Lab 4 Report

Name: Frank Le

UT EID: fpl227

Professor: Dr. Orshansky

Checklist:

Part 1 -

i. Simulation waveform of the flight attendant call system
ii. K-map for next_state for dataflow modelling
iii. Boolean expression for next_state for dataflow modelling
iv. Completed design file (.v) for dataflow modelling

Part 2 -

v. State graph for rising-edge detector
vi. Completed design files (.v) including the top module and clock divider
vii. Testbench code
viii. Simulation waveform screenshot
ix. Constraints file (Just the uncommented portion)

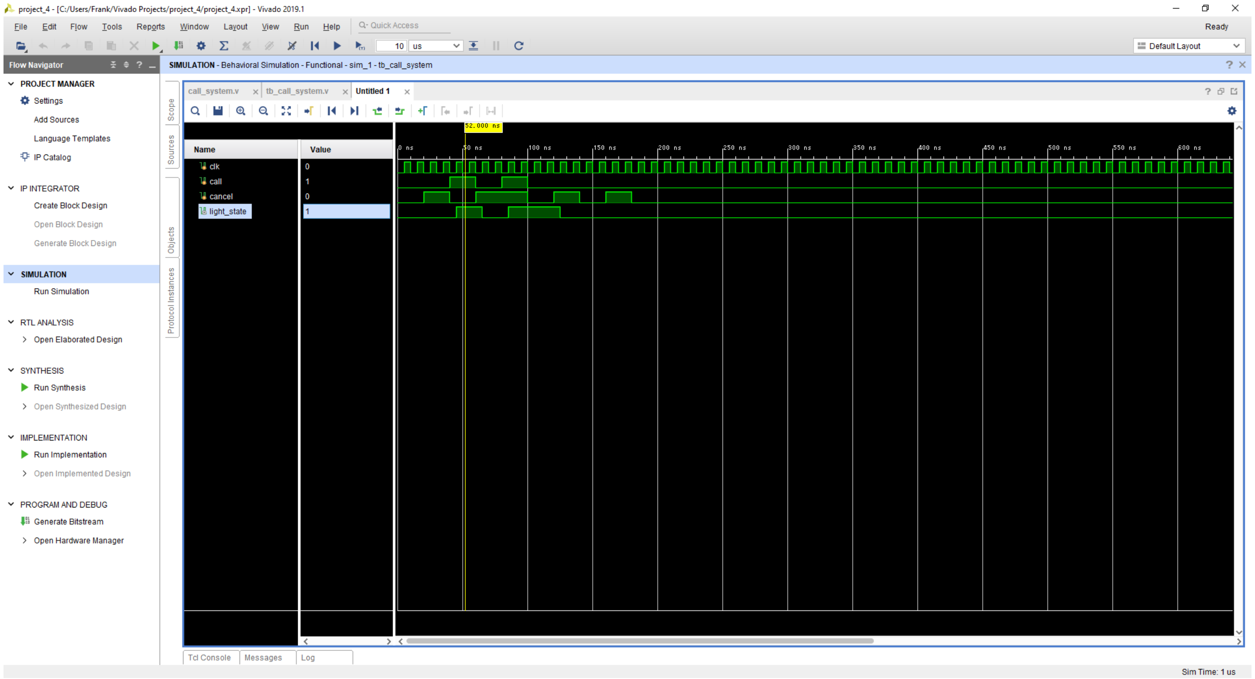
Part 3 -

x. High-level block diagram of the system
xi. Frequency calculations
xii. Completed design files (.v) of all modules in the system
xiii. Testbench code
xiv. Simulation waveform screenshot
xv. Constraints file (Just the uncommented portion)

Note *→ The Verilog codes and the uncommented portions of the constraint files should be copied in your lab report and the* ***actual Verilog (.v), Constraint (.xdc)***

files and Bitstream (.bit) files *need to be zipped and submitted as well on Canvas. You are not allowed to change your codes after final submission as the TAs may download the submitted codes or bitstream files from Canvas during checkouts. For the truth table, K-maps minimizations and algebraic expressions, you are free to draw them on paper and then put the pictures in your lab report, but please make sure it is legible for the TAs to grade it properly.*

i) Simulation waveform of the flight attendant call system



project_4 - [C:/Users/Frank/Vivado Projects/project_4/project_4.xpr] - Vivado 2019.1

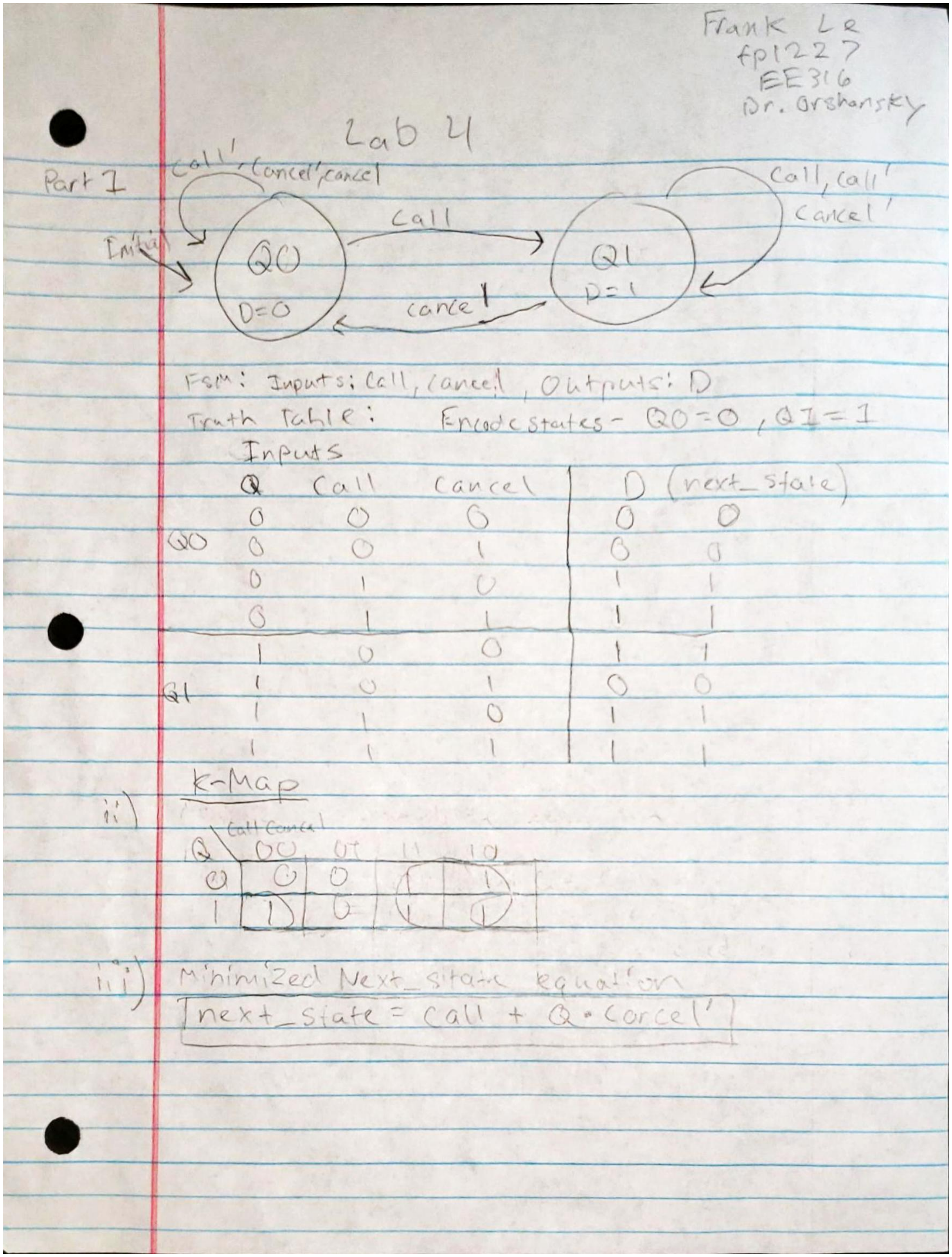
SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_call_system

Name	Value
clk	0
call	1
cancel	0
light_state	1

Sim Time: 1 us

ii) K-map for next_state for dataflow modelling

iii) Boolean expression for next_state for dataflow modelling



Frank Le
fp1227
EE316
Dr. Orshansky

Lab 4

Part 1

call', cancel', cancel
Initial
Q0
D=0
call
Q1
D=1
cancel
Call, call'
Cancel'

FSM: Inputs: Call, cancel, Outputs: D

Truth Table: Encode states - Q0=0, Q1=1

	Inputs				
	Q	Call	Cancel	D	(next_state)
Q0	0	0	0	0	0
	0	0	1	0	0
	0	1	0	1	1
	0	1	1	1	1
Q1	1	0	0	1	1
	1	0	1	0	0
	1	1	0	1	1
	1	1	1	1	1

