1. 실습 목표

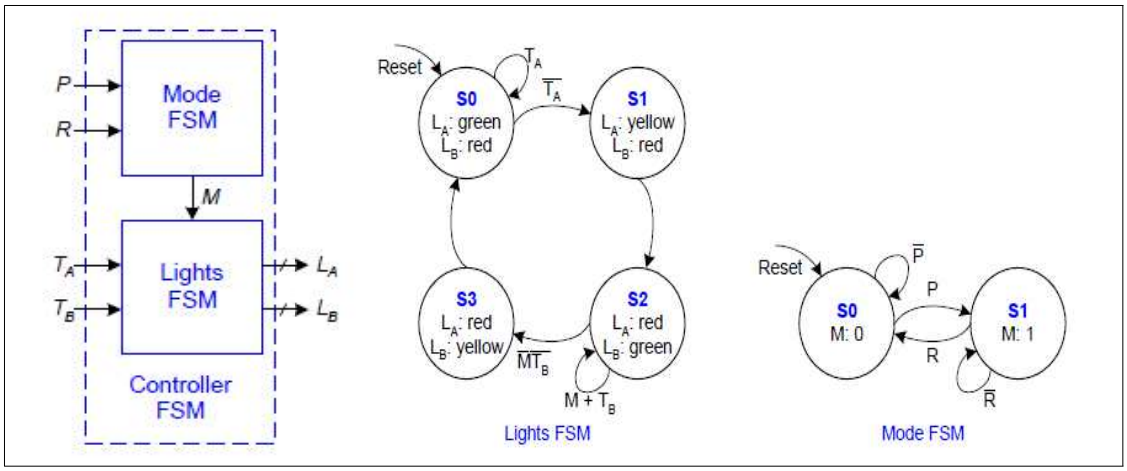
- 사람이 있으면 켜지고 없으면 꺼지는 사거리 신호등에다가 퍼레이드가 열릴 때 초록불이 유지 되는 신호등을 설계한다.

2. Specification

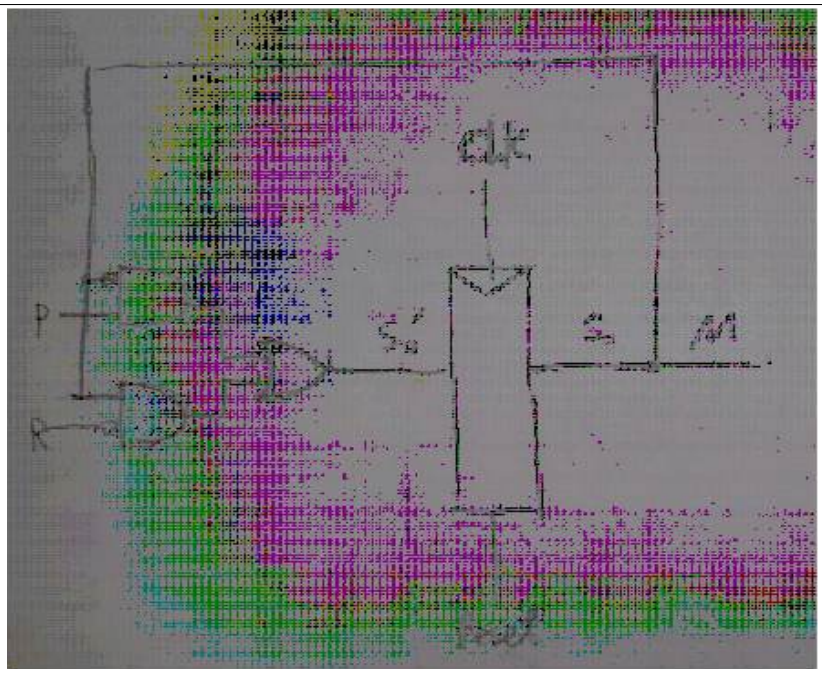
1) I/O port

- Input port : 6개 입력 (T_A, T_B, P, R, clock, reset)
- Output port : 3bit 2개 출력 (L_A, L_B)

