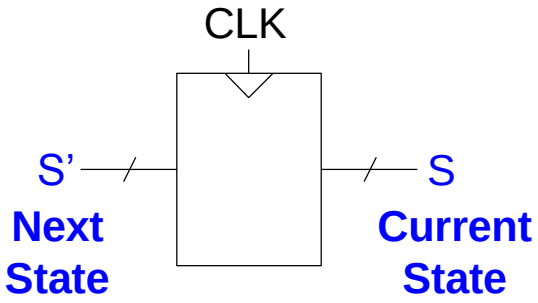
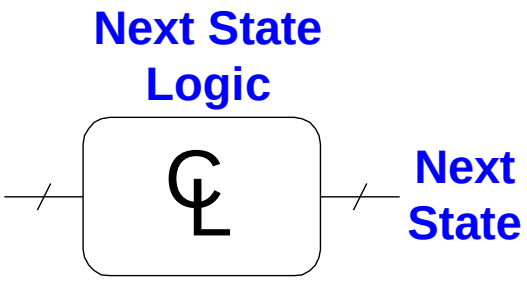
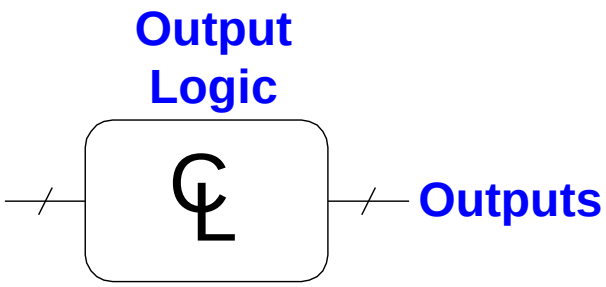
Finite State Machine (FSM)

- Consists of:
  - **State register**
    - Stores current state
    - Loads next state at clock edge
  - **Combinational logic**
    - Computes the next state
    - Computes the outputs



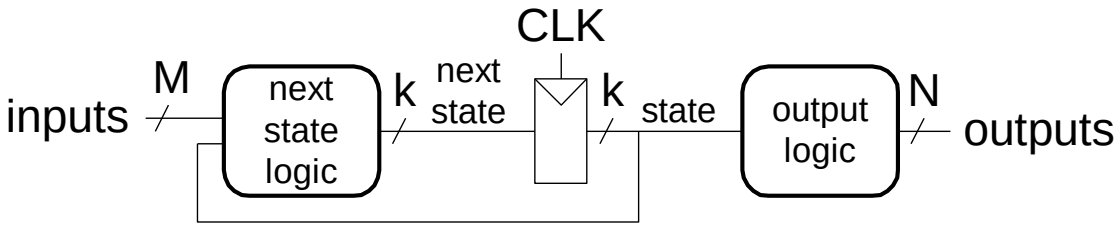




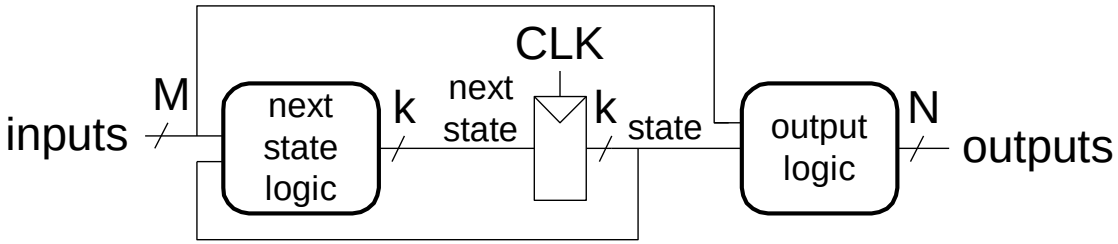
Finite State Machines (FSMs)

- Next state determined by current state and inputs
- Two types of finite state machines differ in output logic:
  - **Moore FSM:** outputs depend only on current state
  - **Mealy FSM:** outputs depend on current state *and* inputs