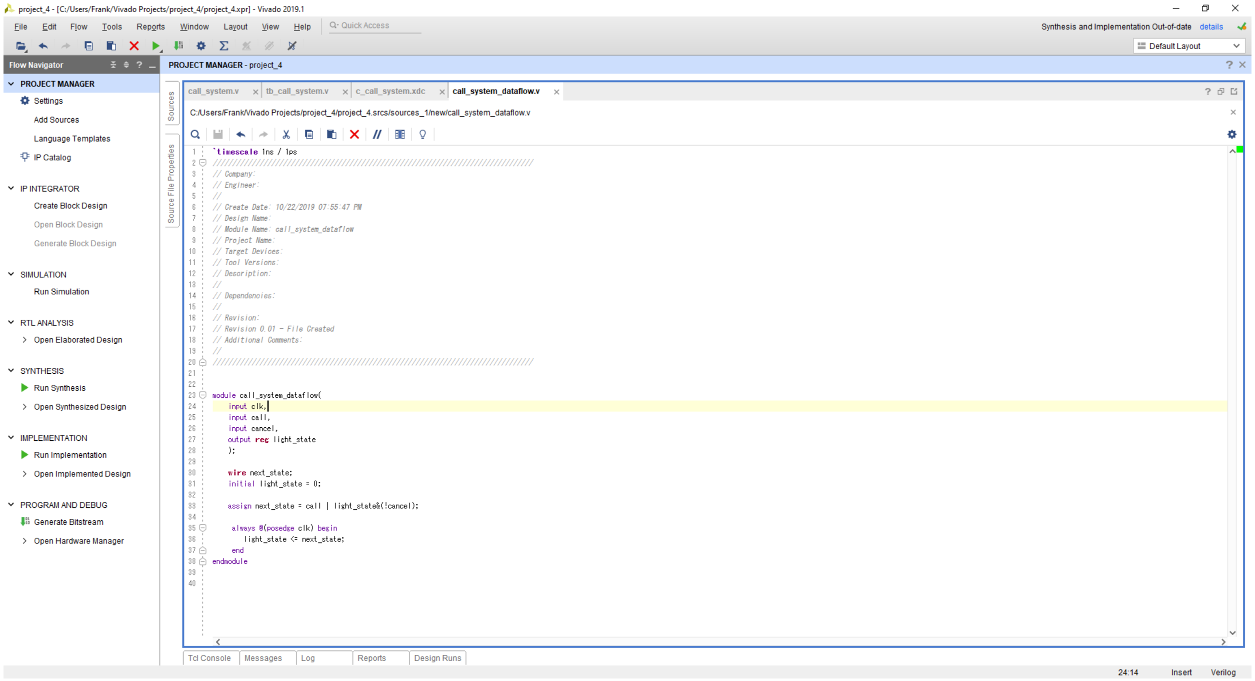
ii) K-Map

Q \ Call Cancel	00	01	11	10
0	0	0	1	1
1	1	0	1	1

iii) Minimized Next_state equation

$$\boxed{\text{next\_state} = \text{Call} + Q \cdot \text{Corcel}'}$$

iv) Completed design file (.v) for dataflow modelling

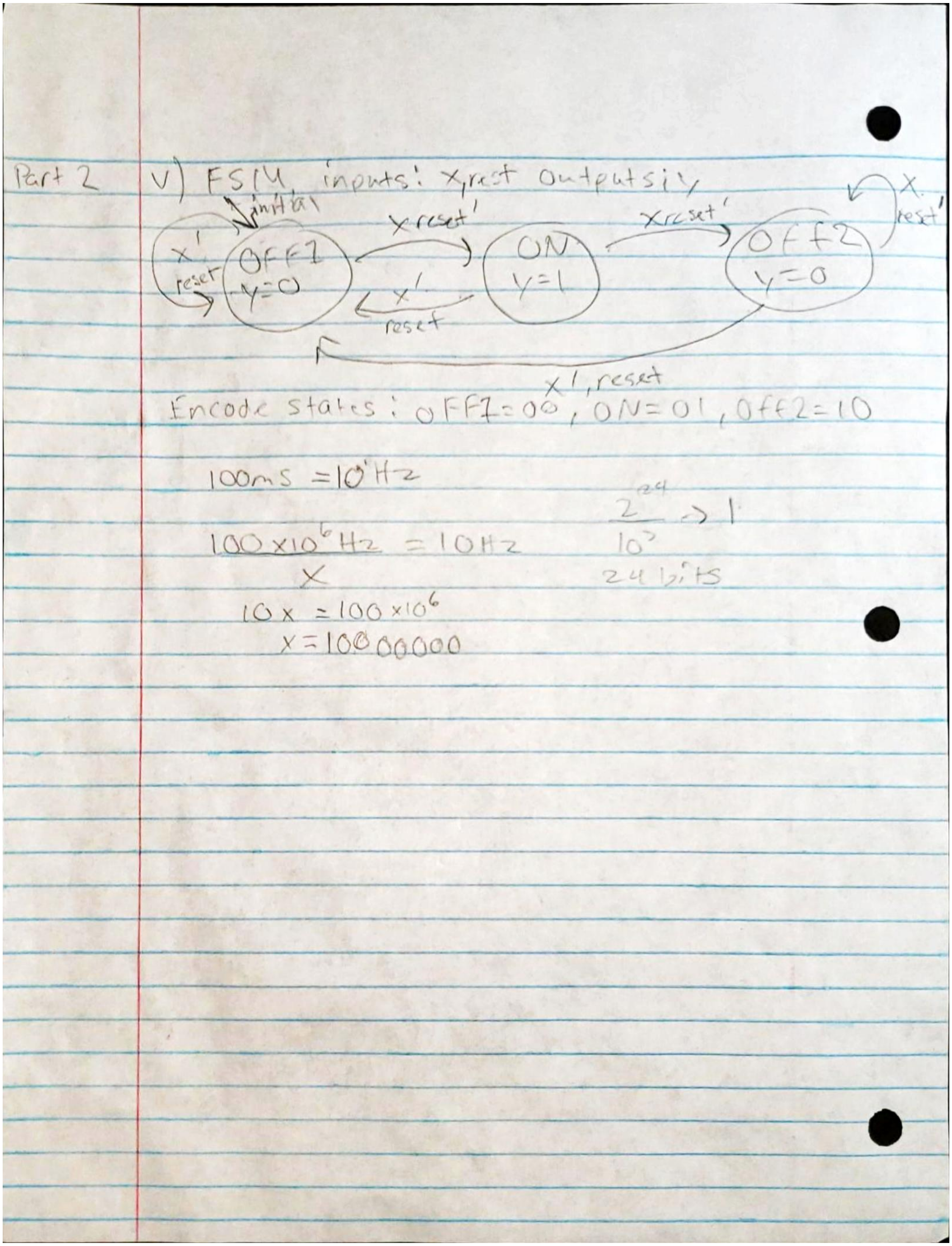


project_4 - [C:/Users/Frank/Vivado Projects/project_4/project_4.xpr] - Vivado 2019.1

C:/Users/Frank/Vivado Projects/project_4/project_4.srcs/sources_1/new/call_system_dataflow.v

```verilog
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 10/22/2019 07:55:47 PM
// Design Name:
// Module Name: call_system_dataflow
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module call_system_dataflow(
    input clk,
    input call,
    input cancel,
    output reg light_state
    );

    wire next_state;
    initial light_state = 0;

    assign next_state = call | light_state&(!cancel);

     always @(posedge clk) begin
        light_state <= next_state;
     end
endmodule
```

v) State graph for rising-edge detector



Part 2

v) FSM inputs: x, rest outputs: y

initial
OFF1
y=0
x'
reset
x reset'
ON
y=1
x'
reset
x reset'
OFF2
y=0
x.
rest'
x', reset