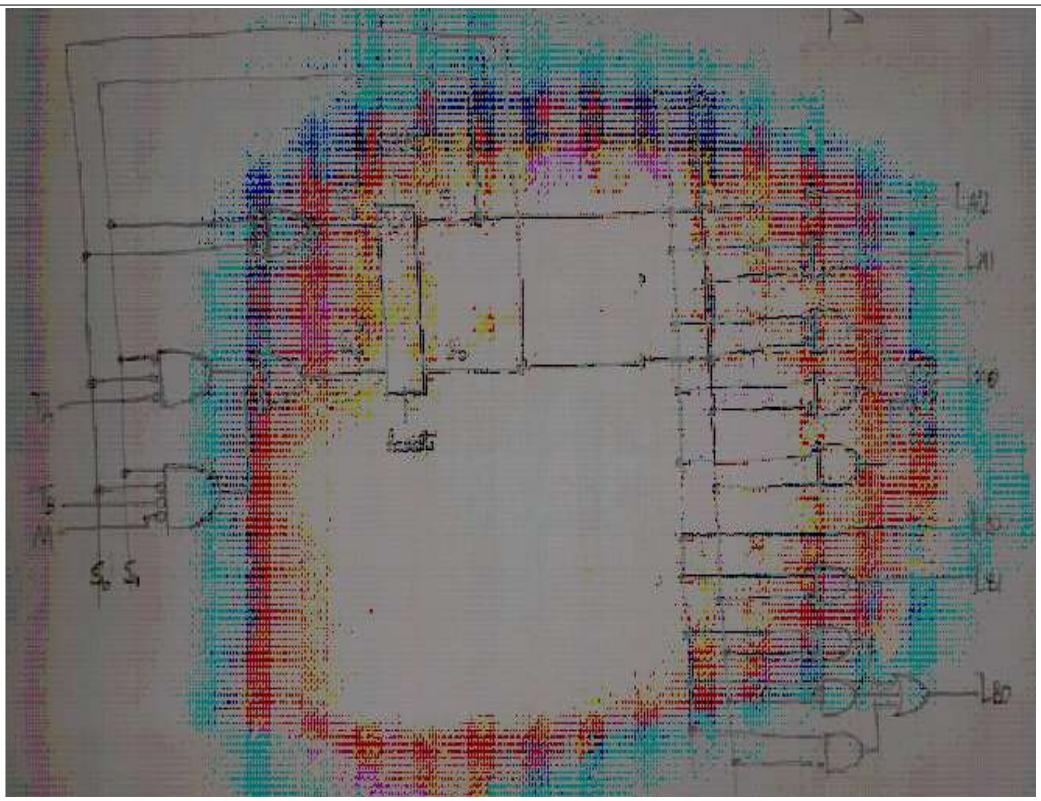
2) 예상 RTL 블록도



3) 예상 내부 회로도



Mode FSM



Light FSM

4) 예상 진리표

state	S_0
S0	0
S1	1

current state	Input		next state
S_0	P	R	S_0'
0	0	x	0
0	1	x	1
1	x	0	1
1	x	1	0

current state	output
S_0	M
0	0
1	1

$$S_0' = \overline{S_0}P + S_0\overline{R}$$

$$M = S_0$$

state	S_1S_0
S0	00
S1	01
S2	10
S3	11

light	$L_2L_1L_0$
green	111
yellow	100
red	001

current state		input			next state	
S_1	S_0	M	T_A	T_B	S_1'	S_0'
0	0	x	0	x	0	1
0	0	x	1	x	0	0
0	1	x	x	x	1	0
1	0	0	x	0	1	1
1	0	0	x	1	1	0
1	0	1	x	0	1	0
1	0	1	x	1	1	0
1	1	x	x	x	0	0

current state		output	
S_1	S_0	$L_{A2}L_{A1}L_{A0}$	$L_{B2}L_{B1}L_{B0}$
0	0	111	001
0	1	100	001
1	0	001	111
1	1	001	100

$$S_1' = S_0 \oplus S_1$$

$$S_0' = \overline{S_1S_0T_A} + S_1\overline{S_0MT_B}$$

$$L_{A2} = \overline{S_1},\ L_{B2} = S_1$$

$$L_{A1} = \overline{S_1S_0},\ L_{B1} = S_1\overline{S_0}$$

$$L_{A0} = \overline{S_1S_0} + S_1\overline{S_0} + S_1S_0,\ L_{B0} = \overline{S_1S_0} + \overline{S_1}S_0 + S_1\overline{S_0}$$

