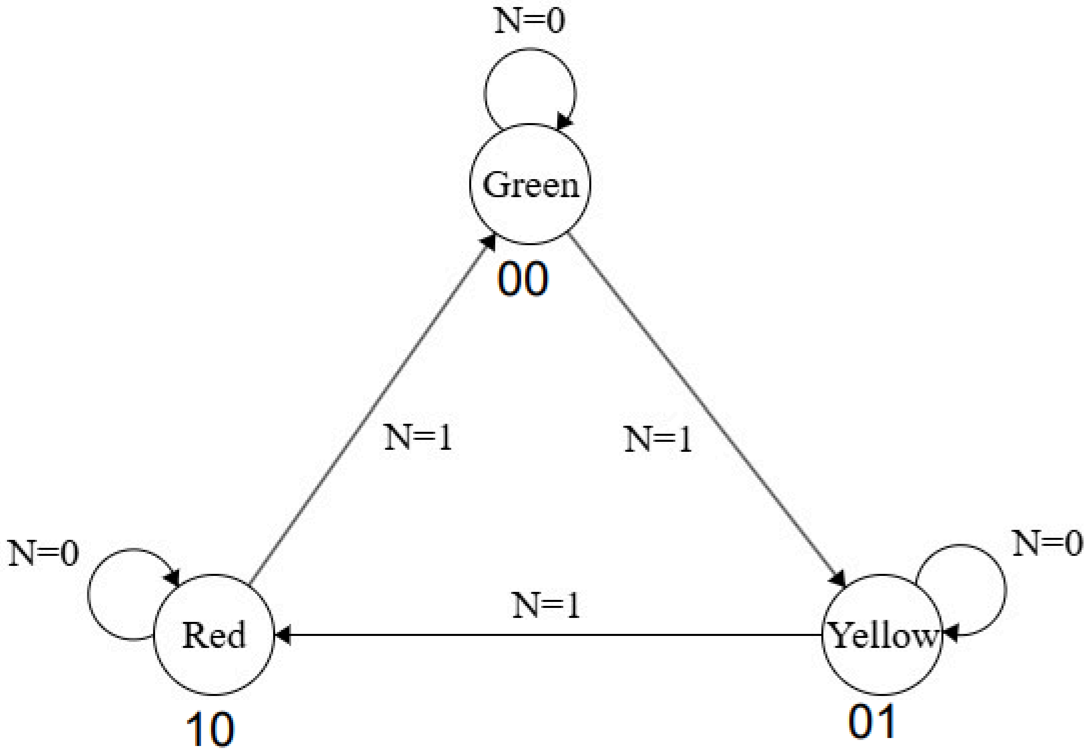
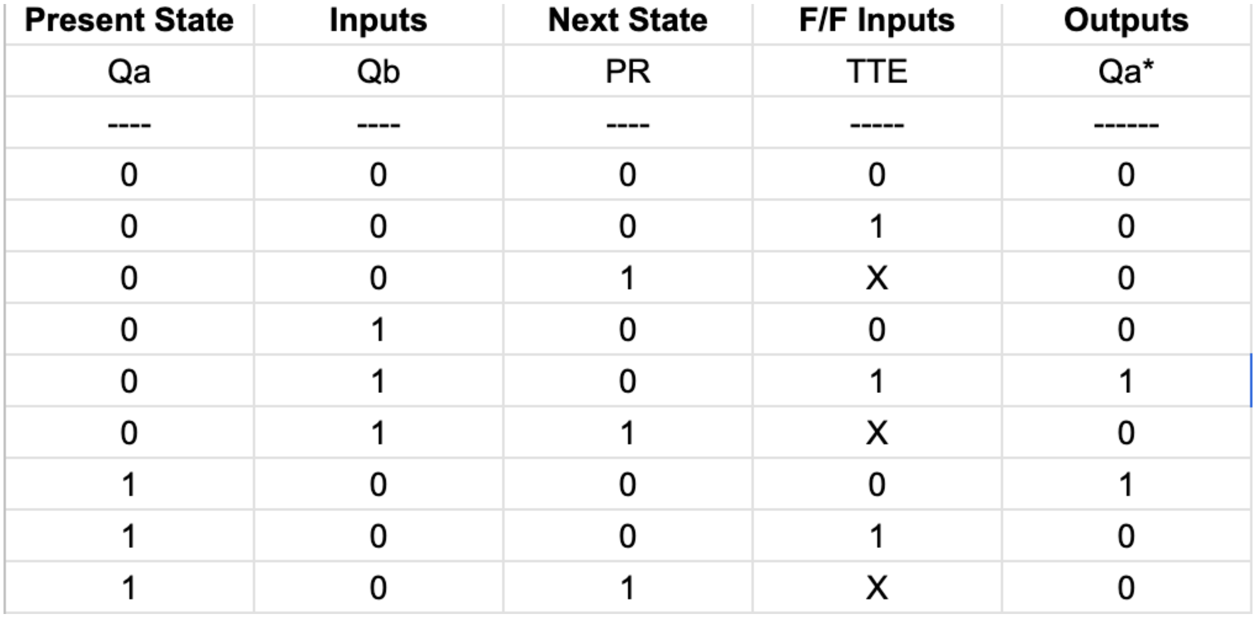
This project aimed to build a traffic light using a finite state machine that cycles through red, green, and yellow lights in the correct order. I used D flip-flops to store state, and logic gates to control the state transitions and LED outputs. A preemption input was added to simulate emergency vehicle priority, letting the system skip directly to green when needed.

