CS 223 Digital Design

Bilkent University

Spring 2021/2022

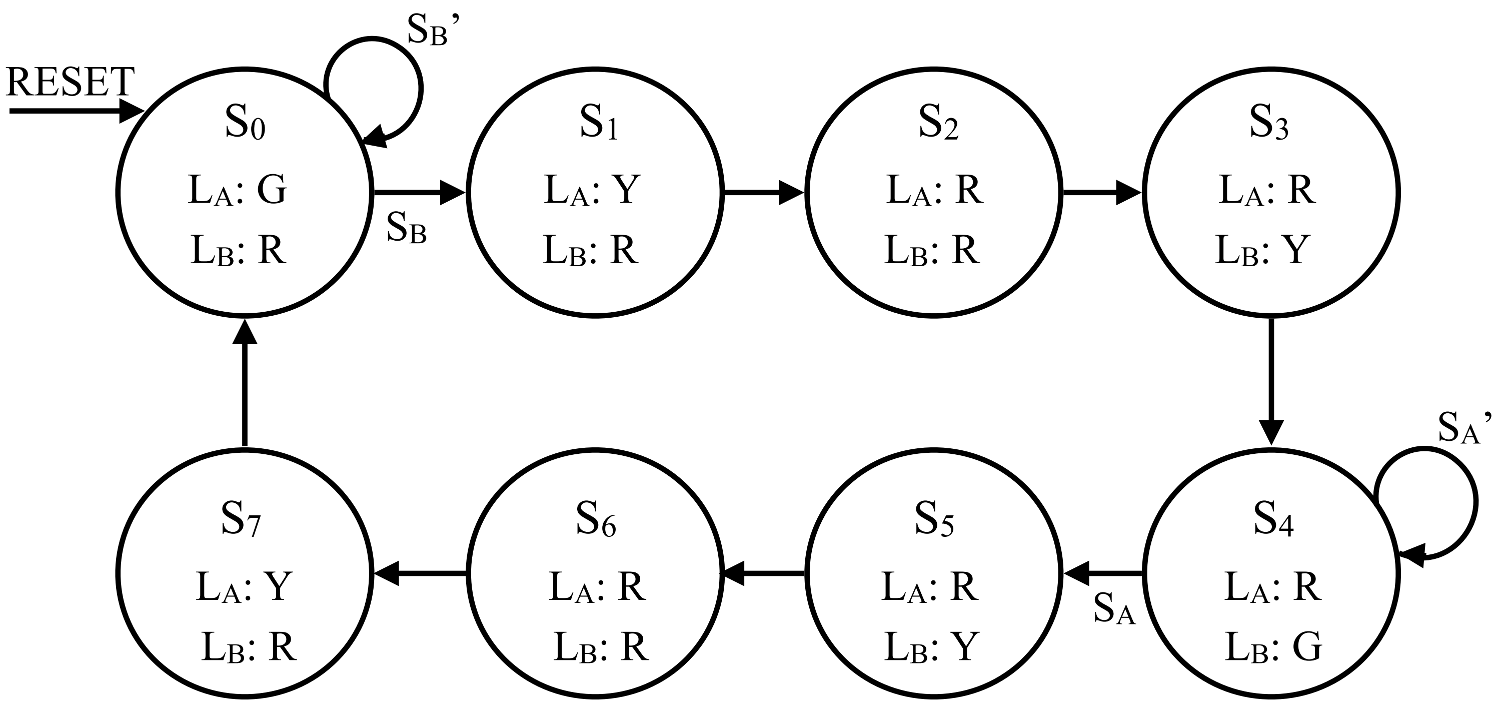
Laboratory Assignment 4
Preliminary Report

Section 2

Deniz Tuna Onguner
22001788

Mon. April 4th, 2022

State Transition Diagram



State Encodings

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

State Transition Table

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

Output Encoding

Output	Encoding $L_{2:0}$
Green	011
Yellow	001
Red	111

Output Table

Current State			Outputs					
S_2	**S_1**	**S_0**	**L_{A2}**	**L_{A1}**	**L_{A0}**	**L_{B2}**	**L_{B1}**	**L_{B0}**
0	0	0	0	1	1	1	1	1
0	0	1	0	0	1	1	1	1
0	1	0	1	1	1	1	1	1
0	1	1	1	1	1	0	0	1
1	0	0	1	1	1	0	1	1
1	0	1	1	1	1	0	0	1
1	1	0	1	1	1	1	1	1
1	1	1	0	0	1	1	1	1