3. 실험내용

```
module parade_mode(rst, clk, P, R, S_mode, M); //Mode FSM의 module
input wire rst,clk;        // clock, reset
input wire P,R;            // 입력 P, R
output wire S_mode,M; // 상태 출력 S_mode와 출력 M
reg next_s;

always@(posedge clk or posedge rst)  // clock이나 reset이 상승edge일 때 작동
begin

if(rst)
 next_s <= 1'b0;  // reset이 입력되면 상태는 S0으로 초기화

else if(P==0 && next_s==1'b0)
 next_s <= 1'b0;
// 현재 상태가 S0이고 R의 값과 상관없이 P가 0이면 다음상태는 S0으로 유지

else if(P==1 && next_s==1'b0)
 next_s <= 1'b1;
// 현재 상태가 S0이고 R의 값과 상관없이 P가 1이면 다음상태는 S1으로 변환

else if(R==0 && next_s==1'b1)
 next_s <= 1'b1;
// 현재 상태가 S1이고 P의 값과 상관없이 R가 0이면 다음상태는 S1으로 유지

else if(R==1 && next_s==1'b1)
 next_s <= 1'b0;
// 현재 상태가 S1이고 P의 값과 상관없이 R가 1이면 다음상태는 S0으로 변환

end

assign S_mode = next_s;    // 조건문안의 상태함수 next_s는 S_mode이다.
assign M = S_mode;         // S_mode는 출력 M과 같다.

endmodule
```

```
module traffic_light(rst, clk, Ta, Tb, M, S_light, La, Lb); //Light FSM의 module
input wire Ta,Tb,M;        // 입력 T_A, T_B, M
input wire rst,clk;        // reset, clock
output wire [2:0]La,Lb;    // 신호등 출력 L_A, L_B
output wire [1:0]S_light;  // 상태 출력 S_light
reg [2:0]la,lb;
reg [1:0]next_s;

always@(posedge clk or posedge rst) // clock이나 reset이 상승edge일 때 작동
begin

if(rst)                 // reset이 입력되면
begin
 next_s = 2'b00;    // 상태는 S0로 초기화
 la= 3'b111;        // L_A는 green
 lb= 3'b001;        // L_B는 red
end

else if(Ta==1'b1 && next_s==2'b00) // 현재상태가 S0일 때 T_A가 1이면
begin
 next_s <= 2'b00;    // 다음상태는 S0으로 유지
 la= 3'b111;         // L_A는 green
 lb= 3'b001;         // L_B는 red
end

else if(Ta==1'b0 && next_s==2'b00) // 현재상태가 S0일 때 T_A가 0이면
begin
 next_s <= 2'b01;    // 다음상태는 S1으로 변환
 la= 3'b100;         // L_A는 yellow
 lb= 3'b001;         // L_B는 red
end

else if(next_s==2'b01) // 현재상태가 S1이면
begin
 next_s <= 2'b10;  // 다음상태는 S2으로 변환
 la= 3'b001;        // L_A는 red
 lb= 3'b111;        // L_B는 green
end
```

```
else if((M|Tb==1) && next_s==2'b10) // 현재상태가 S2일 때 (M+T_B)가 1이면
 begin
  next_s <= 2'b10;    // 다음상태는 S2으로 유지
  la= 3'b001;         // L_A는 red
  lb= 3'b111;         // L_B는 green
 end

else if((M|Tb==0) && next_s==2'b10) // 현재상태가 S2일 때 (M+T_B)가 0이면
 begin
  next_s <= 2'b11;    // 다음상태는 S3으로 변환
  la= 3'b001;         // L_A는 red
  lb= 3'b100;         // L_B는 yellow
 end

else if(next_s==2'b11)  // 현재상태가 S3이면
begin
 next_s <= 2'b00;    // 다음상태는 S0으로 변환
 la= 3'b111;         // L_A는 green
 lb= 3'b001;         // L_B는 red
end
end

assign S_light = next_s;  // 조건문안의 상태함수 next_s는 상태출력 S_light
assign La = la;           // 조건문안의 la는 출력 L_A
assign Lb = lb;           // 조건문안의 lb는 출력 L_B

endmodule
```

