

# Assignment 6

(Under supervision of Eng/ Karim Wassem)

## Question1

```
1  module question1(x,clk,rst,y,count);
2
3  parameter IDLE=2'b00;
4  parameter ZERO=2'b01;
5  parameter ONE=2'b10;
6  parameter STORE=2'b11;
7
8  input x,clk,rst;
9  output reg y;
10 output reg [9:0] count;
11
12 reg [1:0] cs,ns;
13
14 //state memory
15
16 always @(posedge clk or posedge rst) begin
17     if(rst) begin
18         cs<=IDLE;
19         count<=10'b0000000000;
20     end
21     else
22         cs<=ns;
23 end
24
25 // next state logic
26
27 always @(*) begin
28     case(cs)
29     IDLE: begin
30         if(x)
31             ns=IDLE;
32         else
33             ns=ZERO;
34     end
35     ZERO:begin
36         if(x)
37             ns=ONE;
38         else
39             ns=ZERO;
40     end
41     ONE:begin
42         if(x)
43             ns=IDLE;
44         else
45             ns=STORE;
46     end
47     STORE:begin
48         if(x)
49             ns=IDLE;
50         else
51             ns=ZERO;
52     end
53     default: ns=IDLE;
54 endcase
55 end
56
57 // output logic block
58
59 always @(*) begin
60     case(cs)
61     IDLE: y=0;
62     ZERO: y=0;
63     ONE: y=0;
64     STORE: begin
65         y=1;
66         count = count +1;
67     end
68     endcase
69 end
70
71 endmodule
```

Figure (1): design code

```

1  module question1_tb();
2
3  parameter n0=2'b00;
4  parameter n1=2'b01;
5  parameter n2=2'b10;
6  parameter n3=2'b11;
7
8  reg x,clk,rst;
9  wire y;
10 wire [9:0] count;
11 integer i=0;
12
13 question1 #(.IDLE(n0),.ZERO(n1),.ONE(n2),.STORE(n3))DUT(.x(x),.clk(clk),.rst(rst),.y(y),.count(count));
14
15 initial begin
16   clk=0;
17   forever
18     #25 clk=~clk;
19   end
20
21 initial begin
22   rst=1;
23   #100; //delay for two clock periods
24   rst=0;
25   for(i=0;i<50;i=i+1) begin
26     @(negedge clk);
27     x=$random;
28   end
29   $stop;
30 end
31
32 endmodule

```

Figure (2): testbench code

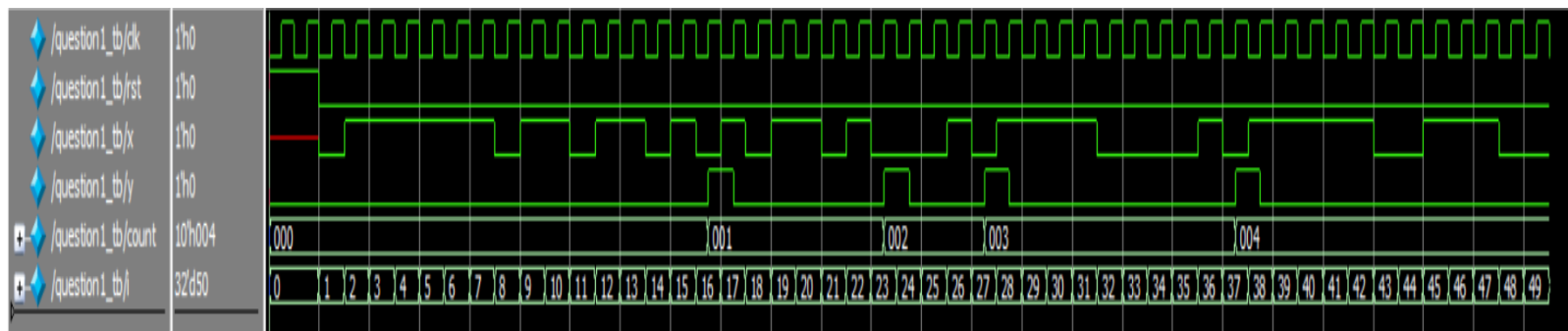


Figure (3): waveform snippet

Utilization				
Hierarchy				
Summary				
▼ Slice Logic				
▼ Slice LUTs (<1%)				
LUT as Logic (<1%)				
▼ Slice Registers (<1%)				
Register as Flip Flop (<1%)				
Memory				
DSP				
▼ IO and GT Specific				
Bonded IOB (13%)				
▼ Clocking				
BUFGCTRL (3%)				
Hierarchy				
Name	1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)
question1		2	2	14