I built and tested the circuit in Multisim, wrote Boolean equations for the outputs and next-state logic, and debugged timing issues to get the light sequence working properly. Through this project, I learned how to design a working state machine from a state diagram and transition table, how to control outputs based on internal state, and how to troubleshoot digital circuits when the logic doesn’t behave as expected.





Present State	Inputs	Next State	F/F Inputs	Outputs
Qa	Qb	PR	TTE	Qa*
----	----	----	-----	------
0	0	0	0	0
0	0	0	1	0
0	0	1	X	0
0	1	0	0	0
0	1	0	1	1
0	1	1	X	0
1	0	0	0	1
1	0	0	1	0
1	0	1	X	0

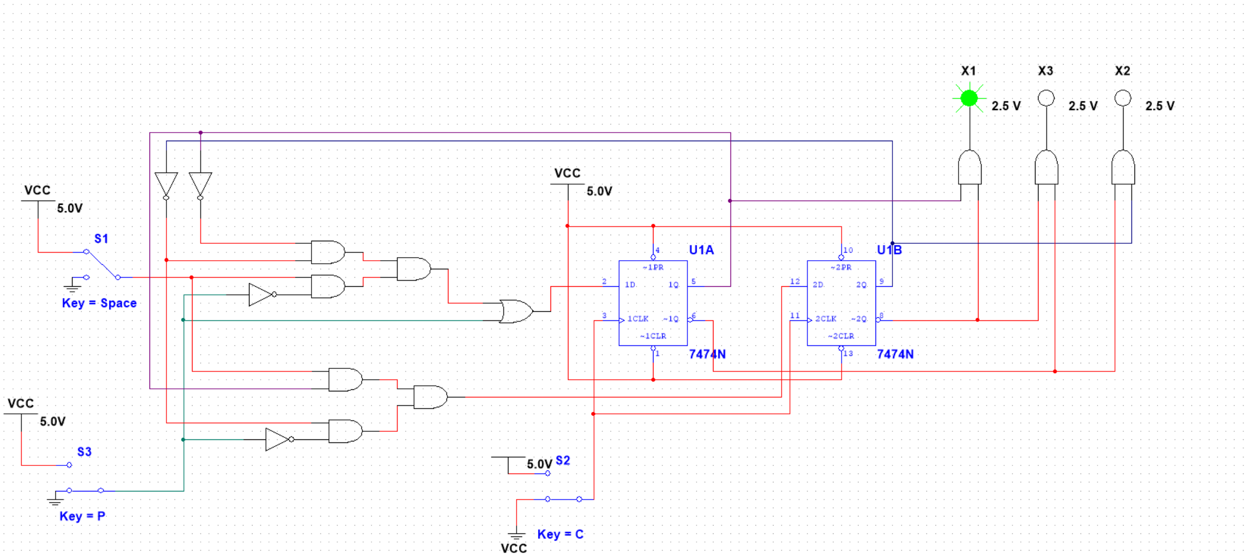
L2 = ~Qa & Qb

L1 = ~Qa & ~Qb

L0 = Qa & ~Qb

Da = Qb·TTE·~Qa + Qa·~Qb·~TTE

Db = PR + (~Qa & ~Qb & TTE & ~PR) + (Qa & ~Qb & TTE & ~PR)


X1
X3
X2
2.5 V
2.5 V
2.5 V
VCC
5.0V
VCC
5.0V
S1
Key = Space
U1A
U1B
4
10
~1PR
~2PR
2
1D
1Q
5
12
2D
2Q
9
3
1CLK
~1Q
6
11
2CLK
~2Q
8
~1CLR
~2CLR
1
13
7474N
7474N
VCC
5.0V
S3
Key = P
5.0V
S2
Key = C
VCC