Moore FSM



Mealy FSM

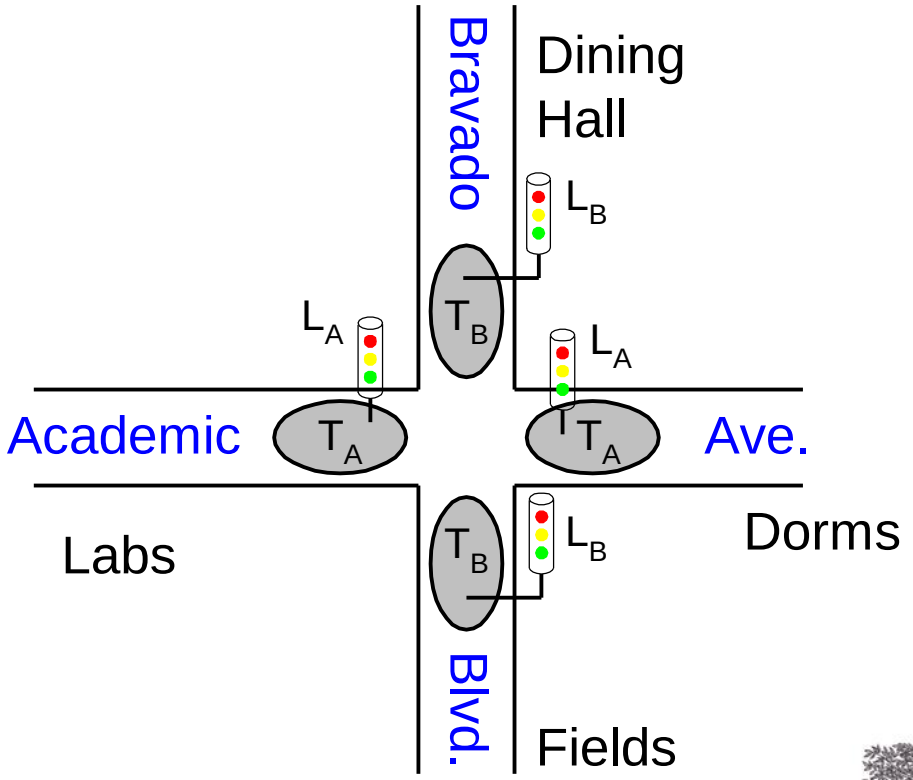


FSM Design Procedure

1. Identify inputs and outputs
2. Sketch state transition diagram
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic

FSM Example

- Traffic light controller
  - Traffic sensors: T_A, T_B (TRUE when there's traffic)
  - Lights: L_A, L_B

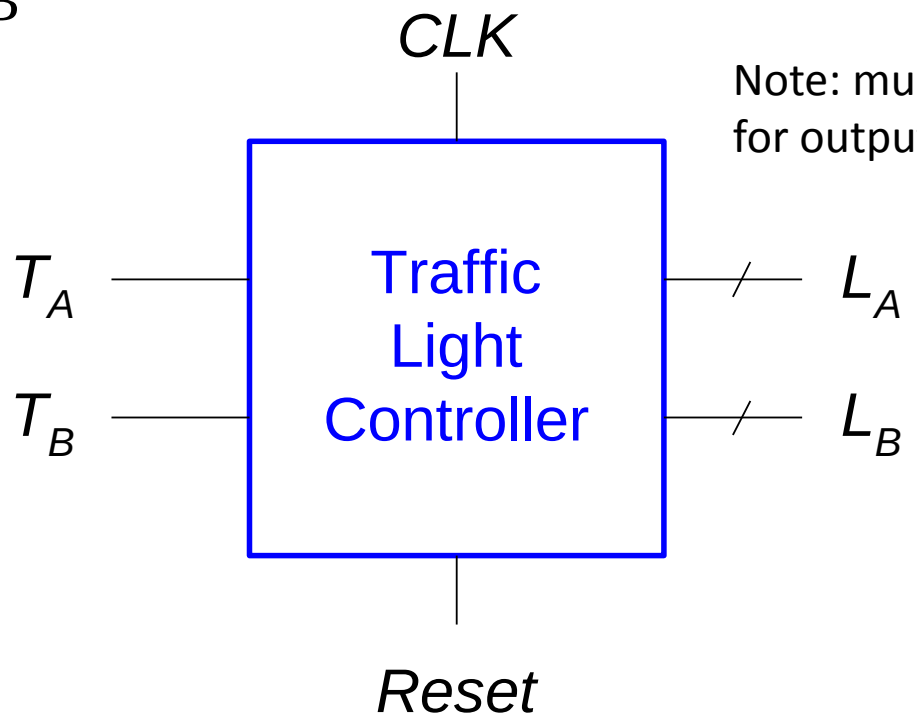


FSM Design Procedure

1. **Identify inputs and outputs**
2. Sketch state transition diagram
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic

FSM Black Box

- Inputs: *CLK*, *Reset*, T_A, T_B
- Outputs: L_A, L_B

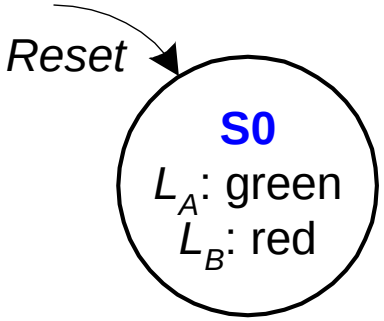


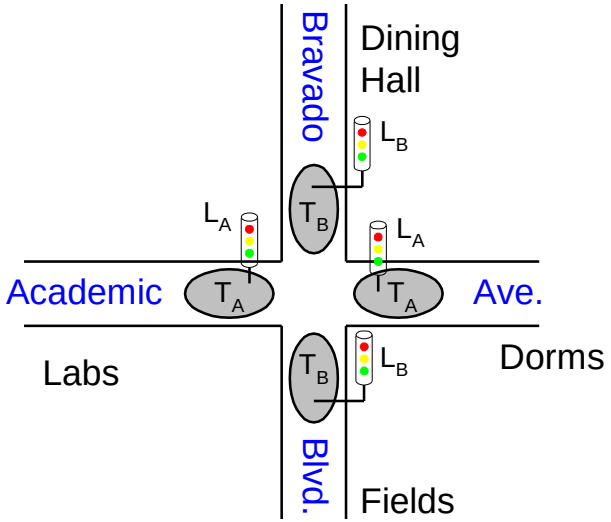
FSM Design Procedure

1. Identify inputs and outputs
2. **Sketch state transition diagram**
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic

FSM State Transition Diagram

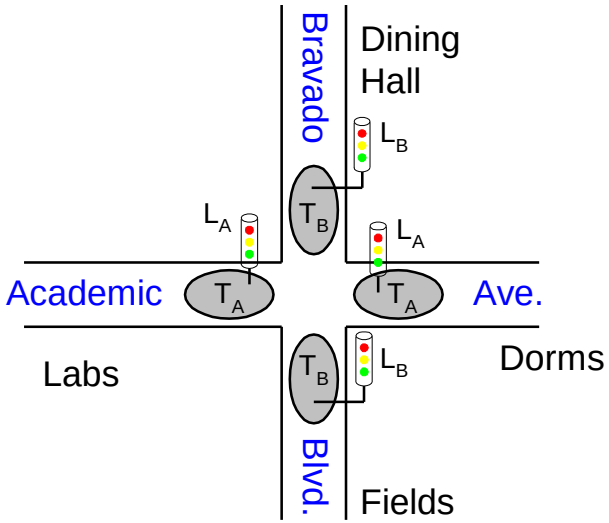
- **Moore FSM:** outputs labeled in each state
- **States:** Circles
- **Transitions:** Arcs