Encode states: OFF1=00, ON=01, Off2=10

$100mS = 10^1 Hz$

$\frac{100 \times 10^6 Hz}{x} = 10 Hz$

$10x = 100 \times 10^6$

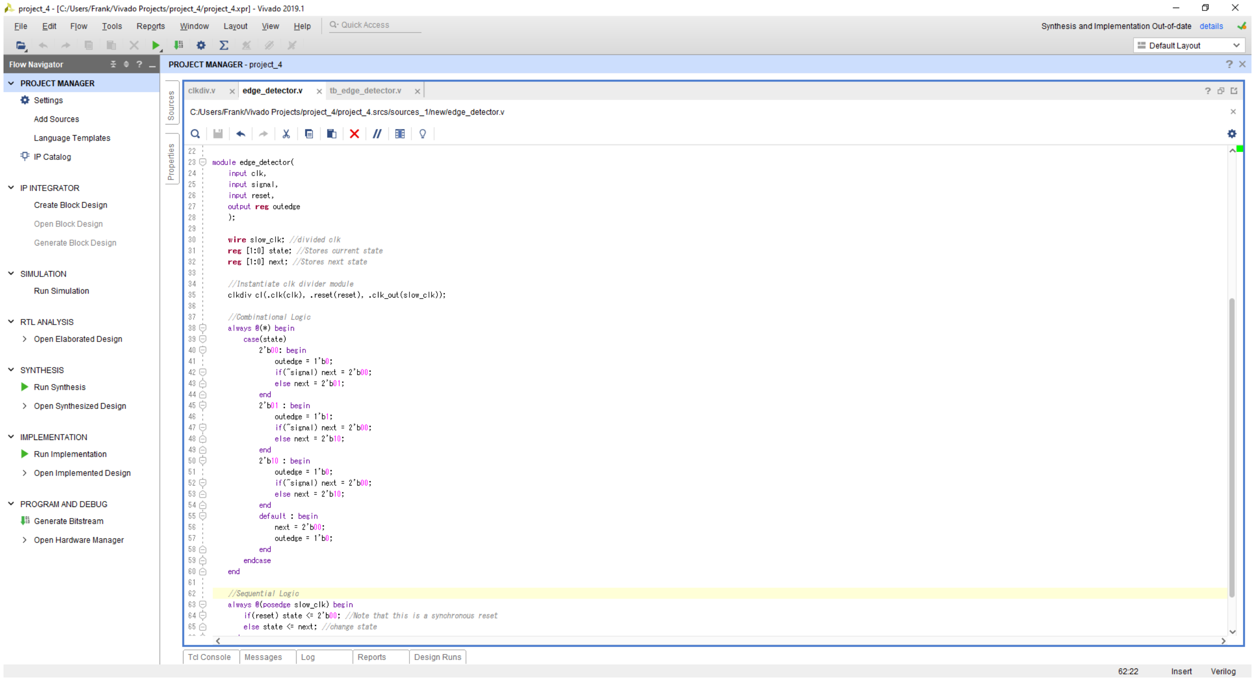
$x = 100000000$

$\frac{2^{24}}{10^3} \to 1$

24 bits

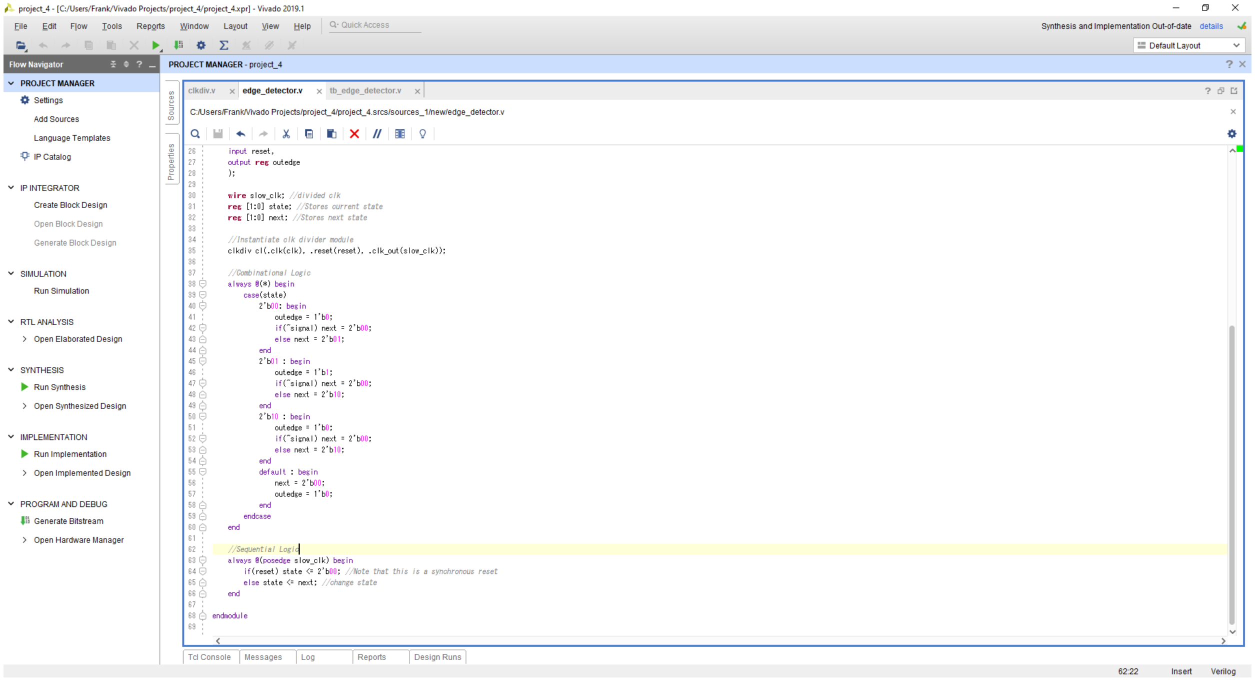
vi) Completed design files (.v) including the top module and clock divider

Top Module:



C:/Users/Frank/Vivado Projects/project_4/project_4.srcs/sources_1/new/edge_detector.v

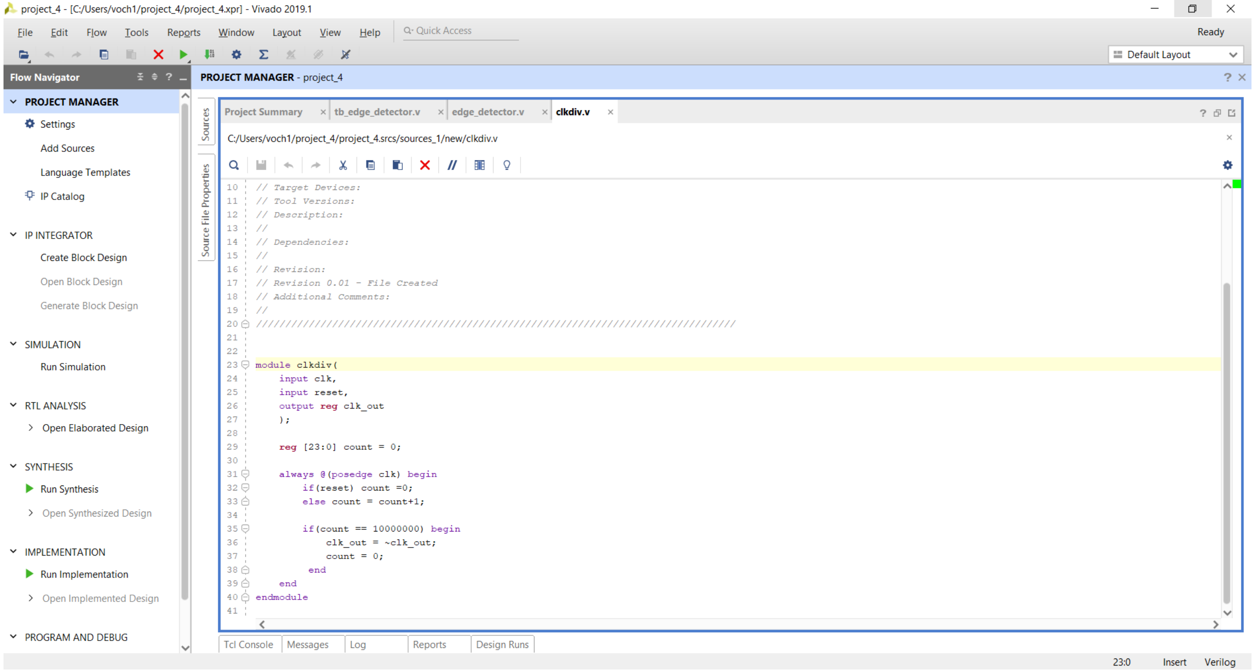
```
module edge_detector(
    input clk,
    input signal,
    input reset,
    output reg outedge
    );

    wire slow_clk; //divided clk
    reg [1:0] state; //Stores current state
    reg [1:0] next; //Stores next state

    //Instantiate clk divider module
    clkdiv cl(.clk(clk), .reset(reset), .clk_out(slow_clk));

    //Combinational Logic
    always @(*) begin
        case(state)
            2'b00: begin
                outedge = 1'b0;
                if(~signal) next = 2'b00;
                else next = 2'b01;
            end
            2'b01 : begin
                outedge = 1'b1;
                if(~signal) next = 2'b00;
                else next = 2'b10;
            end
            2'b10 : begin
                outedge = 1'b0;
                if(~signal) next = 2'b00;
                else next = 2'b10;
            end
            default : begin
                next = 2'b00;
                outedge = 1'b0;
            end
        endcase
    end

    //Sequential Logic
    always @(posedge slow_clk) begin
        if(reset) state <= 2'b00; //Note that this is a synchronous reset
        else state <= next; //change state
```



C:/Users/Frank/Vivado Projects/project_4/project_4.srcs/sources_1/new/edge_detector.v

```
    input reset,
    output reg outedge
    );

    wire slow_clk; //divided clk
    reg [1:0] state; //Stores current state
    reg [1:0] next; //Stores next state

    //Instantiate clk divider module
    clkdiv cl(.clk(clk), .reset(reset), .clk_out(slow_clk));

    //Combinational Logic
    always @(*) begin
        case(state)
            2'b00: begin
                outedge = 1'b0;
                if(~signal) next = 2'b00;
                else next = 2'b01;
            end
            2'b01 : begin
                outedge = 1'b1;
                if(~signal) next = 2'b00;
                else next = 2'b10;
            end
            2'b10 : begin
                outedge = 1'b0;
                if(~signal) next = 2'b00;
                else next = 2'b10;
            end
            default : begin
                next = 2'b00;
                outedge = 1'b0;
            end
        endcase
    end

    //Sequential Logic
    always @(posedge slow_clk) begin
        if(reset) state <= 2'b00; //Note that this is a synchronous reset
        else state <= next; //change state
    end

endmodule
```

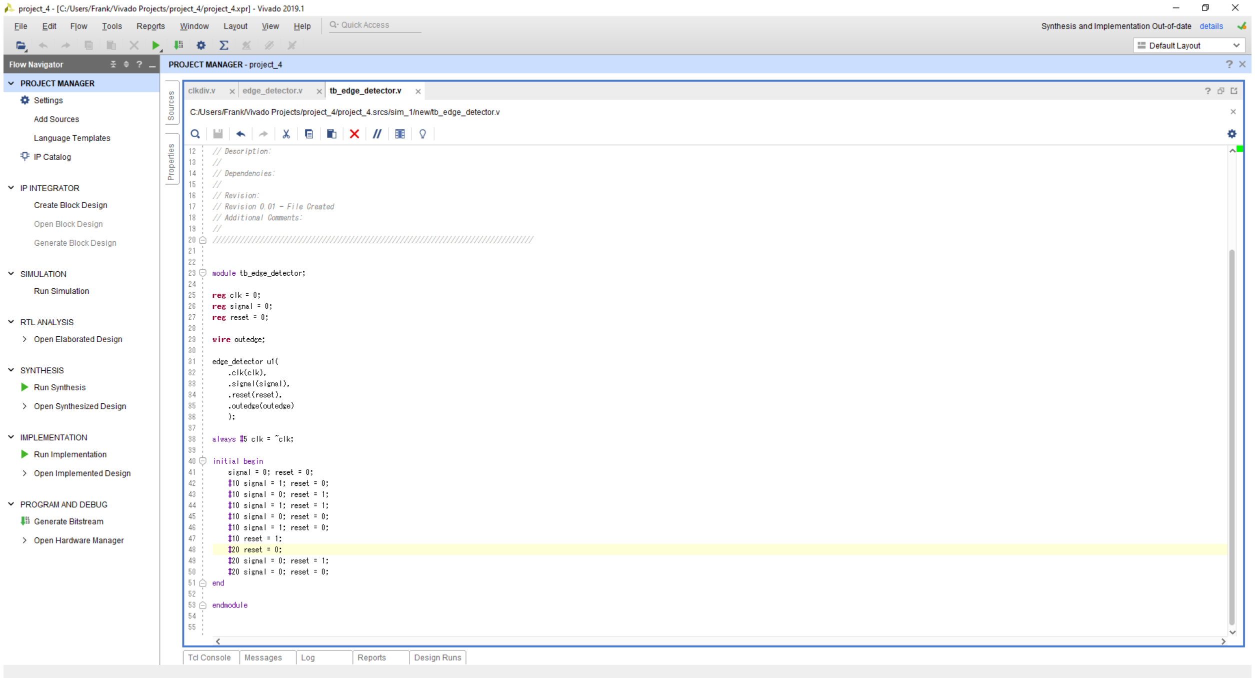
Clock Divider:



C:/Users/voch1/project_4/project_4.srcs/sources_1/new/clkdiv.v

```
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module clkdiv(
    input clk,
    input reset,
    output reg clk_out
    );

    reg [23:0] count = 0;

    always @(posedge clk) begin
        if(reset) count =0;
        else count = count+1;

        if(count == 10000000) begin
            clk_out = ~clk_out;
            count = 0;
        end
    end
endmodule

```

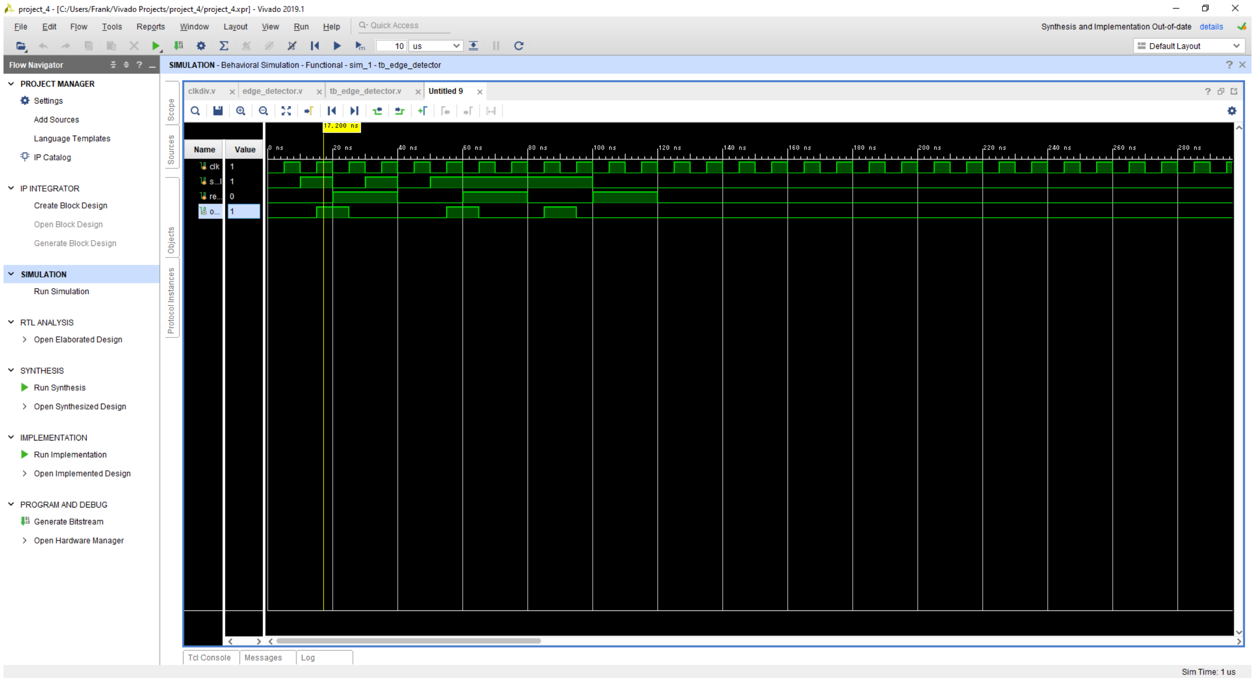
vii) Testbench code



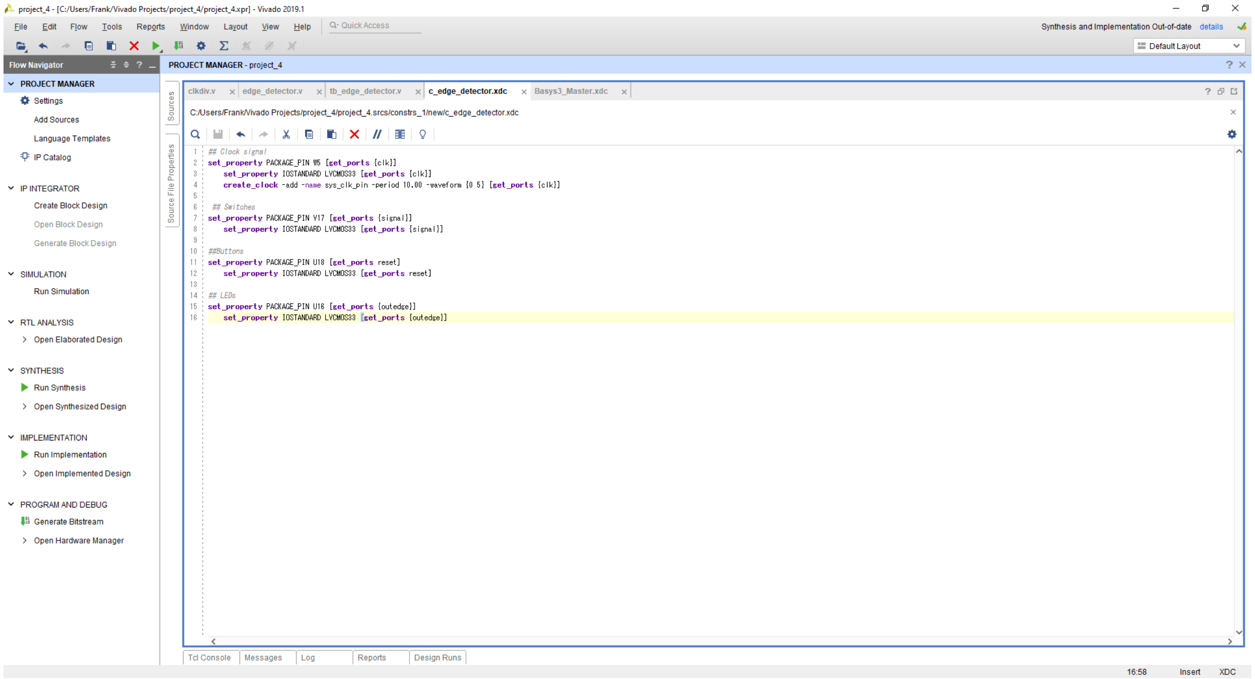
C:/Users/Frank/Vivado Projects/project_4/project_4.srcs/sim_1/new/tb_edge_detector.v

```
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module tb_edge_detector;

reg clk = 0;
reg signal = 0;
reg reset = 0;

wire outedge;

edge_detector u1(
    .clk(clk),
    .signal(signal),
    .reset(reset),
    .outedge(outedge)
    );

always #5 clk = ~clk;

initial begin
    signal = 0; reset = 0;
    #10 signal = 1; reset = 0;
    #10 signal = 0; reset = 1;
    #10 signal = 1; reset = 1;
    #10 signal = 0; reset = 0;
    #10 signal = 1; reset = 0;
    #10 reset = 1;
    #20 reset = 0;
    #20 signal = 0; reset = 1;
    #20 signal = 0; reset = 0;
end

endmodule


```

viii) Simulation waveform screenshot



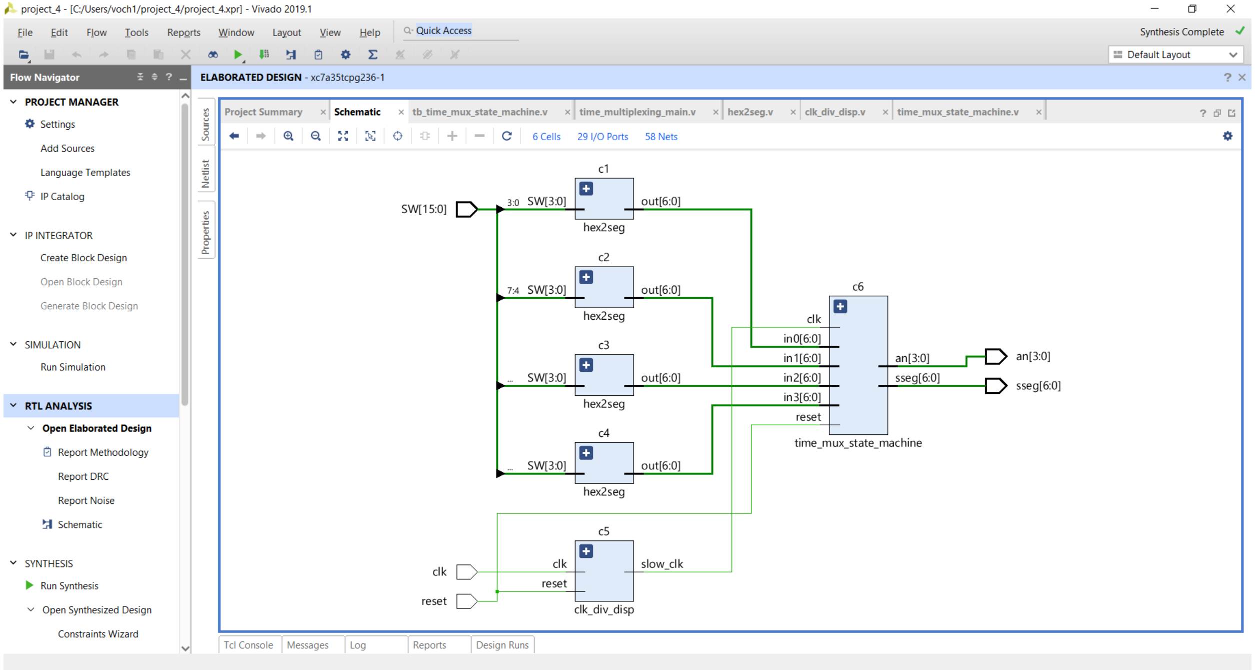
ix) Constraints file (Just the uncommented portion)



