Assignment_6

(Under supervision of Eng/ Karim Wassem)

Question1]

```
module question1(x,clk,rst,y,count);

parameter IDLE=2'b00;
parameter ZERO=2'b01;
parameter ONE=2'b10;
parameter STORE=2'b11;

input x,clk,rst;
output reg y;
output reg [9:0] count;

reg [1:0] cs,ns;

//state memory

always @(posedge clk or posedge rst) begin
    if(rst) begin
    cs<=IDLE;
    count<=10'b0000000000;
    end
    else
    cs<=ns;
end

// next state logic

always @(*) begin
    case(cs)
    IDLE: begin
        if(x)
        ns=IDLE;
        else
        ns=ZERO;
    end
    ZERO:begin
        if(x)
        ns=ONE;
        else
        ns=ZERO;
    end
    ONE:begin
        if(x)
        ns=IDLE;
        else
        ns=STORE;
    end
    STORE:begin
        if(x)
        ns=IDLE;
        else
        ns=ZERO;
    end
    default: ns=IDLE;
    endcase
end

// output logic block

always @(*) begin
    case(cs)
        IDLE: y=0;
        ZERO: y=0;
        ONE: y=0;
        STORE: begin
            y=1;
            count = count +1;
        end
    endcase
end

endmodule
```

Figure (1): design code

```
module question1_tb();

parameter n0=2'b00;
parameter n1=2'b01;
parameter n2=2'b10;
parameter n3=2'b11;

reg x,clk,rst;
wire y;
wire [9:0] count;
integer i=0;

question1 #(.IDLE(n0),.ZERO(n1),.ONE(n2),.STORE(n3))DUT(.x(x),.clk(clk),.rst(rst),.y(y),.count(count));

initial begin
clk=0;
forever
#25 clk=~clk;
end

initial begin
    rst=1;
    #100;    //delay for two clock periods
    rst=0;
    for(i=0;i<50;i=i+1) begin
        @(negedge clk);
        x=$random;
    end
    $stop;
end

endmodule
```

