2021–2022 FALL SEMESTER

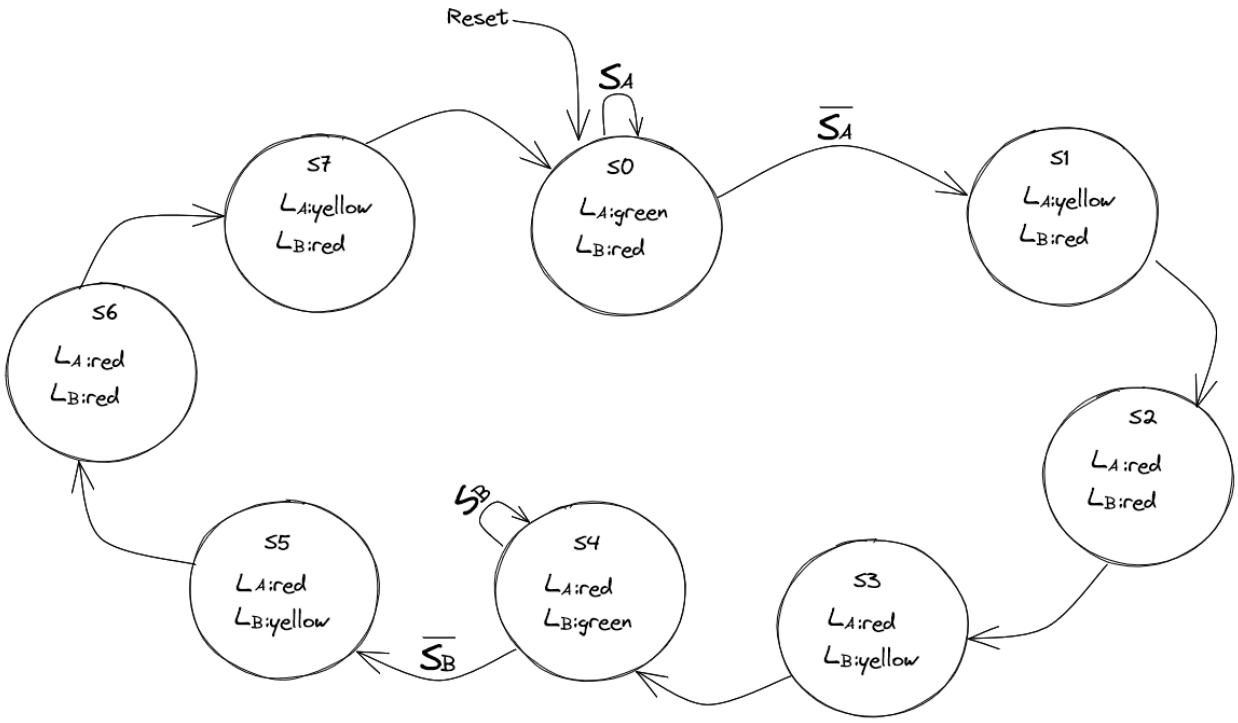
CS223 – LAB 05

Traffic Light System

NAME: ABDULLAH RIAZ
ID : 22001296
COURSE:CS223 – DIGTAL DESGIN
SECTION: 01
DATE: 25/11/2021

a)

Moore machine state transition diagram



State Encodings

State	Encoding $S_{2:0}$
S0	**000**
S1	**001**
S2	**010**
S3	**011**
S4	**100**
S5	**101**
S6	**110**
S7	**111**