Figure (4): report utilization

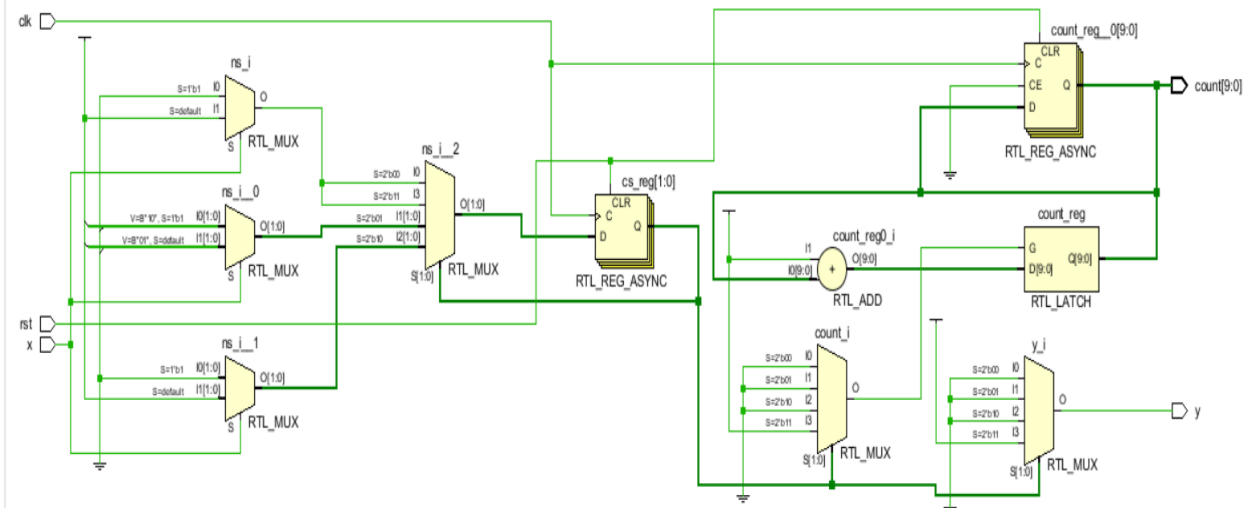


Figure (5): elaborated schematic

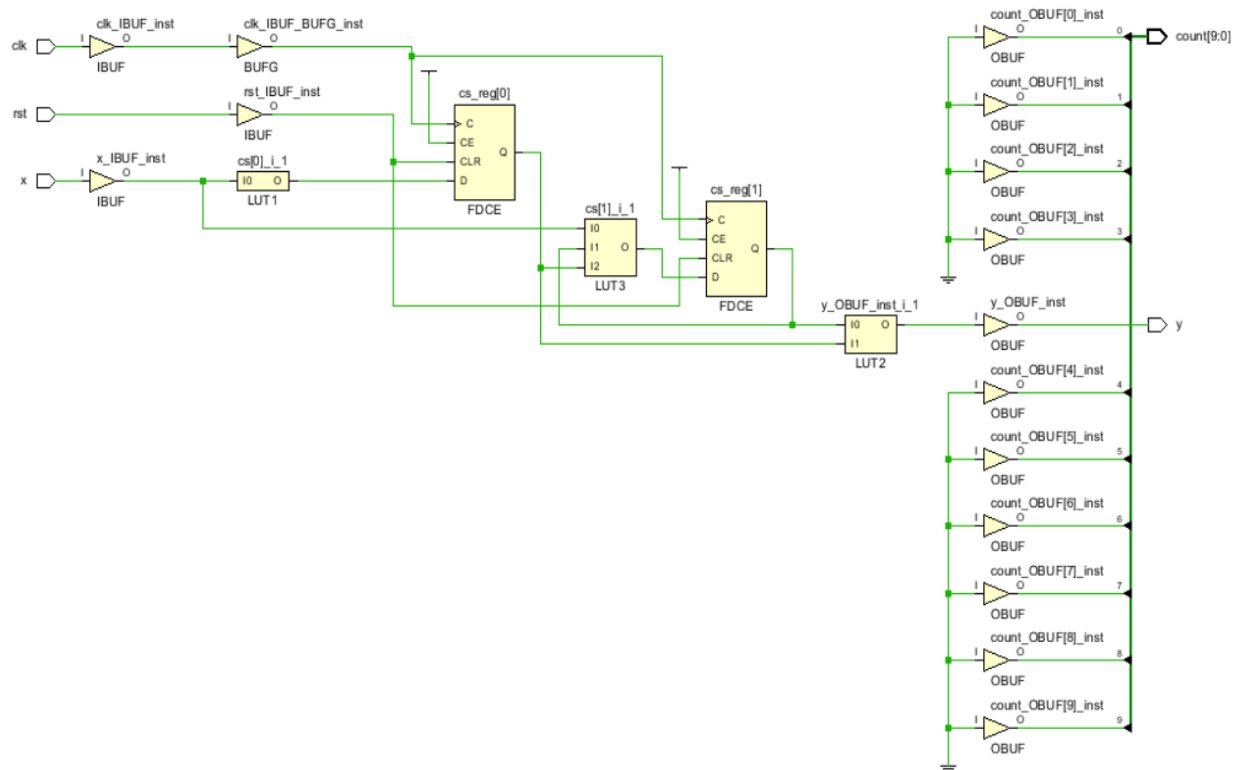


Figure (6): synthesis schematic

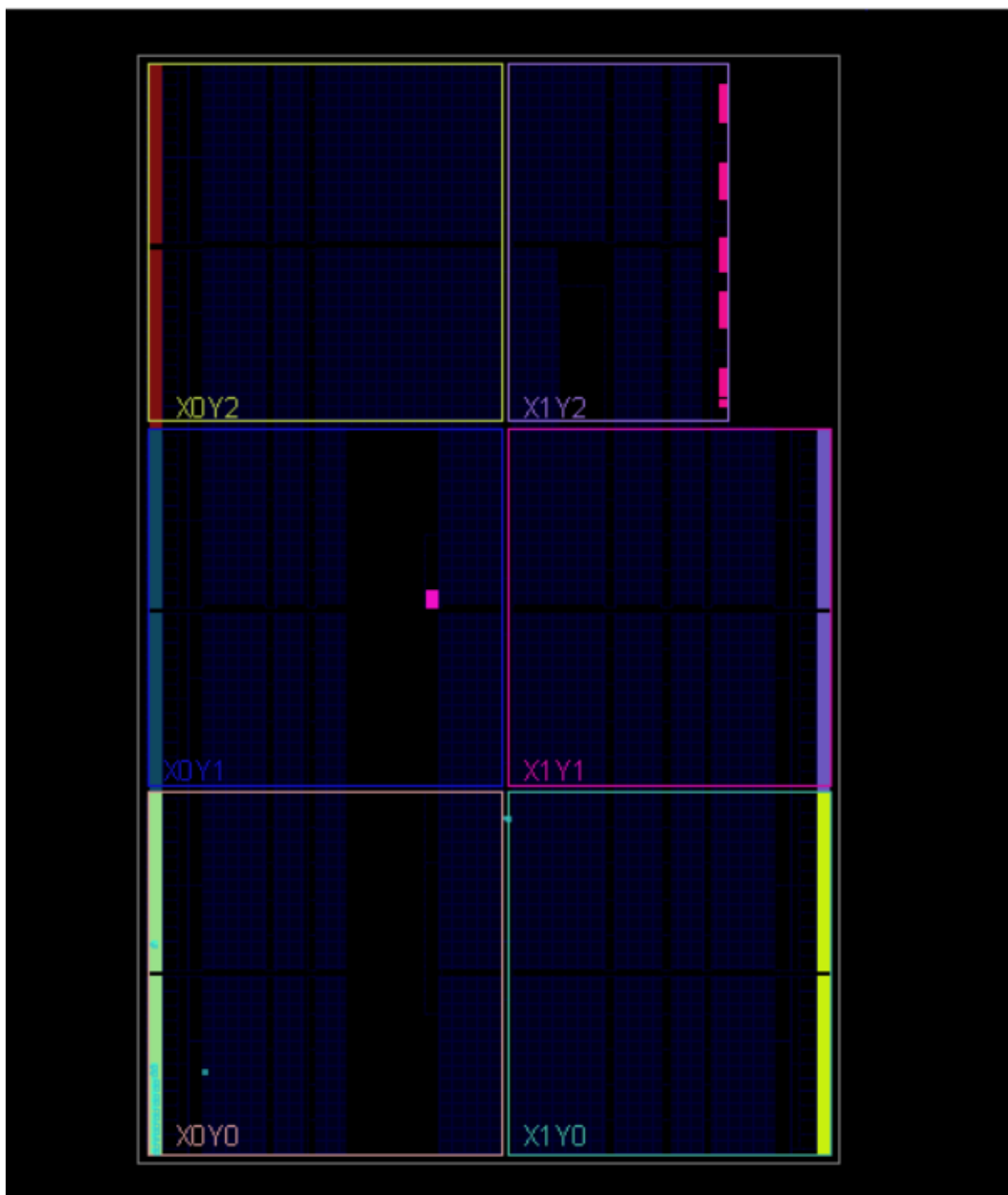


Figure (7): device implementation

## Question2

```
1  module question2(speed_limit,car_speed,leading_distance,clk,rst,unlock_doors,accelerate_car);
2
3  parameter MIN_DISTANCE=7'd40;
4
5  parameter STOP=2'b00;
6  parameter DECELERATE=2'b01;
7  parameter ACCELERATE=2'b10;
8
9  input clk,rst;
10 input [7:0] speed_limit,car_speed;
11 input [6:0] leading_distance;
12 output reg unlock_doors,accelerate_car;
13
14 reg [1:0] cs,ns;
15
16 // state memory
17 always @(posedge clk or posedge rst) begin
18     if(rst)
19         cs<=STOP;
20     else
21         cs<=ns;
22 end
23
24 // nextstate logic block
25 always @(*) begin
26     case(cs)
27     STOP: begin
28         if(leading_distance < MIN_DISTANCE)
29             ns = STOP;
30         else
31             ns = ACCELERATE;
32     end
33     DECELERATE: begin
34         if(car_speed==0)
35             ns = STOP;
36         else if(leading_distance >= MIN_DISTANCE && car_speed < speed_limit)
37             ns = ACCELERATE;
38         else
39             ns = DECELERATE;
40     end
41     ACCELERATE: begin
42         if(leading_distance < MIN_DISTANCE || car_speed > speed_limit)
43             ns = DECELERATE;
44         else
45             ns = ACCELERATE;
46         end
47     endcase
48 end
49
50 // output logic block
51
52 always @(*) begin
53     case(cs)
54     STOP: begin
55         unlock_doors = 1;
56         accelerate_car = 0;
57     end
58     ACCELERATE: begin
59         unlock_doors = 0;
60         accelerate_car = 1;
61     end
62     DECELERATE: begin
63         unlock_doors = 0;
64         accelerate_car = 0;
65     end
66     endcase
67
68 end
69
70 endmodule
```

Figure (8): design code

```

1  module question2_tb();
2
3  parameter N0= 7'd40; // MIN_DISTANCE parameter
4  parameter N1=2'b00; // stop state
5  parameter N2=2'b01; // decelerate state
6  parameter N3=2'b10; // accelerate state
7
8  reg [7:0] speed_limit,car_speed;
9  reg [6:0] leading_distance;
10 reg clk,rst;
11 wire unlock_doors,accelerate_car;
12 integer i=0;
13
14 question2 #(.MIN_DISTANCE(N0),.STOP(N1),.DECELERATE(N2),.ACCELERATE(N3)) DUT(.speed_limit(speed_limit),.car_speed(car_speed),.leading_distance(leading_distance),.clk(clk)
15
16 initial begin
17 clk=0;
18 forever
19 #25 clk=~clk;
20 end
21
22 initial begin
23 rst=1;
24 #100;
25 rst=0;
26 speed_limit=8'd100; // randomization testing if the max speed =100 km/h
27 for(i=0;i<10;i=i+1) begin
28 @(negedge clk);
29 car_speed=$urandom_range(50,150); // random values for car speed between 50 km/h to 150 km/h
30 leading_distance=$urandom_range(0,80); // random values for leading_distance between 0 meters to 80 meters
31 end
32 speed_limit=8'd150; // randomization testing if the max speed =150 km/h
33 for(i=10;i<20;i=i+1) begin
34 @(negedge clk);
35 car_speed=$urandom_range(100,200); // random values for car speed between 100 km/h to 200 km/h
36 leading_distance=$urandom_range(10,60); // random values for leading_distance between 10 meters to 60 meters
37 end
38 $stop;
39 end
40
41 endmodule

```

Figure (9): testbench code

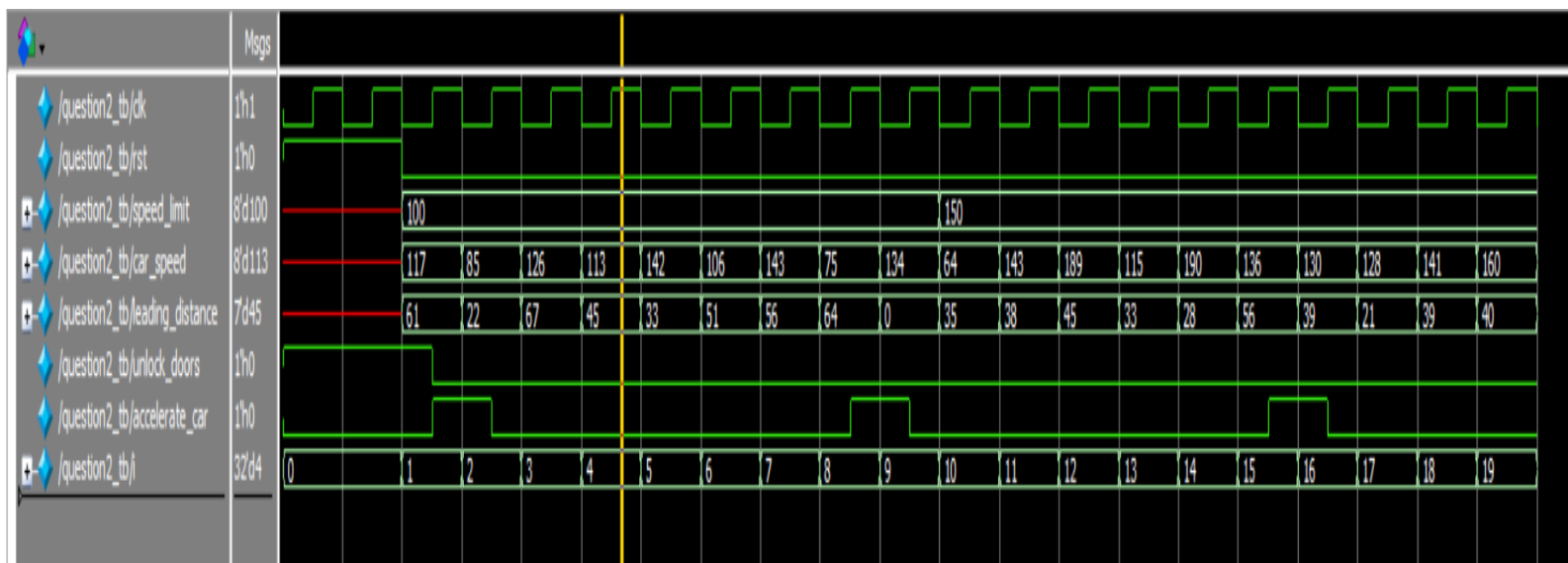


Figure (10): waveform snippet

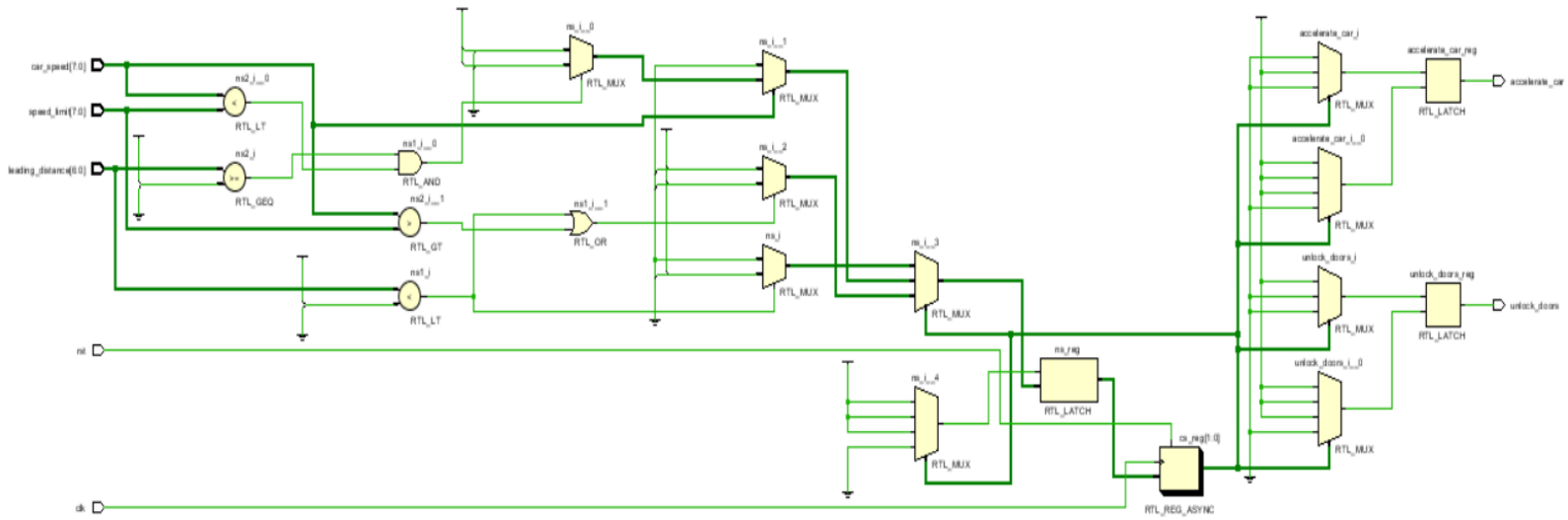


Figure (11): Elaborated schematic

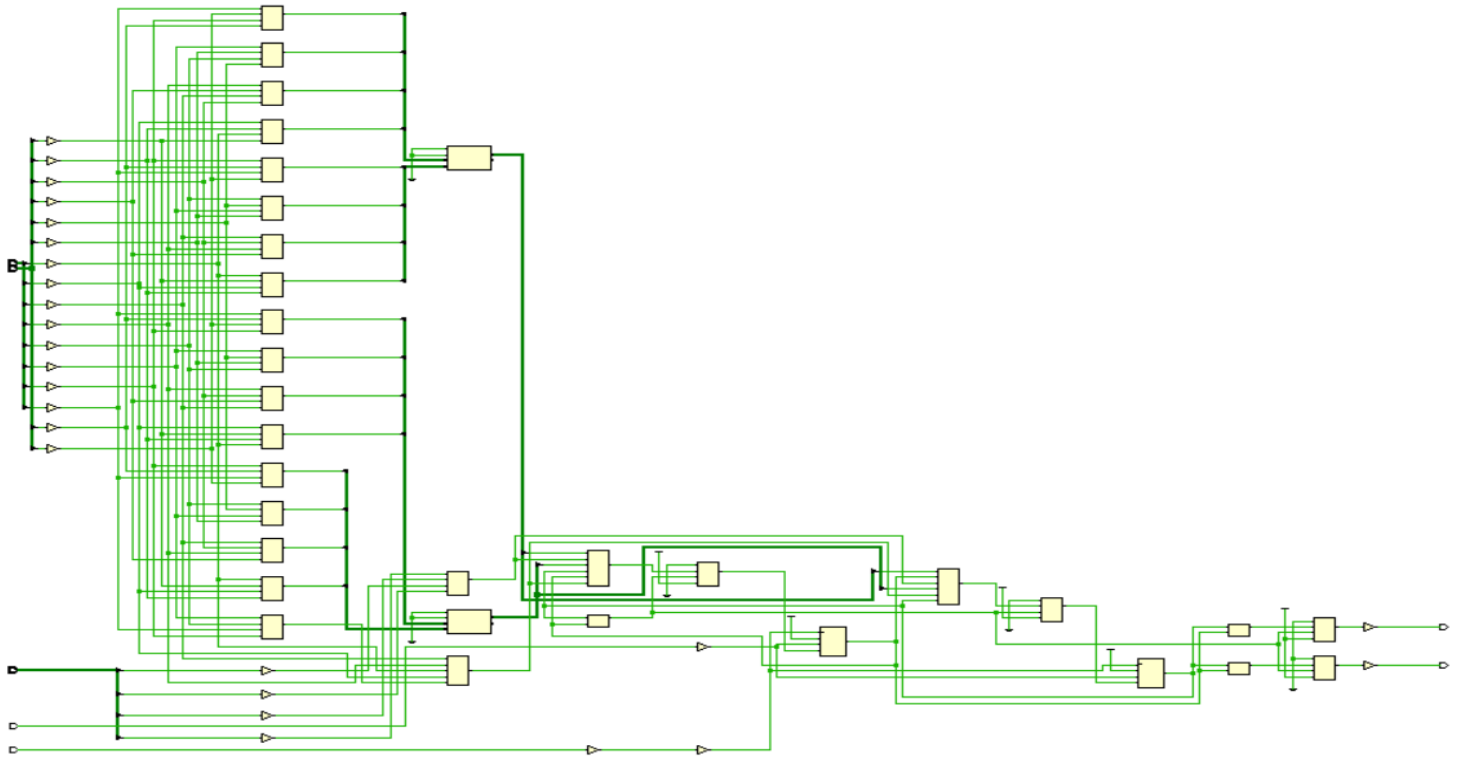


Figure (12): synthesis schematic

Utilization					
Hierarchy					
Summary					
▼ Slice Logic					
▼ Slice LUTs (<1%)					
LUT as Logic (<1%)					
▼ Slice Registers (<1%)					
Register as Latch (<1%)					
Register as Flip Flop					
Memory					
DSP					
▼ IO and GT Specific					
Bonded IOB (23%)					
▼ Clocking					
BUFGCTRL (3%)					
Hierarchy					
Name	1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N question2		16	6	24	1

