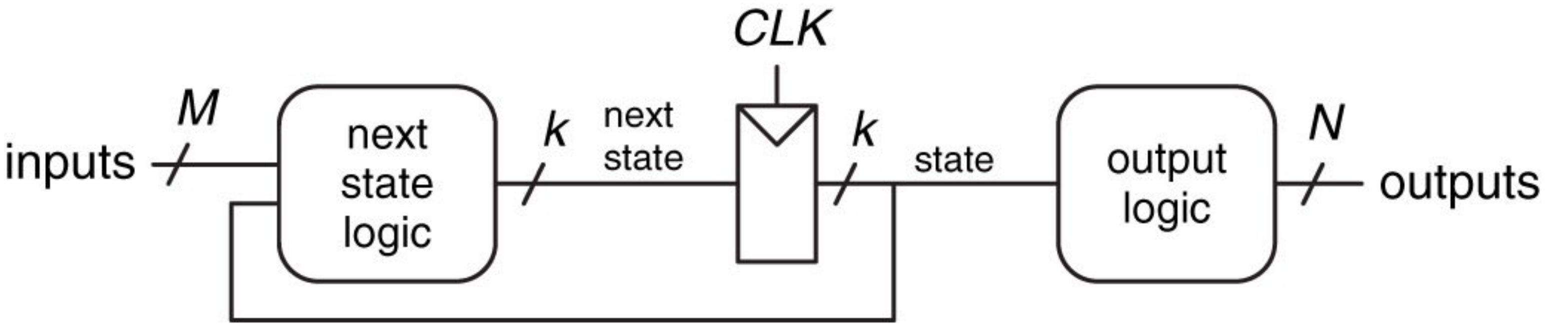
THE UNIVERSITY OF TEXAS AT DALLAS

CS4341 Digital Logic & Computer Design

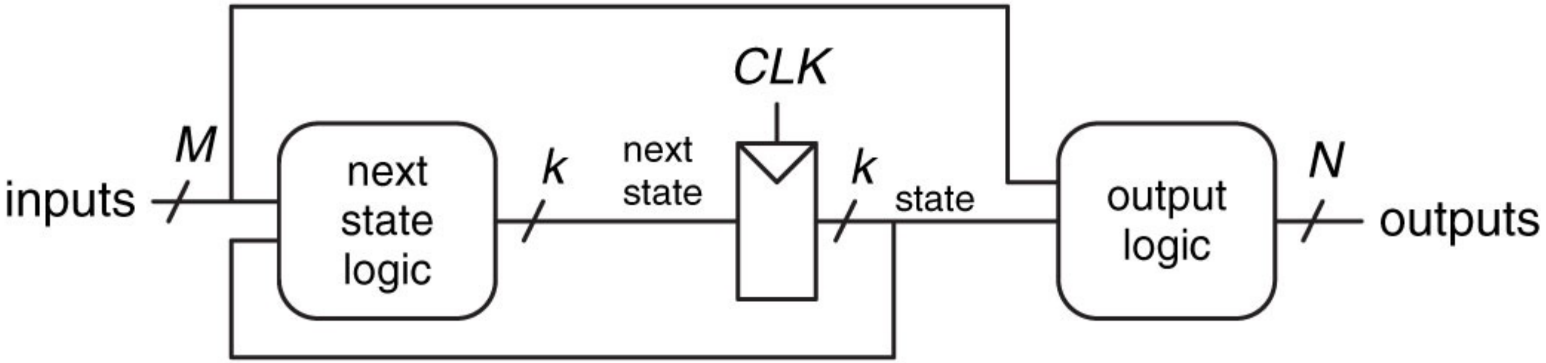
Lecture Notes 15

Omar Hamdy
Assistant Professor
Department of Computer Science

Review: Moore and Mealy Machines



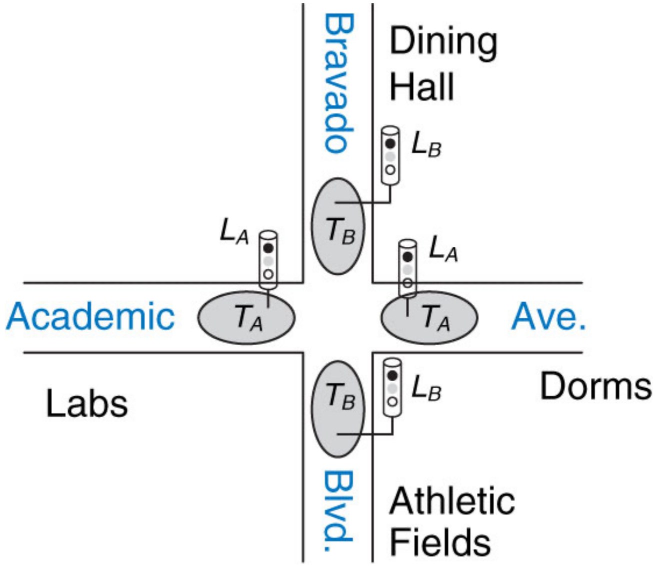
Moore machine

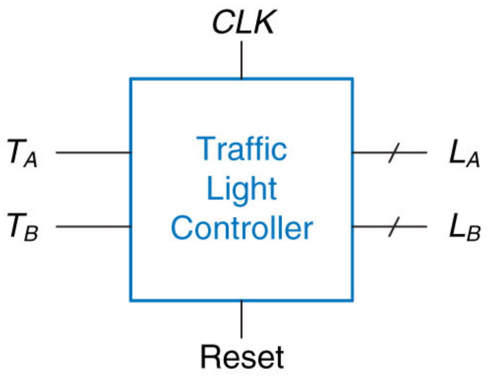


Mealy machine

Review: Traffic Light Design

- Design a system to control two traffic lights L_A and L_B.
- The traffic lights are connected to two sensors T_A and T_B, where each sensor is TRUE if a car is present, and FALSE if the street is empty (inputs)
- Each traffic light receives digital input specifying the color it should display: R, Y, G (outputs)
