**LAB 4 Report**

**Name: Donald Maze-England**

**UT EID: dsm2588**

**Section: 15320**

**Checklist:**

**Part 1 -**

1. Simulation waveform of the Flight Attendant Call System for behavioral as well as dataflow modelling
2. K-map for minimizing the expression for next\_state for dataflow modelling
3. Boolean expression for next\_state for dataflow modelling
4. Completed design file (.v) for dataflow modelling

**Part 2 -**

1. State Diagram of the Rising Edge Detector
2. Completed design files (.v) including the top module and clock divider
3. Test-bench of the system
4. Simulation waveform
5. Constraints File (Just the uncommented portion)

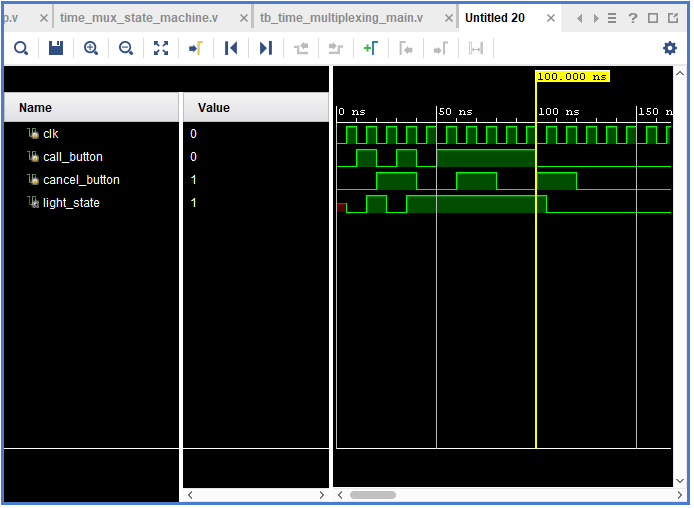
**Part 3 -**

1. Completed design files (.v) of all the modules in the system
2. Test-bench of the system
3. Simulation waveform
4. 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.*

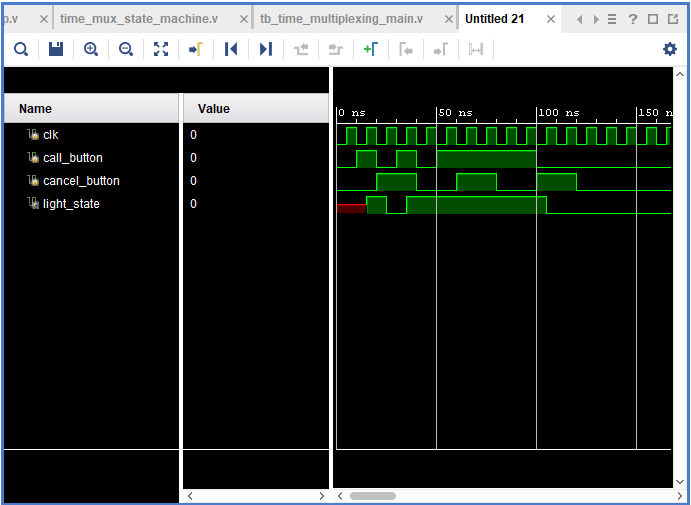
**Part 1:**

**Flight attendant behavioural waveform:**

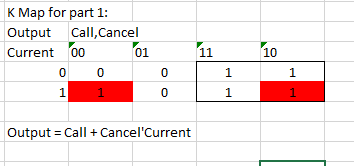


**Test Bench for part 2:**

**Flight Attendant Dataflow waveform:**



**K-Map for Minimizing Expression for Dataflow:**



**Boolean Expression for next\_state for Dateflow Modeling:**

next\_state = call\_button | (light\_state & ~cancel\_button)

**Completed design file (.v) for dataflow modelling:**

`timescale 1ns / 1ps

module flight\_attendent\_call\_system\_dataflow(

input wire clk,

input wire call\_button,

input wire cancel\_button,

output reg light\_state

);

wire next\_state;

assign next\_state = call\_button | (light\_state & ~cancel\_button);

always @(posedge clk) begin

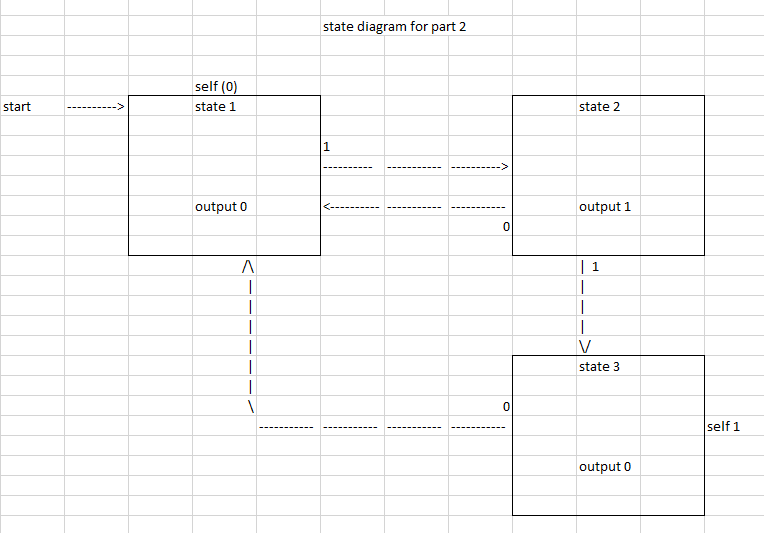
light\_state <= next\_state;