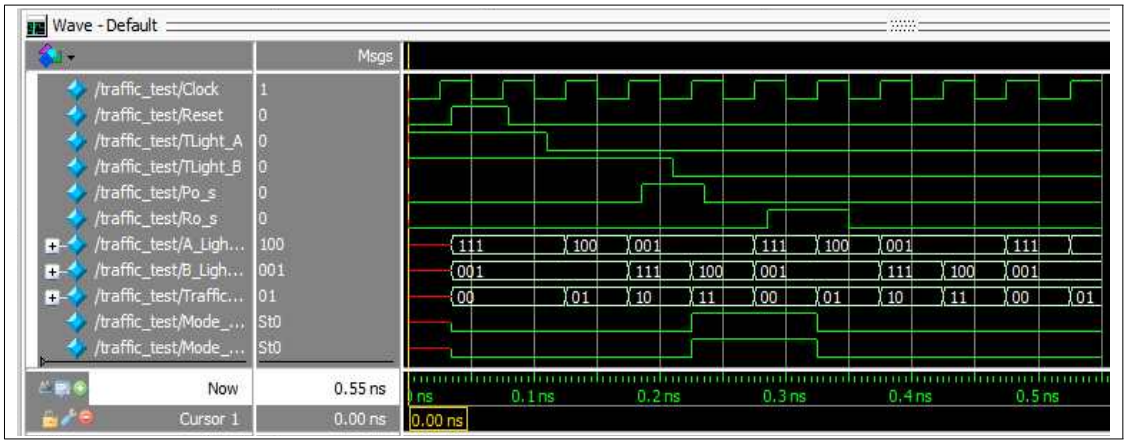
```
module parade_traffic_light
(clk, rst, Ta, Tb, P, R, La, Lb, States_light_out, State_mode_out, M);
// Mode FSM와 Light FSM를 연결시키는 top module

input clk,rst;    // clock, reset
input P,R,Ta,Tb;    // 입력 P, R, T_A, T_B
output [2:0]La,Lb;    // 신호등 출력 L_A, L_B
output [1:0]States_light_out;  // Light FSM의 상태 출력
output State_mode_out,M;    // Mode FSM의 상태 출력과 출력 M

wire m; //Mode FSM의 출력 M을 Light FSM의 입력으로 넣어주기 위한 선언

assign M=m; // m은 출력 M이다

parade_mode    parade(rst, clk, P, R, State_mode_out, m);
traffic_light light (rst, clk, Ta, Tb, m, States_light_out, La, Lb);
// Mode FSM과 Light FSM을 연동

endmodule
```

Testbench 코드

```
module traffic_test();
reg Clock, Reset;  // Input Data to Top Module
reg TLight_A, TLight_B, Po_s, Ro_s;
wire [2:0] A_Light_out, B_Light_out;  //Output Data from Top Module
wire [1:0] Traffic_state;
wire Mode_state;
wire Mode_out;

initial begin         //Input data Initialization.
Clock = 1; Reset = 0; TLight_A = 1; TLight_B = 1; Po_s =0; Ro_s = 0;
#35 Reset = 1;     //Reset Clock Input
#45 Reset = 0;
#30 TLight_A = 0; // Light FSM S0 -> S1 -> S2
#75 Po_s = 1;      // Mode S0 -> S1 -> S0
 #25 TLight_B =0; // Light FSM S2 -> S2 -> S3 -> S0
#25 Po_s = 0;
#50 Ro_s =1;
#65 Ro_s =0;
#200 $stop;
end

always begin       //clock controls
Clock = ~Clock;
#25;
end

//Wiring between Top Module and Signal.
parade_traffic_light
t(.clk(Clock), .rst(Reset), .Ta(TLight_A), .Tb(TLight_B), .P(Po_s),
.R(Ro_s), .La(A_Light_out), .Lb(B_Light_out), .States_light_out(Traffic_state),
.State_mode_out(Mode_state), .M(Mode_out));

endmodule
```