- Rule is simple: a green light stays green as long as there are cars on that street. Otherwise, it switches to the other light.
- The system is linked to a 5-second clock, where at each rising edge of the clock, the output might change based on the input and the current state
- The system has a reset button. When pressed, it resets the outputs to L_A = Green and L_B = Red
- So, what is the system-level design looks like?



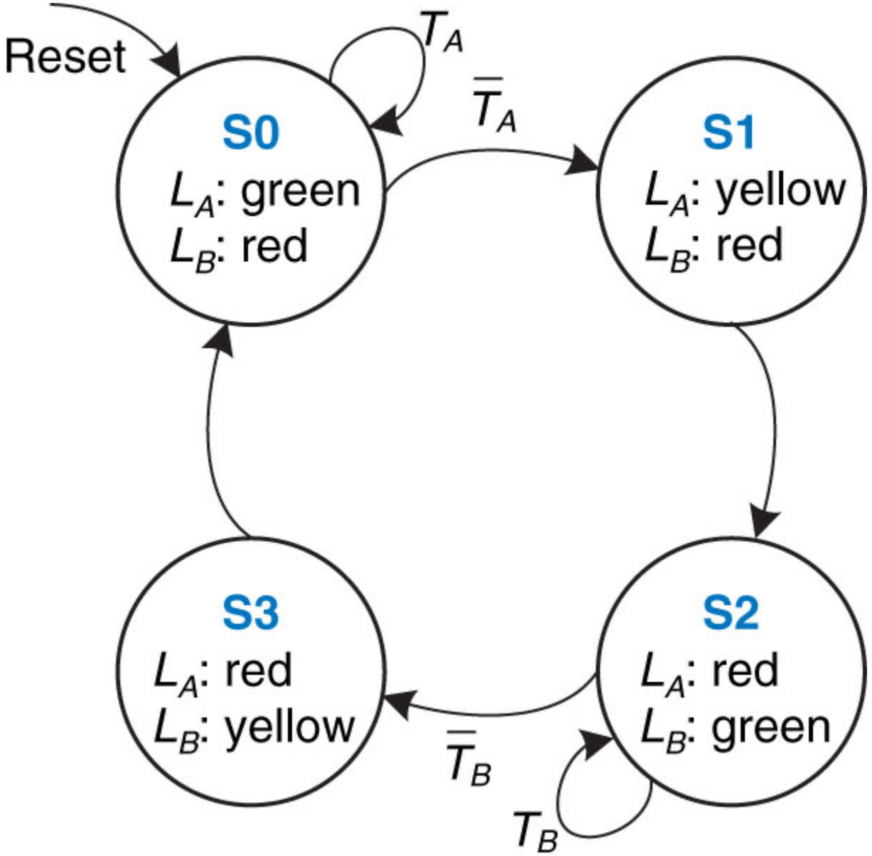


Traffic Light: State Transition Diagram

- How do we determine a state?
  - Using the different possible (legal) outputs the system produces
- How many states do we have?