Figure (2): testbench code

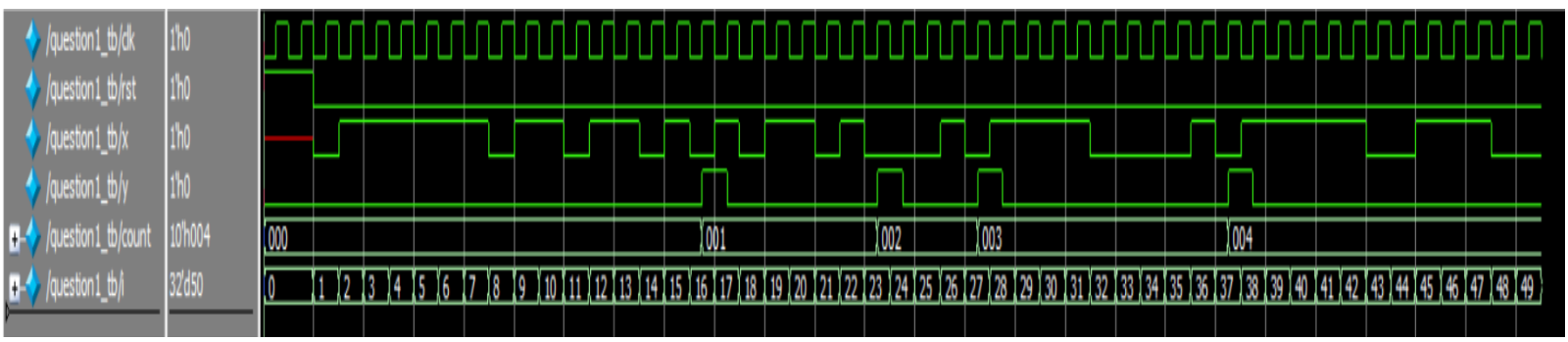

Figure (3): waveform snippet

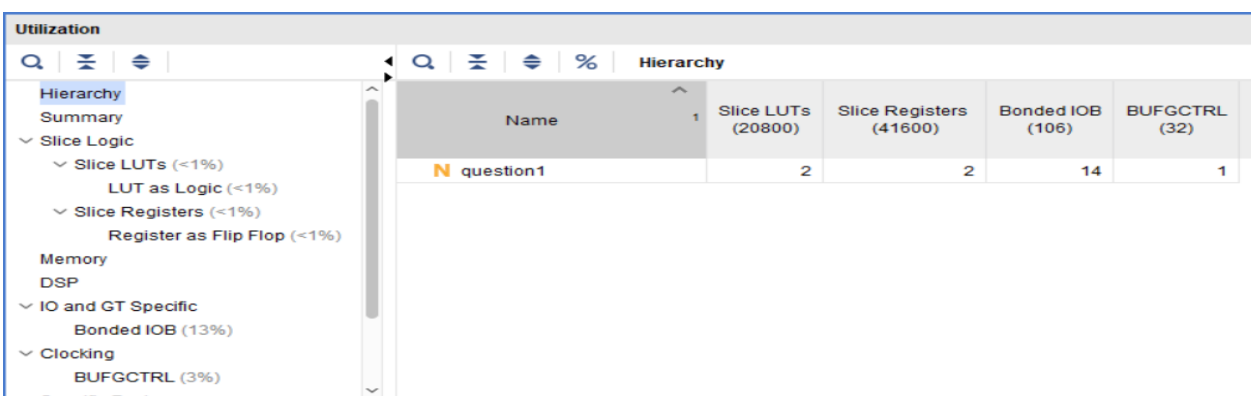

Figure (4): report utilization

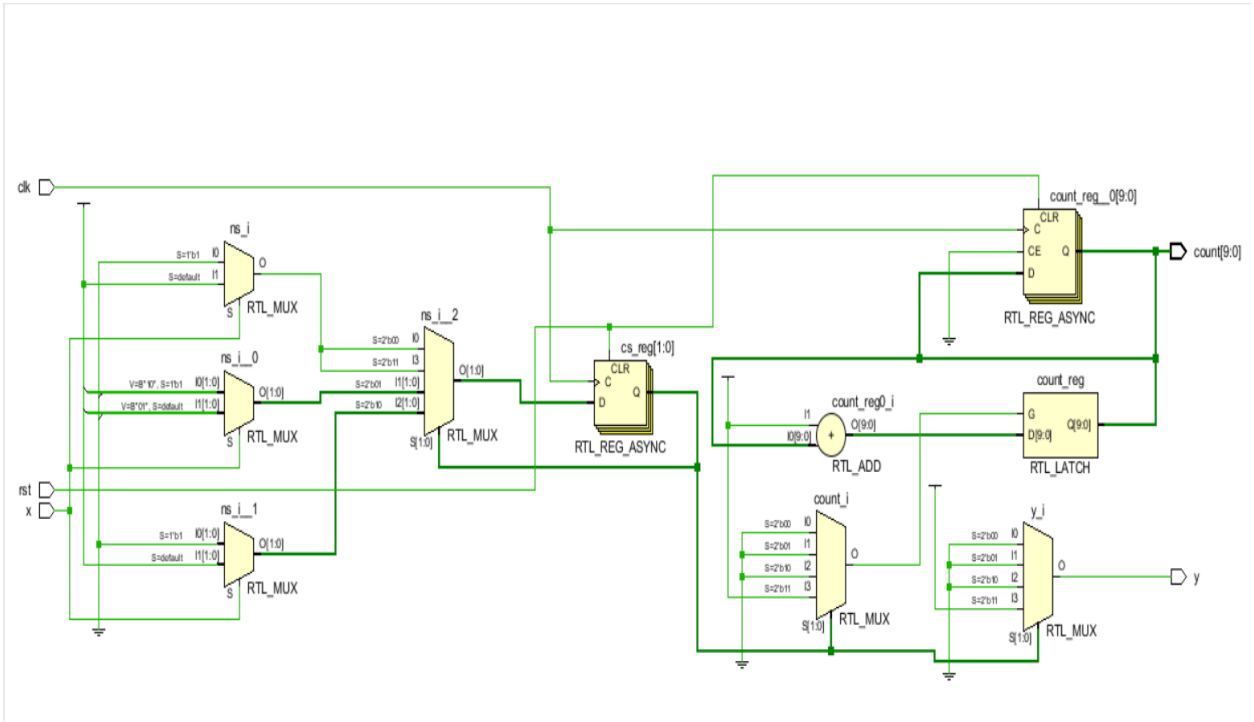

Figure (5): elaborated schematic

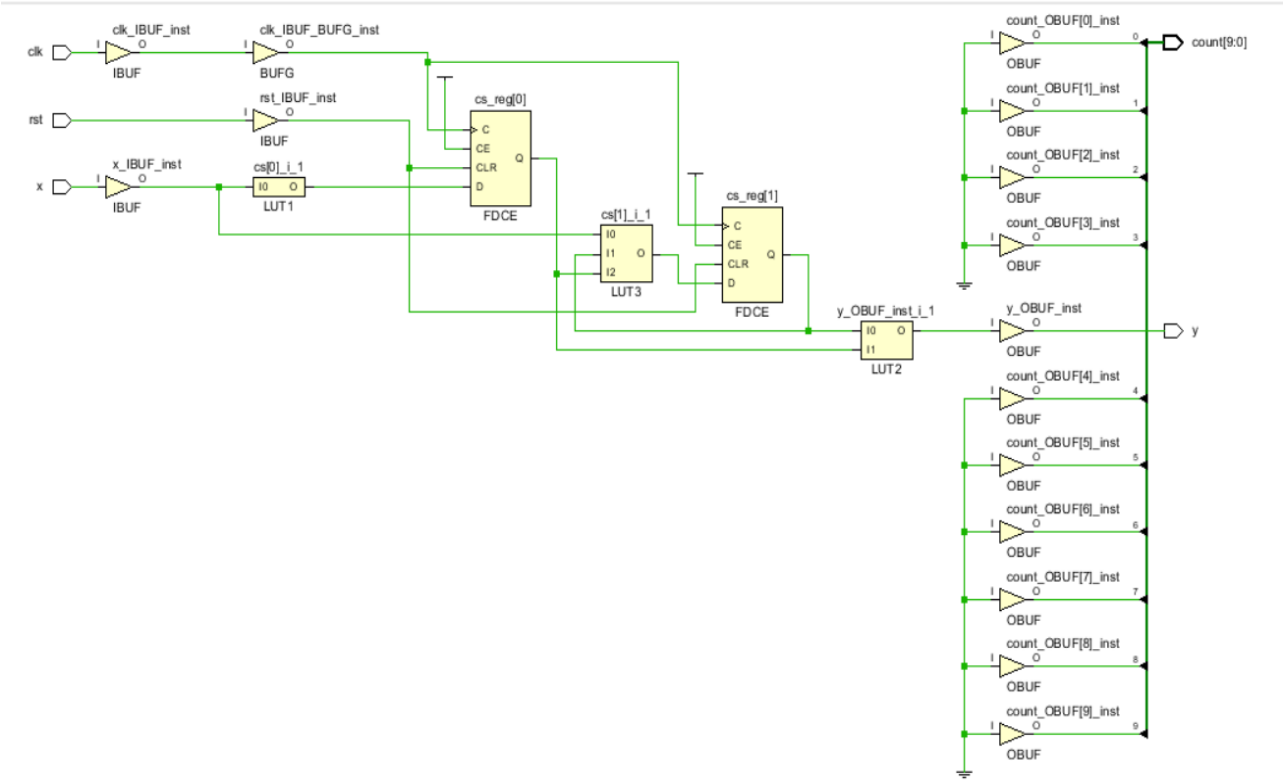

Figure (6): synthesis schematic

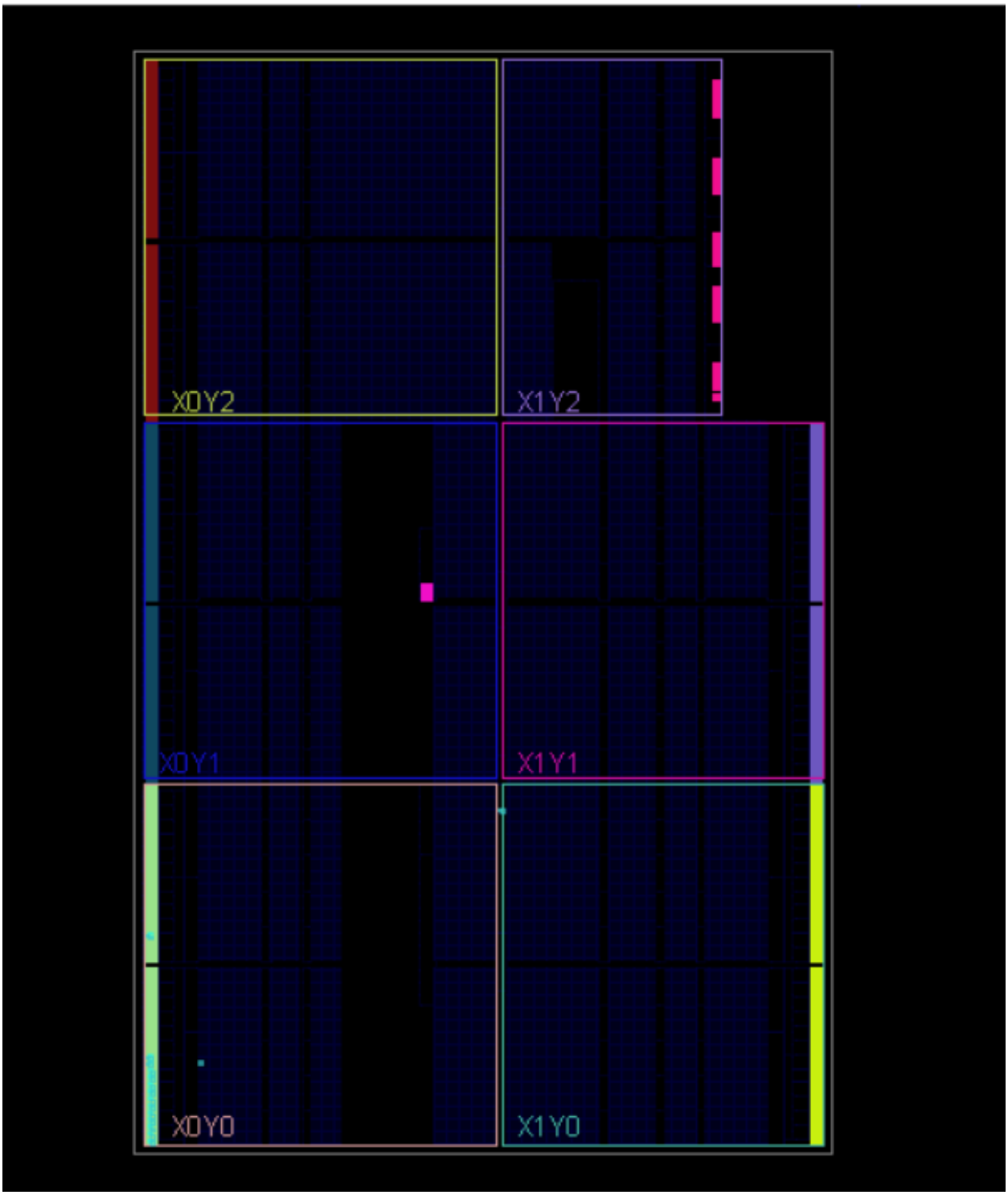

Figure (7): device implementation

Question2]

```
module question2(speed_limit,car_speed,leading_distance,clk,rst,unlock_doors,accelerate_car);

parameter MIN_DISTANCE=7'd40;

parameter STOP=2'b00;
parameter DECELERATE=2'b01;
parameter ACCELERATE=2'b10;

input clk,rst;
input [7:0] speed_limit,car_speed;
input [6:0] leading_distance;
output reg unlock_doors,accelerate_car;

reg [1:0] cs,ns;

// state memory
always @(posedge clk or posedge rst) begin
    if(rst)
        cs<=STOP;
    else
        cs<=ns;
end

// nextstate logic block
always @(*) begin
    case(cs)
    STOP: begin
        if(leading_distance < MIN_DISTANCE)
        ns = STOP;
        else
        ns = ACCELERATE;
    end
    DECELERATE: begin
        if(car_speed==0)
        ns = STOP;
        else if(leading_distance >= MIN_DISTANCE && car_speed < speed_limit)
        ns = ACCELERATE;
        else
        ns = DECELERATE;
    end
    ACCELERATE: begin
        if(leading_distance < MIN_DISTANCE || car_speed > speed_limit)
        ns = DECELERATE;
        else
        ns = ACCELERATE;
        end
    endcase
end

    // output logic block

    always @(*) begin
        case(cs)
            STOP: begin
                unlock_doors = 1;
                accelerate_car = 0;
            end
            ACCELERATE: begin
                unlock_doors = 0;
                accelerate_car = 1;
            end
            DECELERATE: begin
