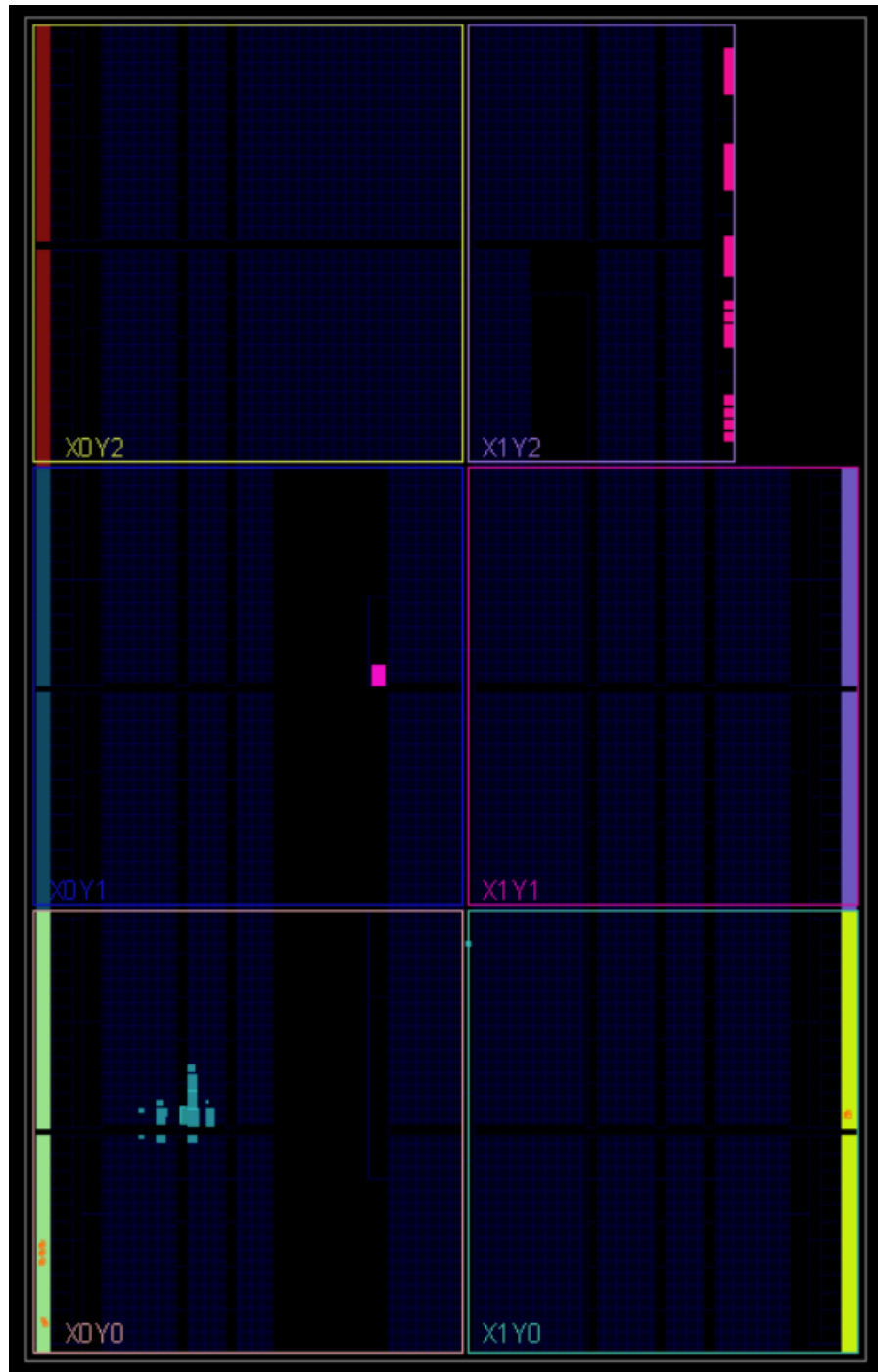
Timing report



Schematic Showing Critical Path

C. Implementation Flows for Encoding Used

- Gray Encoding



Device Implementation

Q

%

Hierarchy

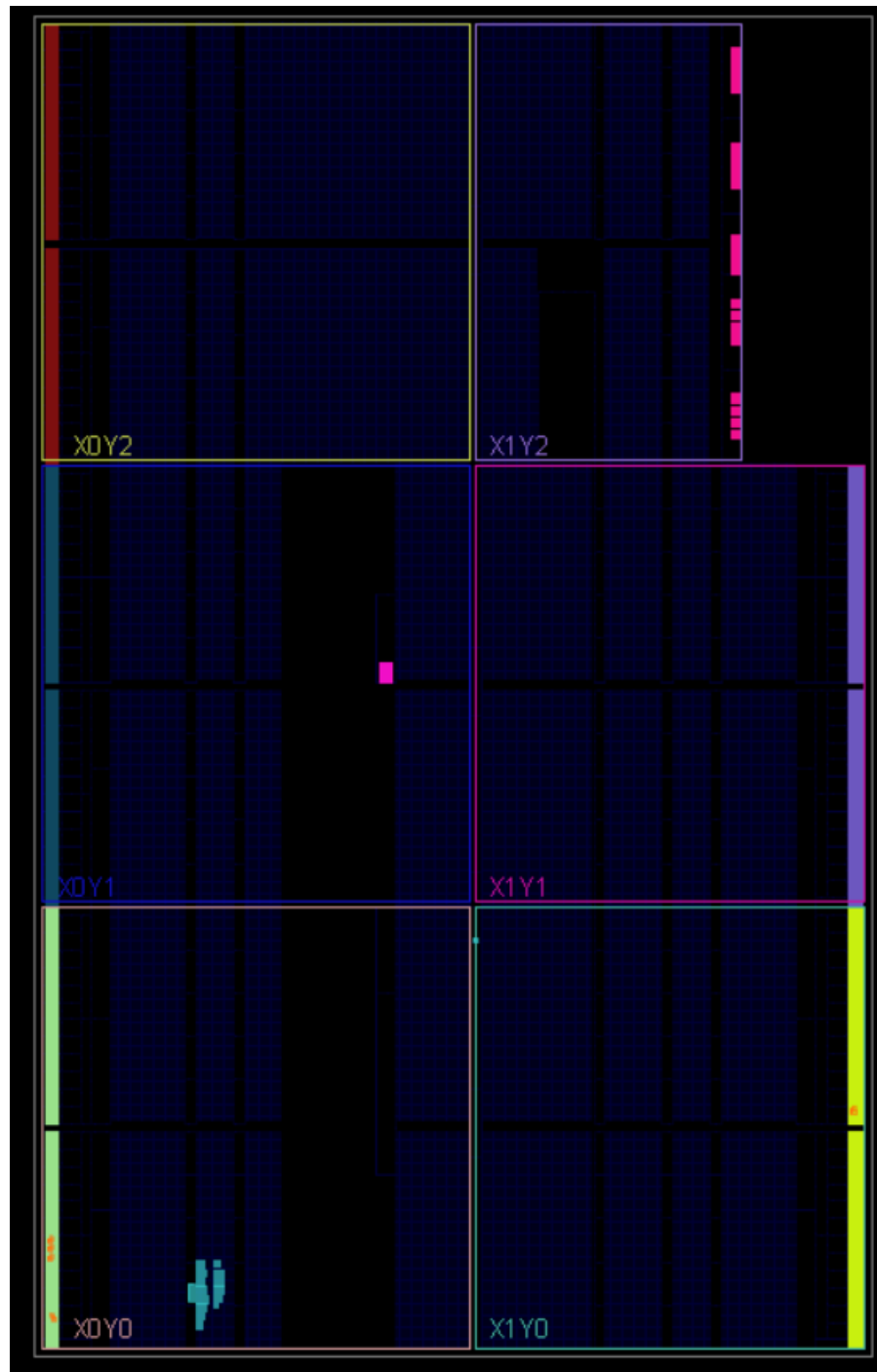
Name	1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Block RAM Tile (50)	Bonded IOB (106)	BUFGCTRL (32)
▼ N Wrapper_module		88	57	30	88	49	0.5	5	1
M1 (ram)		4	9	6	4	0	0.5	0	0
M2 (SPI_FSM_module)		84	48	30	84	47	0	0	0