State transition table/Next State table

Current State S_2	S_1	S_0	Inputs S_A	S_B	Next State $S`_2$	$S`_1$	$S`_0$
0	0	0	0	X	0	0	1
0	0	0	1	X	0	0	0
0	0	1	X	X	0	1	0
0	1	0	X	X	0	1	1
0	1	1	X	X	1	0	0
1	0	0	X	1	1	0	0
1	0	0	X	0	1	0	1
1	0	1	X	X	1	1	0
1	1	0	X	X	1	1	1
1	1	1	X	X	0	0	0

$$S`_2 = \overline{S_2}\, S_1 S_0 + \overline{S_1\, S_0} S_2 S_B + \overline{S_1\, S_0 S_B} S_2 + \overline{S_1}\, S_2 S_0 + \overline{S_0}\, S_2 S_1$$

$$= \overline{S_2}\, S_1 S_0 + \overline{S_0}\, S_2 + \overline{S_1}\, S_2$$

$$S`_1 = \overline{S_2\, S_1} S_0 + \overline{S_2\, S_0} S_1 + \overline{S_1}\, S_2 S_0 + \overline{S_0}\, S_2 S_1 = S_1 \oplus S_0$$

$$S`_0 = \overline{S_2\, S_1 S_0 S_A} + \overline{S_2\, S_0} S_1 + \overline{S_1\, S_0 S_B} S_2 + \overline{S_0}\, S_2 S_1 = \overline{S_0 S_B} S_2 + \overline{S_2\, S_0 S_A} + \overline{S_0}\, S_1$$

Output Encoding

State	Encoding $L_{1:0}$
Green	00
Yellow	01
Red	10

Output Table

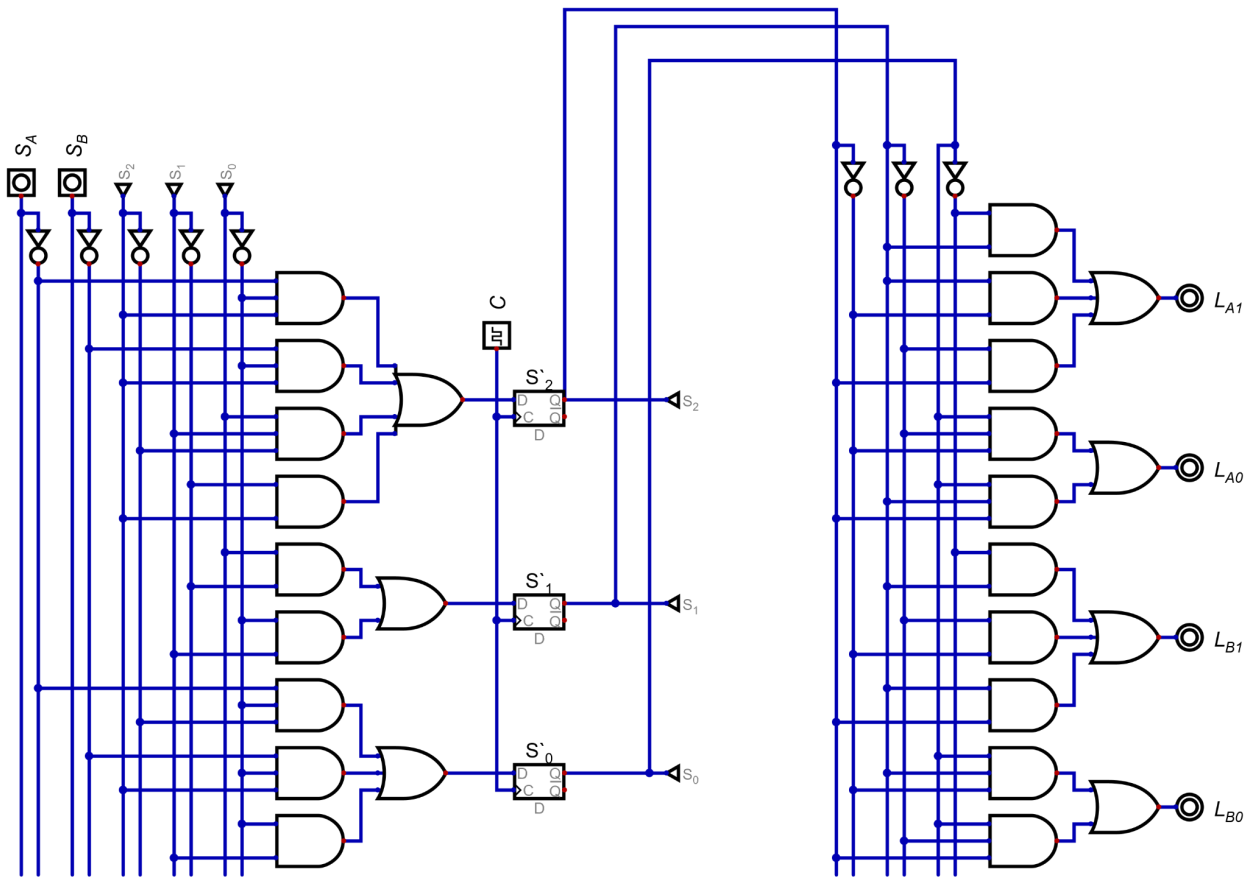
Current State			Output			
S_2	S_1	S_0	L_{A1}	L_{A0}	L_{B1}	L_{B0}
0	0	0	0	0	1	0
0	0	1	0	1	1	0
0	1	0	1	0	1	0
0	1	1	1	0	0	1
1	0	0	1	0	0	0
1	0	1	1	0	0	1
1	1	0	1	0	1	0
1	1	1	0	1	1	0