Figure (13): utilization report

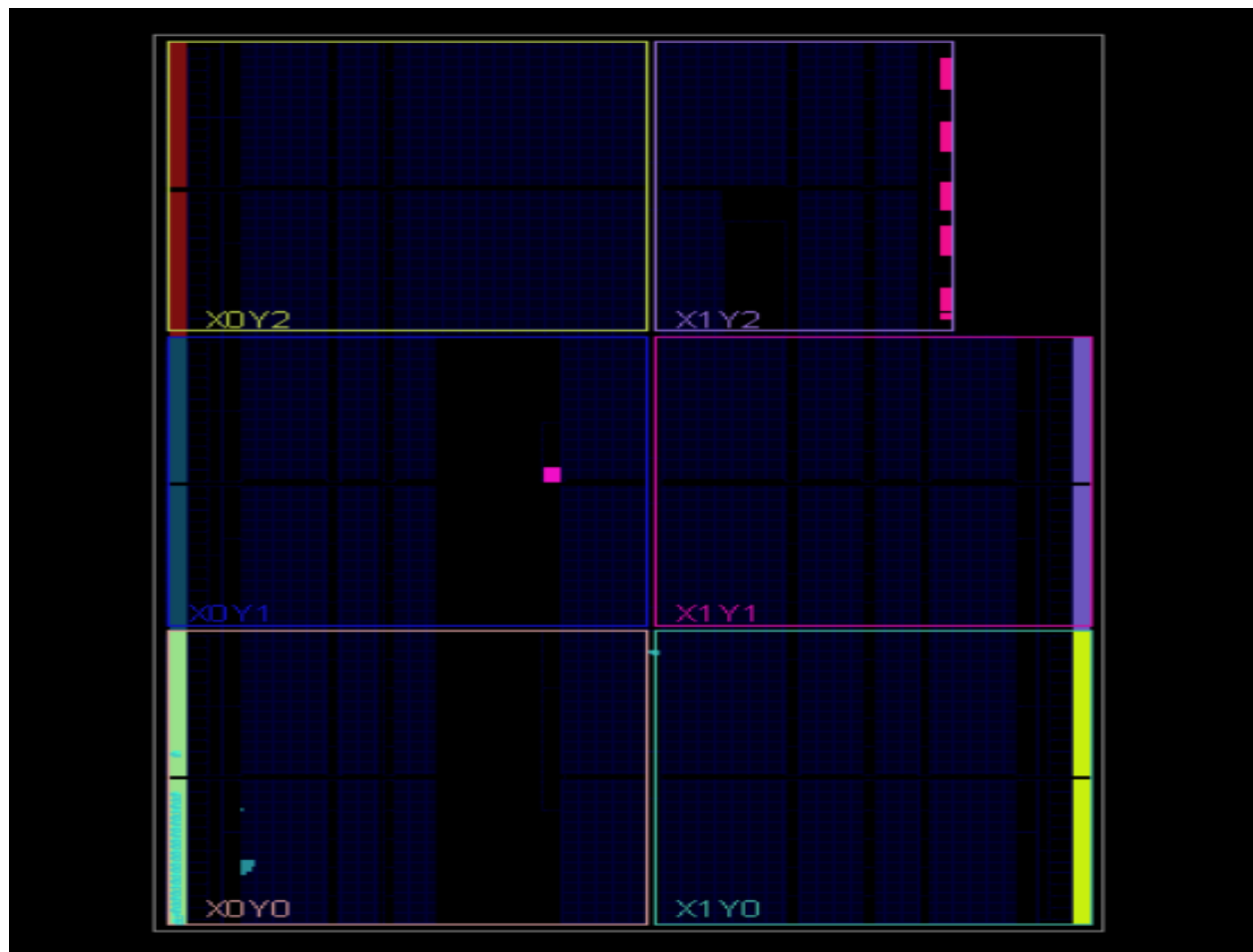


Figure (14): device implementation



### Question3]

```
1 module question3(clk,rst,y);
2
3 parameter A=2'b00;
4 parameter B=2'b01;
5 parameter C=2'b10;
6 parameter D=2'b11;
7
8 input clk,rst;
9 output reg [1:0] y;
10 reg [1:0] cs,ns;
11
12 // state memory block
13 always @(posedge clk or posedge rst) begin
14     if(rst)
15         cs<=0;
16     else
17         cs<=ns;
18 end
19
20 // next state logic block
21 always @(*) begin
22     case(cs)
23         A: ns = B;
24         B: ns = C;
25         C: ns = D;
26         D: ns = A;
27     endcase
28 end
29
30 // output logic block
31 always @(*) begin
32     case(cs)
33         A: y = A;
34         B: y = B;
35         C: y = D;
36         D: y = C;
37     endcase
38 end
39
40 endmodule
```

Figure (15): main design code

```
1 module gray_counter(clk,rst,gray_out);
2
3 input clk,rst;
4
5 output [1:0] gray_out;
6
7 reg [1:0] bin_out;
8
9 always @(posedge clk or posedge rst) begin
10
11     if(rst)
12         bin_out<=0;
13     else
14         bin_out<=bin_out+1;
15
16 end
17
18 assign gray_out[0]=^bin_out;
19
20 assign gray_out[1]=bin_out[1];
21
22 endmodule
```

Figure (16): reference design code

```
1 vlib work
2 vlog design.v refrence_design.v testbench.v
3 vsim -voptargs=+acc work.question3_tb
4 add wave *
5 run -all
```

Figure (17): Do file

```

1  module question3_tb();
2
3  reg clk,rst;
4  wire [1:0] y_dut,y_ref;    // y_dut represent the output of main design
5  integer i=0;              // y_ref represent the output of the reference design
6
7  question3 DUT1(.clk(clk),.rst(rst),.y(y_dut));
8  gray_counter DUT2(.clk(clk),.rst(rst),.gray_out(y_ref));
9
10 initial begin
11   clk=0;
12   forever
13     #25 clk=~clk;
14   end
15
16 initial begin
17   rst=1;
18   #100;    //delay for the first two clock cycles
19   rst=0;
20   for(i=0;i<10;i=i+1) begin
21     @(negedge clk);
22     if(y_dut!=y_ref) begin
23       $display("the counter doesn't act properly!!");
24       $stop;
25     end
26   end
27   $stop;
28 end
29
30 endmodule