C:/Users/Frank/Vivado Projects/project_4/project_4.srcs/constrs_1/new/c_edge_detector.xdc

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

 ## Switches
set_property PACKAGE_PIN V17 [get_ports {signal}]
    set_property IOSTANDARD LVCMOS33 [get_ports {signal}]

##Buttons
set_property PACKAGE_PIN U18 [get_ports reset]
    set_property IOSTANDARD LVCMOS33 [get_ports reset]

## LEDs
set_property PACKAGE_PIN U16 [get_ports {outedge}]
    set_property IOSTANDARD LVCMOS33 [get_ports {outedge}]
```

x) High-level block diagram of the system



xi) Frequency calculations

5 Hz

(100 x 10^6 Hz)/x = 5 Hz

x = (100 x 10^6 Hz) / 5 Hz

x = 20000000

2^25/20000000 = 1.7

25 bits

5 kHz

(100 x 10^6 Hz)/x = 5 kHz

x = (100 x 10^6 Hz) / 5 kHz

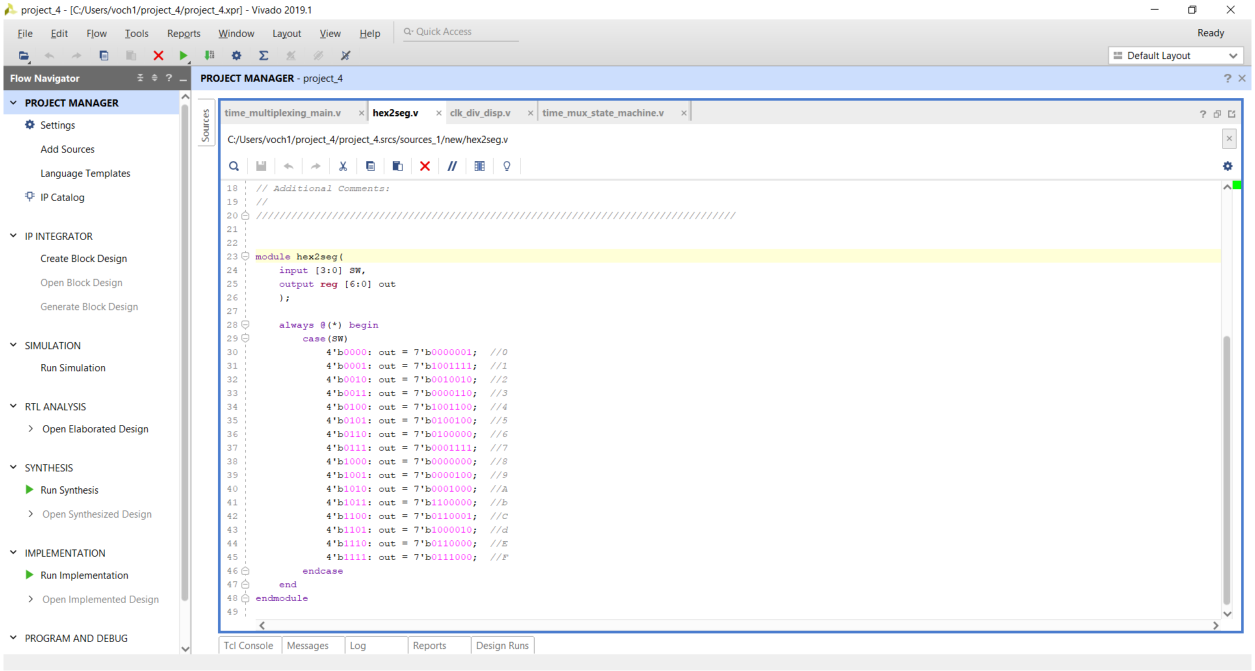
x = 20000

2^15/20000 = 1.6

15 bits

xii) Completed design files (.v) of all modules in the system

hex2seg:

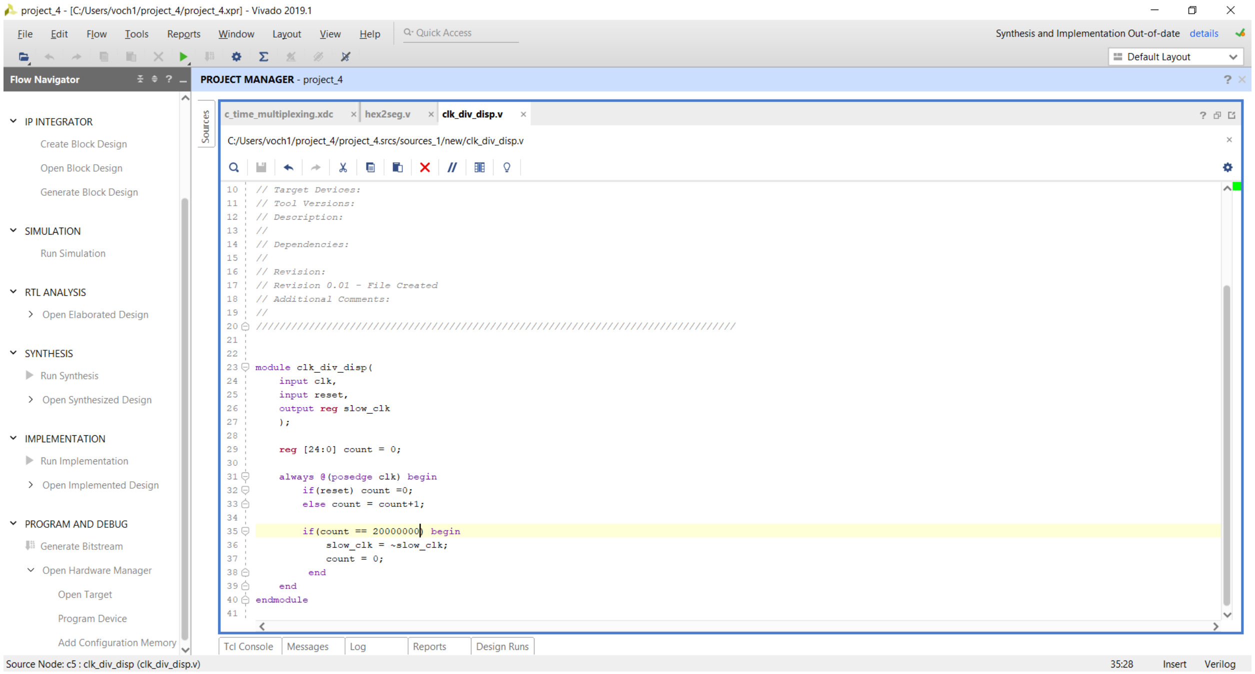


C:/Users/voch1/project_4/project_4.srcs/sources_1/new/hex2seg.v

```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module hex2seg(
    input [3:0] SW,
    output reg [6:0] out
    );