Utilization Report

Design Timing Summary			
Setup		Hold	Pulse Width
Worst Negative Slack (WNS):		Worst Hold Slack (WHS):	Worst Pulse Width Slack (WPWS):
5.150 ns		0.077 ns	4.500 ns
Total Negative Slack (TNS):		Total Hold Slack (THS):	Total Pulse Width Negative Slack (TPWS):
0.000 ns		0.000 ns	0.000 ns
Number of Failing Endpoints:		Number of Failing Endpoints:	Number of Failing Endpoints:
0		0	0
Total Number of Endpoints:		Total Number of Endpoints:	Total Number of Endpoints:
152		152	60
All user specified timing constraints are met.			

Timing Report

- One_Hot Encoding



Device Implementation

Q

≡

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

⬆

⬇

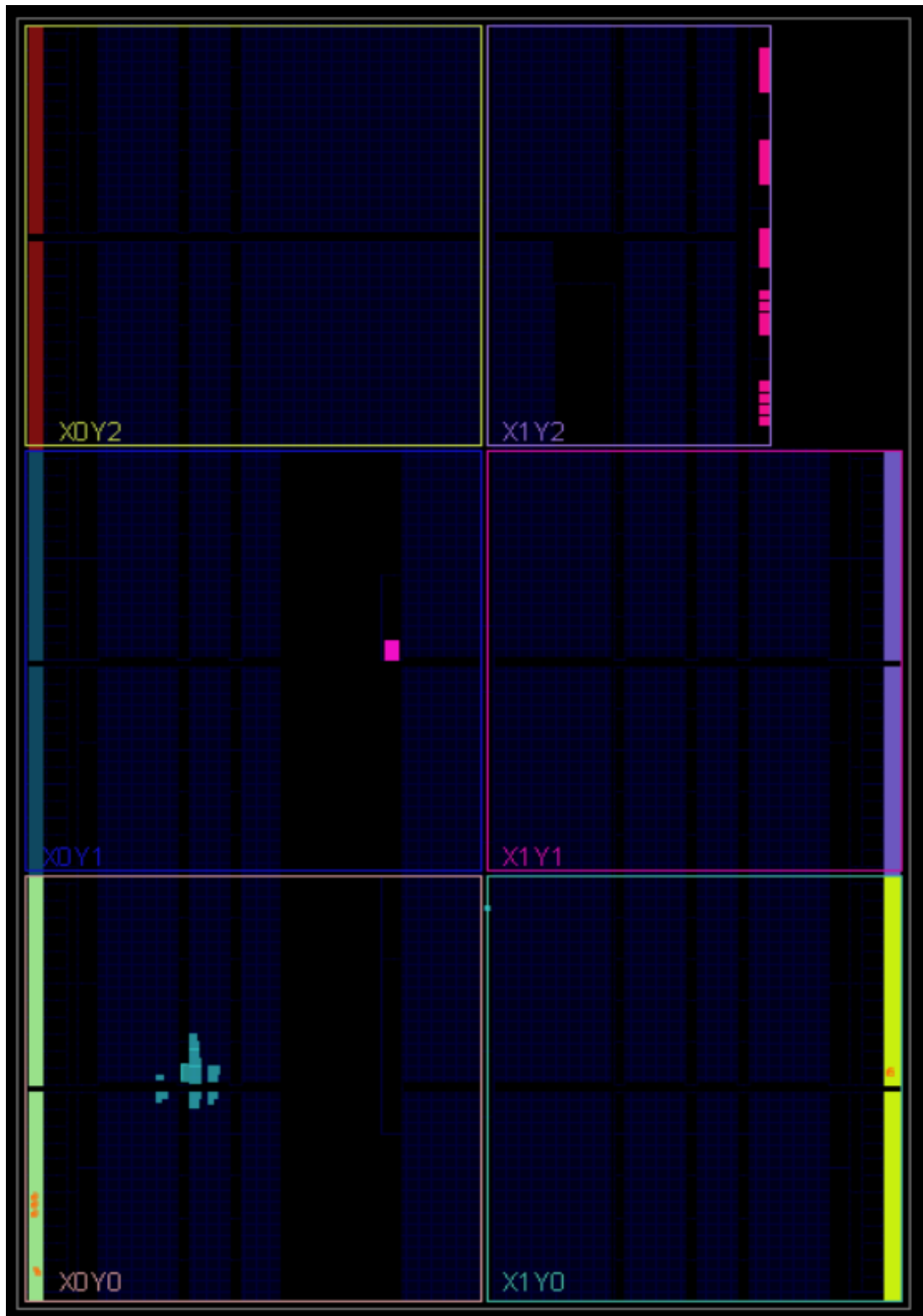
⬆

Utilization Report

Design Timing Summary			
Setup	Hold	Pulse Width	
Worst Negative Slack (WNS): 4.347 ns	Worst Hold Slack (WHS): 0.080 ns	Worst Pulse Width Slack (WPWS): 4.500 ns	
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns	
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	
Total Number of Endpoints: 154	Total Number of Endpoints: 154	Total Number of Endpoints: 62	
All user specified timing constraints are met.			

Timing Report

- Sequential Encoding



Device Implementation

Name	Slice LUTs (20800)	Slice Registers (41600)	F7 Muxes (16300)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Block RAM Tile (50)	Bonded IOB (106)	BUFGCTRL (32)
▼ N Wrapper_module	85	57	1	28	85	49	0.5	5	1
M1 (ram)	3	9	1	3	3	0	0.5	0	0