                unlock_doors = 0;
                accelerate_car = 0;
            end
        endcase

    end

    endmodule
```

Figure (8): design code

```
module question2_tb();

parameter N0= 7'd40;    // MIN_DISTANCE parameter
parameter N1=2'b00;    // stop state
parameter N2=2'b01;   // decelerate state
parameter N3=2'b10;  // accelerate state

reg [7:0] speed_limit,car_speed;
reg [6:0] leading_distance;
reg clk,rst;
wire unlock_doors,accelerate_car;
integer i=0;

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)

initial begin
clk=0;
forever
#25 clk=~clk;
end

initial begin
    rst=1;
    #100;
    rst=0;
    speed_limit=8'd100;      // randomization testing if the max speed =100 km/h
    for(i=0;i<10;i=i+1) begin
        @(negedge clk);
        car_speed=$urandom_range(50,150);      // random values for car speed between 50 km/h to 150 km/h
        leading_distance=$urandom_range(0,80);       // random values for leading_distance between 0 meters to 80 meters
    end
    speed_limit=8'd150;      // randomization testing if the max speed =150 km/h
    for(i=10;i<20;i=i+1) begin
        @(negedge clk);
        car_speed=$urandom_range(100,200);     // random values for car speed between 100 km/h to 200 km/h
        leading_distance=$urandom_range(10,60);        // random values for leading_distance between 10 meters to 60 meters
    end
    $stop;
end

endmodule
```

Figure (9): testbench code

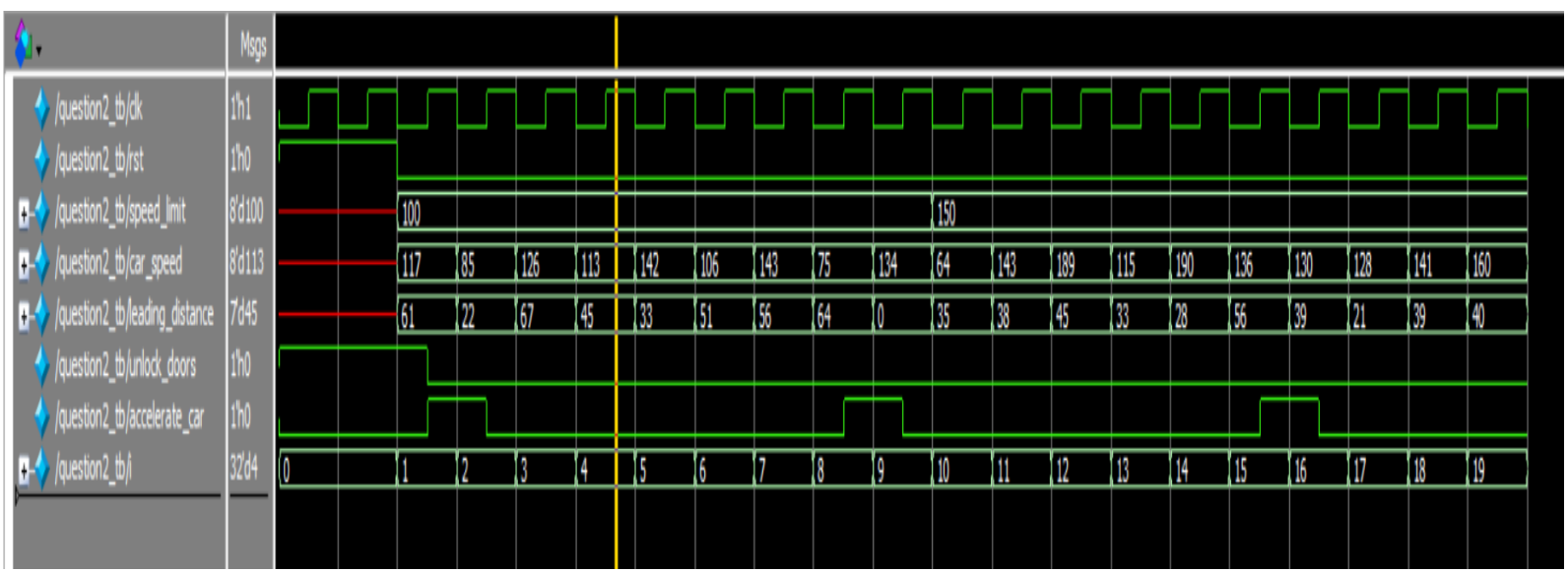

Figure (10): waveform snippet

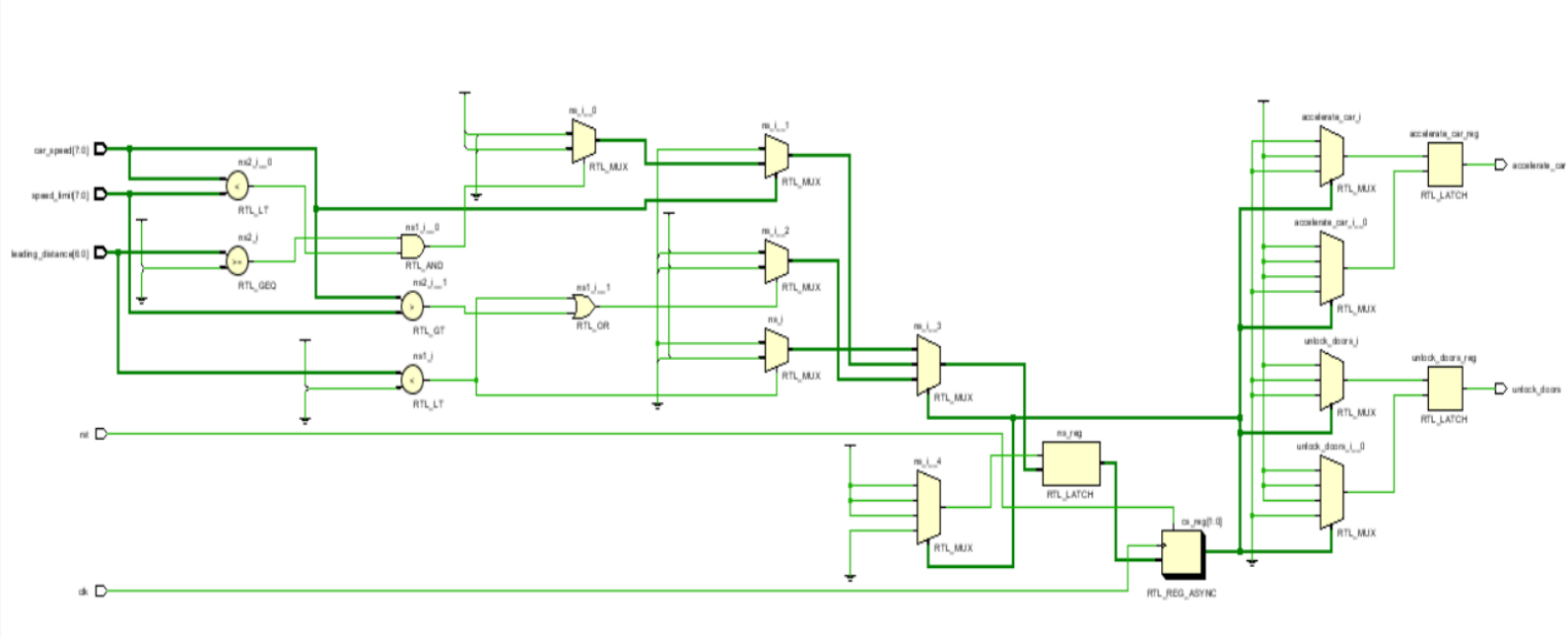

Figure (11): Elaborated schematic

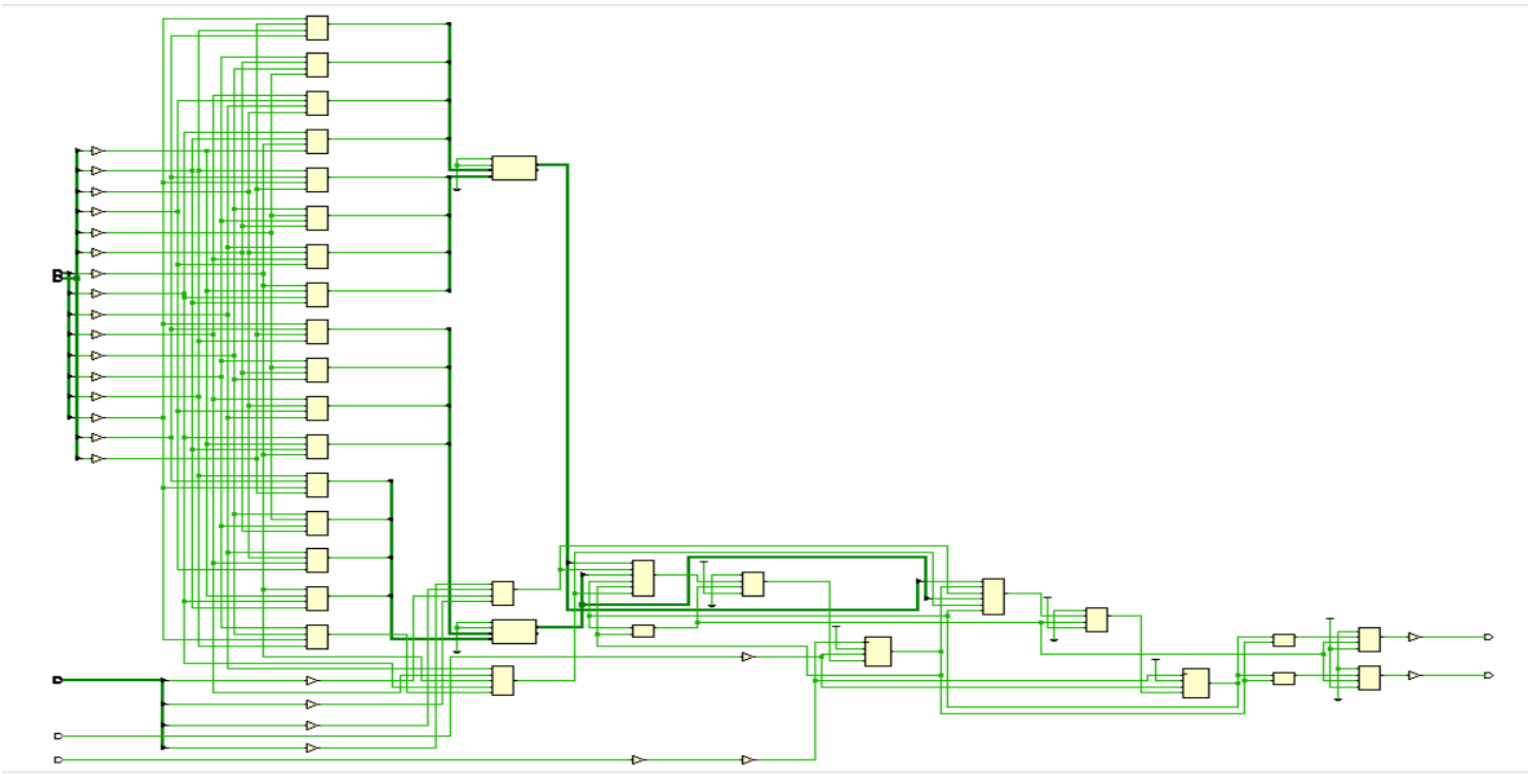

Figure (12): synthesis schematic

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	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
question2	16	6	24	1