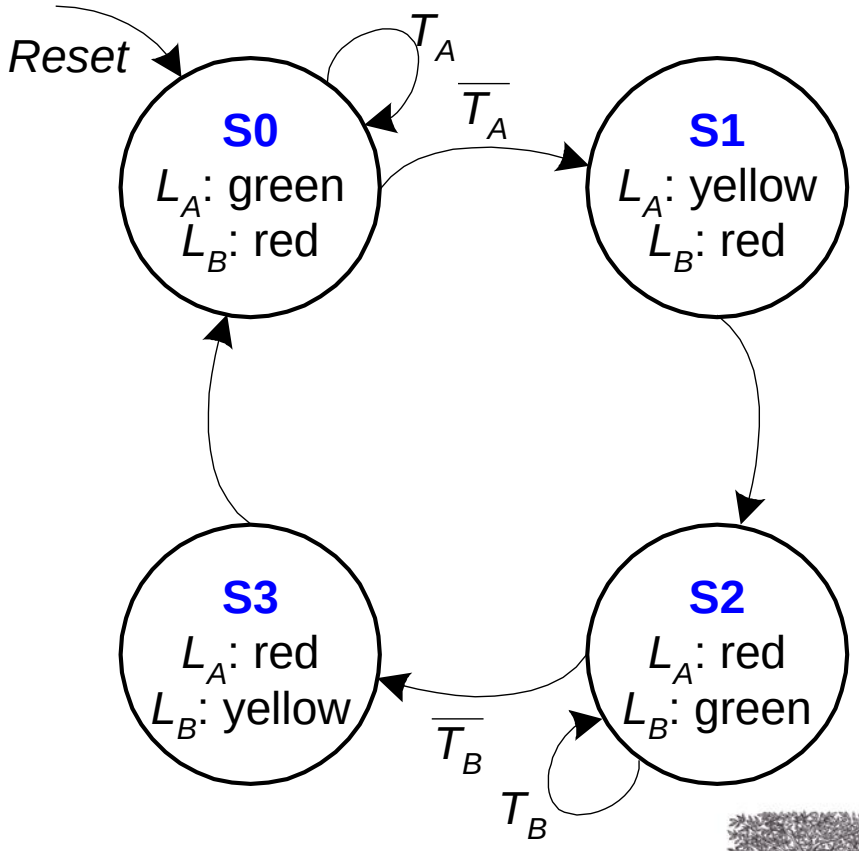




FSM State Transition Diagram

- **Moore FSM:** outputs labeled in each state
- **States:** Circles
- **Transitions:** Arcs

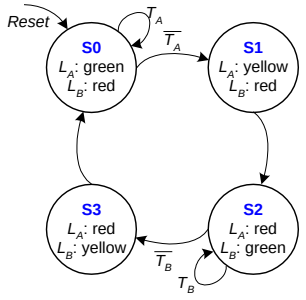




FSM Design Procedure

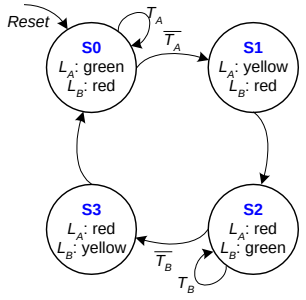
1. Identify inputs and outputs
2. Sketch state transition diagram
3. **Write state transition table**
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic

FSM State Transition Table



Current State S	Inputs T_A	Inputs T_B	Next State S'
S0	0	X	
S0	1	X	
S1	X	X	
S2	X	0	
S2	X	1	
S3	X	X	

FSM State Transition Table



Current State S	Inputs T_A	Inputs T_B	Next State S'
S0	0	X	S1
S0	1	X	S0
S1	X	X	S2
S2	X	0	S3
S2	X	1	S2
S3	X	X	S0

FSM Design Procedure

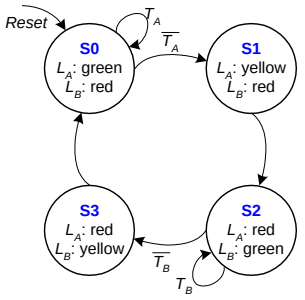
1. Identify inputs and outputs
2. Sketch state transition diagram
3. Write state transition table
4. **Select state encodings**
5. **Rewrite state transition table with state encodings**
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic

FSM Encoded State Transition Table

Current State		Inputs		Next State	
S_1	S_0	T_A	T_B	S'_1	S'_0
0	0	0	X		
0	0	1	X		
0	1	X	X		
1	0	X	0		
1	0	X	1		
1	1	X	X		

State	Encoding
S0	00
S1	01
S2	10
S3	11

Two bits required for 4 states



FSM Encoded State Transition Table

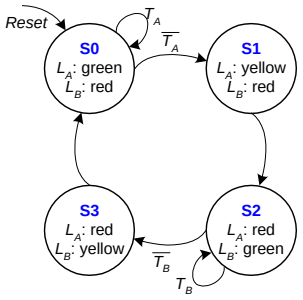
Current State		Inputs		Next State	
S_1	S_0	T_A	T_B	S'_1	S'_0
0	0	0	X	0	1
0	0	1	X	0	0
0	1	X	X	1	0
1	0	X	0	1	1
1	0	X	1	1	0
1	1	X	X	0	0

State	Encoding
S0	00
S1	01
S2	10
S3	11

Two bits required for 4 states

$$S'_1 = S_1 \oplus S_0$$

$$S'_0 = \overline{S_1}\,\overline{S_0}\,\overline{T_A} + S_1\overline{S_0}\,\overline{T_B}$$