$$L_{A1} = \overline{S_2}\,\overline{S_0}S_1 + \overline{S_2}\,S_1S_0 + \overline{S_1}\,\overline{S_0}S_2 + \overline{S_1}\,S_2S_0 + \overline{S_0}\,S_2S_1 = S_2 \oplus S_1 + \overline{S_0}\,S_2$$

$$L_{A0} = \overline{S_2}\,\overline{S_1}S_0 + S_2\,S_1S_0 = (\overline{S_2 \oplus S_1}) * S_0$$

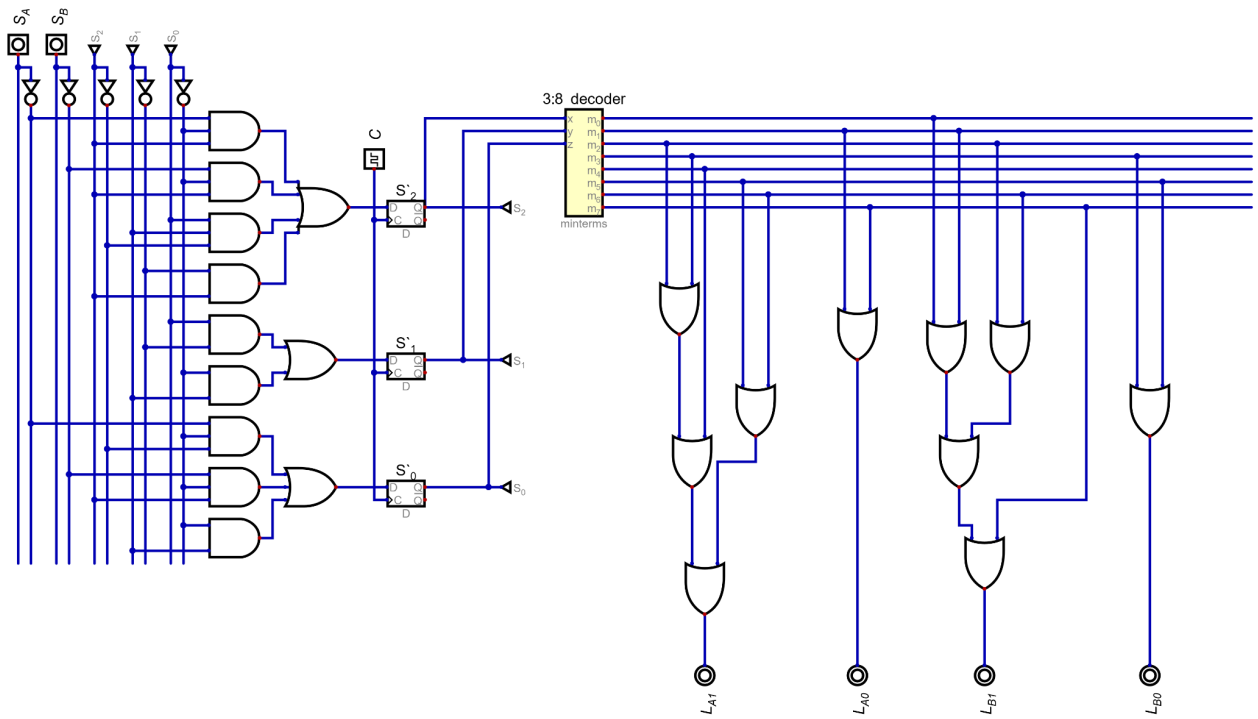
$$L_{B1} = \overline{S_2}\,\overline{S_1S_0} + \overline{S_2}\,S_1S_0 + \overline{S_2}\,\overline{S_0}S_1 + \overline{S_0}\,S_2S_1 + S_2\,S_1S_0 = \overline{S_2 \oplus S_1} + \overline{S_0}\,S_1$$