```

Figure (18): testbench code

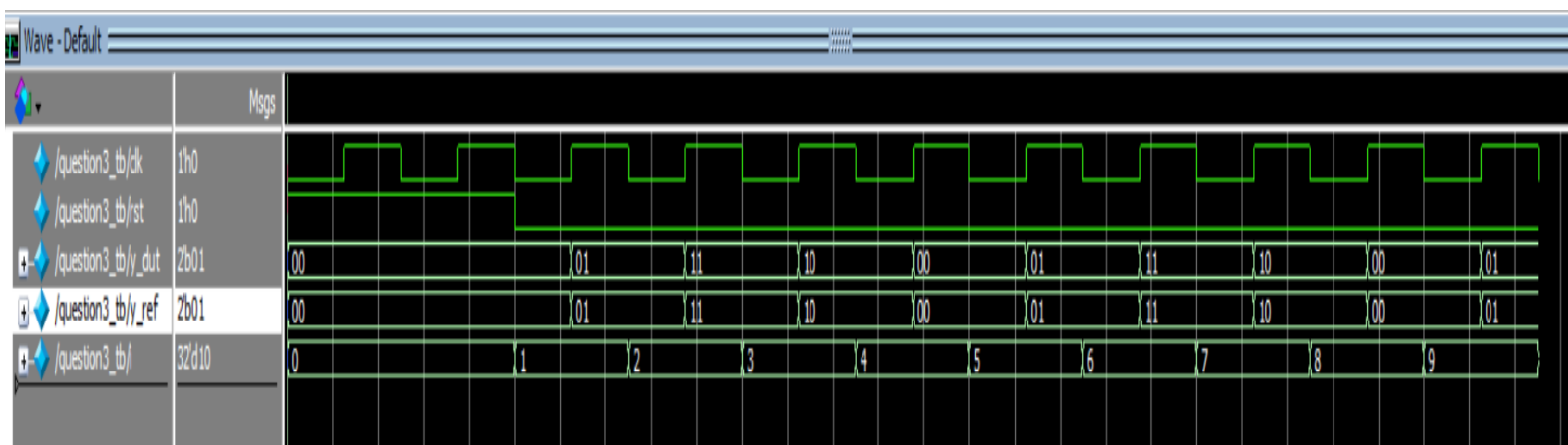


Figure (19): waveform snippet

```

1  ## This file is a general .xdc for the Basys3 rev B board
2  ## To use it in a project:
3  ## - uncomment the lines corresponding to used pins
4  ## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project
5
6  ## Clock signal
7  set_property -dict { PACKAGE_PIN W5      IOSTANDARD LVCMOS33 } [get_ports clk]
8  create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
9
10
11  ## Switches
12  #set_property -dict { PACKAGE_PIN V17      IOSTANDARD LVCMOS33 } [get_ports {sw[0]}]
13  #set_property -dict { PACKAGE_PIN V16      IOSTANDARD LVCMOS33 } [get_ports {sw[1]}]
14  #set_property -dict { PACKAGE_PIN W16      IOSTANDARD LVCMOS33 } [get_ports {sw[2]}]
15  #set_property -dict { PACKAGE_PIN W17      IOSTANDARD LVCMOS33 } [get_ports {sw[3]}]
16  #set_property -dict { PACKAGE_PIN W15      IOSTANDARD LVCMOS33 } [get_ports {sw[4]}]
17  #set_property -dict { PACKAGE_PIN V15      IOSTANDARD LVCMOS33 } [get_ports {sw[5]}]
18  #set_property -dict { PACKAGE_PIN W14      IOSTANDARD LVCMOS33 } [get_ports {sw[6]}]
19  #set_property -dict { PACKAGE_PIN W13      IOSTANDARD LVCMOS33 } [get_ports {sw[7]}]
20  #set_property -dict { PACKAGE_PIN V2       IOSTANDARD LVCMOS33 } [get_ports {sw[8]}]
21  #set_property -dict { PACKAGE_PIN T3       IOSTANDARD LVCMOS33 } [get_ports {sw[9]}]
22  #set_property -dict { PACKAGE_PIN T2       IOSTANDARD LVCMOS33 } [get_ports {sw[10]}]
23  #set_property -dict { PACKAGE_PIN R3       IOSTANDARD LVCMOS33 } [get_ports {sw[11]}]
24  #set_property -dict { PACKAGE_PIN W2       IOSTANDARD LVCMOS33 } [get_ports {sw[12]}]
25  #set_property -dict { PACKAGE_PIN U1       IOSTANDARD LVCMOS33 } [get_ports {sw[13]}]
26  #set_property -dict { PACKAGE_PIN T1       IOSTANDARD LVCMOS33 } [get_ports {sw[14]}]
27  #set_property -dict { PACKAGE_PIN R2       IOSTANDARD LVCMOS33 } [get_ports {sw[15]}]
28
29
30  ## LEDs
31  set_property -dict { PACKAGE_PIN U16      IOSTANDARD LVCMOS33 } [get_ports {y[0]}]
32  set_property -dict { PACKAGE_PIN E19      IOSTANDARD LVCMOS33 } [get_ports {y[1]}]
33  #set_property -dict { PACKAGE_PIN U19      IOSTANDARD LVCMOS33 } [get_ports {led[2]}]
34  #set_property -dict { PACKAGE_PIN V19      IOSTANDARD LVCMOS33 } [get_ports {led[3]}]
35  #set_property -dict { PACKAGE_PIN W18      IOSTANDARD LVCMOS33 } [get_ports {led[4]}]
36  #set_property -dict { PACKAGE_PIN U15      IOSTANDARD LVCMOS33 } [get_ports {led[5]}]
37  #set_property -dict { PACKAGE_PIN U14      IOSTANDARD LVCMOS33 } [get_ports {led[6]}]
38  #set_property -dict { PACKAGE_PIN V14      IOSTANDARD LVCMOS33 } [get_ports {led[7]}]
39  #set_property -dict { PACKAGE_PIN V13      IOSTANDARD LVCMOS33 } [get_ports {led[8]}]
40  #set_property -dict { PACKAGE_PIN V3       IOSTANDARD LVCMOS33 } [get_ports {led[9]}]
41  #set_property -dict { PACKAGE_PIN W3       IOSTANDARD LVCMOS33 } [get_ports {led[10]}]
42  #set_property -dict { PACKAGE_PIN U3       IOSTANDARD LVCMOS33 } [get_ports {led[11]}]
43  #set_property -dict { PACKAGE_PIN P3       IOSTANDARD LVCMOS33 } [get_ports {led[12]}]
44  #set_property -dict { PACKAGE_PIN N3       IOSTANDARD LVCMOS33 } [get_ports {led[13]}]
45  #set_property -dict { PACKAGE_PIN P1       IOSTANDARD LVCMOS33 } [get_ports {led[14]}]
46  #set_property -dict { PACKAGE_PIN L1       IOSTANDARD LVCMOS33 } [get_ports {led[15]}]
47
48
49  ##7 Segment Display
50  #set_property -dict { PACKAGE_PIN W7       IOSTANDARD LVCMOS33 } [get_ports {seg[0]}]
51  #set_property -dict { PACKAGE_PIN W6       IOSTANDARD LVCMOS33 } [get_ports {seg[1]}]
52  #set_property -dict { PACKAGE_PIN U8       IOSTANDARD LVCMOS33 } [get_ports {seg[2]}]
53  #set_property -dict { PACKAGE_PIN V8       IOSTANDARD LVCMOS33 } [get_ports {seg[3]}]
54  #set_property -dict { PACKAGE_PIN U5       IOSTANDARD LVCMOS33 } [get_ports {seg[4]}]
55  #set_property -dict { PACKAGE_PIN V5       IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]
56  #set_property -dict { PACKAGE_PIN U7       IOSTANDARD LVCMOS33 } [get_ports {seg[6]}]
57
58  #set_property -dict { PACKAGE_PIN V7       IOSTANDARD LVCMOS33 } [get_ports dp]
59
60  #set_property -dict { PACKAGE_PIN U2       IOSTANDARD LVCMOS33 } [get_ports {an[0]}]
61  #set_property -dict { PACKAGE_PIN U4       IOSTANDARD LVCMOS33 } [get_ports {an[1]}]
62  #set_property -dict { PACKAGE_PIN V4       IOSTANDARD LVCMOS33 } [get_ports {an[2]}]
63  #set_property -dict { PACKAGE_PIN W4       IOSTANDARD LVCMOS33 } [get_ports {an[3]}]
64
65
66  ##Buttons
67  set_property -dict { PACKAGE_PIN U18      IOSTANDARD LVCMOS33 } [get_ports rst]
68  #set_property -dict { PACKAGE_PIN T18      IOSTANDARD LVCMOS33 } [get_ports btnU]
69  #set_property -dict { PACKAGE_PIN W19      IOSTANDARD LVCMOS33 } [get_ports btnL]
70  #set_property -dict { PACKAGE_PIN T17      IOSTANDARD LVCMOS33 } [get_ports btnR]
71  #set_property -dict { PACKAGE_PIN U17      IOSTANDARD LVCMOS33 } [get_ports btnD]
72
73
74  ##Pmod Header JA
75  #set_property -dict { PACKAGE_PIN J1       IOSTANDARD LVCMOS33 } [get_ports {JA[0]}];#Sch name = JA1
76  #set_property -dict { PACKAGE_PIN L2       IOSTANDARD LVCMOS33 } [get_ports {JA[1]}];#Sch name = JA2
77  #set_property -dict { PACKAGE_PIN J2       IOSTANDARD LVCMOS33 } [get_ports {JA[2]}];#Sch name = JA3
78  #set_property -dict { PACKAGE_PIN G2       IOSTANDARD LVCMOS33 } [get_ports {JA[3]}];#Sch name = JA4
79  #set_property -dict { PACKAGE_PIN H1       IOSTANDARD LVCMOS33 } [get_ports {JA[4]}];#Sch name = JA7
80  #set_property -dict { PACKAGE_PIN K2       IOSTANDARD LVCMOS33 } [get_ports {JA[5]}];#Sch name = JA8
81  #set_property -dict { PACKAGE_PIN H2       IOSTANDARD LVCMOS33 } [get_ports {JA[6]}];#Sch name = JA9
82  #set_property -dict { PACKAGE_PIN G3       IOSTANDARD LVCMOS33 } [get_ports {JA[7]}];#Sch name = JA10
83
84  ##Pmod Header JB
85  #set_property -dict { PACKAGE_PIN A14      IOSTANDARD LVCMOS33 } [get_ports {JB[0]}];#Sch name = JB1
86  #set_property -dict { PACKAGE_PIN A16      IOSTANDARD LVCMOS33 } [get_ports {JB[1]}];#Sch name = JB2
87  #set_property -dict { PACKAGE_PIN B15      IOSTANDARD LVCMOS33 } [get_ports {JB[2]}];#Sch name = JB3
88  #set_property -dict { PACKAGE_PIN B16      IOSTANDARD LVCMOS33 } [get_ports {JB[3]}];#Sch name = JB4
89  #set_property -dict { PACKAGE_PIN A15      IOSTANDARD LVCMOS33 } [get_ports {JB[4]}];#Sch name = JB7
90  #set_property -dict { PACKAGE_PIN A17      IOSTANDARD LVCMOS33 } [get_ports {JB[5]}];#Sch name = JB8
91  #set_property -dict { PACKAGE_PIN C15      IOSTANDARD LVCMOS33 } [get_ports {JB[6]}];#Sch name = JB9
92  #set_property -dict { PACKAGE_PIN C16      IOSTANDARD LVCMOS33 } [get_ports {JB[7]}];#Sch name = JB10
93
94  ##Pmod Header JC
95  #set_property -dict { PACKAGE_PIN K17      IOSTANDARD LVCMOS33 } [get_ports {JC[0]}];#Sch name = JC1
96  #set_property -dict { PACKAGE_PIN M18      IOSTANDARD LVCMOS33 } [get_ports {JC[1]}];#Sch name = JC2
97  #set_property -dict { PACKAGE_PIN N17      IOSTANDARD LVCMOS33 } [get_ports {JC[2]}];#Sch name = JC3
98  #set_property -dict { PACKAGE_PIN P18      IOSTANDARD LVCMOS33 } [get_ports {JC[3]}];#Sch name = JC4
99  #set_property -dict { PACKAGE_PIN L17      IOSTANDARD LVCMOS33 } [get_ports {JC[4]}];#Sch name = JC7
100 #set_property -dict { PACKAGE_PIN M19      IOSTANDARD LVCMOS33 } [get_ports {JC[5]}];#Sch name = JC8
101 #set_property -dict { PACKAGE_PIN P17      IOSTANDARD LVCMOS33 } [get_ports {JC[6]}];#Sch name = JC9
102 #set_property -dict { PACKAGE_PIN R18      IOSTANDARD LVCMOS33 } [get_ports {JC[7]}];#Sch name = JC10
103
104  ##Pmod Header JXADC
105 #set_property -dict { PACKAGE_PIN J3       IOSTANDARD LVCMOS33 } [get_ports {JXADC[0]}];#Sch name = XA1_P
106 #set_property -dict { PACKAGE_PIN L3       IOSTANDARD LVCMOS33 } [get_ports {JXADC[1]}];#Sch name = XA2_P
107 #set_property -dict { PACKAGE_PIN M2       IOSTANDARD LVCMOS33 } [get_ports {JXADC[2]}];#Sch name = XA3_P
108 #set_property -dict { PACKAGE_PIN N2       IOSTANDARD LVCMOS33 } [get_ports {JXADC[3]}];#Sch name = XA4_P
109 #set_property -dict { PACKAGE_PIN K3       IOSTANDARD LVCMOS33 } [get_ports {JXADC[4]}];#Sch name = XA1_N

```

Figure (20): constraint file after edit some ports

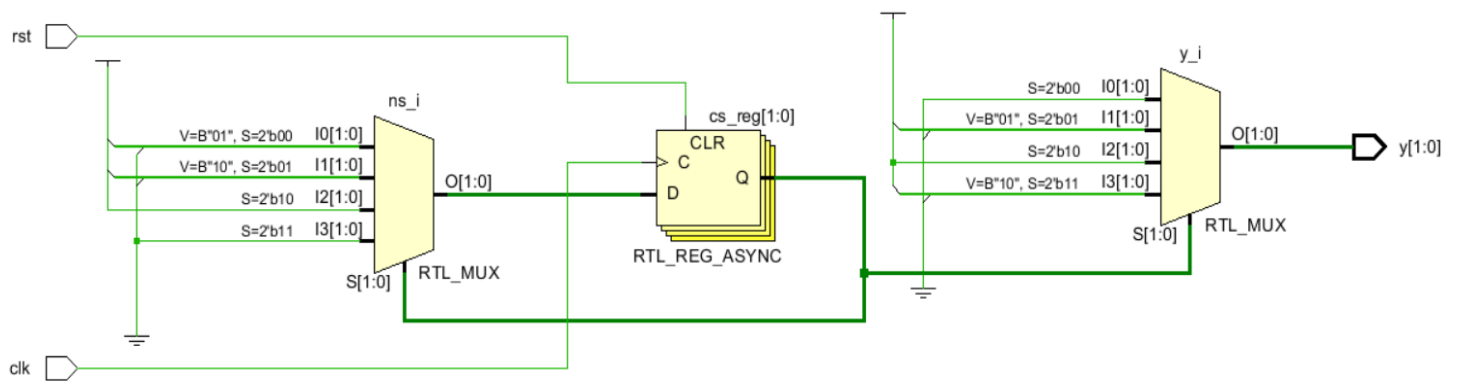


Figure (21): elaborated schematic

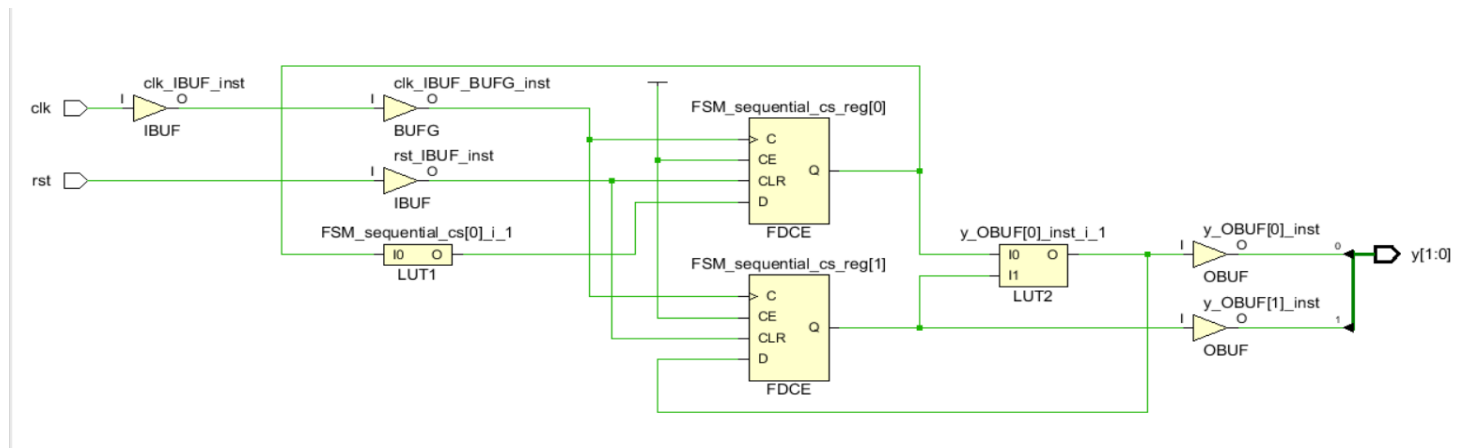


Figure (22): synthesis schematic

Utilization				
Hierarchy				
Summary				
▼ Slice Logic				
▼ Slice LUTs (<1%)				
LUT as Logic (<1%)				
▼ Slice Registers (<1%)				
Register as Flip Flop				
Memory				
DSP				
▼ IO and GT Specific				
▼ Bonded IOB (4%)				
IOB Master Pads				
IOB Slave Pads				
▼ Clocking				
BUFGCTRL (3%)				
Name	1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)
question3		2	2	4
				1