Figure (13): utilization report

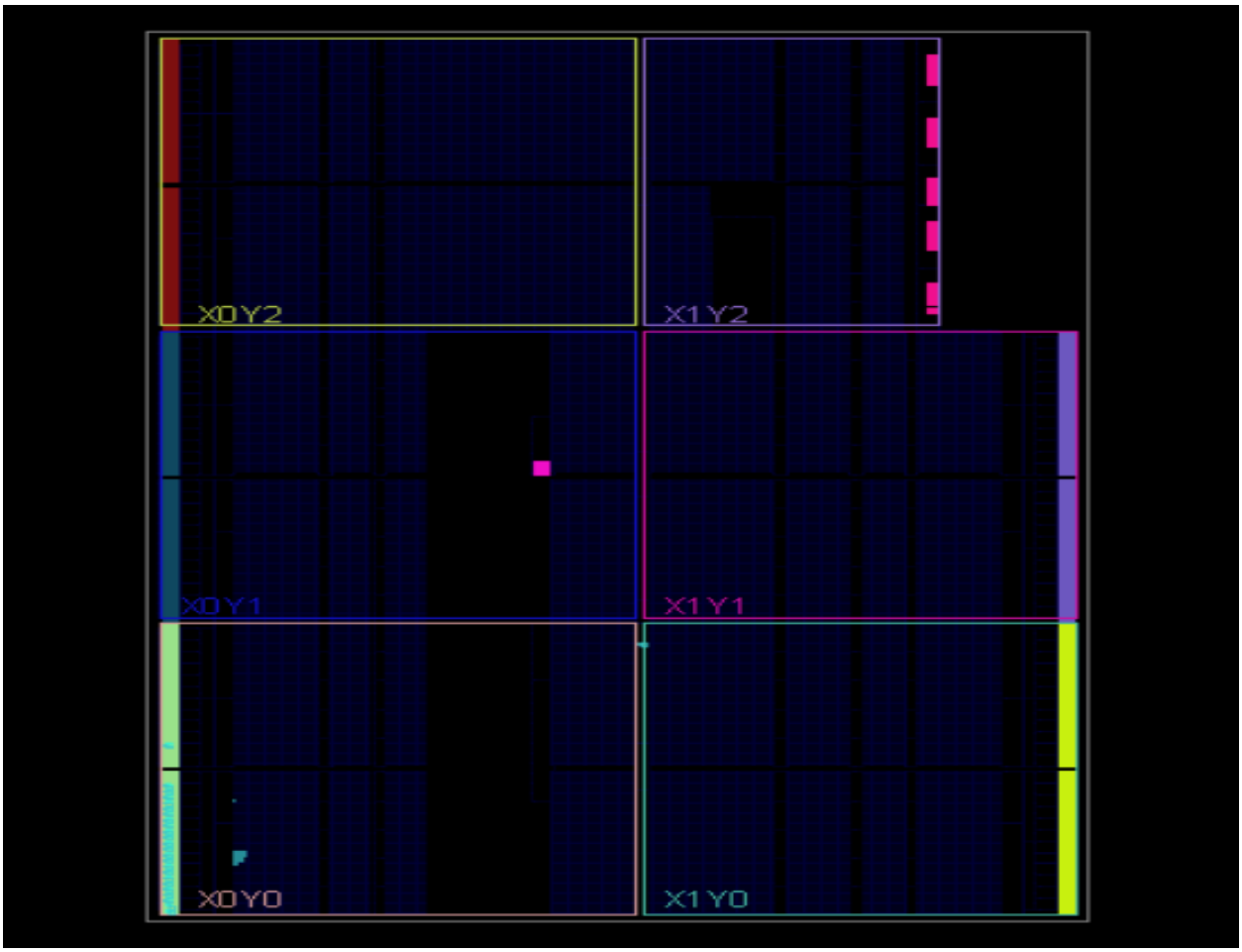

Figure (14): device implementation

Question3]

```
module question3(clk,rst,y);

parameter A=2'b00;
parameter B=2'b01;
parameter C=2'b10;
parameter D=2'b11;

input clk,rst;
output reg [1:0] y;
reg [1:0] cs,ns;

// state memory block
always @(posedge clk or posedge rst) begin
    if(rst)
    cs<=0;
    else
    cs<=ns;
end

// next state logic block
always @(*) begin
    case(cs)
        A: ns = B;
        B: ns = C;
        C: ns = D;
        D: ns = A;
    endcase
end

// output logic block
always @(*) begin
    case(cs)
        A: y = A;
        B: y = B;
        C: y = D;
        D: y = C;
    endcase
end

endmodule
```

Figure (15): main design code

```
module gray_counter(clk,rst,gray_out);

input clk,rst;

output [1:0] gray_out;

reg [1:0] bin_out;

always @(posedge clk or posedge rst) begin

    if(rst)
    bin_out<=0;
    else
    bin_out<=bin_out+1;

end

assign gray_out[0]=^bin_out;

assign gray_out[1]=bin_out[1];

endmodule
```

Figure (16): reference design code

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

Figure (17): Do file

```
module question3_tb();

reg clk,rst;
wire [1:0] y_dut,y_ref;   // y_dut represent the output of main design
integer i=0;             // y_ref represent the output of the reference design

question3 DUT1(.clk(clk),.rst(rst),.y(y_dut));
gray_counter DUT2(.clk(clk),.rst(rst),.gray_out(y_ref));

initial begin
clk=0;
forever
#25 clk=~clk;
end

initial begin
    rst=1;
    #100;      //delay for the first two clock cycles
    rst=0;
    for(i=0;i<10;i=i+1) begin
        @(negedge clk);
        if(y_dut!=y_ref) begin
            $display("the counter doesn't act properly!!");
            $stop;
        end
    end
    $stop;
end

endmodule
```

Figure (18): testbench code

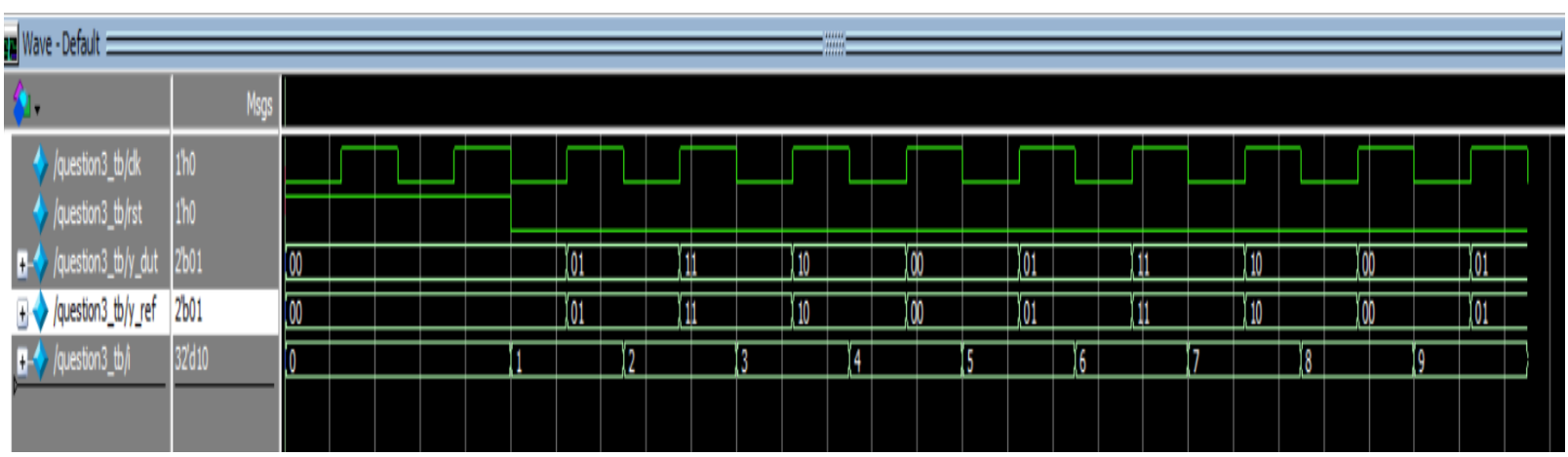

Figure (19): waveform snippet

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]


## Switches
#set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports {sw[0]}]
#set_property -dict { PACKAGE_PIN V16   IOSTANDARD LVCMOS33 } [get_ports {sw[1]}]
#set_property -dict { PACKAGE_PIN W16   IOSTANDARD LVCMOS33 } [get_ports {sw[2]}]
#set_property -dict { PACKAGE_PIN W17   IOSTANDARD LVCMOS33 } [get_ports {sw[3]}]
#set_property -dict { PACKAGE_PIN W15   IOSTANDARD LVCMOS33 } [get_ports {sw[4]}]
#set_property -dict { PACKAGE_PIN V15   IOSTANDARD LVCMOS33 } [get_ports {sw[5]}]
#set_property -dict { PACKAGE_PIN W14   IOSTANDARD LVCMOS33 } [get_ports {sw[6]}]
#set_property -dict { PACKAGE_PIN W13   IOSTANDARD LVCMOS33 } [get_ports {sw[7]}]
#set_property -dict { PACKAGE_PIN V2    IOSTANDARD LVCMOS33 } [get_ports {sw[8]}]
#set_property -dict { PACKAGE_PIN T3    IOSTANDARD LVCMOS33 } [get_ports {sw[9]}]
#set_property -dict { PACKAGE_PIN T2    IOSTANDARD LVCMOS33 } [get_ports {sw[10]}]
#set_property -dict { PACKAGE_PIN R3    IOSTANDARD LVCMOS33 } [get_ports {sw[11]}]
#set_property -dict { PACKAGE_PIN W2    IOSTANDARD LVCMOS33 } [get_ports {sw[12]}]
#set_property -dict { PACKAGE_PIN U1    IOSTANDARD LVCMOS33 } [get_ports {sw[13]}]
#set_property -dict { PACKAGE_PIN T1    IOSTANDARD LVCMOS33 } [get_ports {sw[14]}]
#set_property -dict { PACKAGE_PIN R2    IOSTANDARD LVCMOS33 } [get_ports {sw[15]}]


## LEDs
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports {y[0]}]
set_property -dict { PACKAGE_PIN E19   IOSTANDARD LVCMOS33 } [get_ports {y[1]}]
#set_property -dict { PACKAGE_PIN U19   IOSTANDARD LVCMOS33 } [get_ports {led[2]}]
#set_property -dict { PACKAGE_PIN V19   IOSTANDARD LVCMOS33 } [get_ports {led[3]}]
#set_property -dict { PACKAGE_PIN W18   IOSTANDARD LVCMOS33 } [get_ports {led[4]}]
#set_property -dict { PACKAGE_PIN U15   IOSTANDARD LVCMOS33 } [get_ports {led[5]}]
#set_property -dict { PACKAGE_PIN U14   IOSTANDARD LVCMOS33 } [get_ports {led[6]}]
#set_property -dict { PACKAGE_PIN V14   IOSTANDARD LVCMOS33 } [get_ports {led[7]}]
#set_property -dict { PACKAGE_PIN V13   IOSTANDARD LVCMOS33 } [get_ports {led[8]}]
#set_property -dict { PACKAGE_PIN V3    IOSTANDARD LVCMOS33 } [get_ports {led[9]}]
#set_property -dict { PACKAGE_PIN W3    IOSTANDARD LVCMOS33 } [get_ports {led[10]}]
#set_property -dict { PACKAGE_PIN U3    IOSTANDARD LVCMOS33 } [get_ports {led[11]}]
#set_property -dict { PACKAGE_PIN P3    IOSTANDARD LVCMOS33 } [get_ports {led[12]}]
#set_property -dict { PACKAGE_PIN N3    IOSTANDARD LVCMOS33 } [get_ports {led[13]}]
#set_property -dict { PACKAGE_PIN P1    IOSTANDARD LVCMOS33 } [get_ports {led[14]}]
#set_property -dict { PACKAGE_PIN L1    IOSTANDARD LVCMOS33 } [get_ports {led[15]}]