    always @(*) begin
        case(SW)
            4'b0000: out = 7'b0000001;  //0
            4'b0001: out = 7'b1001111;  //1
            4'b0010: out = 7'b0010010;  //2
            4'b0011: out = 7'b0000110;  //3
            4'b0100: out = 7'b1001100;  //4
            4'b0101: out = 7'b0100100;  //5
            4'b0110: out = 7'b0100000;  //6
            4'b0111: out = 7'b0001111;  //7
            4'b1000: out = 7'b0000000;  //8
            4'b1001: out = 7'b0000100;  //9
            4'b1010: out = 7'b0001000;  //A
            4'b1011: out = 7'b1100000;  //b
            4'b1100: out = 7'b0110001;  //C
            4'b1101: out = 7'b1000010;  //d
            4'b1110: out = 7'b0110000;  //E
            4'b1111: out = 7'b0111000;  //F
        endcase
    end
endmodule

```

clk_div_disp:

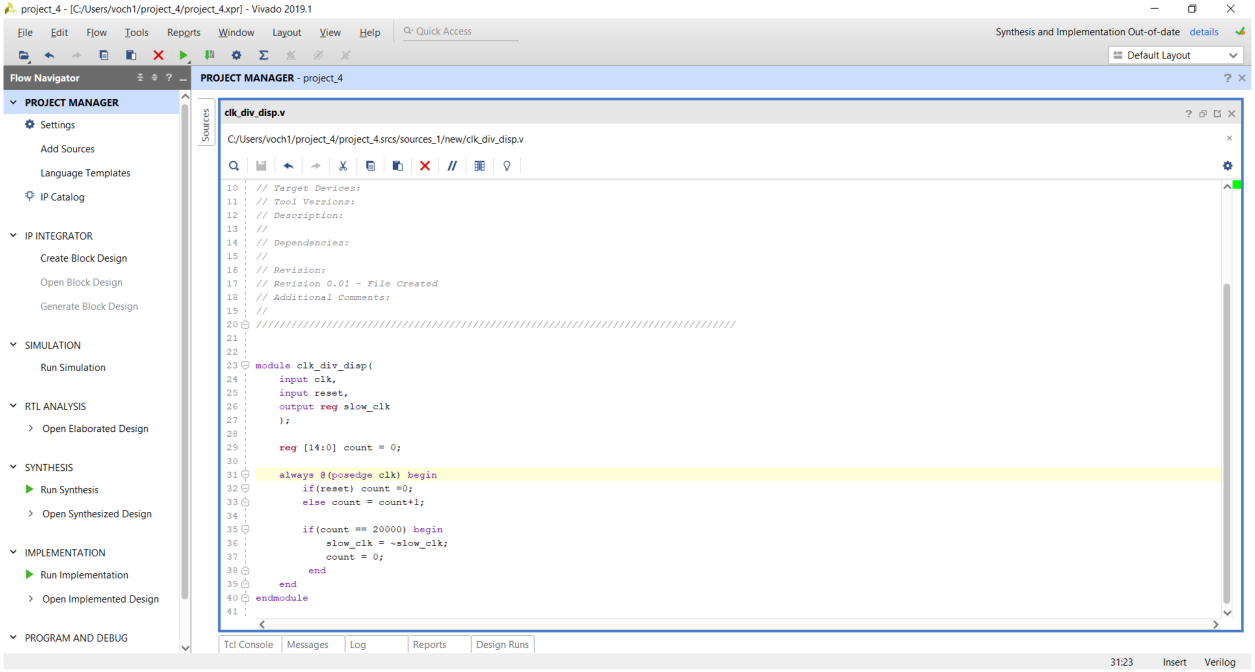
5 Hz



C:/Users/voch1/project_4/project_4.srcs/sources_1/new/clk_div_disp.v

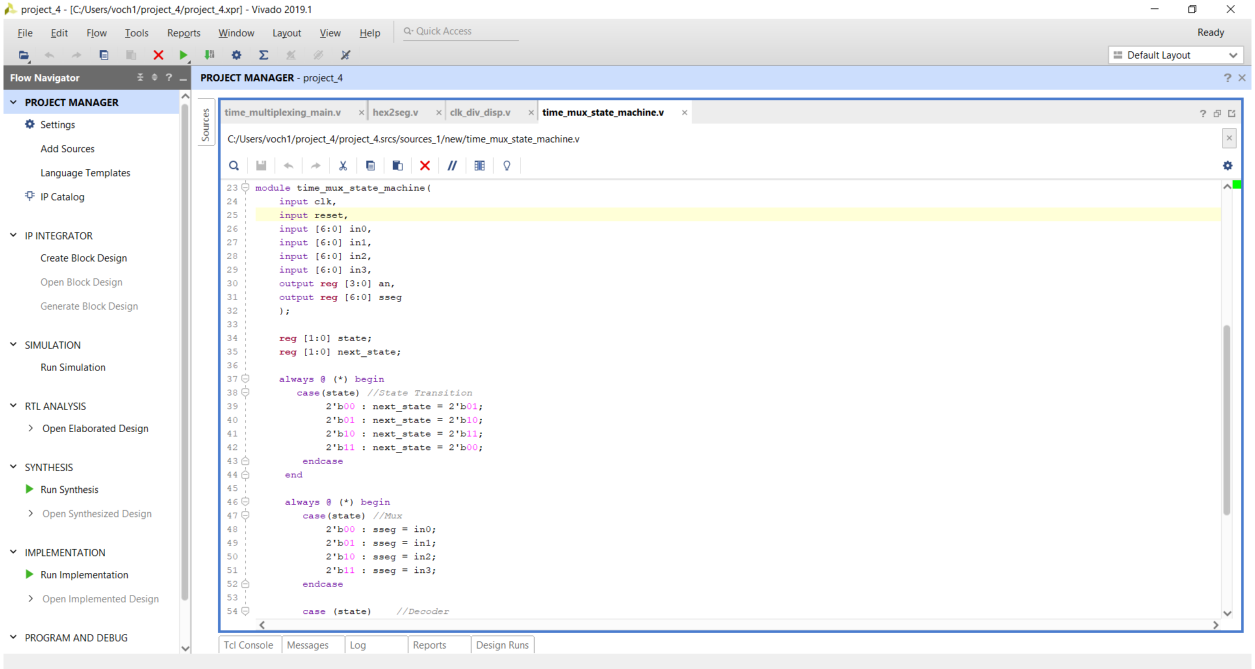
```
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module clk_div_disp(
    input clk,
    input reset,
    output reg slow_clk
    );

    reg [24:0] count = 0;

    always @(posedge clk) begin
        if(reset) count =0;
        else count = count+1;

        if(count == 20000000) begin
            slow_clk = ~slow_clk;
            count = 0;
         end
    end
endmodule

```

5 kHz

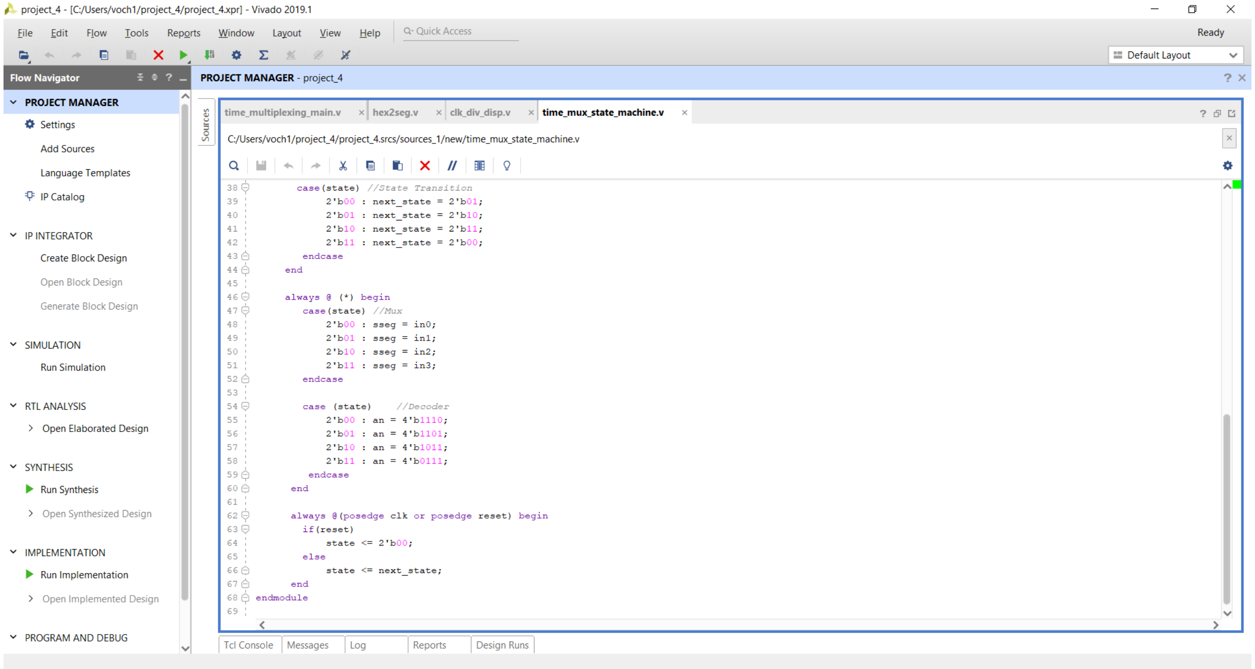


```verilog
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module clk_div_disp(
    input clk,
    input reset,
    output reg slow_clk
    );

    reg [14:0] count = 0;

    always @(posedge clk) begin
        if(reset) count =0;
        else count = count+1;

        if(count == 20000) begin
            slow_clk = ~slow_clk;
            count = 0;
         end
    end
endmodule

```

time_mux_state_machine:



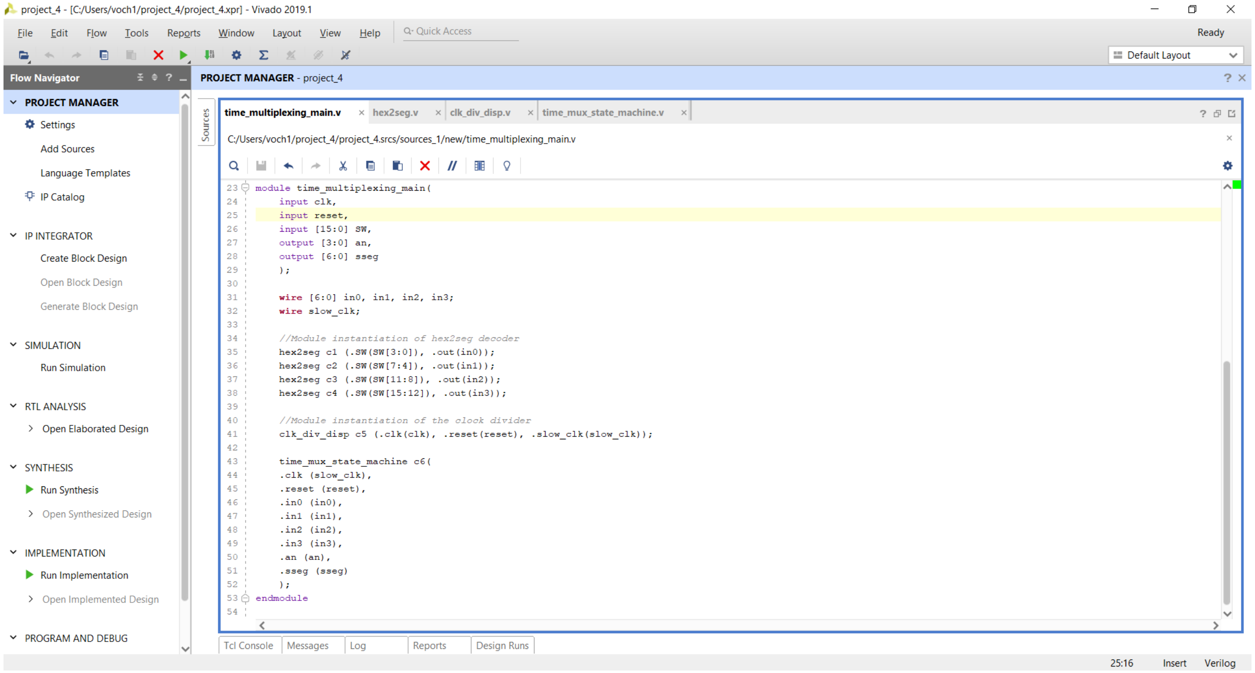
```verilog
module time_mux_state_machine(
    input clk,
    input reset,
    input [6:0] in0,
    input [6:0] in1,
    input [6:0] in2,
    input [6:0] in3,
    output reg [3:0] an,
    output reg [6:0] sseg
    );

    reg [1:0] state;
    reg [1:0] next_state;

    always @ (*) begin
       case(state) //State Transition
            2'b00 : next_state = 2'b01;
            2'b01 : next_state = 2'b10;
            2'b10 : next_state = 2'b11;
            2'b11 : next_state = 2'b00;
         endcase
     end

     always @ (*) begin
         case(state) //Mux
            2'b00 : sseg = in0;
            2'b01 : sseg = in1;
            2'b10 : sseg = in2;
            2'b11 : sseg = in3;
         endcase

         case (state)    //Decoder
```



C:/Users/voch1/project_4/project_4.srcs/sources_1/new/time_mux_state_machine.v