Figure (23): utilization report

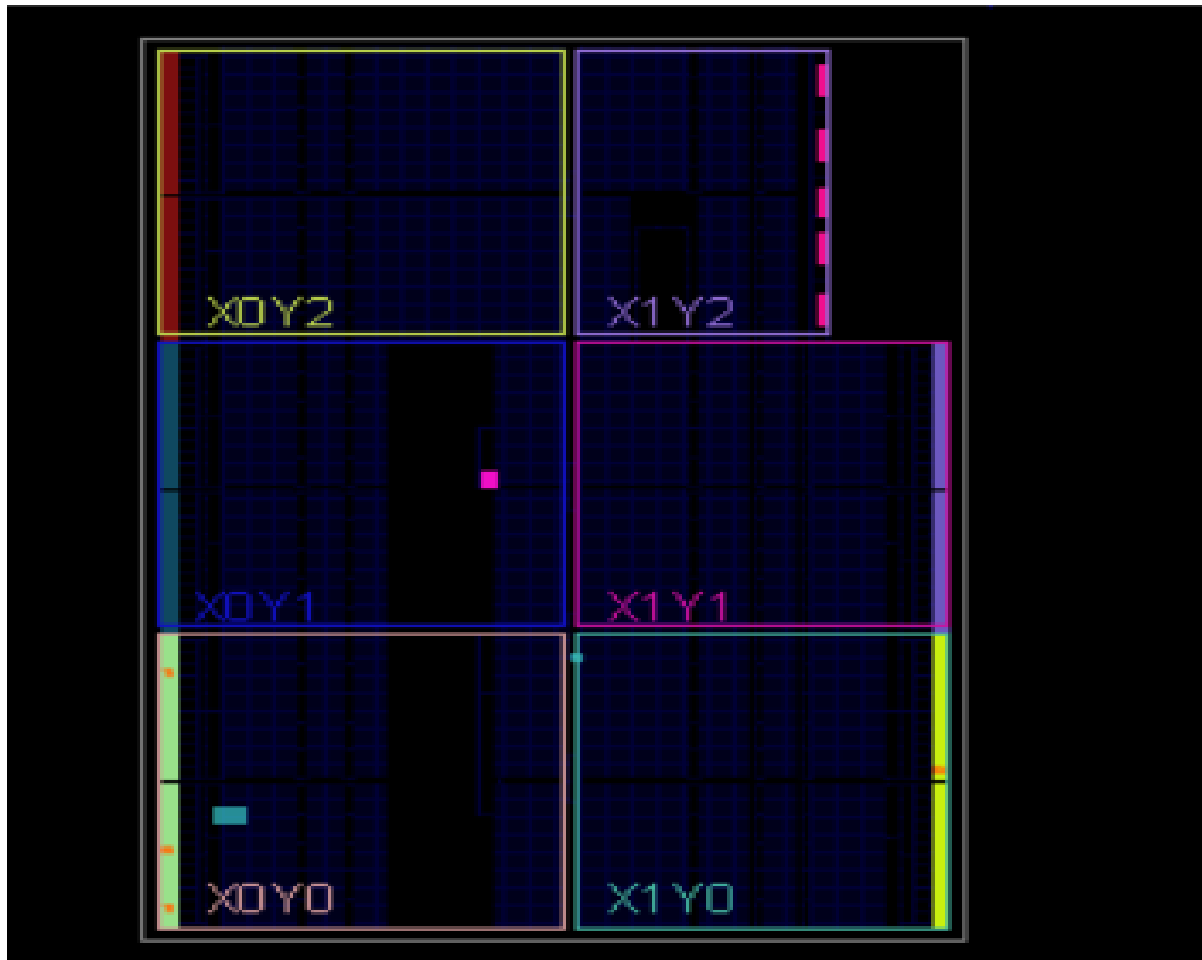


Figure (24): device implementation

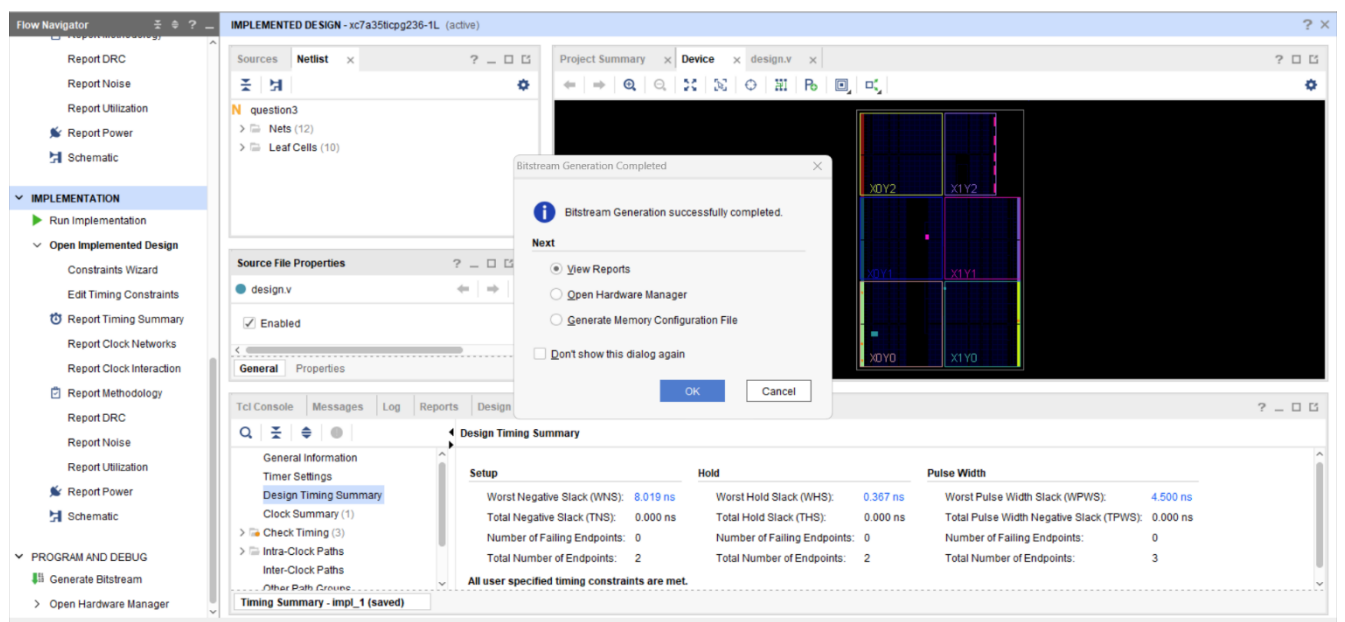


Figure (25): successful bitstream generation

#### Question4]

```
1  module question4(clk,rst,in,y);
2
3  parameter S0=2'b00;
4  parameter S1=2'b01;
5  parameter S2=2'b10;
6  parameter S3=2'b11;
7  input clk,rst,in;
8  output reg y;
9  reg [1:0] cs,ns;
10 // state memory block
11 always @(posedge clk or posedge rst) begin
12     if(rst)
13         cs<=S0;
14     else
15         cs<=ns;
16 end
17 // next state logic block
18 always @(*) begin
19     case(cs)
20         S0: begin
21             if(in)
22                 ns = S1;
23             else
24                 ns = S0;
25         end
26         S1: begin
27             if(in)
28                 ns = S2;
29             else
30                 ns = S1;
31         end
32         S2: begin
33             if(in)
34                 ns = S3;
35             else
36                 ns = S2;
37         end
38         S3: begin
39             if(in)
40                 ns = S1;
41             else
42                 ns = S0;
43         end
44     endcase
45 end
46 // output logic block
47 always @(*) begin
48     if(cs==S2 && in==1)
49         y=1;
50     else
51         y=0;
52 end
53 endmodule
```

Figure (26): design code

```
1  module question4_tb();
2
3  reg clk,rst,in;
4  wire out;
5  integer i=0;
6
7  question4 DUT1(.clk(clk),.rst(rst),.in(in),.y(out));
8
9  initial begin
10     clk=0;
11     forever
12         #25 clk=~clk;
13     end
14
15     initial begin
16         rst=1;
17         #100; //delay for the first two clock cycles
18         rst=0;
19         for(i=0;i<50;i=i+1) begin
20             @(negedge clk);
21             in=$random;
22         end
23         $stop;
24     end
25
26 endmodule
```

Figure (27): testbench code



```

1  ## This file is a general .xdc for the Basys3 rev B board
2  ## To use it in a project:
3  ## - uncomment the lines corresponding to used pins
4  ## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project
5
6  ## Clock signal
7  set_property -dict { PACKAGE_PIN W5      IOSTANDARD LVCMOS33 } [get_ports clk]
8  create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
9
10
11 ## Switches
12 set_property -dict { PACKAGE_PIN V17      IOSTANDARD LVCMOS33 } [get_ports {in}]
13 #set_property -dict { PACKAGE_PIN V16      IOSTANDARD LVCMOS33 } [get_ports {sw[1]}]
14 #set_property -dict { PACKAGE_PIN W16      IOSTANDARD LVCMOS33 } [get_ports {sw[2]}]
15 #set_property -dict { PACKAGE_PIN W17      IOSTANDARD LVCMOS33 } [get_ports {sw[3]}]
16 #set_property -dict { PACKAGE_PIN W15      IOSTANDARD LVCMOS33 } [get_ports {sw[4]}]
17 #set_property -dict { PACKAGE_PIN V15      IOSTANDARD LVCMOS33 } [get_ports {sw[5]}]
18 #set_property -dict { PACKAGE_PIN W14      IOSTANDARD LVCMOS33 } [get_ports {sw[6]}]
19 #set_property -dict { PACKAGE_PIN W13      IOSTANDARD LVCMOS33 } [get_ports {sw[7]}]
20 #set_property -dict { PACKAGE_PIN V2       IOSTANDARD LVCMOS33 } [get_ports {sw[8]}]
21 #set_property -dict { PACKAGE_PIN T3       IOSTANDARD LVCMOS33 } [get_ports {sw[9]}]
22 #set_property -dict { PACKAGE_PIN T2       IOSTANDARD LVCMOS33 } [get_ports {sw[10]}]
23 #set_property -dict { PACKAGE_PIN R3       IOSTANDARD LVCMOS33 } [get_ports {sw[11]}]
24 #set_property -dict { PACKAGE_PIN W2       IOSTANDARD LVCMOS33 } [get_ports {sw[12]}]
25 #set_property -dict { PACKAGE_PIN U1       IOSTANDARD LVCMOS33 } [get_ports {sw[13]}]
26 #set_property -dict { PACKAGE_PIN T1       IOSTANDARD LVCMOS33 } [get_ports {sw[14]}]
27 #set_property -dict { PACKAGE_PIN R2       IOSTANDARD LVCMOS33 } [get_ports {sw[15]}]
28
29
30 ## LEDs
31 set_property -dict { PACKAGE_PIN U16      IOSTANDARD LVCMOS33 } [get_ports {y}]
32 #set_property -dict { PACKAGE_PIN E19      IOSTANDARD LVCMOS33 } [get_ports {led[1]}]
33 #set_property -dict { PACKAGE_PIN U19      IOSTANDARD LVCMOS33 } [get_ports {led[2]}]
34 #set_property -dict { PACKAGE_PIN V19      IOSTANDARD LVCMOS33 } [get_ports {led[3]}]
35 #set_property -dict { PACKAGE_PIN W18      IOSTANDARD LVCMOS33 } [get_ports {led[4]}]
36 #set_property -dict { PACKAGE_PIN U15      IOSTANDARD LVCMOS33 } [get_ports {led[5]}]
37 #set_property -dict { PACKAGE_PIN U14      IOSTANDARD LVCMOS33 } [get_ports {led[6]}]
38 #set_property -dict { PACKAGE_PIN V14      IOSTANDARD LVCMOS33 } [get_ports {led[7]}]
39 #set_property -dict { PACKAGE_PIN V13      IOSTANDARD LVCMOS33 } [get_ports {led[8]}]
40 #set_property -dict { PACKAGE_PIN V3       IOSTANDARD LVCMOS33 } [get_ports {led[9]}]
41 #set_property -dict { PACKAGE_PIN W3       IOSTANDARD LVCMOS33 } [get_ports {led[10]}]
42 #set_property -dict { PACKAGE_PIN U3       IOSTANDARD LVCMOS33 } [get_ports {led[11]}]
43 #set_property -dict { PACKAGE_PIN P3       IOSTANDARD LVCMOS33 } [get_ports {led[12]}]
44 #set_property -dict { PACKAGE_PIN N3       IOSTANDARD LVCMOS33 } [get_ports {led[13]}]
45 #set_property -dict { PACKAGE_PIN P1       IOSTANDARD LVCMOS33 } [get_ports {led[14]}]
46 #set_property -dict { PACKAGE_PIN L1       IOSTANDARD LVCMOS33 } [get_ports {led[15]}]
47
48
49 ##7 Segment Display
50 #set_property -dict { PACKAGE_PIN W7       IOSTANDARD LVCMOS33 } [get_ports {seg[0]}]
51 #set_property -dict { PACKAGE_PIN W6       IOSTANDARD LVCMOS33 } [get_ports {seg[1]}]
52 #set_property -dict { PACKAGE_PIN U8       IOSTANDARD LVCMOS33 } [get_ports {seg[2]}]
53 #set_property -dict { PACKAGE_PIN V8       IOSTANDARD LVCMOS33 } [get_ports {seg[3]}]
54 #set_property -dict { PACKAGE_PIN U5       IOSTANDARD LVCMOS33 } [get_ports {seg[4]}]
55 #set_property -dict { PACKAGE_PIN V5       IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]
56
57 #set_property -dict { PACKAGE_PIN V5       IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]
58 #set_property -dict { PACKAGE_PIN U7       IOSTANDARD LVCMOS33 } [get_ports {seg[6]}]
59
60 #set_property -dict { PACKAGE_PIN V7       IOSTANDARD LVCMOS33 } [get_ports dp]
61
62 #set_property -dict { PACKAGE_PIN U2       IOSTANDARD LVCMOS33 } [get_ports {an[0]}]
63 #set_property -dict { PACKAGE_PIN U4       IOSTANDARD LVCMOS33 } [get_ports {an[1]}]
64 #set_property -dict { PACKAGE_PIN V4       IOSTANDARD LVCMOS33 } [get_ports {an[2]}]
65 #set_property -dict { PACKAGE_PIN W4       IOSTANDARD LVCMOS33 } [get_ports {an[3]}]
66
67 ##Buttons
68 set_property -dict { PACKAGE_PIN U18      IOSTANDARD LVCMOS33 } [get_ports rst]
69 #set_property -dict { PACKAGE_PIN T18      IOSTANDARD LVCMOS33 } [get_ports btnA]
70 #set_property -dict { PACKAGE_PIN T17      IOSTANDARD LVCMOS33 } [get_ports btnB]
71 #set_property -dict { PACKAGE_PIN U17      IOSTANDARD LVCMOS33 } [get_ports btnD]
72
73
74 ##Pmod Header JA
75 #set_property -dict { PACKAGE_PIN J1       IOSTANDARD LVCMOS33 } [get_ports {JA[0]}];#Sch name = JA1
76 #set_property -dict { PACKAGE_PIN L2       IOSTANDARD LVCMOS33 } [get_ports {JA[1]}];#Sch name = JA2
77 #set_property -dict { PACKAGE_PIN J2       IOSTANDARD LVCMOS33 } [get_ports {JA[2]}];#Sch name = JA3
78 #set_property -dict { PACKAGE_PIN G2       IOSTANDARD LVCMOS33 } [get_ports {JA[3]}];#Sch name = JA4
79 #set_property -dict { PACKAGE_PIN H1       IOSTANDARD LVCMOS33 } [get_ports {JA[4]}];#Sch name = JA7
80 #set_property -dict { PACKAGE_PIN K2       IOSTANDARD LVCMOS33 } [get_ports {JA[5]}];#Sch name = JA8
81 #set_property -dict { PACKAGE_PIN H2       IOSTANDARD LVCMOS33 } [get_ports {JA[6]}];#Sch name = JA9
82 #set_property -dict { PACKAGE_PIN G3       IOSTANDARD LVCMOS33 } [get_ports {JA[7]}];#Sch name = JA10
83
84 ##Pmod Header JB
85 #set_property -dict { PACKAGE_PIN A14      IOSTANDARD LVCMOS33 } [get_ports {JB[0]}];#Sch name = JB1
86 #set_property -dict { PACKAGE_PIN A16      IOSTANDARD LVCMOS33 } [get_ports {JB[1]}];#Sch name = JB2
87 #set_property -dict { PACKAGE_PIN B15      IOSTANDARD LVCMOS33 } [get_ports {JB[2]}];#Sch name = JB3
88 #set_property -dict { PACKAGE_PIN B16      IOSTANDARD LVCMOS33 } [get_ports {JB[3]}];#Sch name = JB4
89 #set_property -dict { PACKAGE_PIN A15      IOSTANDARD LVCMOS33 } [get_ports {JB[4]}];#Sch name = JB7
90 #set_property -dict { PACKAGE_PIN A17      IOSTANDARD LVCMOS33 } [get_ports {JB[5]}];#Sch name = JB8
91 #set_property -dict { PACKAGE_PIN C15      IOSTANDARD LVCMOS33 } [get_ports {JB[6]}];#Sch name = JB9
92 #set_property -dict { PACKAGE_PIN C16      IOSTANDARD LVCMOS33 } [get_ports {JB[7]}];#Sch name = JB10
93
94 ##Pmod Header JC
95 #set_property -dict { PACKAGE_PIN K17      IOSTANDARD LVCMOS33 } [get_ports {JC[0]}];#Sch name = JC1
96 #set_property -dict { PACKAGE_PIN M18      IOSTANDARD LVCMOS33 } [get_ports {JC[1]}];#Sch name = JC2
97 #set_property -dict { PACKAGE_PIN N17      IOSTANDARD LVCMOS33 } [get_ports {JC[2]}];#Sch name = JC3
98 #set_property -dict { PACKAGE_PIN P18      IOSTANDARD LVCMOS33 } [get_ports {JC[3]}];#Sch name = JC4
99 #set_property -dict { PACKAGE_PIN L17      IOSTANDARD LVCMOS33 } [get_ports {JC[4]}];#Sch name = JC7
100 #set_property -dict { PACKAGE_PIN M19      IOSTANDARD LVCMOS33 } [get_ports {JC[5]}];#Sch name = JC8
101 #set_property -dict { PACKAGE_PIN P17      IOSTANDARD LVCMOS33 } [get_ports {JC[6]}];#Sch name = JC9
102 #set_property -dict { PACKAGE_PIN R18      IOSTANDARD LVCMOS33 } [get_ports {JC[7]}];#Sch name = JC10
103
104 ##Pmod Header JXADC
105 #set_property -dict { PACKAGE_PIN J3       IOSTANDARD LVCMOS33 } [get_ports {JXADC[0]}];#Sch name = XA1_P
106 #set_property -dict { PACKAGE_PIN L3       IOSTANDARD LVCMOS33 } [get_ports {JXADC[1]}];#Sch name = XA2_P
107 #set_property -dict { PACKAGE_PIN M2       IOSTANDARD LVCMOS33 } [get_ports {JXADC[2]}];#Sch name = XA3_P
108 #set_property -dict { PACKAGE_PIN N2       IOSTANDARD LVCMOS33 } [get_ports {JXADC[3]}];#Sch name = XA4_P
109 #set_property -dict { PACKAGE_PIN K3       IOSTANDARD LVCMOS33 } [get_ports {JXADC[4]}];#Sch name = XA1_N