##7 Segment Display
#set_property -dict { PACKAGE_PIN W7   IOSTANDARD LVCMOS33 } [get_ports {seg[0]}]
#set_property -dict { PACKAGE_PIN W6   IOSTANDARD LVCMOS33 } [get_ports {seg[1]}]
#set_property -dict { PACKAGE_PIN U8   IOSTANDARD LVCMOS33 } [get_ports {seg[2]}]
#set_property -dict { PACKAGE_PIN V8   IOSTANDARD LVCMOS33 } [get_ports {seg[3]}]
#set_property -dict { PACKAGE_PIN U5   IOSTANDARD LVCMOS33 } [get_ports {seg[4]}]
#set_property -dict { PACKAGE_PIN V5   IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]
#set_property -dict { PACKAGE_PIN U7   IOSTANDARD LVCMOS33 } [get_ports {seg[6]}]

#set_property -dict { PACKAGE_PIN V7   IOSTANDARD LVCMOS33 } [get_ports dp]

#set_property -dict { PACKAGE_PIN U2   IOSTANDARD LVCMOS33 } [get_ports {an[0]}]
#set_property -dict { PACKAGE_PIN U4   IOSTANDARD LVCMOS33 } [get_ports {an[1]}]
#set_property -dict { PACKAGE_PIN V4   IOSTANDARD LVCMOS33 } [get_ports {an[2]}]
#set_property -dict { PACKAGE_PIN W4   IOSTANDARD LVCMOS33 } [get_ports {an[3]}]


##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]


##Pmod Header JA
#set_property -dict { PACKAGE_PIN J1   IOSTANDARD LVCMOS33 } [get_ports {JA[0]}];#Sch name = JA1
#set_property -dict { PACKAGE_PIN L2   IOSTANDARD LVCMOS33 } [get_ports {JA[1]}];#Sch name = JA2
#set_property -dict { PACKAGE_PIN J2   IOSTANDARD LVCMOS33 } [get_ports {JA[2]}];#Sch name = JA3
#set_property -dict { PACKAGE_PIN G2   IOSTANDARD LVCMOS33 } [get_ports {JA[3]}];#Sch name = JA4
#set_property -dict { PACKAGE_PIN H1   IOSTANDARD LVCMOS33 } [get_ports {JA[4]}];#Sch name = JA7
#set_property -dict { PACKAGE_PIN K2   IOSTANDARD LVCMOS33 } [get_ports {JA[5]}];#Sch name = JA8
#set_property -dict { PACKAGE_PIN H2   IOSTANDARD LVCMOS33 } [get_ports {JA[6]}];#Sch name = JA9
#set_property -dict { PACKAGE_PIN G3   IOSTANDARD LVCMOS33 } [get_ports {JA[7]}];#Sch name = JA10

##Pmod Header JB
#set_property -dict { PACKAGE_PIN A14   IOSTANDARD LVCMOS33 } [get_ports {JB[0]}];#Sch name = JB1
#set_property -dict { PACKAGE_PIN A16   IOSTANDARD LVCMOS33 } [get_ports {JB[1]}];#Sch name = JB2
#set_property -dict { PACKAGE_PIN B15   IOSTANDARD LVCMOS33 } [get_ports {JB[2]}];#Sch name = JB3
#set_property -dict { PACKAGE_PIN B16   IOSTANDARD LVCMOS33 } [get_ports {JB[3]}];#Sch name = JB4
#set_property -dict { PACKAGE_PIN A15   IOSTANDARD LVCMOS33 } [get_ports {JB[4]}];#Sch name = JB7
#set_property -dict { PACKAGE_PIN A17   IOSTANDARD LVCMOS33 } [get_ports {JB[5]}];#Sch name = JB8
#set_property -dict { PACKAGE_PIN C15   IOSTANDARD LVCMOS33 } [get_ports {JB[6]}];#Sch name = JB9
#set_property -dict { PACKAGE_PIN C16   IOSTANDARD LVCMOS33 } [get_ports {JB[7]}];#Sch name = JB10

##Pmod Header JC
#set_property -dict { PACKAGE_PIN K17   IOSTANDARD LVCMOS33 } [get_ports {JC[0]}];#Sch name = JC1
#set_property -dict { PACKAGE_PIN M18   IOSTANDARD LVCMOS33 } [get_ports {JC[1]}];#Sch name = JC2
#set_property -dict { PACKAGE_PIN N17   IOSTANDARD LVCMOS33 } [get_ports {JC[2]}];#Sch name = JC3
#set_property -dict { PACKAGE_PIN P18   IOSTANDARD LVCMOS33 } [get_ports {JC[3]}];#Sch name = JC4
#set_property -dict { PACKAGE_PIN L17   IOSTANDARD LVCMOS33 } [get_ports {JC[4]}];#Sch name = JC7
#set_property -dict { PACKAGE_PIN M19   IOSTANDARD LVCMOS33 } [get_ports {JC[5]}];#Sch name = JC8
#set_property -dict { PACKAGE_PIN P17   IOSTANDARD LVCMOS33 } [get_ports {JC[6]}];#Sch name = JC9
#set_property -dict { PACKAGE_PIN R18   IOSTANDARD LVCMOS33 } [get_ports {JC[7]}];#Sch name = JC10

##Pmod Header JXADC
#set_property -dict { PACKAGE_PIN J3   IOSTANDARD LVCMOS33 } [get_ports {JXADC[0]}];#Sch name = XA1_P
#set_property -dict { PACKAGE_PIN L3   IOSTANDARD LVCMOS33 } [get_ports {JXADC[1]}];#Sch name = XA2_P
#set_property -dict { PACKAGE_PIN M2   IOSTANDARD LVCMOS33 } [get_ports {JXADC[2]}];#Sch name = XA3_P
#set_property -dict { PACKAGE_PIN N2   IOSTANDARD LVCMOS33 } [get_ports {JXADC[3]}];#Sch name = XA4_P
#set_property -dict { PACKAGE_PIN K3   IOSTANDARD LVCMOS33 } [get_ports {JXADC[4]}];#Sch name = XA1_N
```

Figure (20): constraint file after edit some ports

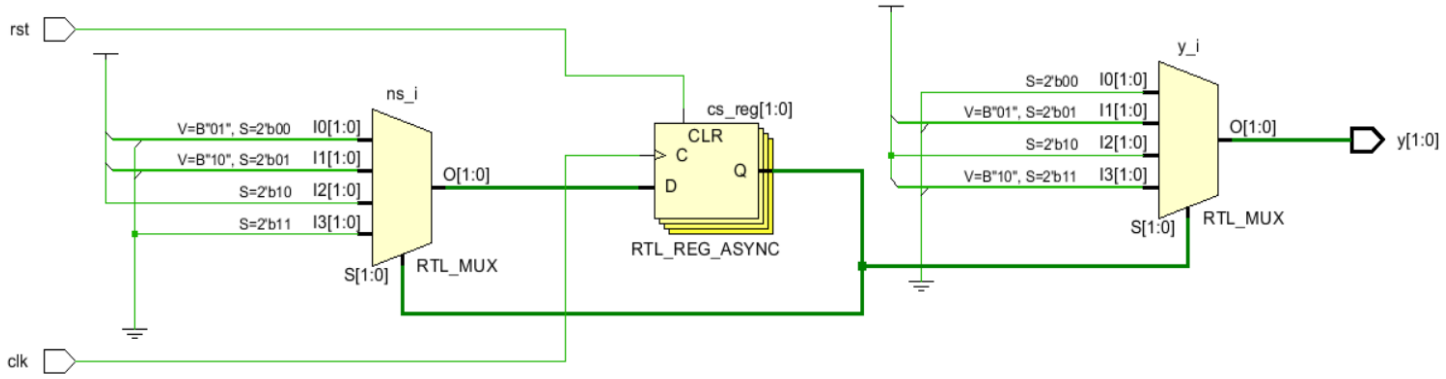

Figure (21): elaborated schematic

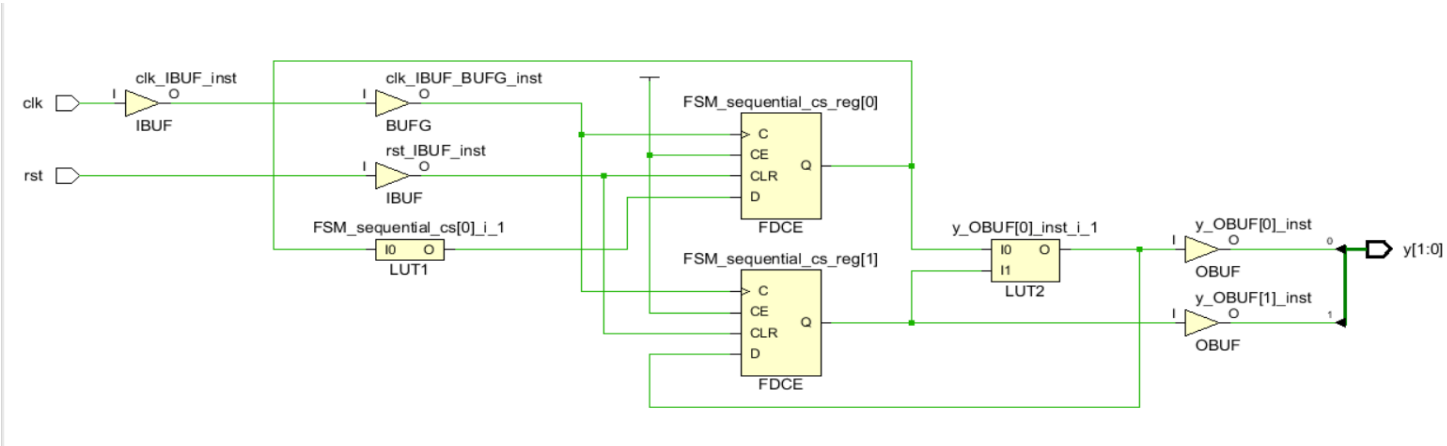

Figure (22): synthesis schematic

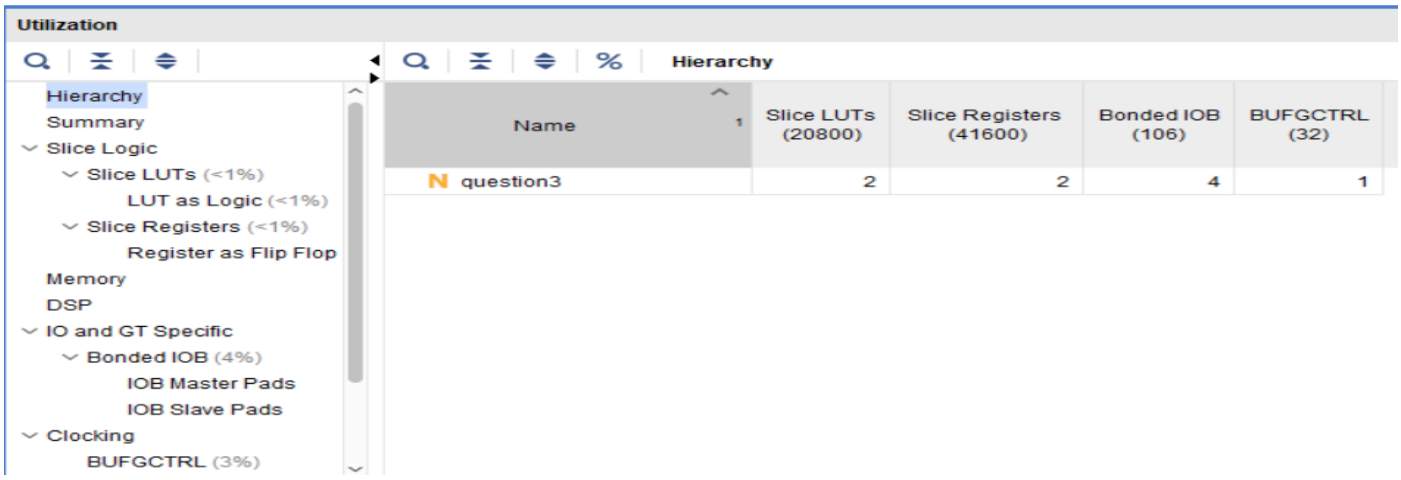

Figure (23): utilization report

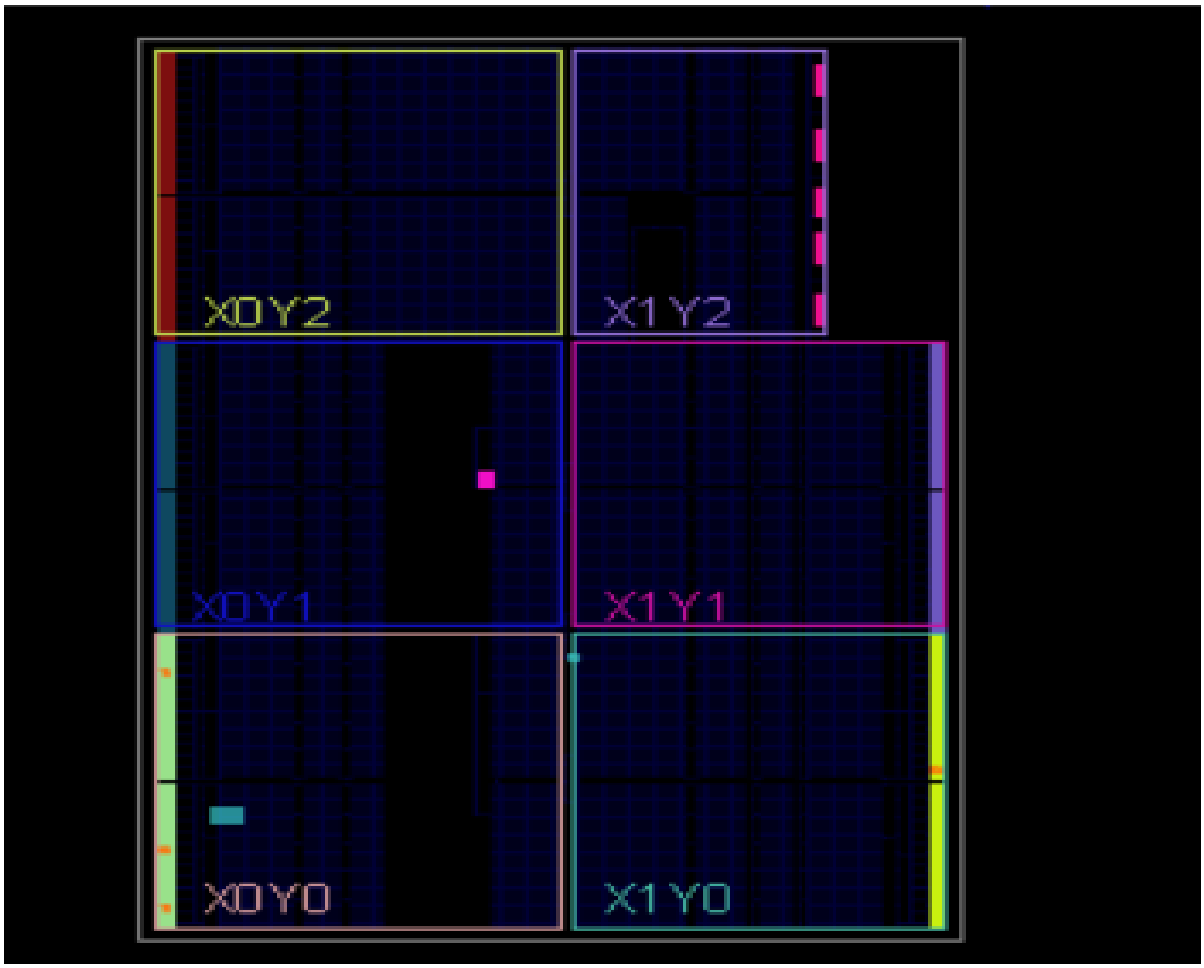

Figure (24): device implementation

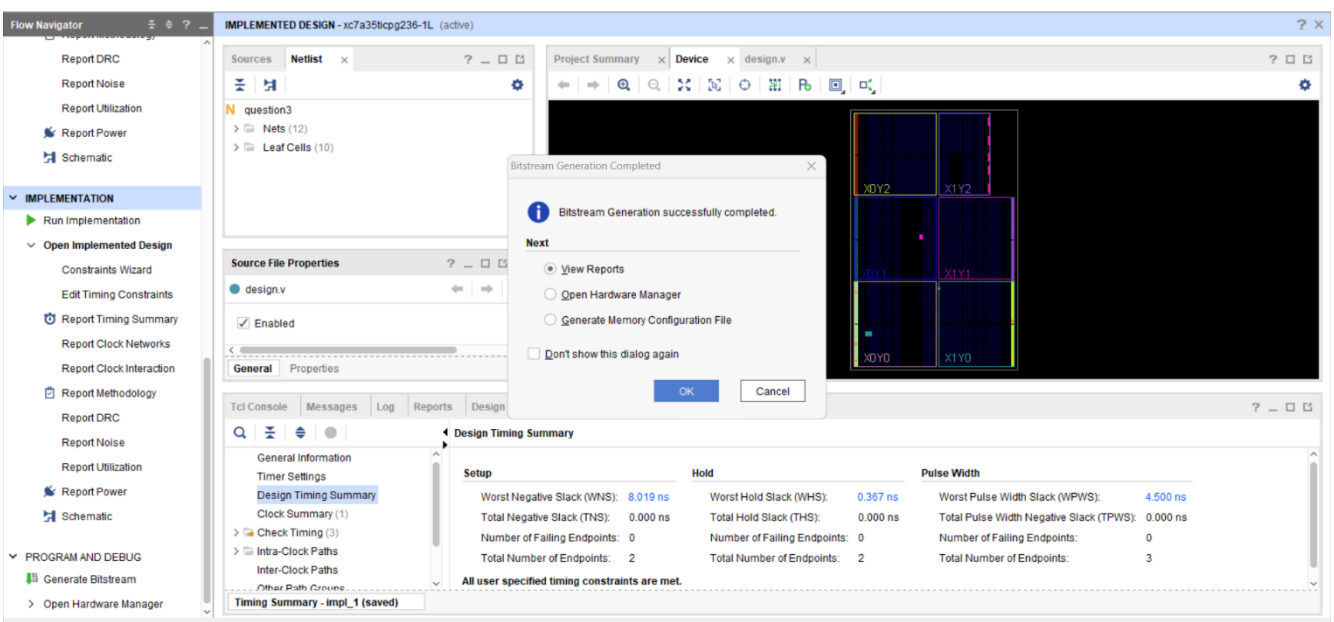

Figure (25): successful bitstream generation

Question4]