```verilog
         case(state) //State Transition
              2'b00 : next_state = 2'b01;
              2'b01 : next_state = 2'b10;
              2'b10 : next_state = 2'b11;
              2'b11 : next_state = 2'b00;
          endcase
      end

      always @ (*) begin
          case(state) //Mux
              2'b00 : sseg = in0;
              2'b01 : sseg = in1;
              2'b10 : sseg = in2;
              2'b11 : sseg = in3;
          endcase

          case (state)    //Decoder
              2'b00 : an = 4'b1110;
              2'b01 : an = 4'b1101;
              2'b10 : an = 4'b1011;
              2'b11 : an = 4'b0111;
           endcase
       end

       always @(posedge clk or posedge reset) begin
          if(reset)
              state <= 2'b00;
          else
              state <= next_state;
       end
endmodule
```

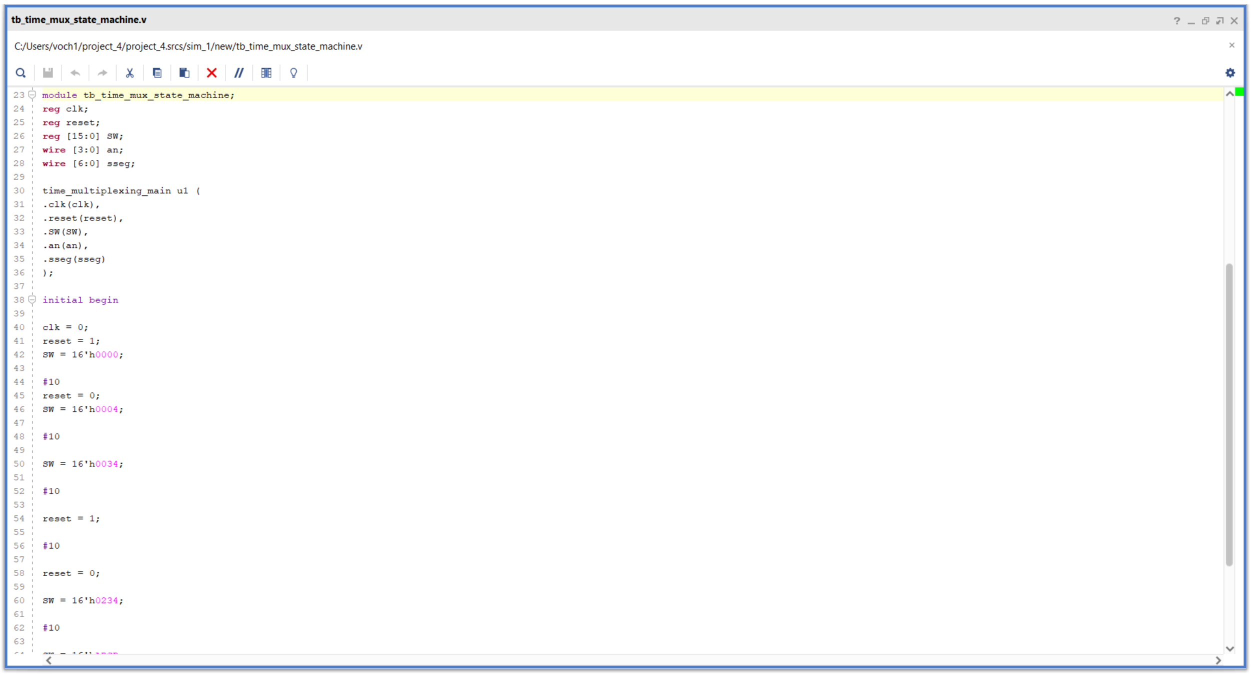
time_multiplexing_main:



C:/Users/voch1/project_4/project_4.srcs/sources_1/new/time_multiplexing_main.v

```verilog
module time_multiplexing_main(
    input clk,
    input reset,
    input [15:0] SW,
    output [3:0] an,
    output [6:0] sseg
    );

    wire [6:0] in0, in1, in2, in3;
    wire slow_clk;

    //Module instantiation of hex2seg decoder
    hex2seg c1 (.SW(SW[3:0]), .out(in0));
    hex2seg c2 (.SW(SW[7:4]), .out(in1));
    hex2seg c3 (.SW(SW[11:8]), .out(in2));
    hex2seg c4 (.SW(SW[15:12]), .out(in3));

    //Module instantiation of the clock divider
    clk_div_disp c5 (.clk(clk), .reset(reset), .slow_clk(slow_clk));

    time_mux_state_machine c6(
    .clk (slow_clk),
    .reset (reset),
    .in0 (in0),
    .in1 (in1),
    .in2 (in2),
    .in3 (in3),
    .an (an),
    .sseg (sseg)
    );
endmodule
```

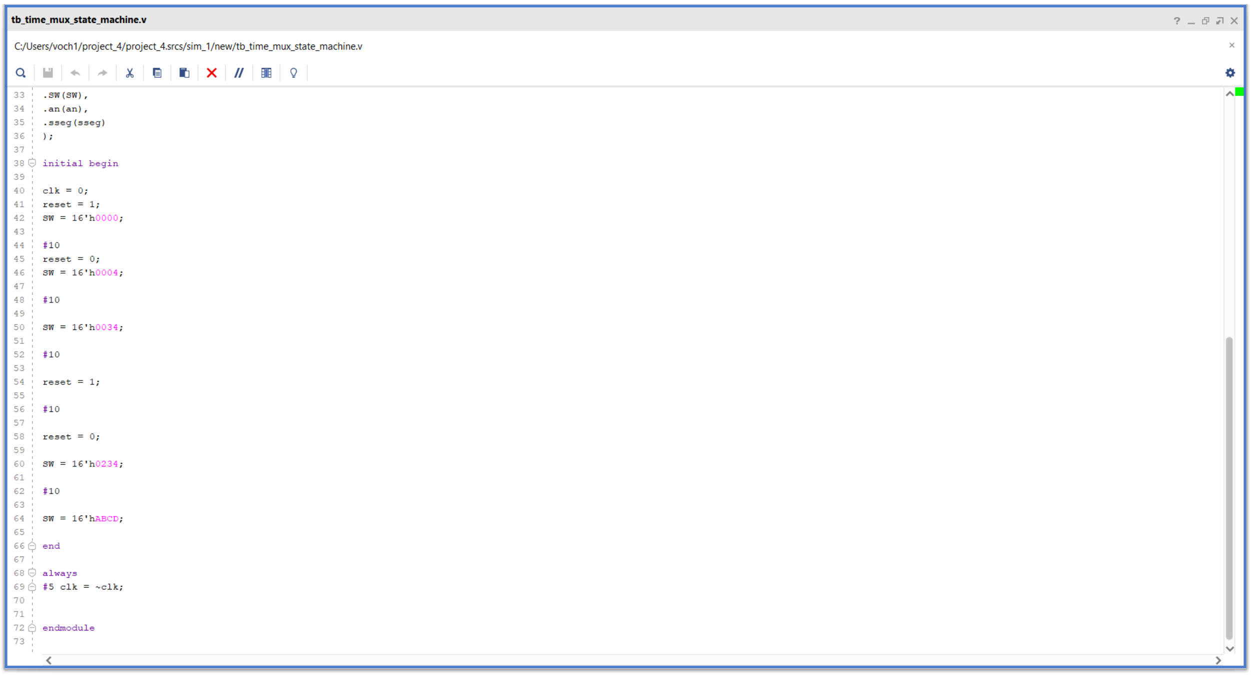
xiii) Testbench code



tb_time_mux_state_machine.v

C:/Users/voch1/project_4/project_4.srcs/sim_1/new/tb_time_mux_state_machine.v

```
module tb_time_mux_state_machine;
reg clk;
reg reset;
reg [15:0] SW;
wire [3:0] an;
wire [6:0] sseg;

time_multiplexing_main u1 (
.clk(clk),
.reset(reset),
.SW(SW),
.an(an),
.sseg(sseg)
);

initial begin

clk = 0;
reset = 1;
SW = 16'h0000;

#10
reset = 0;
SW = 16'h0004;

#10

SW = 16'h0034;

#10

reset = 1;

#10

reset = 0;

SW = 16'h0234;