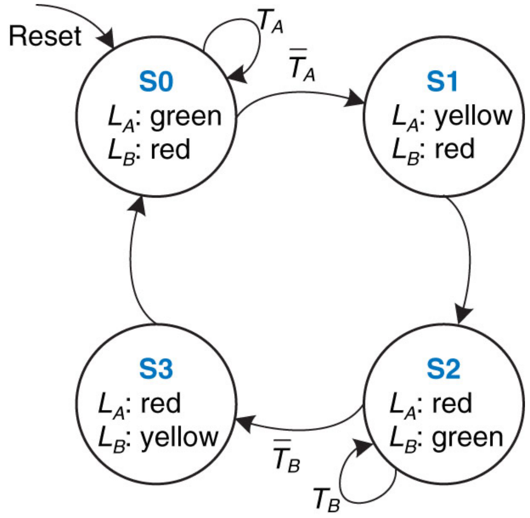
State	L_A	L_B
S0	Green	Red
S1	Yellow	Red
S2	Red	Green
S3	Red	Yellow

- What are the transitioning conditions?
- How many bits of memory are needed?



Traffic Light: State Transition Truth Table

- The first truth table needed is the state transition table, which determines all the next state S' given current state S and input.
- There are two ways: either we build the full table (2 state bits and 2 input sensors = 2^4 = 16 lines), or we build from the state diagram (easier)



Current State S	T_A	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

Traffic Light: State Encoding

- Before we proceed in the analysis, we need to give a binary code for each of the 4 states using 2 binary bits.

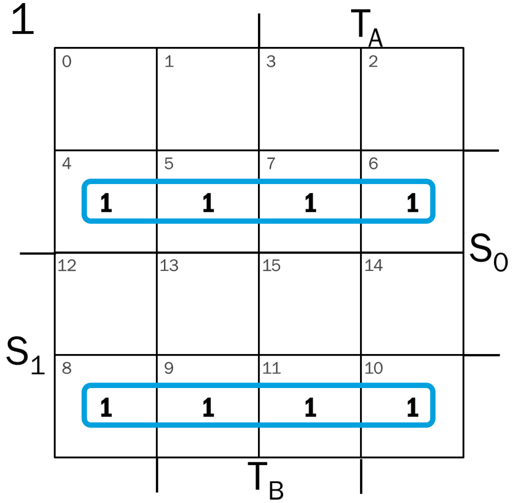
State	Encoding $S_{1:0}$	
S0	0	0
S1	0	1
S2	1	0
S3	1	1

Traffic Light: State Transition Truth Table

- Properly represent current state, next state and inputs as individual bits
- Simplify using K-Map and express S'_1 and S'_0 algebraically

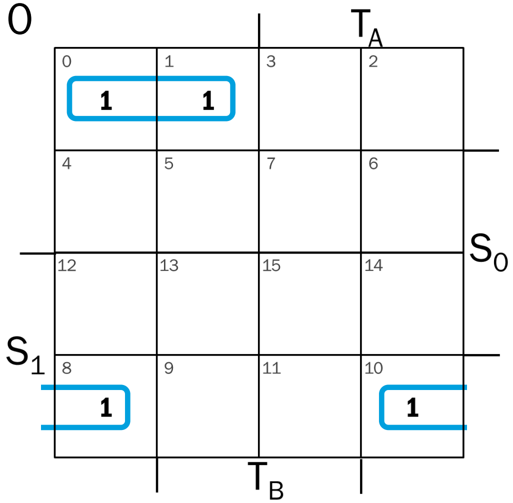
Current State		Input		Next State	
S_1	S_0	T_A	T_B	S'_1	S'_0
0	0	0	X	0	1
0	0	1	X	0	0
0	1	X	X	1	0
1	0	X	0	1	1
1	0	X	1	1	0
1	1	X	X	0	0