```
module question4(clk,rst,in,y);

parameter S0=2'b00;
parameter S1=2'b01;
parameter S2=2'b10;
parameter S3=2'b11;
input clk,rst,in;
output reg  y;
reg [1:0] cs,ns;
// state memory block
always @(posedge clk or posedge rst) begin
    if(rst)
    cs<=S0;
    else
    cs<=ns;
end
// next state logic block
always @(*) begin
    case(cs)
        S0: begin
            if(in)
            ns = S1;
            else
            ns = S0;
        end
        S1: begin
            if(in)
            ns = S2;
            else
            ns = S1;
        end
        S2: begin
            if(in)
            ns = S3;
            else
            ns = S2;
        end
        S3: begin
            if(in)
            ns = S1;
            else
            ns = S0;
        end

    endcase
end
// output logic block
always @(*) begin
    if(cs==S2 && in==1)
    y=1;
    else
    y=0;
end

endmodule
```

Figure (26): design code

```
module question4_tb();

reg clk,rst,in;
wire out;
integer i=0;

question4 DUT1(.clk(clk),.rst(rst),.in(in),.y(out));

initial begin
clk=0;
forever
#25 clk=~clk;
end

initial begin
    rst=1;
    #100;      //delay for the first two clock cycles
    rst=0;
    for(i=0;i<50;i=i+1) begin
        @(negedge clk);
        in=$random;
    end
    $stop;
end

endmodule
```

Figure (27): testbench code

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]


## Switches
set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports {in}]
#set_property -dict { PACKAGE_PIN V16   IOSTANDARD LVCMOS33 } [get_ports {sw[1]}]
#set_property -dict { PACKAGE_PIN W16   IOSTANDARD LVCMOS33 } [get_ports {sw[2]}]
#set_property -dict { PACKAGE_PIN W17   IOSTANDARD LVCMOS33 } [get_ports {sw[3]}]
#set_property -dict { PACKAGE_PIN W15   IOSTANDARD LVCMOS33 } [get_ports {sw[4]}]
#set_property -dict { PACKAGE_PIN V15   IOSTANDARD LVCMOS33 } [get_ports {sw[5]}]
#set_property -dict { PACKAGE_PIN W14   IOSTANDARD LVCMOS33 } [get_ports {sw[6]}]
#set_property -dict { PACKAGE_PIN W13   IOSTANDARD LVCMOS33 } [get_ports {sw[7]}]
#set_property -dict { PACKAGE_PIN V2    IOSTANDARD LVCMOS33 } [get_ports {sw[8]}]
#set_property -dict { PACKAGE_PIN T3    IOSTANDARD LVCMOS33 } [get_ports {sw[9]}]
#set_property -dict { PACKAGE_PIN T2    IOSTANDARD LVCMOS33 } [get_ports {sw[10]}]
#set_property -dict { PACKAGE_PIN R3    IOSTANDARD LVCMOS33 } [get_ports {sw[11]}]
#set_property -dict { PACKAGE_PIN W2    IOSTANDARD LVCMOS33 } [get_ports {sw[12]}]
#set_property -dict { PACKAGE_PIN U1    IOSTANDARD LVCMOS33 } [get_ports {sw[13]}]
#set_property -dict { PACKAGE_PIN T1    IOSTANDARD LVCMOS33 } [get_ports {sw[14]}]
#set_property -dict { PACKAGE_PIN R2    IOSTANDARD LVCMOS33 } [get_ports {sw[15]}]


## LEDs
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports {y}]
#set_property -dict { PACKAGE_PIN E19   IOSTANDARD LVCMOS33 } [get_ports {led[1]}]
#set_property -dict { PACKAGE_PIN U19   IOSTANDARD LVCMOS33 } [get_ports {led[2]}]
#set_property -dict { PACKAGE_PIN V19   IOSTANDARD LVCMOS33 } [get_ports {led[3]}]
#set_property -dict { PACKAGE_PIN W18   IOSTANDARD LVCMOS33 } [get_ports {led[4]}]
#set_property -dict { PACKAGE_PIN U15   IOSTANDARD LVCMOS33 } [get_ports {led[5]}]
#set_property -dict { PACKAGE_PIN U14   IOSTANDARD LVCMOS33 } [get_ports {led[6]}]
#set_property -dict { PACKAGE_PIN V14   IOSTANDARD LVCMOS33 } [get_ports {led[7]}]
#set_property -dict { PACKAGE_PIN V13   IOSTANDARD LVCMOS33 } [get_ports {led[8]}]
#set_property -dict { PACKAGE_PIN V3    IOSTANDARD LVCMOS33 } [get_ports {led[9]}]
#set_property -dict { PACKAGE_PIN W3    IOSTANDARD LVCMOS33 } [get_ports {led[10]}]
#set_property -dict { PACKAGE_PIN U3    IOSTANDARD LVCMOS33 } [get_ports {led[11]}]
#set_property -dict { PACKAGE_PIN P3    IOSTANDARD LVCMOS33 } [get_ports {led[12]}]
#set_property -dict { PACKAGE_PIN N3    IOSTANDARD LVCMOS33 } [get_ports {led[13]}]
#set_property -dict { PACKAGE_PIN P1    IOSTANDARD LVCMOS33 } [get_ports {led[14]}]
#set_property -dict { PACKAGE_PIN L1    IOSTANDARD LVCMOS33 } [get_ports {led[15]}]


##7 Segment Display
#set_property -dict { PACKAGE_PIN W7   IOSTANDARD LVCMOS33 } [get_ports {seg[0]}]
#set_property -dict { PACKAGE_PIN W6   IOSTANDARD LVCMOS33 } [get_ports {seg[1]}]
#set_property -dict { PACKAGE_PIN U8   IOSTANDARD LVCMOS33 } [get_ports {seg[2]}]
#set_property -dict { PACKAGE_PIN V8   IOSTANDARD LVCMOS33 } [get_ports {seg[3]}]
#set_property -dict { PACKAGE_PIN U5   IOSTANDARD LVCMOS33 } [get_ports {seg[4]}]
#set_property -dict { PACKAGE_PIN V5   IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]
#set_property -dict { PACKAGE_PIN V5   IOSTANDARD LVCMOS33 } [get_ports {seg[5]}]
#set_property -dict { PACKAGE_PIN U7   IOSTANDARD LVCMOS33 } [get_ports {seg[6]}]

#set_property -dict { PACKAGE_PIN V7   IOSTANDARD LVCMOS33 } [get_ports dp]

#set_property -dict { PACKAGE_PIN U2   IOSTANDARD LVCMOS33 } [get_ports {an[0]}]
#set_property -dict { PACKAGE_PIN U4   IOSTANDARD LVCMOS33 } [get_ports {an[1]}]
#set_property -dict { PACKAGE_PIN V4   IOSTANDARD LVCMOS33 } [get_ports {an[2]}]
#set_property -dict { PACKAGE_PIN W4   IOSTANDARD LVCMOS33 } [get_ports {an[3]}]


##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]


##Pmod Header JA
#set_property -dict { PACKAGE_PIN J1   IOSTANDARD LVCMOS33 } [get_ports {JA[0]}];#Sch name = JA1
#set_property -dict { PACKAGE_PIN L2   IOSTANDARD LVCMOS33 } [get_ports {JA[1]}];#Sch name = JA2
#set_property -dict { PACKAGE_PIN J2   IOSTANDARD LVCMOS33 } [get_ports {JA[2]}];#Sch name = JA3
#set_property -dict { PACKAGE_PIN G2   IOSTANDARD LVCMOS33 } [get_ports {JA[3]}];#Sch name = JA4
#set_property -dict { PACKAGE_PIN H1   IOSTANDARD LVCMOS33 } [get_ports {JA[4]}];#Sch name = JA7
#set_property -dict { PACKAGE_PIN K2   IOSTANDARD LVCMOS33 } [get_ports {JA[5]}];#Sch name = JA8
#set_property -dict { PACKAGE_PIN H2   IOSTANDARD LVCMOS33 } [get_ports {JA[6]}];#Sch name = JA9
#set_property -dict { PACKAGE_PIN G3   IOSTANDARD LVCMOS33 } [get_ports {JA[7]}];#Sch name = JA10

##Pmod Header JB
#set_property -dict { PACKAGE_PIN A14   IOSTANDARD LVCMOS33 } [get_ports {JB[0]}];#Sch name = JB1
#set_property -dict { PACKAGE_PIN A16   IOSTANDARD LVCMOS33 } [get_ports {JB[1]}];#Sch name = JB2
#set_property -dict { PACKAGE_PIN B15   IOSTANDARD LVCMOS33 } [get_ports {JB[2]}];#Sch name = JB3
#set_property -dict { PACKAGE_PIN B16   IOSTANDARD LVCMOS33 } [get_ports {JB[3]}];#Sch name = JB4
#set_property -dict { PACKAGE_PIN A15   IOSTANDARD LVCMOS33 } [get_ports {JB[4]}];#Sch name = JB7
#set_property -dict { PACKAGE_PIN A17   IOSTANDARD LVCMOS33 } [get_ports {JB[5]}];#Sch name = JB8
#set_property -dict { PACKAGE_PIN C15   IOSTANDARD LVCMOS33 } [get_ports {JB[6]}];#Sch name = JB9
#set_property -dict { PACKAGE_PIN C16   IOSTANDARD LVCMOS33 } [get_ports {JB[7]}];#Sch name = JB10

##Pmod Header JC
#set_property -dict { PACKAGE_PIN K17   IOSTANDARD LVCMOS33 } [get_ports {JC[0]}];#Sch name = JC1
#set_property -dict { PACKAGE_PIN M18   IOSTANDARD LVCMOS33 } [get_ports {JC[1]}];#Sch name = JC2
#set_property -dict { PACKAGE_PIN N17   IOSTANDARD LVCMOS33 } [get_ports {JC[2]}];#Sch name = JC3
#set_property -dict { PACKAGE_PIN P18   IOSTANDARD LVCMOS33 } [get_ports {JC[3]}];#Sch name = JC4
#set_property -dict { PACKAGE_PIN L17   IOSTANDARD LVCMOS33 } [get_ports {JC[4]}];#Sch name = JC7
#set_property -dict { PACKAGE_PIN M19   IOSTANDARD LVCMOS33 } [get_ports {JC[5]}];#Sch name = JC8
#set_property -dict { PACKAGE_PIN P17   IOSTANDARD LVCMOS33 } [get_ports {JC[6]}];#Sch name = JC9
#set_property -dict { PACKAGE_PIN R18   IOSTANDARD LVCMOS33 } [get_ports {JC[7]}];#Sch name = JC10

##Pmod Header JXADC
#set_property -dict { PACKAGE_PIN J3   IOSTANDARD LVCMOS33 } [get_ports {JXADC[0]}];#Sch name = XA1_P
#set_property -dict { PACKAGE_PIN L3   IOSTANDARD LVCMOS33 } [get_ports {JXADC[1]}];#Sch name = XA2_P
#set_property -dict { PACKAGE_PIN M2   IOSTANDARD LVCMOS33 } [get_ports {JXADC[2]}];#Sch name = XA3_P
#set_property -dict { PACKAGE_PIN N2   IOSTANDARD LVCMOS33 } [get_ports {JXADC[3]}];#Sch name = XA4_P
#set_property -dict { PACKAGE_PIN K3   IOSTANDARD LVCMOS33 } [get_ports {JXADC[4]}];#Sch name = XA1_N
```