M2 (SPI_FSM_module)	82	48	0	28	82	47	0	0	0

Utilization Report





















Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 4.968 ns	Worst Hold Slack (WHS): 0.080 ns	Worst Pulse Width Slack (WPWS): 4.500 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 152	Total Number of Endpoints: 152	Total Number of Endpoints: 60

All user specified timing constraints are met.














Timing Report

After comparing timing reports, seeing that gray encoding has the highest setup/hold slack after implementation, proceeding in adding debugging core.

- Debugging core added

Debug			
     			
Name	Driver Cell	Driver Pin	Probe Type
▼  dbg_hub(labtools_xsdbm_v3)			
▼  u_ila_0(labtools_ila_v6)			
>  clk (1)			
▼  probe0 (1)			Data and Trigger ▼
 Ch 0 (clk_IBUF)	IBUF	O	
▼  probe1 (1)			Data and Trigger ▼
 Ch 0 (MISO_OBUF)	FDRE	Q	
▼  probe2 (1)			Data and Trigger ▼
 Ch 0 (MOSI_IBUF)	IBUF	O	
▼  probe3 (1)			Data and Trigger ▼
 Ch 0 (rst_n_IBUF)	IBUF	O	
▼  probe4 (1)			Data and Trigger ▼
 Ch 0 (SS_n_IBUF)	IBUF	O	
 Unassigned Debug Nets (0)			

- Messages Tab showing no critical warnings or errors

Messages	
     	
<input checked="" type="checkbox"/>  Warning (4)	<input checked="" type="checkbox"/>  Info (265)
<input type="checkbox"/>  Status (503)	<input type="button" value="Show All"/>
>  Vivado Commands (3 infos)	
>  Synthesis (1 warning, 38 infos)	
>  Implementation (1 warning, 107 infos)	
>  Implemented Design (1 warning, 10 infos)	

Bitstream Generation