end

endmodule

**Part 2:**

**State Diagram for the Rising Edge Detector:**

****

**Completed Verilog Files:**

**Top Module:**

`timescale 1ns / 1ps

module rising\_edge\_detector(

input clk,

input signal,

input reset,

output reg outedge

);

wire slow\_clk;

reg [1:0] state = 2'b00;

reg [1:0] next\_state;

clkdiv c1(clk, reset, slow\_clk);

//combinational logic

always@(\*) begin

case (state)

2'b00 : begin

outedge = 1'b0;

if(~signal)

next\_state = 2'b00;

else

next\_state = 2'b01;

end

2'b01 : begin

outedge = 1'b1;

if(~signal)

next\_state = 2'b00;

else

next\_state = 2'b10;

end

2'b10 : begin

outedge = 1'b0;

if(~signal)

next\_state = 2'b00;

else

next\_state = 2'b10;

end

default : begin

next\_state = 2'b00;

outedge = 1'b0;

end

endcase

end

//sequential logic

always @(posedge slow\_clk) begin

if (reset)

state <= 2'b00;

else

state <= next\_state;

end

endmodule

**Clock Divider:**

`timescale 1ns / 1ps

module clkdiv(

input clk,

input reset,

output clk\_out

);

reg [1:0] COUNT = 2'b00;

assign clk\_out = COUNT[1];

always @(posedge clk) begin

if (reset)

COUNT = 0;

else

COUNT = COUNT+1;

end

endmodule

**Test Bench for part 2:**

`timescale 1ns / 1ps

module tb\_rising\_edge\_detector;

reg clk;

reg signal;

reg reset;

wire outedge;

wire slow\_clk;

clkdiv c1(clk, reset, slow\_clk);

rising\_edge\_detector ut(

.clk(clk),

.signal(signal),

.reset(reset),

.outedge(outedge)

);

initial begin

clk = 0;

signal = 0;

reset = 0;

#100;

signal = 1;

reset = 0;

#100

signal = 0;

reset = 0;

#100

signal = 1;

reset = 1;

#100

reset = 0;

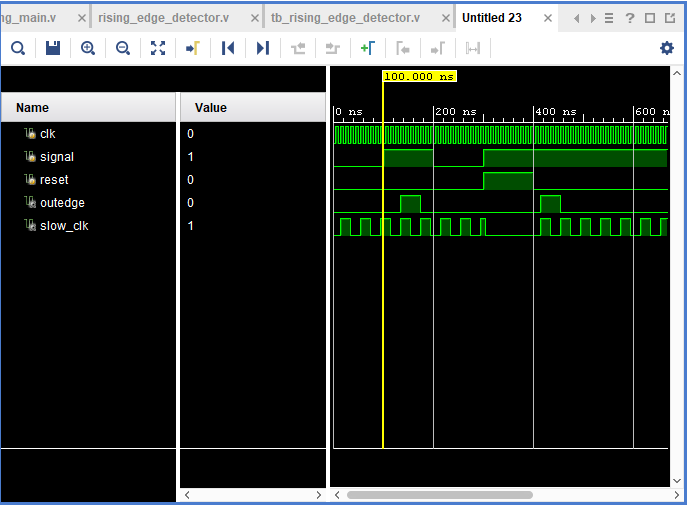
end

always

#5 clk = ~clk;

endmodule

**Simulation Wave form for part 2:**



**Constraints file for Part 2:**

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]

set\_property PACKAGE\_PIN V17 [get\_ports {signal}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {signal}]

set\_property PACKAGE\_PIN U16 [get\_ports {outedge}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {outedge}]

set\_property PACKAGE\_PIN U18 [get\_ports reset]

set\_property IOSTANDARD LVCMOS33 [get\_ports reset]

**Part 3:**

**Verilog Files:**

**Time\_multiplexing\_main:**

`timescale 1ns / 1ps

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 hexTo7Segment decoder

hexTo7Segment c1 (.x(sw[3:0]), .r(in0));

hexTo7Segment c2 (.x(sw[7:4]), .r(in1));

hexTo7Segment c3 (.x(sw[11:8]), .r(in2));

hexTo7Segment c4 (.x(sw[15:12]), .r(in3));

//module instantiation of the clock divider

clk\_div\_disp c5 (.clk(clk), .reset(reset), .clk\_out(slow\_clk));