Figure (28): constraint file

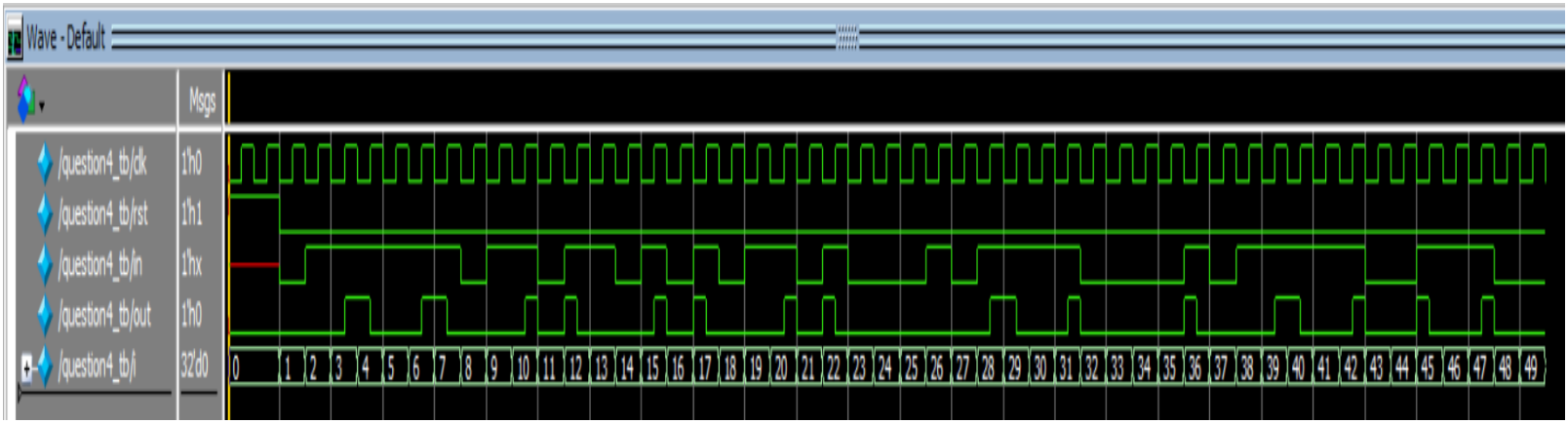

Figure (29): waveform snippet

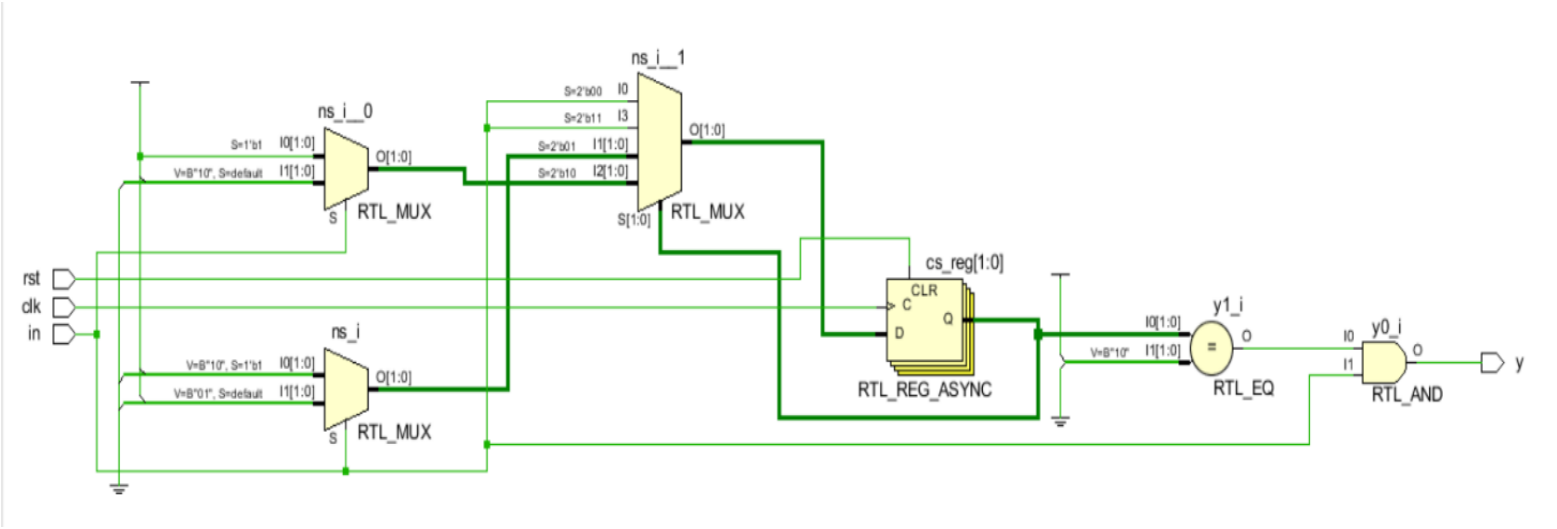

Figure (30): elaborated schematic

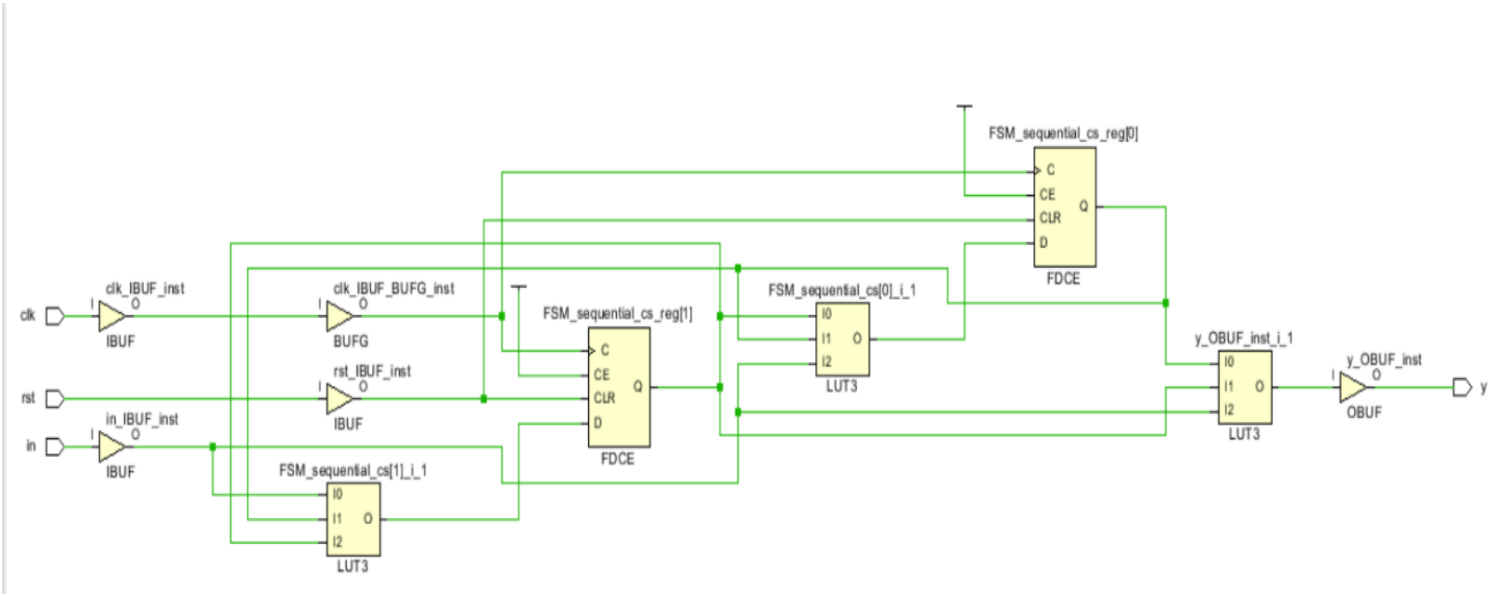

Figure (30): synthesis schematic

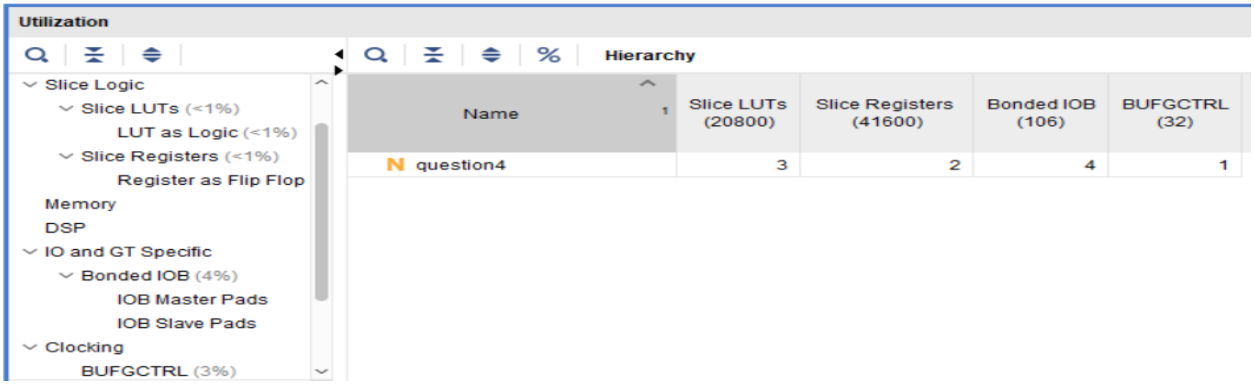

Figure (31): report utilization

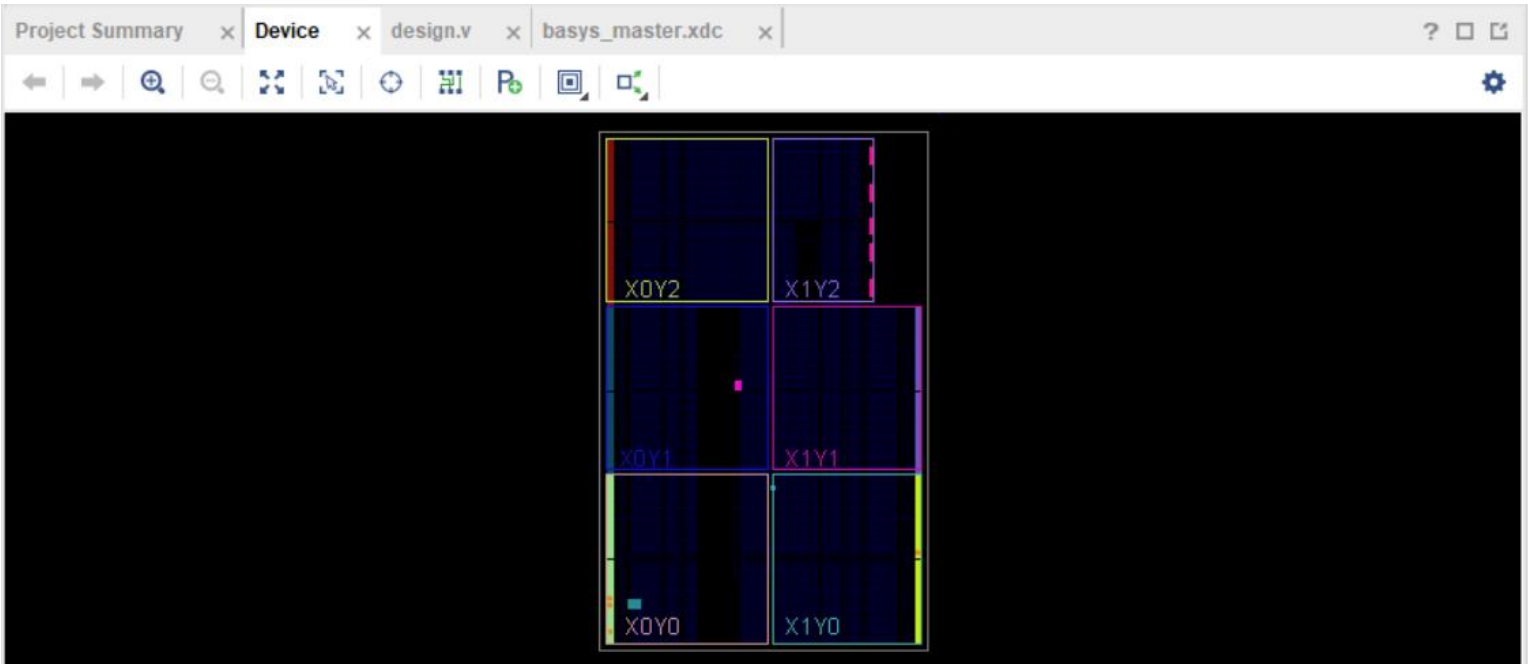

Figure (32): device implementation

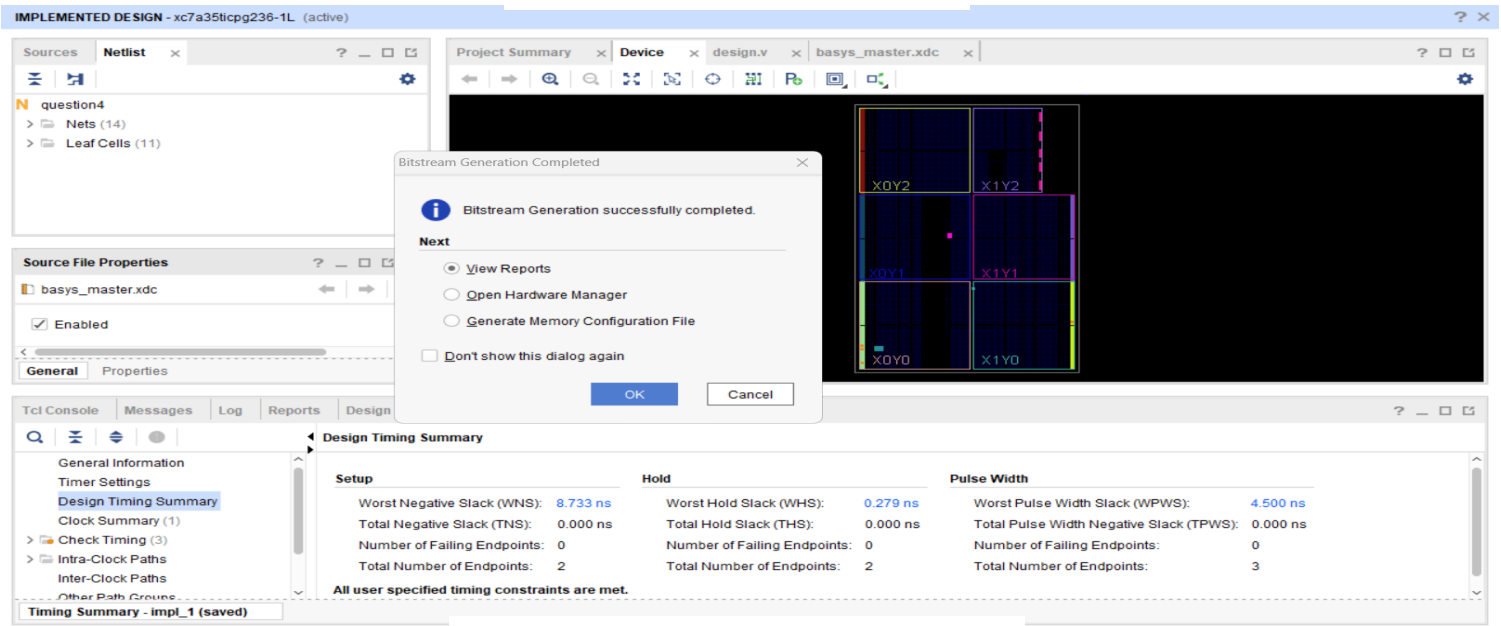

Figure (33): successful bitstream generation

Question5]

```
module ALSU(A,B,Cin,opcode,serial_in,direction,red_op_A,red_op_B,bypass_A,bypass_B,clk,rst,leds);

parameter INPUT_PRIORITY ="A";
parameter FULL_ADDER="ON";

input [2:0] A,B,opcode;
input Cin,serial_in,direction,red_op_A,red_op_B,bypass_A,bypass_B,clk,rst;

reg [2:0] A_ff,B_ff,opcode_ff;
reg Cin_ff,serial_in_ff,direction_ff,red_op_A_ff,red_op_B_ff,bypass_A_ff,bypass_B_ff;

 reg [5:0] out;
output reg [15:0] leds;

wire invalid_opcode,invalid_red_op,invalid_overall;
wire [5:0] mult_out;
wire [3:0] sum_h;
wire [3:0] sum_f;

assign invalid_opcode = opcode_ff[2] & opcode_ff[1];
assign invalid_red_op =(red_op_A_ff | red_op_B_ff) & (opcode_ff[2] | opcode_ff[1]);
assign invalid_overall = invalid_opcode | invalid_red_op;

mult_gen_0 Multiplier (
  .A(A_ff),  // input wire [2 : 0] A
  .B(B_ff),  // input wire [2 : 0] B
  .P(mult_out)  // output wire [5 : 0] P
);

c_addsub_0 half_adder (
  .A(A_ff),  // input wire [2 : 0] A
  .B(B_ff),  // input wire [2 : 0] B
  .S(sum_h)  // output wire [3 : 0] S
);

c_addsub_1 full_adder (
  .A(A_ff),        // input wire [2 : 0] A
  .B(B_ff),        // input wire [2 : 0] B
  .C_IN(Cin_ff),  // input wire C_IN
  .S(sum_f)        // output wire [3 : 0] S
);


// always block specilaized for inputs flip flops

always @(posedge clk or posedge rst) begin
        if(rst) begin
            A_ff<=0;
            B_ff<=0;
            opcode_ff<=0;
            Cin_ff<=0;
            serial_in_ff<=0;
            direction_ff<=0;
            red_op_A_ff<=0;
            red_op_B_ff<=0;
```

Figure (34): design code

```
                bypass_A_ff<=0;
                bypass_B_ff<=0;
                end
            else begin
                A_ff<=A;
                B_ff<=B;
                opcode_ff<=opcode;
                Cin_ff<=Cin;
                serial_in_ff<=serial_in;
                direction_ff<=direction;
                red_op_A_ff<=red_op_A;
                red_op_B_ff<=red_op_B;
                bypass_A_ff<=bypass_A;
                bypass_B_ff<=bypass_B;
            end
    end

    // always block specialized for leds

    always @(posedge clk or posedge rst) begin

        if(rst)
            leds<=0;
        else if(invalid_overall==1)
            leds<=~leds;
            else
            leds<=0;
    end

    // always block for ALSU output

    always @(posedge clk or posedge rst) begin
        if(rst)
            out<=0;
        else if(invalid_overall)
            out<=0;
        else if(bypass_A_ff && bypass_B_ff)
            out<= (INPUT_PRIORITY =="A")?A_ff:B_ff;
            else if(bypass_A_ff)
            out<=A_ff;
            else if(bypass_B_ff)
            out<=B_ff;
            else begin
                case(opcode_ff)

                    3'b000:begin
                    case({red_op_A_ff,red_op_B_ff})
                    2'b00:out <= A_ff & B_ff;
                    2'b01:out <= & B_ff;
                    2'b10:out <= & A_ff;
                    2'b11:out <= (INPUT_PRIORITY=="A")?(& A_ff):(& B_ff);
                    endcase
                    end

                    3'b001:begin
                    case({red_op_A_ff,red_op_B_ff})
                    2'b00:out <= A_ff ^ B_ff;
                    2'b01:out <= ^ B_ff;
                    2'b10:out <= ^ A_ff;
                    2'b11:out <= (INPUT_PRIORITY=="A")?(^ A_ff):(^ B_ff);
                    endcase
                    end

                    3'b010:begin
                        case(FULL_ADDER)
                            "OFF" : out<= {2'b00 , sum_h};
                            "ON" : out<= {2'b00 , sum_f};
                        endcase
                    end

                    3'b011: out<= mult_out;

                    3'b100:begin
                        if(direction_ff)
                            out <= {out[4:0] , serial_in_ff};
                        else
                            out <= {serial_in_ff , out[5:1]};
                    end

                    3'b101:begin
                        if(direction_ff)
                            out <= {out[4:0] , out[5]};
                        else
                            out <= {out[0] , out[5:1]};
                    end

                    default:out<=0;

                endcase
            end
        end

endmodule
```

Figure (35): continue_design_code.

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]


## Switches
set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports {opcode[0]}]
set_property -dict { PACKAGE_PIN V16   IOSTANDARD LVCMOS33 } [get_ports {opcode[1]}]
set_property -dict { PACKAGE_PIN W16   IOSTANDARD LVCMOS33 } [get_ports {opcode[2]}]
set_property -dict { PACKAGE_PIN W17   IOSTANDARD LVCMOS33 } [get_ports {A[0]}]
set_property -dict { PACKAGE_PIN W15   IOSTANDARD LVCMOS33 } [get_ports {A[1]}]
set_property -dict { PACKAGE_PIN V15   IOSTANDARD LVCMOS33 } [get_ports {A[2]}]
set_property -dict { PACKAGE_PIN W14   IOSTANDARD LVCMOS33 } [get_ports {B[0]}]
set_property -dict { PACKAGE_PIN W13   IOSTANDARD LVCMOS33 } [get_ports {B[1]}]
set_property -dict { PACKAGE_PIN V2    IOSTANDARD LVCMOS33 } [get_ports {B[2]}]
set_property -dict { PACKAGE_PIN T3    IOSTANDARD LVCMOS33 } [get_ports {Cin}]
set_property -dict { PACKAGE_PIN T2    IOSTANDARD LVCMOS33 } [get_ports {red_op_A}]
set_property -dict { PACKAGE_PIN R3    IOSTANDARD LVCMOS33 } [get_ports {red_op_B}]
set_property -dict { PACKAGE_PIN W2    IOSTANDARD LVCMOS33 } [get_ports {bypass_A}]
set_property -dict { PACKAGE_PIN U1    IOSTANDARD LVCMOS33 } [get_ports {bypass_B}]
set_property -dict { PACKAGE_PIN T1    IOSTANDARD LVCMOS33 } [get_ports {direction}]
set_property -dict { PACKAGE_PIN R2    IOSTANDARD LVCMOS33 } [get_ports {serial_in}]


## LEDs
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports {leds[0]}]
set_property -dict { PACKAGE_PIN E19   IOSTANDARD LVCMOS33 } [get_ports {leds[1]}]
set_property -dict { PACKAGE_PIN U19   IOSTANDARD LVCMOS33 } [get_ports {leds[2]}]
set_property -dict { PACKAGE_PIN V19   IOSTANDARD LVCMOS33 } [get_ports {leds[3]}]
set_property -dict { PACKAGE_PIN W18   IOSTANDARD LVCMOS33 } [get_ports {leds[4]}]
set_property -dict { PACKAGE_PIN U15   IOSTANDARD LVCMOS33 } [get_ports {leds[5]}]
set_property -dict { PACKAGE_PIN U14   IOSTANDARD LVCMOS33 } [get_ports {leds[6]}]
set_property -dict { PACKAGE_PIN V14   IOSTANDARD LVCMOS33 } [get_ports {leds[7]}]
set_property -dict { PACKAGE_PIN V13   IOSTANDARD LVCMOS33 } [get_ports {leds[8]}]
set_property -dict { PACKAGE_PIN V3    IOSTANDARD LVCMOS33 } [get_ports {leds[9]}]
set_property -dict { PACKAGE_PIN W3    IOSTANDARD LVCMOS33 } [get_ports {leds[10]}]
set_property -dict { PACKAGE_PIN U3    IOSTANDARD LVCMOS33 } [get_ports {leds[11]}]
set_property -dict { PACKAGE_PIN P3    IOSTANDARD LVCMOS33 } [get_ports {leds[12]}]
set_property -dict { PACKAGE_PIN N3    IOSTANDARD LVCMOS33 } [get_ports {leds[13]}]
set_property -dict { PACKAGE_PIN P1    IOSTANDARD LVCMOS33 } [get_ports {leds[14]}]
set_property -dict { PACKAGE_PIN L1    IOSTANDARD LVCMOS33 } [get_ports {leds[15]}]


##7 Segment Display
#set_property -dict { PACKAGE_PIN W7   IOSTANDARD LVCMOS33 } [get_ports {seg[0]}]
#set_property -dict { PACKAGE_PIN W6   IOSTANDARD LVCMOS33 } [get_ports {seg[1]}]
#set_property -dict { PACKAGE_PIN U8   IOSTANDARD LVCMOS33 } [get_ports {seg[2]}]
#set_property -dict { PACKAGE_PIN V8   IOSTANDARD LVCMOS33 } [get_ports {seg[3]}]
#set_property -dict { PACKAGE_PIN U5   IOSTANDARD LVCMOS33 } [get_ports {seg[4]}]
#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.