FSM Design Procedure

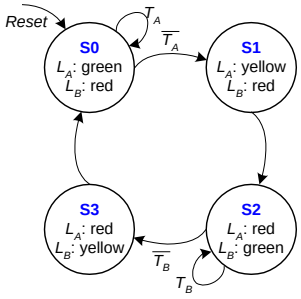
1. Identify inputs and outputs
2. Sketch state transition diagram
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. **Write output table**
7. **Write Boolean equations for next state and output logic**
8. Sketch the circuit schematic

FSM Output Table

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

Output	Encoding
green	00
yellow	01
red	10

Two bits required for 3 outputs



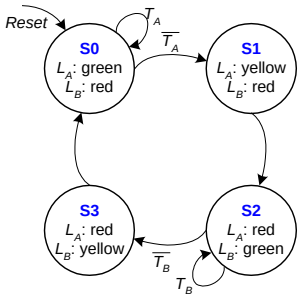
FSM Output Table

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

Output	Encoding
green	00
yellow	01
red	10

Two bits required for 3 outputs

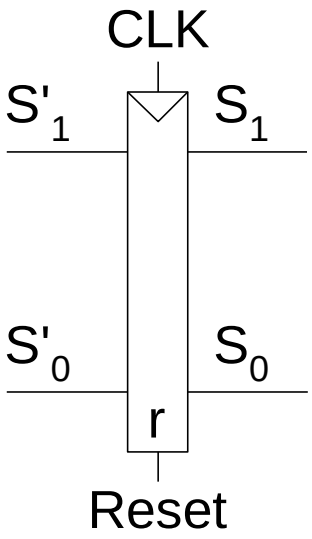
$$L_{A1} = S_1$$
$$L_{A0} = \overline{S_1} S_0$$
$$L_{B1} = \overline{S_1}$$
$$L_{B0} = S_1 S_0$$



FSM Design Procedure

1. Identify inputs and outputs
2. Sketch state transition diagram
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. **Sketch the circuit schematic**

FSM Schematic: State Register

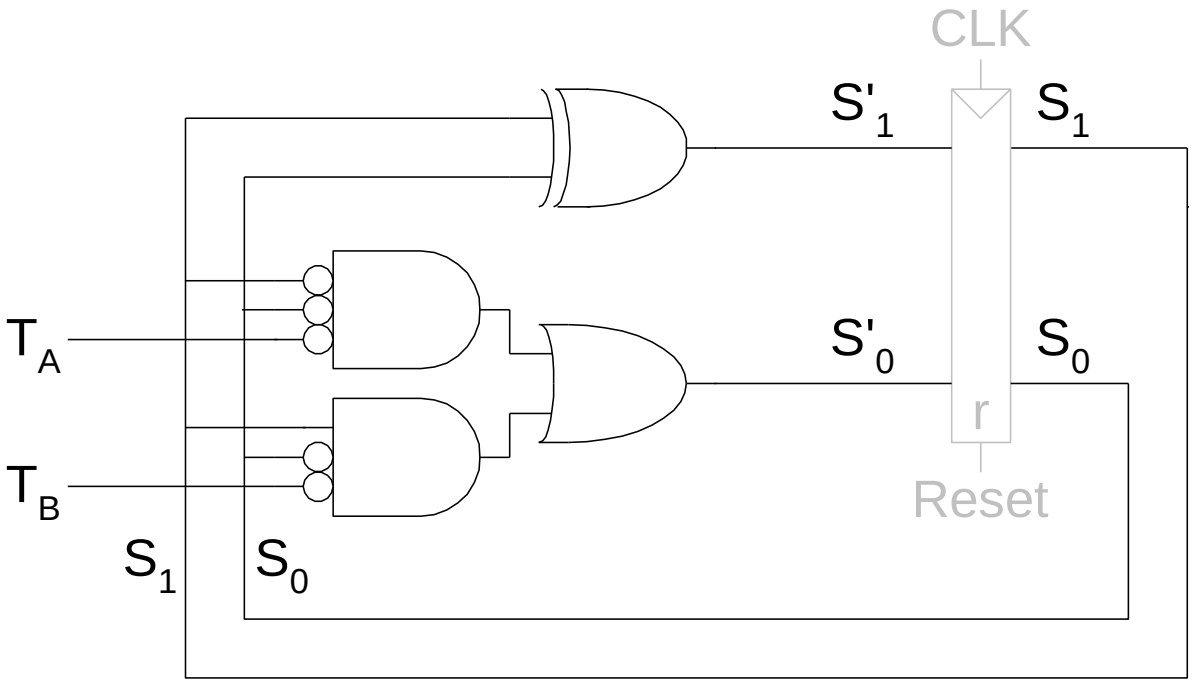


state register

$$S'_1 = S_1 \oplus S_0 \qquad L_{A1} = S_1 \qquad L_{B1} = S_1$$

$$S'_0 = S_1 S_0 T_A + S_1 S_0 T_B \qquad L_{A0} = S_1 S_0 \qquad L_{B0} = S_1 S_0$$