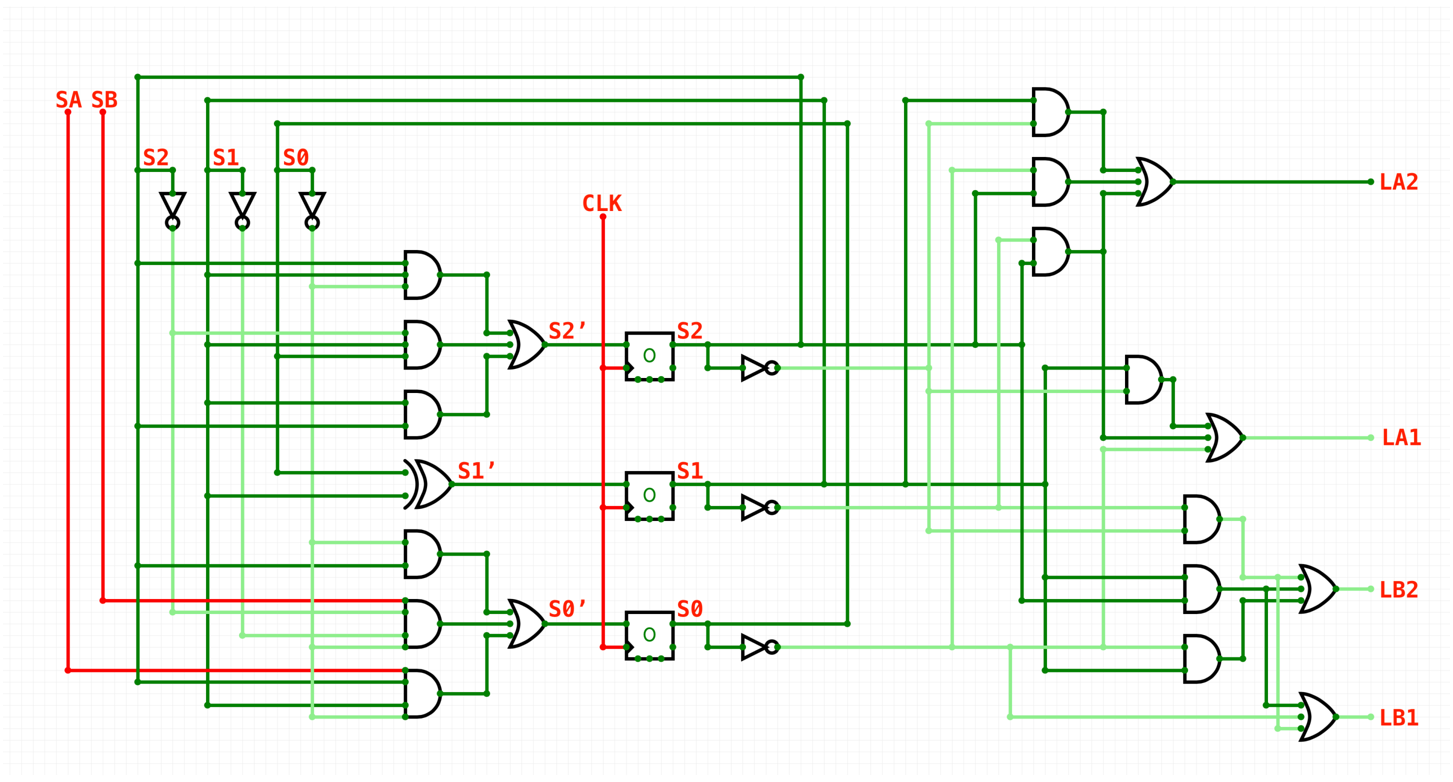
Output Equations

$S'_2 = S_2S_1S_0' + S_2'S_1S_0 + S_2S_1'$
$S'_1 = S_1 \oplus S_0$
$S'_0 = S_1S_0' + S_BS_2'S_1'S_0' + S_AS_2S_1'S_0'$

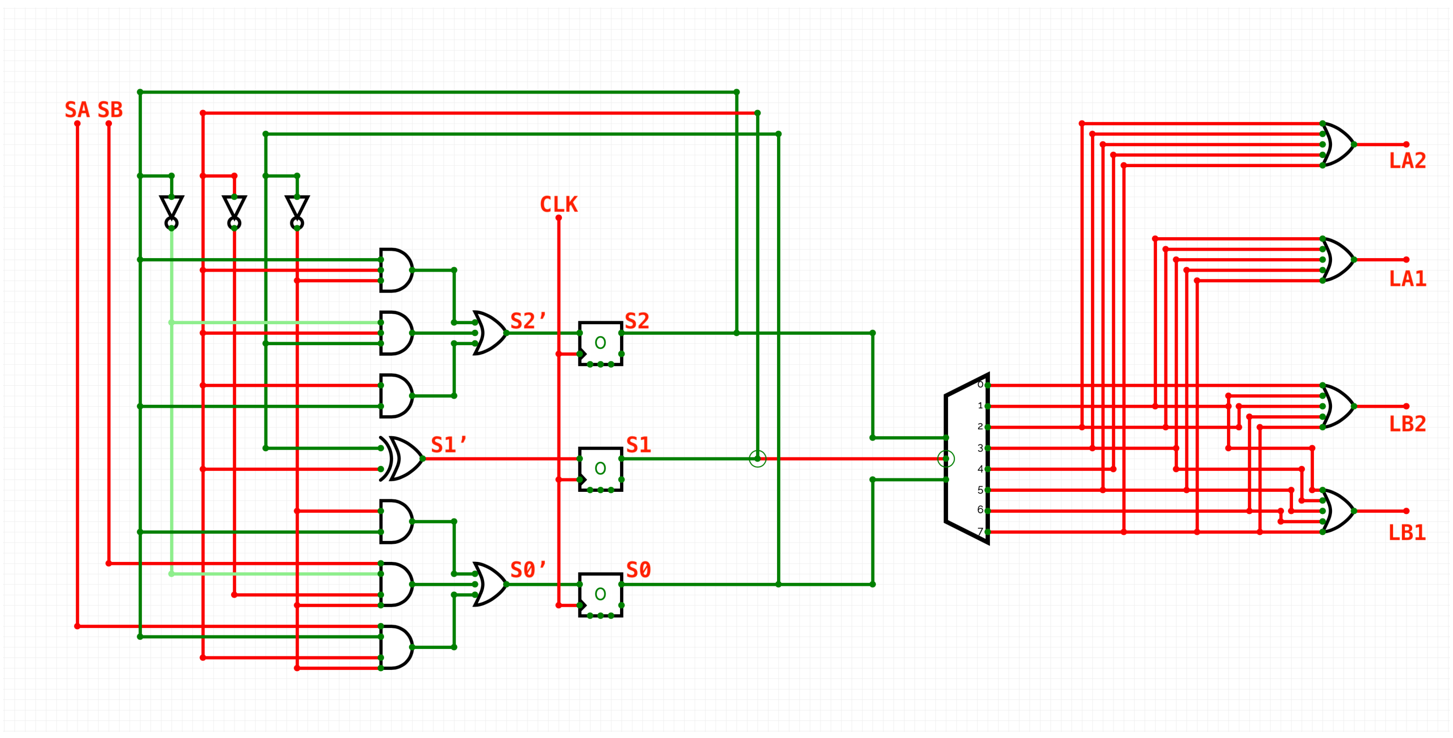
$L_{A2} = S_1S_2' + S_0'S_2 + S_1'S_2$
$L_{A1} = S_0' + S_1S_2' + S_1'S_2$
$L_{A0} = 1$

$L_{B2} = S_1'S_2' + S_0'S_1 + S_1S_2$
$L_{B1} = S_0' + S_1S_2 + S_1'S_2'$
$L_{B0} = 1$

Circuit Diagrams



With Decoder



Only 3 flip-flops are enough for the implementation.

System Verilog Module

```
module TrafficLights(input logic clk, reset, sa, sb,
                     output logic [2:0] la, lb);

    typedef enum logic [2:0] {s0, s1, s2, s3, s4, s5, s6, s7} Statetype;
    Statetype [1:0] state, nextstate;

    parameter g = 3'b011;
    parameter y = 3'b001;
    parameter r = 3'b111;

    always_ff @(posedge clk, posedge reset)
        if(reset) state <= s0;
        else state <= nextstate;

    always_comb
    case(state)
        s0: if(sb) nextstate = s1;
            else   nextstate = s0;
        s1:        nextstate = s2;
        s2:        nextstate = s3;
        s3:        nextstate = s4;
        s4: if(sa) nextstate = s5;
            else   nextstate = s4;
        s5:        nextstate = s6;
        s6:        nextstate = s7;