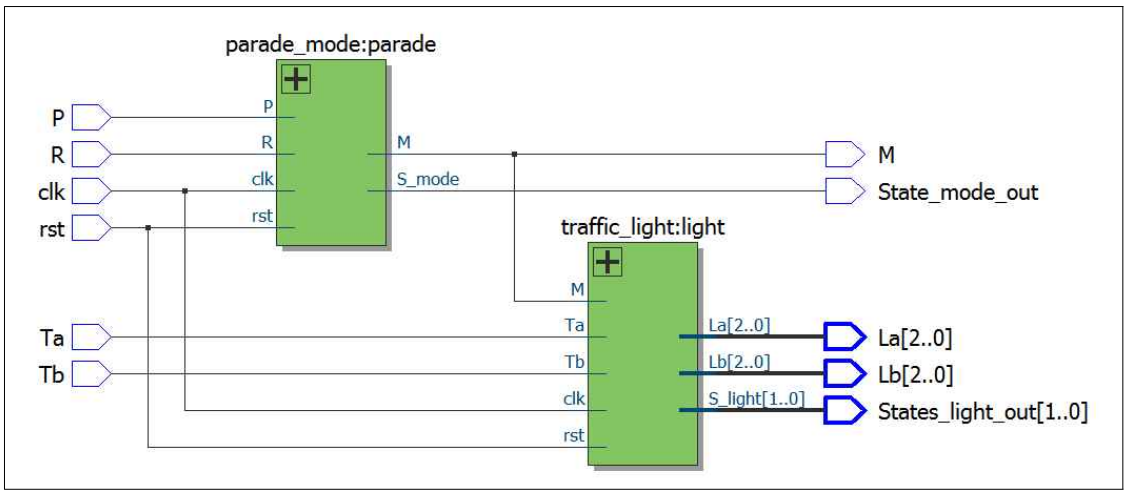
4. 실험결과

1) modelsim simulation 결과 화면



- 처음에 reset으로 Mode FSM과 Light FSM을 S0으로 초기화 시킨다. 그 다음 0.225ns 까지 Light FSM의 상태가 S0 -> S1 -> S2이 순차적으로 이루어지도록 하였다. 각각의 상태에 맞게 신호등색이 변하는 것을 알 수 있다. 그 다음 Mode FSM의 상태가 S0 -> S1 -> S0 으로 변화하도록 선언하고 Light FSM의 상태가 S2 -> S3 -> S0이 되도록 입력 값을 주고 신호등의 출력을 확인한다.

2) RTL 회로 결과



5. 결과

1) 실험 내용 요약

- 이전 traffic light 과제에서 상태 encoding을 one hot으로 설정해줘서 다시 binary encoding으로 바꾸어 주었고, Mode FSM을 따로 만들어서 parade가 지나갈 때와 안지나갈 때를 구분하였다. 그리고 이전 traffic light에서 입력M을 더 넣어 줌으로써 parade가 지나갈 때와 안지나갈 때에 따라서 신호등의 상태가 바뀌도록 설정해준 다음 Mode FSM과 Light FSM을 연결하기위한 top module을 만들어서 Mode FSM의 출력 M이 Light FSM의 입력으로 입력되도록 설정해 주었다.

2) 실험 결과 요약

- 조교님이 올려준 testbench 코드로 RTL simulation을 하면 결과 값이 위와 같이 나오게 된다. 조교님이 올려준 testbench의 결과와는 조금 다르게 나오게 되는데, 0.225ns시점에서 상태가 S2로 유지가 안 되는 것을 볼 수 가 있다. 조교님이 잘못된 결과화면을 올려주셨기 때문에 다르다고 생각이 든다. 이유는 조교님의 결과 화면에는 Mode_state와 Mode_out의 값이 다르기 때문이다. 조교님과 같은 결과를 도출하려면 위의 코드에서 (M|Tb==1) 대신에 (M|Tb==0)을, (M|Tb==0)대신 (M|Tb==1)을 써주면 된다. 하지만 이와 같이 코드를 작성하면 교제에 있는 상태도에서 S2에서 S3로 넘어가는 조건이 반대도 되어버리기 때문에 틀린 것을 알 수 가있다. 따라서 조교님이 올려주신 결과 화면과는 다르지만 위의 결과 화면으로 보고서를 작성하였다.