//module instantiation of the multiplexer

time\_mux\_state\_machine c6(

.clk (slow\_clk),

.reset (reset),

.in0 (in0),

.in1 (in1),

.in2 (in2),

.in3 (in3),

.an (an),

.sseg (sseg)

);

endmodule

**hexTo7Segment:**

`timescale 1ns / 1ps

module hexTo7Segment(

input [3:0] x,

output reg [6:0] r

);

always @(\*)

case (x)

4'b0000 : r = 7'b0000001;

4'b0001 : r = 7'b1001111;

4'b0010 : r = 7'b0010010;

4'b0011 : r = 7'b0000110;

4'b0100 : r = 7'b1001100;

4'b0101 : r = 7'b0100100;

4'b0110 : r = 7'b0100000;

4'b0111 : r = 7'b0001111;

4'b1000 : r = 7'b0000000;

4'b1001 : r = 7'b0001100;

4'b1010 : r = 7'b0001000;

4'b1011 : r = 7'b1100000;

4'b1100 : r = 7'b0110001;

4'b1101 : r = 7'b1000010;

4'b1110 : r = 7'b0110000;

4'b1111 : r = 7'b0111000;

endcase

endmodule

**clk\_div\_disp:**

`timescale 1ns / 1ps

module clk\_div\_disp(

input clk,

input reset,

output clk\_out

);

reg [1:0] COUNT = 2'b00;

assign clk\_out = COUNT[1];

always @(posedge clk) begin

if (reset)

COUNT = 0;

else

COUNT = COUNT+1;

end

endmodule

**time\_mux\_state\_machine:**

`timescale 1ns / 1ps

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 = 2'b00;

reg [1:0] next\_state;

always @(\*) begin

case (state)

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)

2'b00: sseg = in0;

2'b01: sseg = in1;

2'b10: sseg = in2;

2'b11: sseg = in3;

endcase

case(state)

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

**Testbench for part 3:**

`timescale 1ns / 1ps

module tb\_time\_multiplexing\_main;

reg clk;

reg reset;

reg [15:0] sw;

wire [3:0] an;

wire [6:0] sseg;

wire slow\_clk;

time\_multiplexing\_main u7(

.clk(clk),

.reset(reset),

.sw(sw),

.an(an),

.sseg(sseg)

);

//module instantiation of the clock divider

clk\_div\_disp u5 (.clk(clk), .reset(reset), .clk\_out(slow\_clk));

initial begin

clk = 1;

#10

reset = 0;

sw = 16'b0011001000010000;

#160;

sw = 16'b0111011001010100;

#160;

sw = 16'b1011101010011000;

#160;

sw = 16'b1111111011011100;

#160;

reset = 1;

#160;

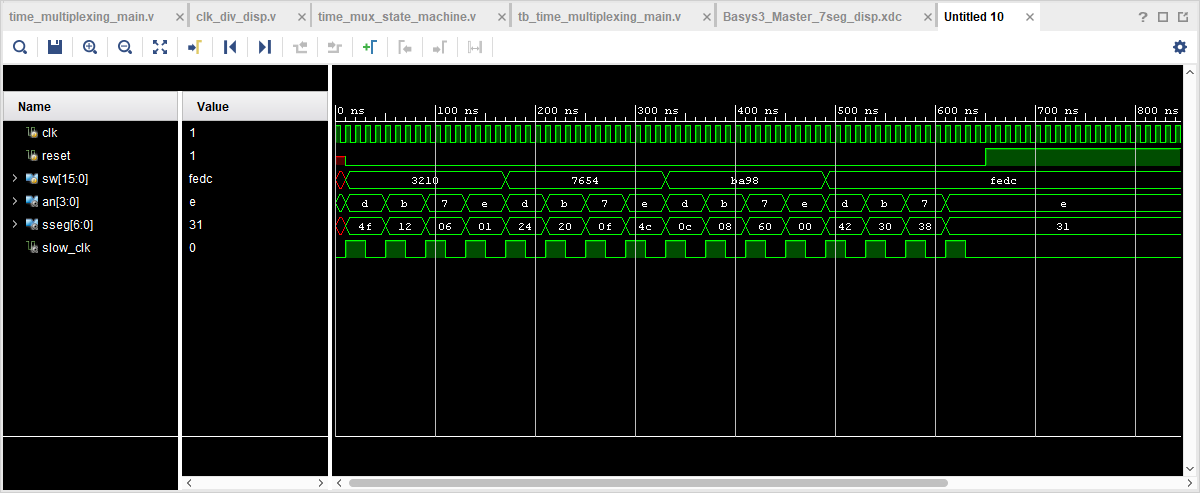
end

always

#5 clk = ~clk;

endmodule

**Simulation Waveform for part 3:**

****

**Constraints File for Part 3:**

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]

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]}]

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]}]

set\_property PACKAGE\_PIN U18 [get\_ports reset]

set\_property IOSTANDARD LVCMOS33 [get\_ports reset]