$$L_{B0} = \overline{S_2}\,S_1S_0 + \overline{S_1}\,S_2S_0 = (S_2 \oplus S_1) * S_0$$



b) 3 flip-flops

c) Redesign your outputs using decoders.



Verilog Code

```
//Clock Divider
 module div_clock(input logic clk_in,
        output clk_out
        );

 logic temp;
 logic [27:0] count = 0;

 always@ (posedge clk_in)
 begin
     count <= count + 1;
     if(count==150000000)
     begin
     count<=0;
     temp = ~temp;
        end
 end

 assign clk_out = temp;
 endmodule
```

```
//Main Module
module traffic_system(input logic clk,
                      input logic reset,
                      input logic sa,
                      input logic sb,
                      output logic [2:0]ta,
                      output logic [2:0]tb);

       logic clk_out;

       typedef enum logic [2:0] {S0, S1, S2, S3, S4, S5, S6, S7} statetype;
       statetype state, nextstate;

       div_clock div(clk,clk_out);
       // state register
       always_ff @(posedge clk_out, posedge reset)
       if (reset) state <= S0;
       else state <= nextstate;

      //Next State
       always_comb
       case (state)
       S0: if (~sa) nextstate = S1;
             else nextstate = S0;
       S1: nextstate <= S2;
       S2: nextstate <= S3;
       S3: nextstate <= S4;
       S4: if (sb) nextstate = S4;
             else nextstate = S5;
       S5: nextstate <= S6;
       S6: nextstate <= S7;
       S7: nextstate <= S0;
       default: nextstate <= S0;
       endcase

      //Output Logic
       always_comb
       case (state)
             S0: begin ta = 3'b011; tb = 3'b111; end
             S1: begin ta = 3'b001; tb = 3'b111; end
             S2: begin ta = 3'b111; tb = 3'b111; end
             S3: begin ta = 3'b111; tb = 3'b001; end
             S4: begin ta = 3'b111; tb = 3'b011; end
             S5: begin ta = 3'b111; tb = 3'b001; end
             S6: begin ta = 3'b111; tb = 3'b111; end
             S7: begin ta = 3'b001; tb = 3'b111; end
       endcase
endmodule
```

Testbench

```
module traffic_tb();
		logic clk;
		logic reset;
		logic sa;
		logic sb;
		logic [2:0]ta;
		logic [2:0]tb;

	traffic_system dut(clk,reset,sa,sb,ta[2:0],tb[2:0]);

	initial
	begin
	reset = 1;
	#13
	reset = 0;
	end

	always #5 clk = ~clk;
	initial
		begin
		clk = 1;
		sa  = 0; sb = 0; #40;
		sa  = 0; sb = 1; #40;
		sa  = 1; sb = 0; #40;
		sa  = 1; sb = 1; #40;
		sa  = 0; sb = 0; #40;
	end

endmodule
```