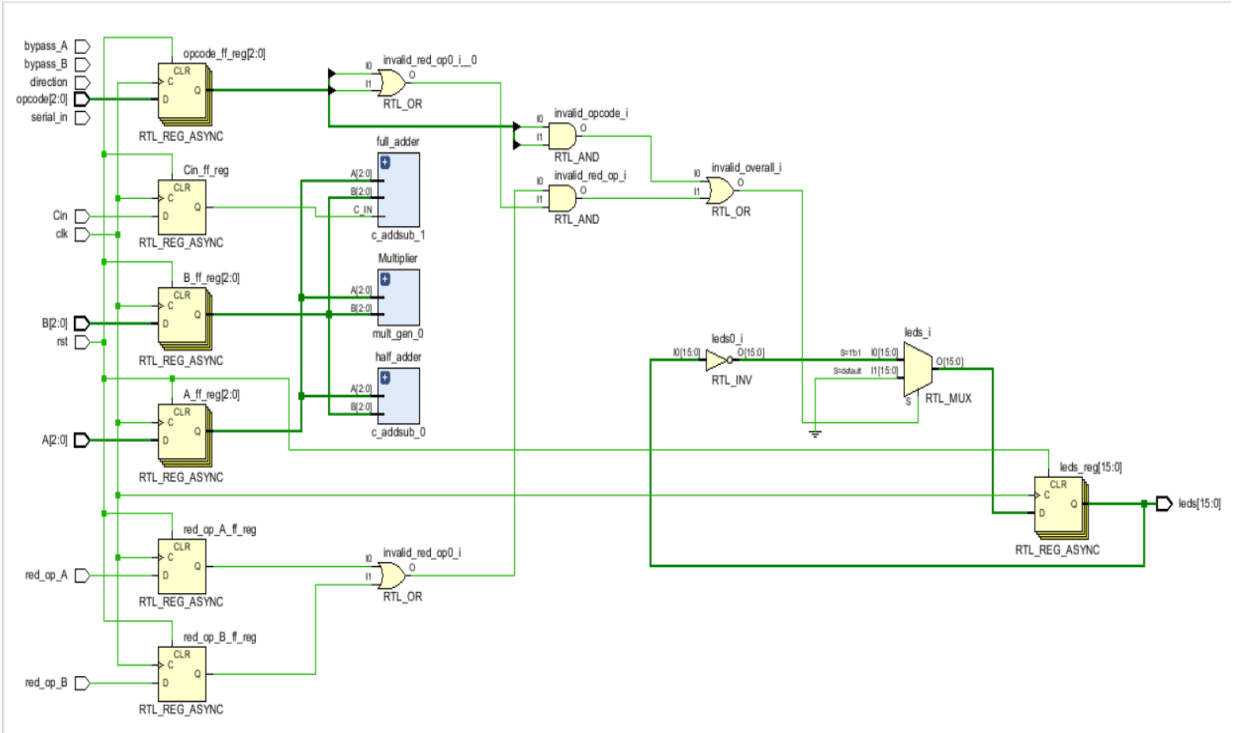

Figure (37): generic schematic (Elaborated)

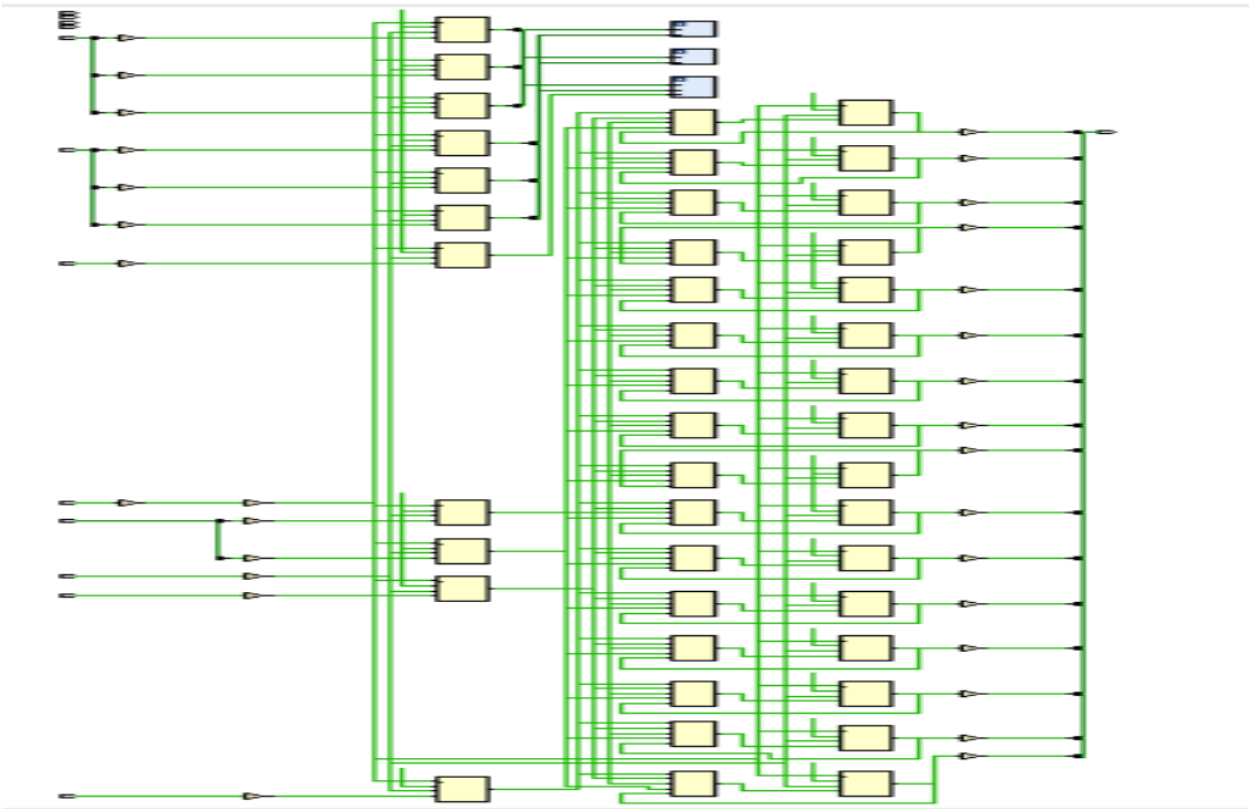

Figure (38): Synthesized schematic

Name	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
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

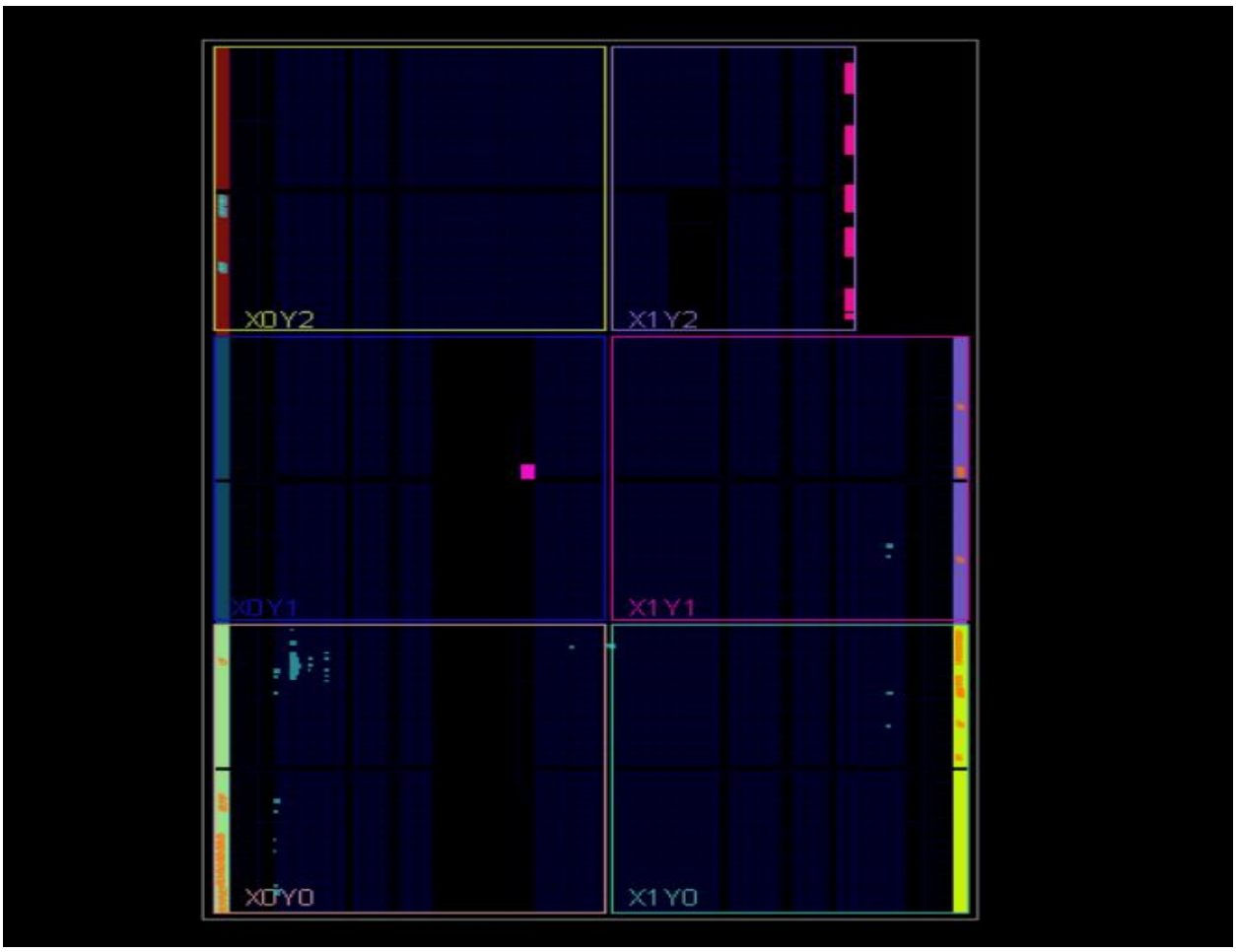

Figure (40): device after implementation

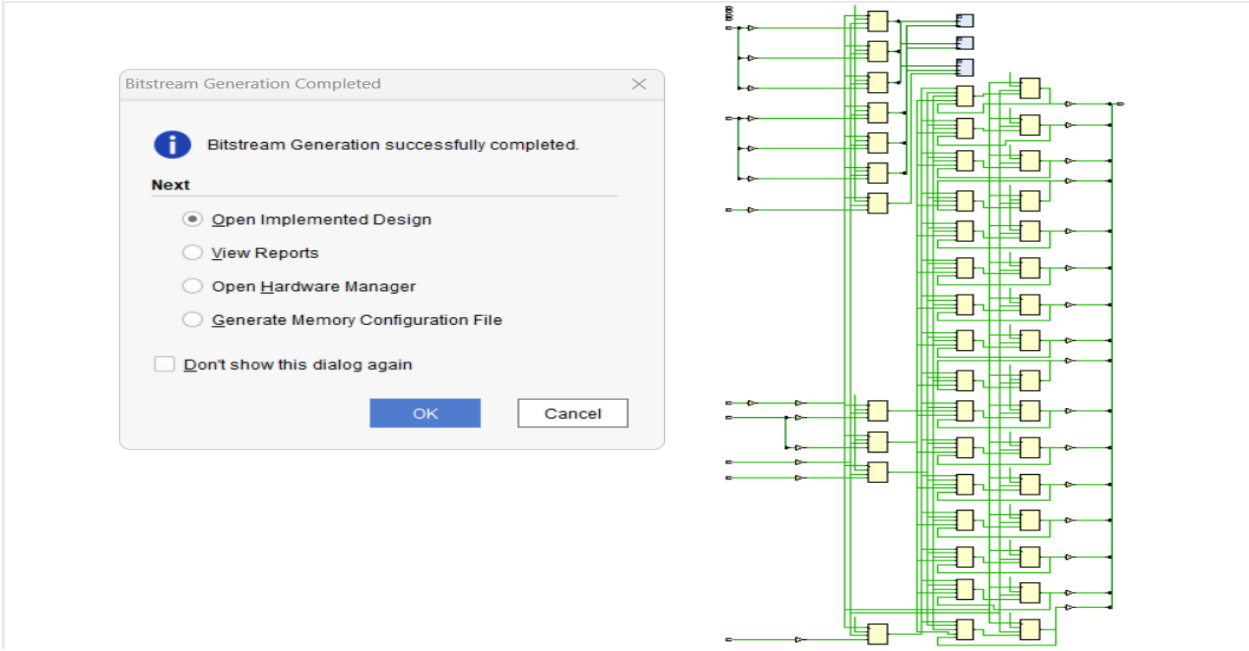

Figure (41): successful bitstream generation

==========