```

Figure (28): constraint file

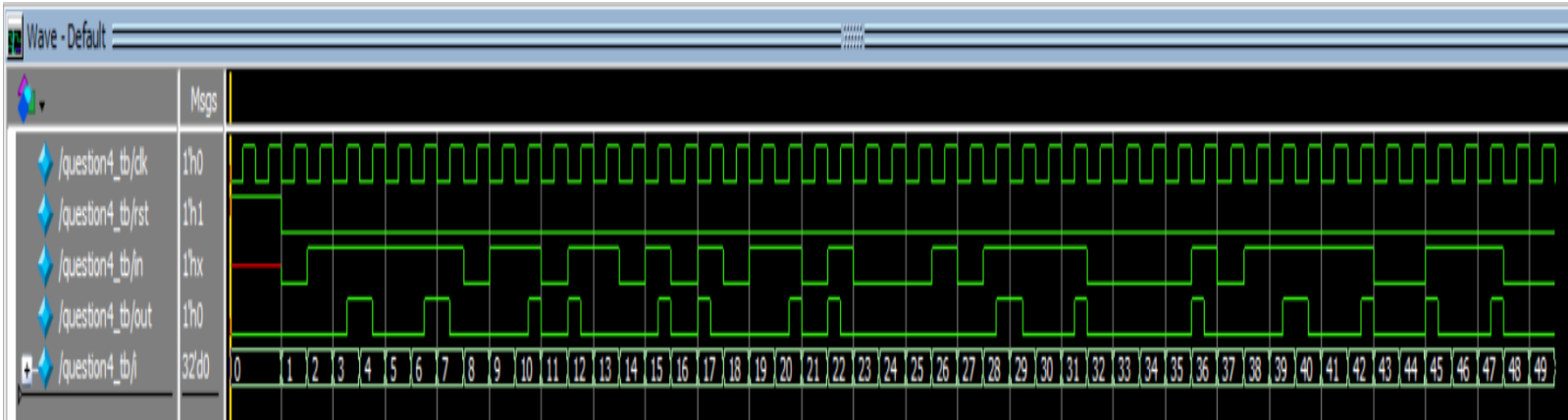


Figure (29): waveform snippet

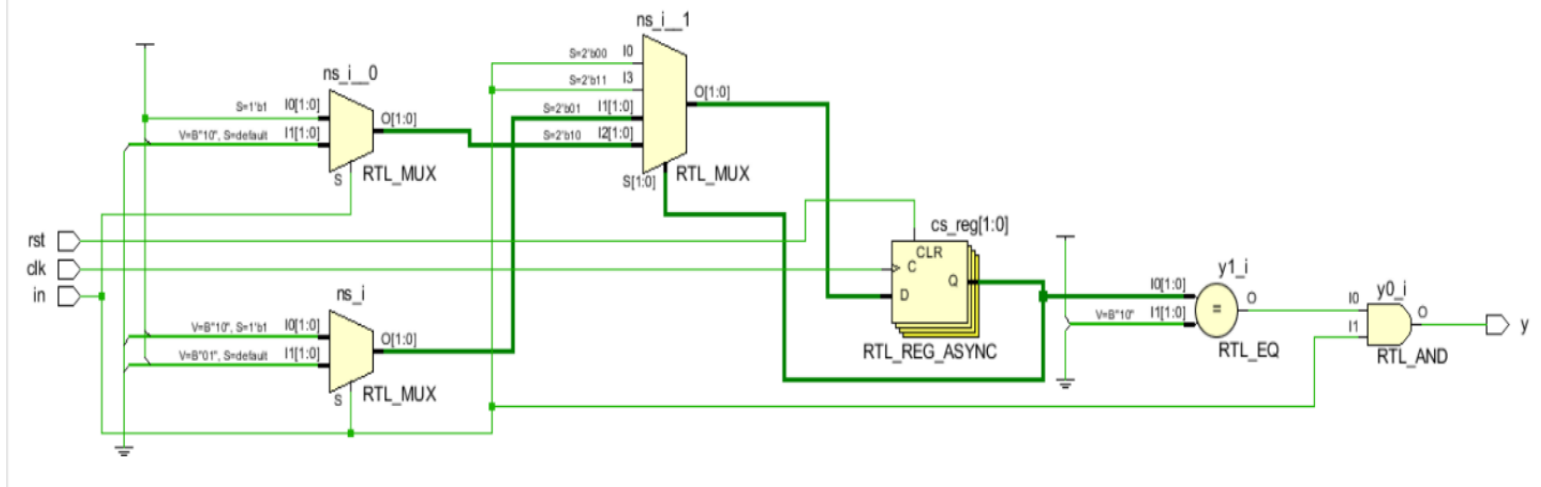


Figure (30): elaborated schematic

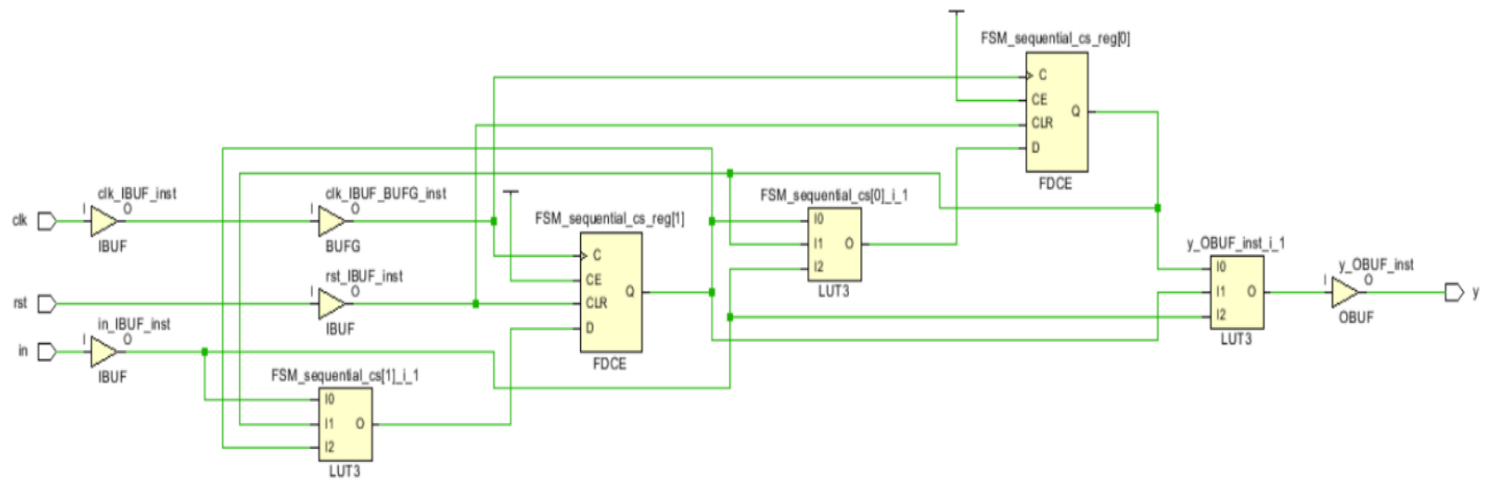


Figure (30): synthesis schematic



Utilization				
Hierarchy				
Name	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
question4	3	2	4	1