FSM Schematic: Next State Logic

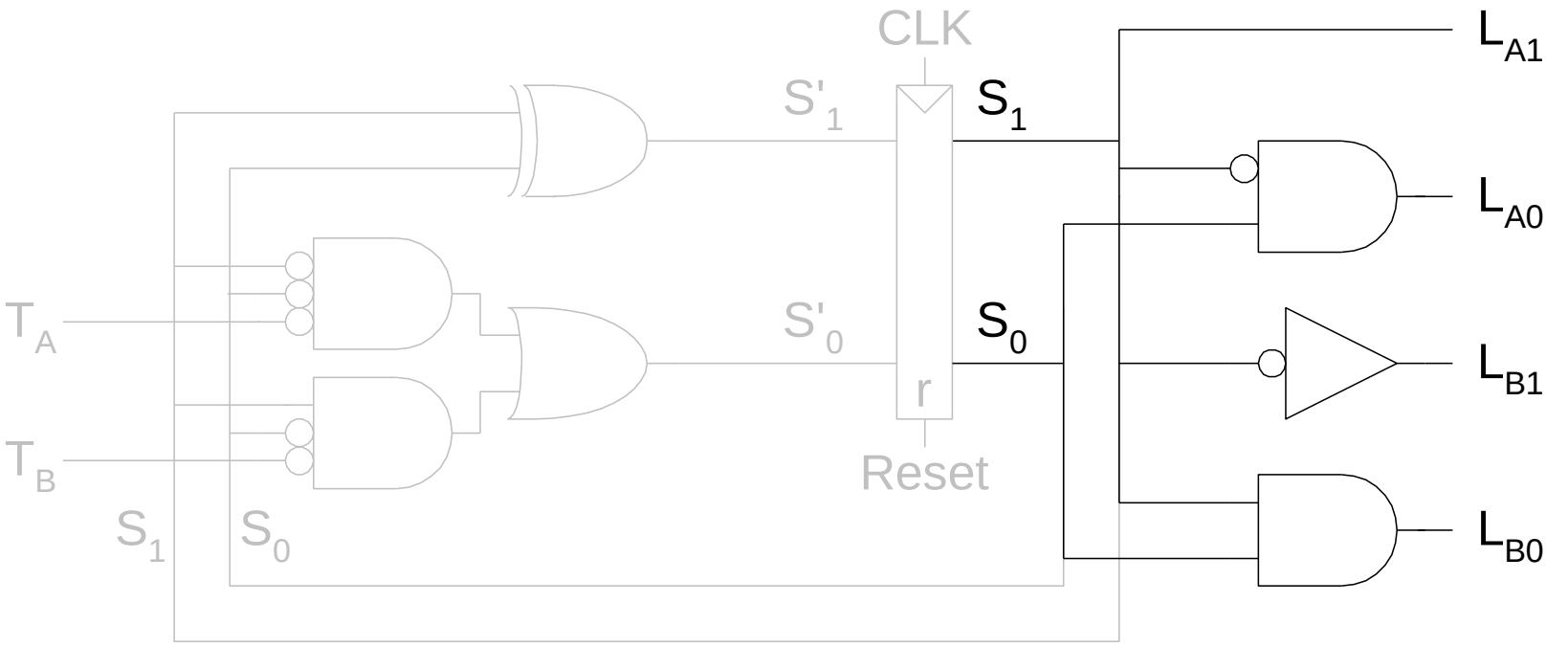


inputs next state logic state register

$$S'_1 = S_1 \oplus S_0$$

$$S'_0 = S_1 S_0 T_A + S_1 S_0 T_B$$

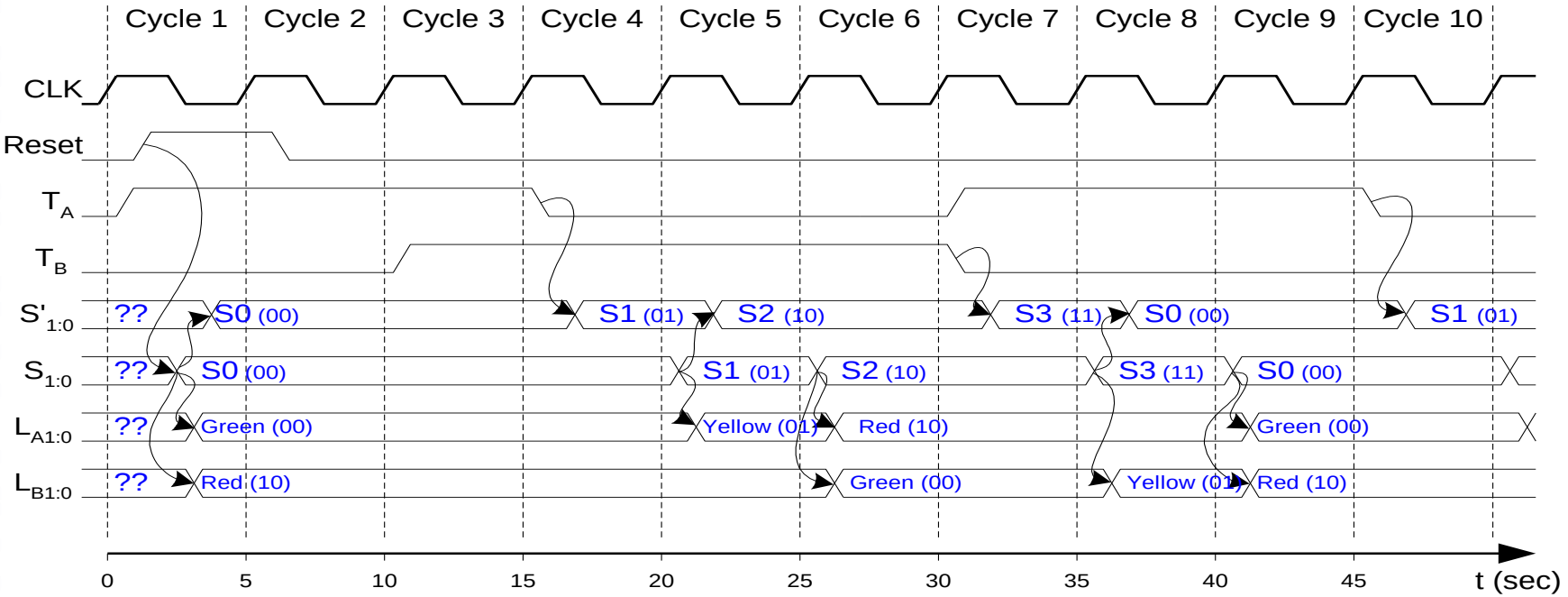
FSM Schematic: Output Logic

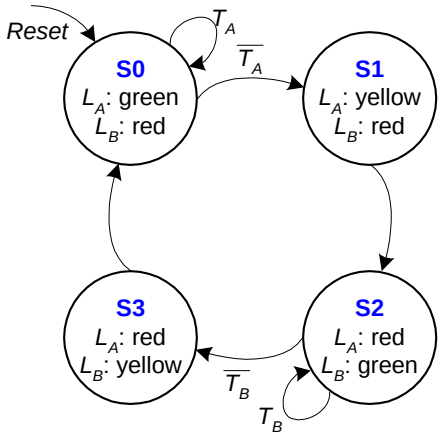


$$L_{A1} = S_1 \qquad L_{B1} = S_1$$

$$L_{A0} = S_1 S_0 \qquad L_{B0} = S_1 S_0$$

FSM Timing Diagram





FSM State Encoding

- **Binary** encoding:
  - i.e., for four states, 00, 01, 10, 11
- **One-hot** encoding
  - One state bit per state
  - Only one state bit HIGH at once
  - i.e., for 4 states, 0001, 0010, 0100, 1000
  - Requires more flip-flops
  - Often next state and output logic is simpler

FSM Design Procedure

1. Identify inputs and outputs
2. Sketch state transition diagram
3. Write state transition table
4. Select state encodings
5. Rewrite state transition table with state encodings
6. Write output table
7. Write Boolean equations for next state and output logic
8. Sketch the circuit schematic