#10

```

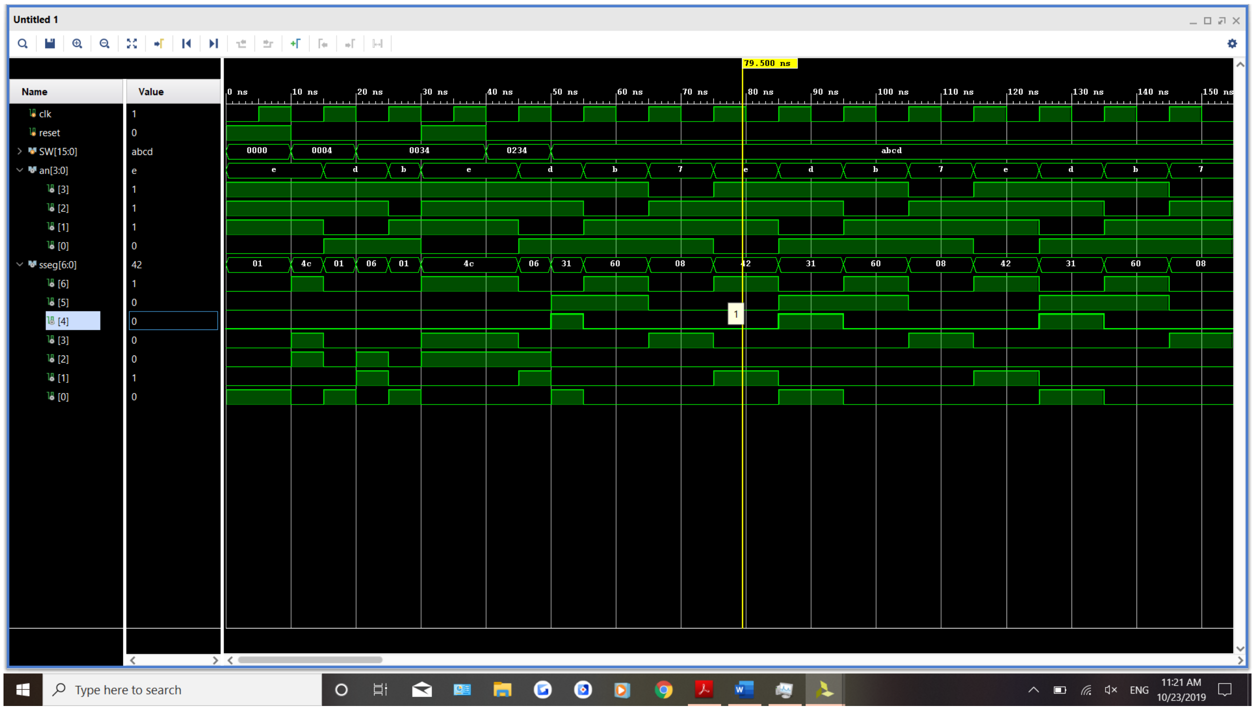


tb_time_mux_state_machine.v

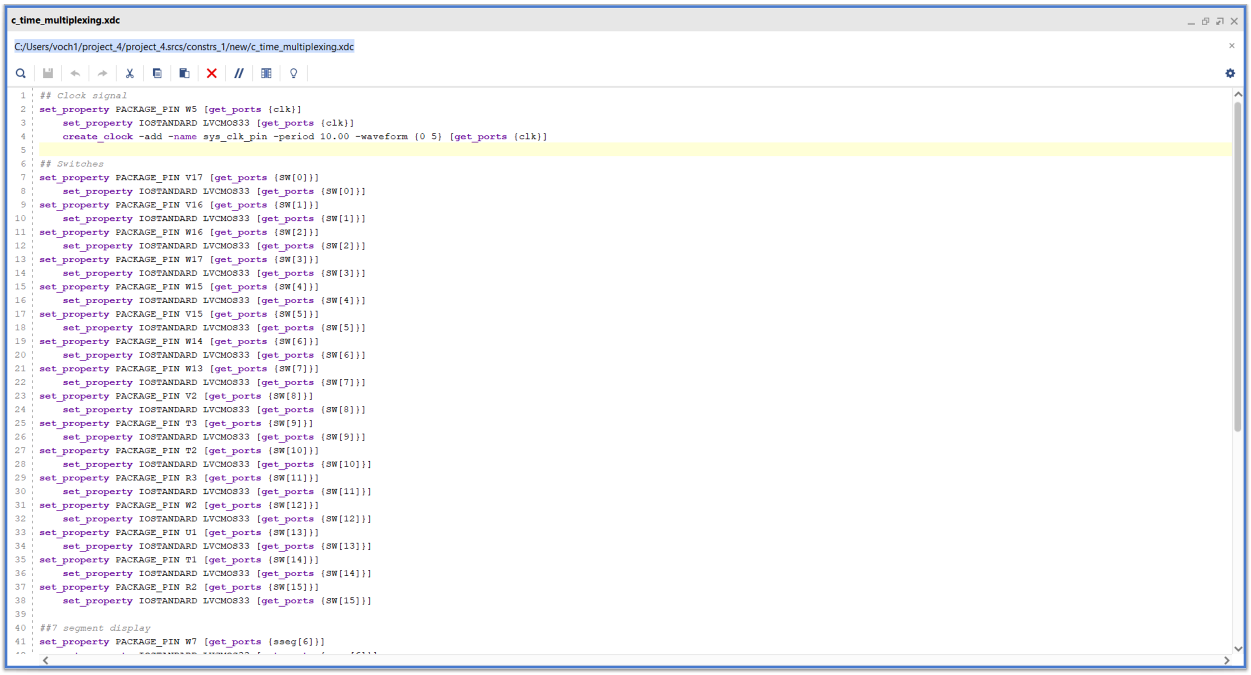
C:/Users/voch1/project_4/project_4.srcs/sim_1/new/tb_time_mux_state_machine.v

```
.SW(SW),
.an(an),
.sseg(sseg)
);

initial begin

clk = 0;
reset = 1;
SW = 16'h0000;

#10
reset = 0;
SW = 16'h0004;

#10

SW = 16'h0034;

#10

reset = 1;

#10

reset = 0;

SW = 16'h0234;

#10

SW = 16'hABCD;

end

always
#5 clk = ~clk;


endmodule

```

xiv) Simulation waveform screenshot

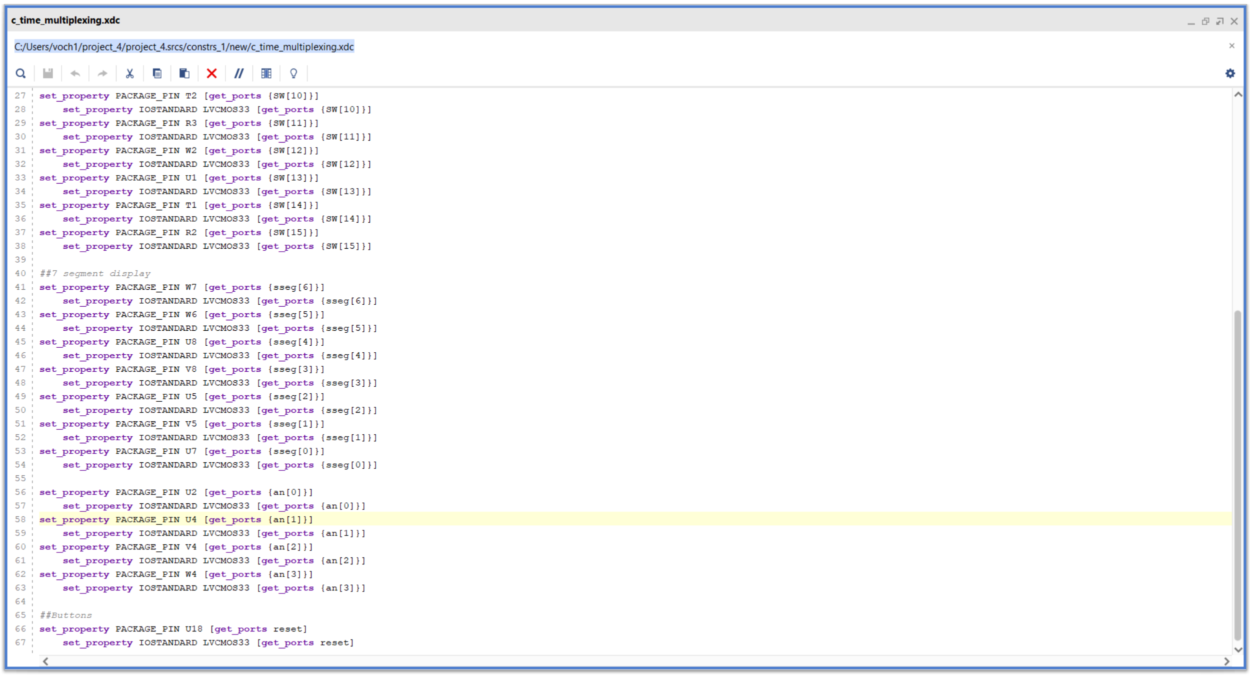


xv) Constraints file (Just the uncommented portion)


c_time_multiplexing.xdc

C:/Users/voch1/project_4/project_4.srcs/constrs_1/new/c_time_multiplexing.xdc

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

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

##7 segment display
set_property PACKAGE_PIN W7 [get_ports {sseg[6]}]
```



C:/Users/voch1/project_4/project_4.srcs/constrs_1/new/c_time_multiplexing.xdc

```
set_property PACKAGE_PIN T2 [get_ports {SW[10]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[10]}]
set_property PACKAGE_PIN R3 [get_ports {SW[11]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[11]}]
set_property PACKAGE_PIN W2 [get_ports {SW[12]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[12]}]
set_property PACKAGE_PIN U1 [get_ports {SW[13]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[13]}]
set_property PACKAGE_PIN T1 [get_ports {SW[14]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[14]}]
set_property PACKAGE_PIN R2 [get_ports {SW[15]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[15]}]

##7 segment display
set_property PACKAGE_PIN W7 [get_ports {sseg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[6]}]
set_property PACKAGE_PIN W6 [get_ports {sseg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[5]}]
set_property PACKAGE_PIN U8 [get_ports {sseg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[4]}]
set_property PACKAGE_PIN V8 [get_ports {sseg[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[3]}]
set_property PACKAGE_PIN U5 [get_ports {sseg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[2]}]
set_property PACKAGE_PIN V5 [get_ports {sseg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[1]}]
set_property PACKAGE_PIN U7 [get_ports {sseg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[0]}]

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

##Buttons
set_property PACKAGE_PIN U18 [get_ports reset]
    set_property IOSTANDARD LVCMOS33 [get_ports reset]
```