Figure (31): report utilization

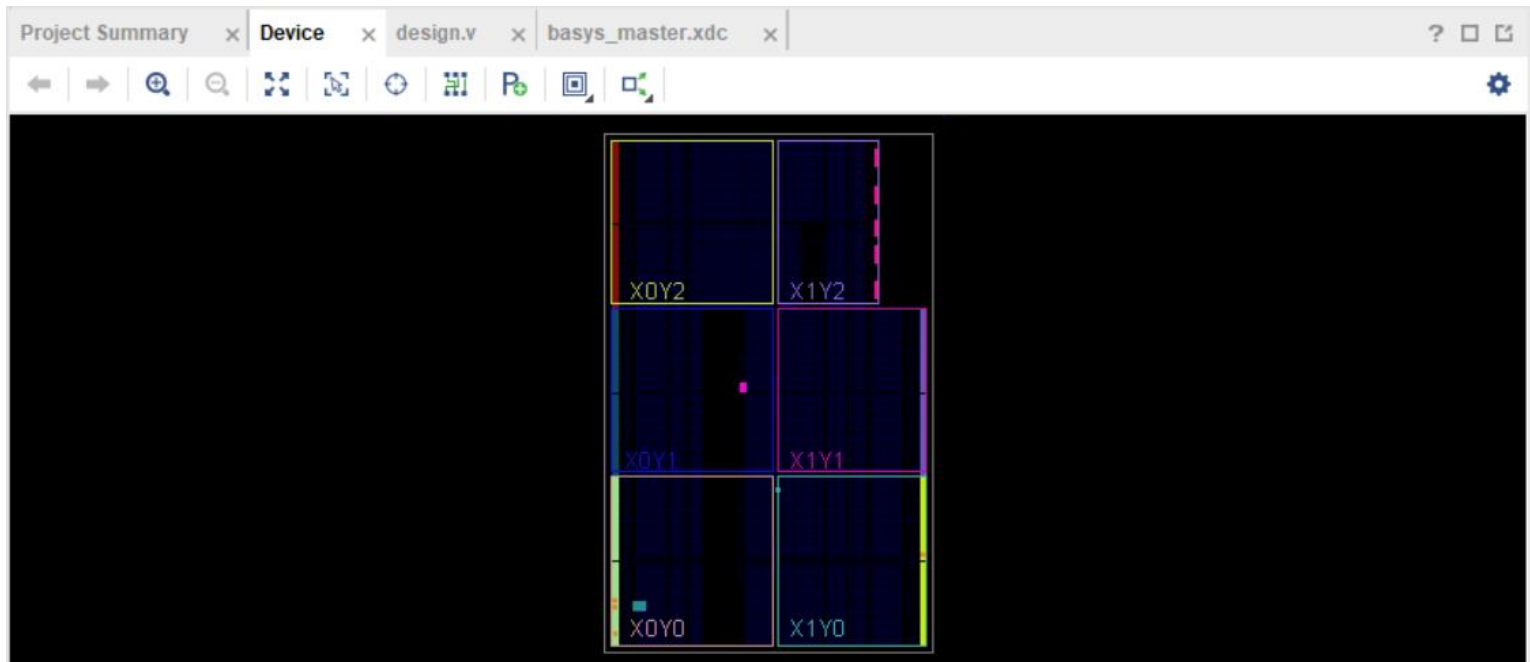


Figure (32): device implementation

IMPLEMENTED DESIGN - xc7a35ticipg236-1L (active)

Sources: Netlist x

question4

- Nets (14)
- Leaf Cells (11)

Source File Properties

basys\_master.xdc

Enabled

General Properties

Tcl Console Messages Log Reports Design

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 8.733 ns	Worst Hold Slack (WHS): 0.279 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: 2	Total Number of Endpoints: 2	Total Number of Endpoints: 3

All user specified timing constraints are met.

Timing Summary - impl\_1 (saved)

Figure (33): successful bitstream generation

## Question5]

```
1  module ALSU(A,B,Cin,opcode,serial_in,direction,red_op_A,red_op_B,bypass_A,bypass_B,clk,rst,leds);
2
3  parameter INPUT_PRIORITY = "A";
4  parameter FULL_ADDER="ON";
5
6  input [2:0] A,B,opcode;
7  input Cin,serial_in,direction,red_op_A,red_op_B,bypass_A,bypass_B,clk,rst;
8
9  reg [2:0] A_ff,B_ff,opcode_ff;
10 ▼ reg Cin_ff,serial_in_ff,direction_ff,red_op_A_ff,red_op_B_ff,bypass_A_ff,bypass_B_ff;
11
12     reg [5:0] out;
13     output reg [15:0] leds;
14
15     wire invalid_opcode,invalid_red_op,invalid_overall;
16     wire [5:0] mult_out;
17     wire [3:0] sum_h;
18     wire [3:0] sum_f;
19
20     assign invalid_opcode = opcode_ff[2] & opcode_ff[1];
21     assign invalid_red_op =(red_op_A_ff | red_op_B_ff) & (opcode_ff[2] | opcode_ff[1]);
22     assign invalid_overall = invalid_opcode | invalid_red_op;
23
24 ▼ mult_gen_0 Multiplier (
25     .A(A_ff), // input wire [2 : 0] A
26     .B(B_ff), // input wire [2 : 0] B
27     .P(mult_out) // output wire [5 : 0] P
28 );
29
30 ▼ c_addsub_0 half_adder (
31     .A(A_ff), // input wire [2 : 0] A
32     .B(B_ff), // input wire [2 : 0] B
33     .S(sum_h) // output wire [3 : 0] S
34 );
35
36 ▼ c_addsub_1 full_adder (
37     .A(A_ff), // input wire [2 : 0] A
38     .B(B_ff), // input wire [2 : 0] B
39     .C_IN(Cin_ff), // input wire C_IN
40     .S(sum_f) // output wire [3 : 0] S
41 );
42
43
44 // always block specilaized for inputs flip flops
45
46 ▼ always @(posedge clk or posedge rst) begin
47 ▼     if(rst) begin
48         A_ff<=0;
49         B_ff<=0;
50         opcode_ff<=0;
51         Cin_ff<=0;
52         serial_in_ff<=0;
53         direction_ff<=0;
54         red_op_A_ff<=0;
55         red_op_B_ff<=0;
```

Figure (34): design code

```

56         bypass_A_ff<=0;
57         bypass_B_ff<=0;
58     end
59     else begin
60         A_ff<=A;
61         B_ff<=B;
62         opcode_ff<=opcode;
63         Cin_ff<=Cin;
64         serial_in_ff<=serial_in;
65         direction_ff<=direction;
66         red_op_A_ff<=red_op_A;
67         red_op_B_ff<=red_op_B;
68         bypass_A_ff<=bypass_A;
69         bypass_B_ff<=bypass_B;
70     end
71 end
72
73 // always block specialized for leds
74
75 always @(posedge clk or posedge rst) begin
76
77     if(rst)
78         leds<=0;
79     else if(invalid_overall==1)
80         leds<=~leds;
81     else
82         leds<=0;
83 end
84
85 // always block for ALSU output
86
87 always @(posedge clk or posedge rst) begin
88     if(rst)
89         out<=0;
90     else if(invalid_overall)
91         out<=0;
92     else if(bypass_A_ff && bypass_B_ff)
93         out<= (INPUT_PRIORITY=="A")?A_ff:B_ff;
94     else if(bypass_A_ff)
95         out<=A_ff;
96     else if(bypass_B_ff)
97         out<=B_ff;
98     else begin
99         case(opcode_ff)
100
101             3'b000:begin
102                 case({red_op_A_ff,red_op_B_ff})
103                     2'b00:out <= A_ff & B_ff;
104                     2'b01:out <= & B_ff;
105                     2'b10:out <= & A_ff;
106                     2'b11:out <= (INPUT_PRIORITY=="A")?(& A_ff):(& B_ff);
107                 endcase
108             end
109
110             3'b001:begin
111                 case({red_op_A_ff,red_op_B_ff})
112                     2'b00:out <= A_ff ^ B_ff;
113                     2'b01:out <= ^ B_ff;
114                     2'b10:out <= ^ A_ff;
115                     2'b11:out <= (INPUT_PRIORITY=="A")?(^ A_ff):(^ B_ff);
116                 endcase
117             end
118
119             3'b010:begin
120                 case(FULL_ADDER)
121                     "OFF" : out<= {2'b00 , sum_h};
122                     "ON"  : out<= {2'b00 , sum_f};
123                 endcase
124             end
125
126             3'b011: out<= mult_out;
127
128             3'b100:begin
129                 if(direction_ff)
130                     out <= {out[4:0] , serial_in_ff};
131                 else
132                     out <= {serial_in_ff , out[5:1]};
133             end
134
135             3'b101:begin
136                 if(direction_ff)
137                     out <= {out[4:0] , out[5]};
138                 else
139                     out <= {out[0] , out[5:1]};
140             end
141
142             default:out<=0;
143         endcase
144     end
145 end
146
147 endmodule
148

```

Figure (35): continue\_design\_code.

```

1  ## This file is a general .xdc for the Basys3 rev B board
2  ## To use it in a project:
3  ## - uncomment the lines corresponding to used pins
4  ## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project
5
6  ## Clock signal
7  set_property -dict { PACKAGE_PIN W5    IOSTANDARD LVCMOS33 } [get_ports clk]
8  create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
9
10
11  ## Switches
12  set_property -dict { PACKAGE_PIN V17    IOSTANDARD LVCMOS33 } [get_ports {opcode[0]}]
13  set_property -dict { PACKAGE_PIN V16    IOSTANDARD LVCMOS33 } [get_ports {opcode[1]}]
14  set_property -dict { PACKAGE_PIN W16    IOSTANDARD LVCMOS33 } [get_ports {opcode[2]}]
15  set_property -dict { PACKAGE_PIN W17    IOSTANDARD LVCMOS33 } [get_ports {A[0]}]
16  set_property -dict { PACKAGE_PIN W15    IOSTANDARD LVCMOS33 } [get_ports {A[1]}]
17  set_property -dict { PACKAGE_PIN V15    IOSTANDARD LVCMOS33 } [get_ports {A[2]}]
18  set_property -dict { PACKAGE_PIN W14    IOSTANDARD LVCMOS33 } [get_ports {B[0]}]
19  set_property -dict { PACKAGE_PIN W13    IOSTANDARD LVCMOS33 } [get_ports {B[1]}]
20  set_property -dict { PACKAGE_PIN V2     IOSTANDARD LVCMOS33 } [get_ports {B[2]}]
21  set_property -dict { PACKAGE_PIN T3     IOSTANDARD LVCMOS33 } [get_ports {Cin}]
22  set_property -dict { PACKAGE_PIN T2     IOSTANDARD LVCMOS33 } [get_ports {red_op_A}]
23  set_property -dict { PACKAGE_PIN R3     IOSTANDARD LVCMOS33 } [get_ports {red_op_B}]
24  set_property -dict { PACKAGE_PIN W2     IOSTANDARD LVCMOS33 } [get_ports {bypass_A}]
25  set_property -dict { PACKAGE_PIN U1     IOSTANDARD LVCMOS33 } [get_ports {bypass_B}]
26  set_property -dict { PACKAGE_PIN T1     IOSTANDARD LVCMOS33 } [get_ports {direction}]
27  set_property -dict { PACKAGE_PIN R2     IOSTANDARD LVCMOS33 } [get_ports {serial_in}]
28
29
30  ## LEDs
31  set_property -dict { PACKAGE_PIN U16    IOSTANDARD LVCMOS33 } [get_ports {leds[0]}]
32  set_property -dict { PACKAGE_PIN E19    IOSTANDARD LVCMOS33 } [get_ports {leds[1]}]
33  set_property -dict { PACKAGE_PIN U19    IOSTANDARD LVCMOS33 } [get_ports {leds[2]}]
34  set_property -dict { PACKAGE_PIN V19    IOSTANDARD LVCMOS33 } [get_ports {leds[3]}]
35  set_property -dict { PACKAGE_PIN W18    IOSTANDARD LVCMOS33 } [get_ports {leds[4]}]
36  set_property -dict { PACKAGE_PIN U15    IOSTANDARD LVCMOS33 } [get_ports {leds[5]}]
37  set_property -dict { PACKAGE_PIN U14    IOSTANDARD LVCMOS33 } [get_ports {leds[6]}]
38  set_property -dict { PACKAGE_PIN V14    IOSTANDARD LVCMOS33 } [get_ports {leds[7]}]
39  set_property -dict { PACKAGE_PIN V13    IOSTANDARD LVCMOS33 } [get_ports {leds[8]}]
40  set_property -dict { PACKAGE_PIN V3     IOSTANDARD LVCMOS33 } [get_ports {leds[9]}]
41  set_property -dict { PACKAGE_PIN W3     IOSTANDARD LVCMOS33 } [get_ports {leds[10]}]
42  set_property -dict { PACKAGE_PIN U3     IOSTANDARD LVCMOS33 } [get_ports {leds[11]}]
43  set_property -dict { PACKAGE_PIN P3     IOSTANDARD LVCMOS33 } [get_ports {leds[12]}]
44  set_property -dict { PACKAGE_PIN N3     IOSTANDARD LVCMOS33 } [get_ports {leds[13]}]
45  set_property -dict { PACKAGE_PIN P1     IOSTANDARD LVCMOS33 } [get_ports {leds[14]}]
46  set_property -dict { PACKAGE_PIN L1     IOSTANDARD LVCMOS33 } [get_ports {leds[15]}]
47
48
49  ##7 Segment Display
50  #set_property -dict { PACKAGE_PIN W7    IOSTANDARD LVCMOS33 } [get_ports {seg[0]}]
51  #set_property -dict { PACKAGE_PIN W6    IOSTANDARD LVCMOS33 } [get_ports {seg[1]}]
52  #set_property -dict { PACKAGE_PIN U8    IOSTANDARD LVCMOS33 } [get_ports {seg[2]}]
53  #set_property -dict { PACKAGE_PIN V8    IOSTANDARD LVCMOS33 } [get_ports {seg[3]}]
54  #set_property -dict { PACKAGE_PIN U5    IOSTANDARD LVCMOS33 } [get_ports {seg[4]}]
55  #set_property -dict { PACKAGE_PIN V5    IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]

```

```

##Buttons
set_property -dict { PACKAGE_PIN U18    IOSTANDARD LVCMOS33 } [get_ports rst]
#set_property -dict { PACKAGE_PIN T18    IOSTANDARD LVCMOS33 } [get_ports btnU]
#set_property -dict { PACKAGE_PIN W19    IOSTANDARD LVCMOS33 } [get_ports btnL]
#set_property -dict { PACKAGE_PIN T17    IOSTANDARD LVCMOS33 } [get_ports btnR]
#set_property -dict { PACKAGE_PIN U17    IOSTANDARD LVCMOS33 } [get_ports btnD]

```

Figure (36): constraint file after editing the switches and LEDs and push button for reset.