        s7:        nextstate = s0;
        default:   nextstate = s0;
    endcase

    always_comb
    case(state)
        s0:      {la, lb} = {g, r};
        s1:      {la, lb} = {y, r};
        s2:      {la, lb} = {r, r};
        s3:      {la, lb} = {r, y};
        s4:      {la, lb} = {r, g};
        s5:      {la, lb} = {r, y};
        s6:      {la, lb} = {r, r};
        s7:      {la, lb} = {y, r};
        default: {la, lb} = {g, r};
    endcase
endmodule
```

Clock Module for Timing

```
module Clock(input clk, output sclk);
    logic [31:0] timer = 0;
    logic out;

    always @(posedge clk)
        begin
        timer <= timer + 1;
        if (timer == 150000000)
            begin
                timer <= 0;
                out = ~out;
            end
        end
    assign sclk = out;
endmodule
```

Testbench

```
module TestBench();
    logic clk, reset, sa, sb;
    logic [2:0] la, lb;
    TrafficLights dut(clk, reset, sa, sb, la, lb);
    initial begin
        reset = 0;
            sa = 0; sb = 1; #10;
            sa = 1; sb = 0; #10;
            sa = 0; sb = 0; #10;
            sa = 1; sb = 1; #10;
    end

    always
        begin
            clk <= 1; #10;
            clk <= 0; #10;
        end
endmodule
```