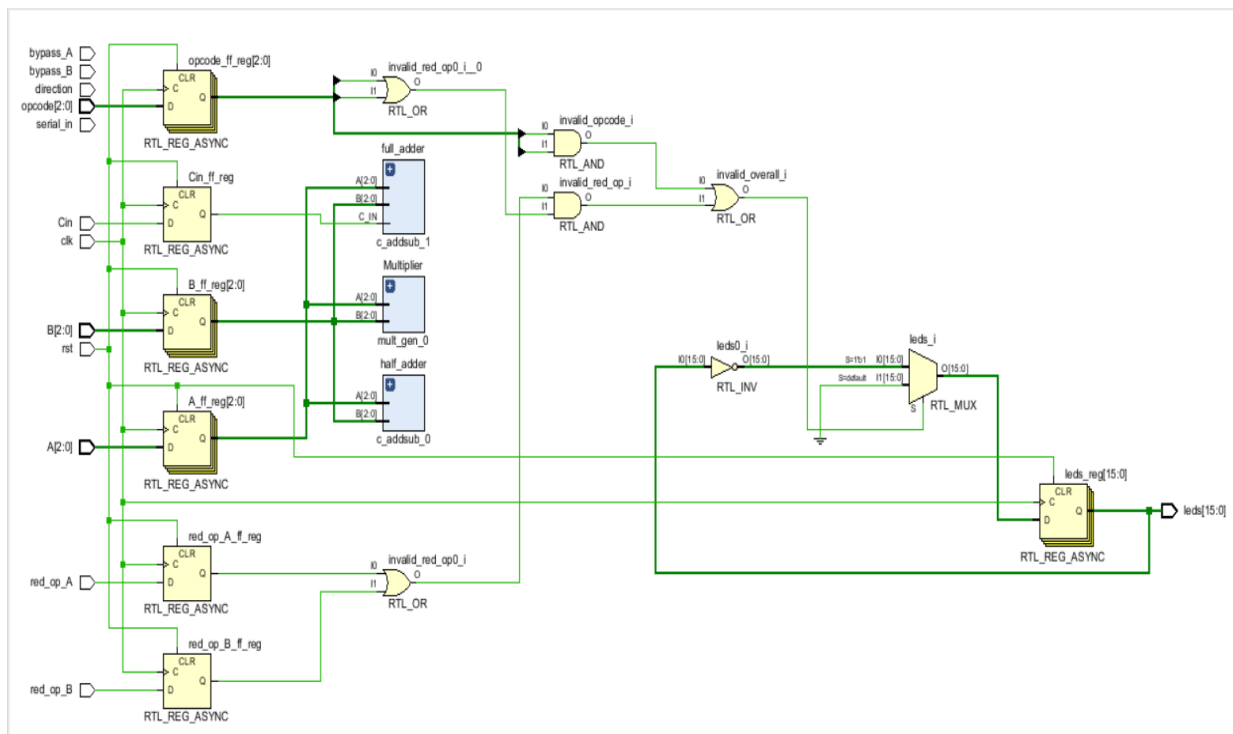


Figure (37): generic schematic (Elaborated)

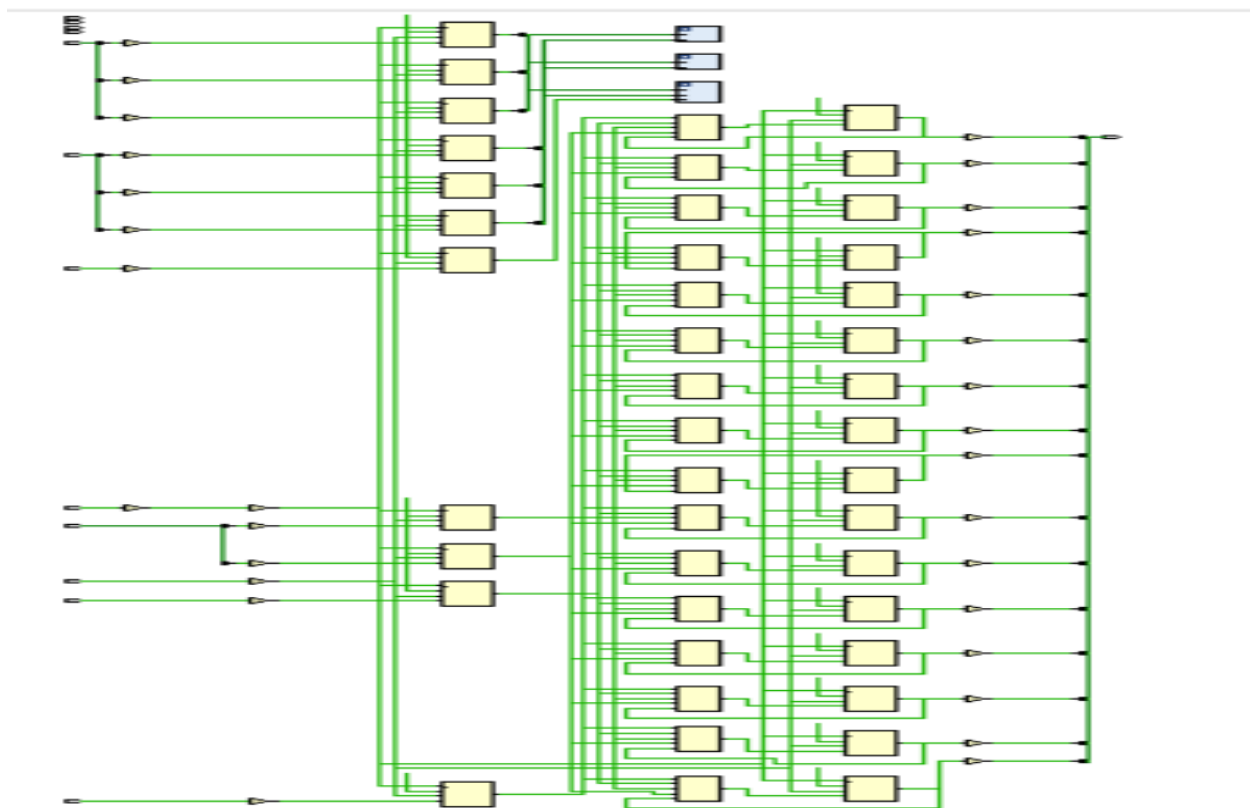


Figure (38): Synthesized schematic




Name	1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
✓ <b>N</b> ALSU		57	38	40	1
>  full_adder (c_addsub_1)		3	0	0	0
>  half_adder (c_addsub...		3	0	0	0
>  Multiplier (mult_gen_0)		11	0	0	0

Figure (39): utilization report

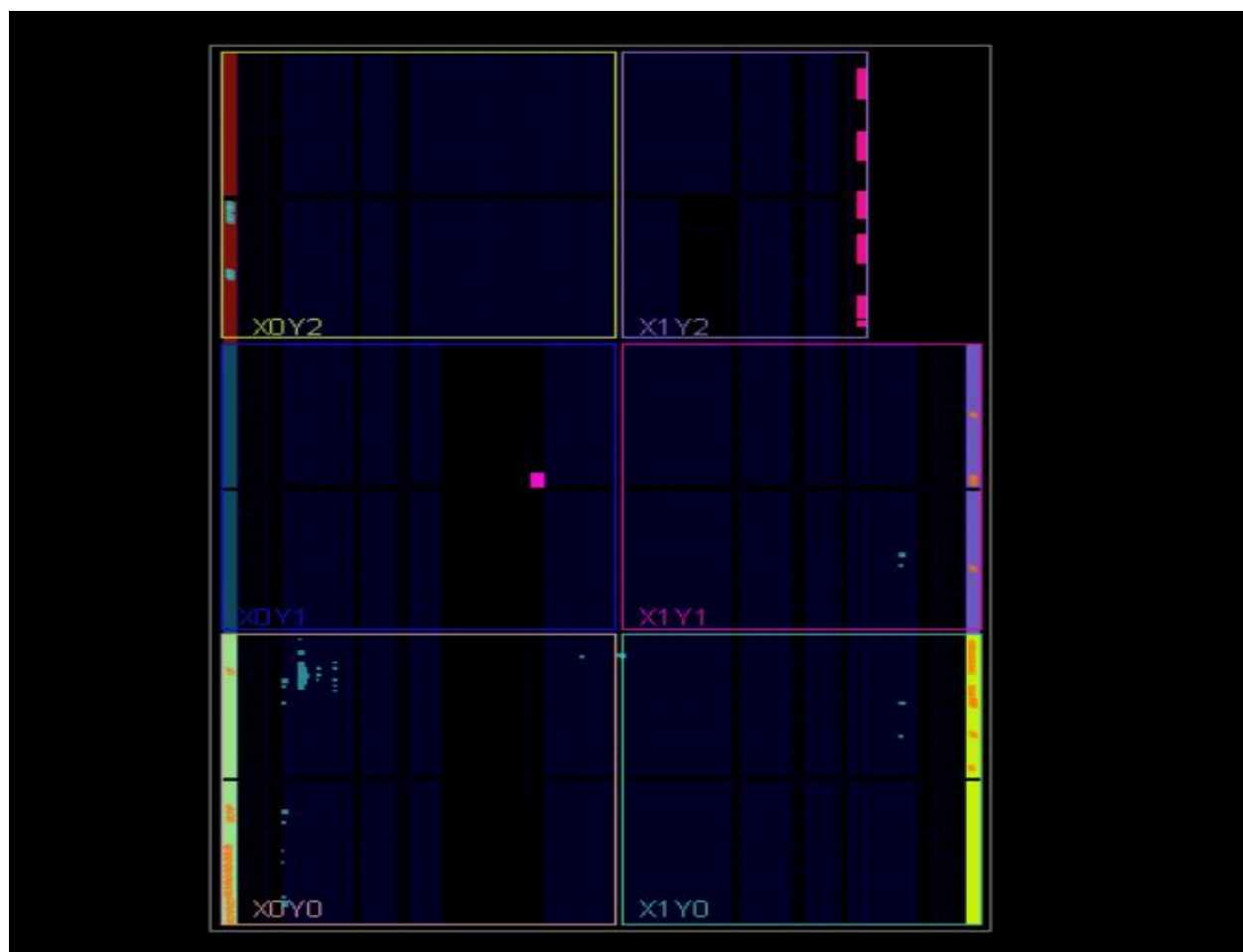


Figure (40): device after implementation

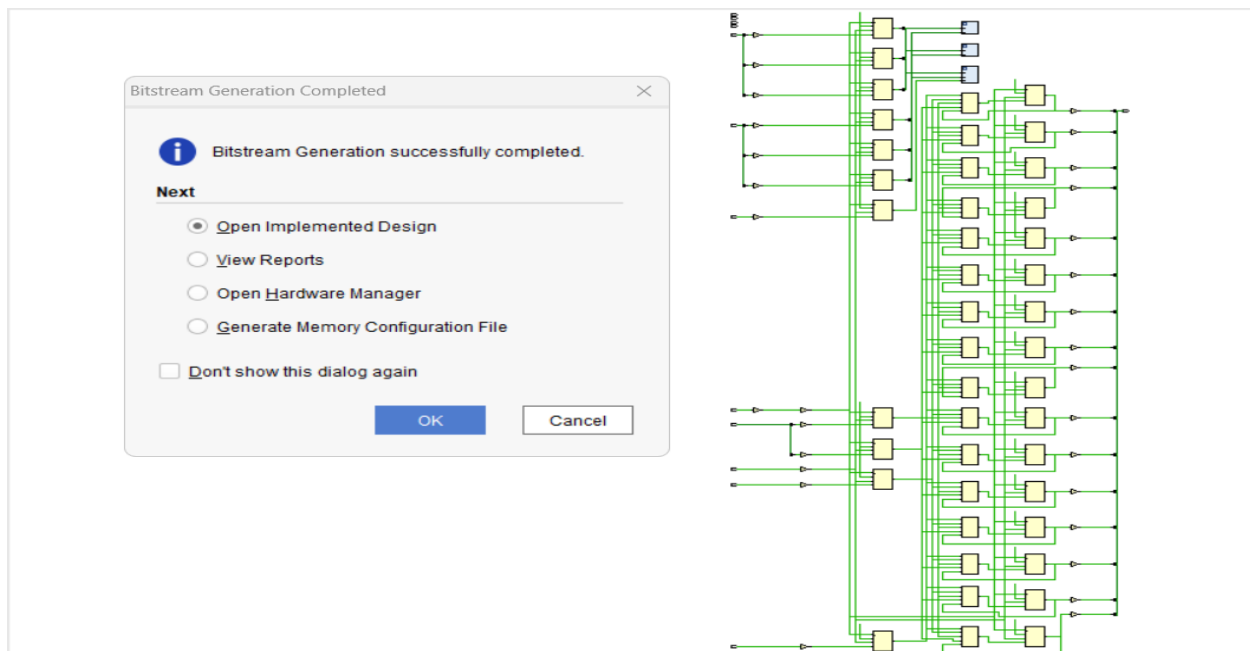


Figure (41): successful bitstream generation