S'_1



$$S'_1 = \bar{S}_1 S_0 + S_1 \bar{S}_0 = S_1 \oplus S_0$$

S'_0



$$S'_0 = \bar{S}_1 \bar{S}_0 \bar{T}_A + S_1 \bar{S}_0 \bar{T}_B$$

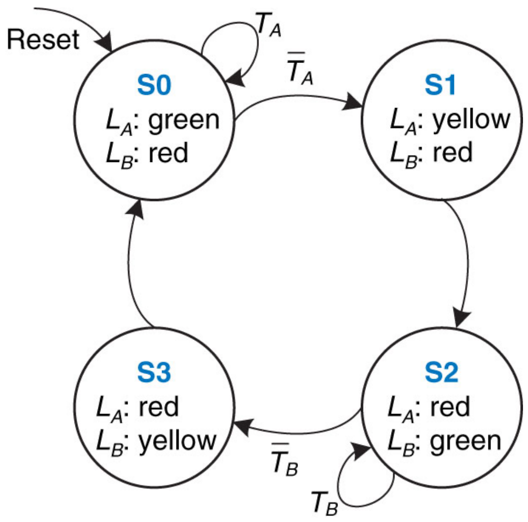
Traffic Light: Output Encoding

- To analyze the output, we first need to assign each output a binary code.
- Since we have 3 colors for each traffic light, then 2 bits are sufficient.

Output	Encoding $L_{1:0}$	
Green	0	0
Yellow	0	1
Red	1	0

Traffic Light: Output Truth Table

- The second truth table needed is the output table, which determines all the output L_A and L_B given current state S.



Output	Encoding $L_{1:0}$	
Green	0	0
Yellow	0	1
Red	1	0

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

What is the algebraic expression for each output?

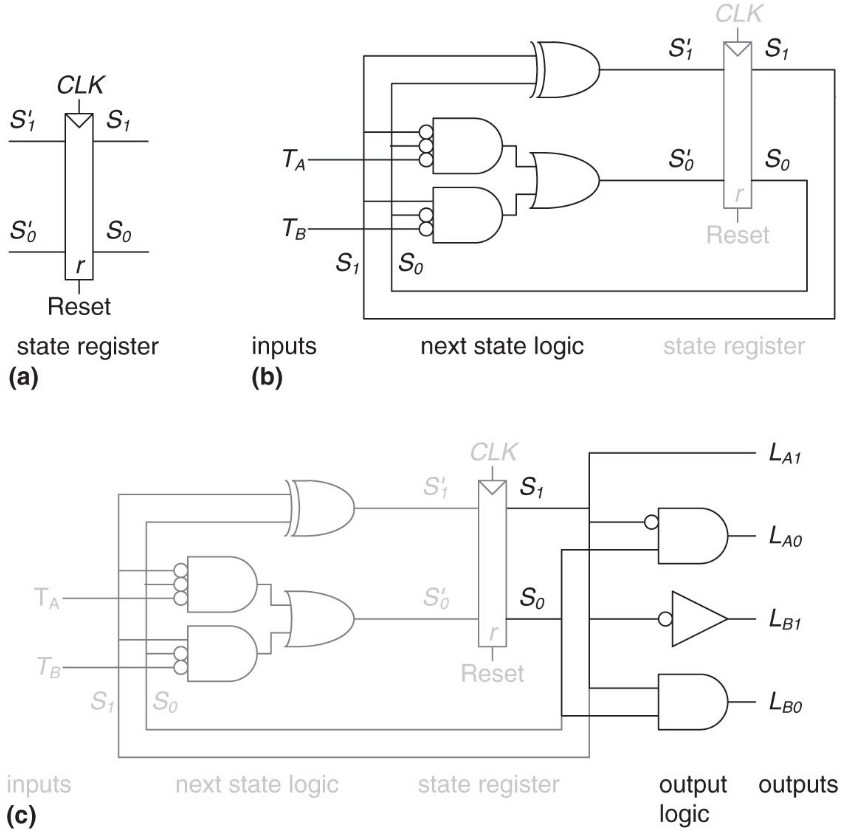
Traffic Light: Output Truth Table

- How to represent each output algebraically?
- Using K-Map, row reduction, or just by looking at the table, we can say:
  - $L_{A1} = S1$
  - $L_{A0} = \overline{S}_1 S_0$
  - $L_{B1} = \overline{S}_1$
  - $L_{B0} = S_1 S_0$

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

Traffic Light: Circuit Design

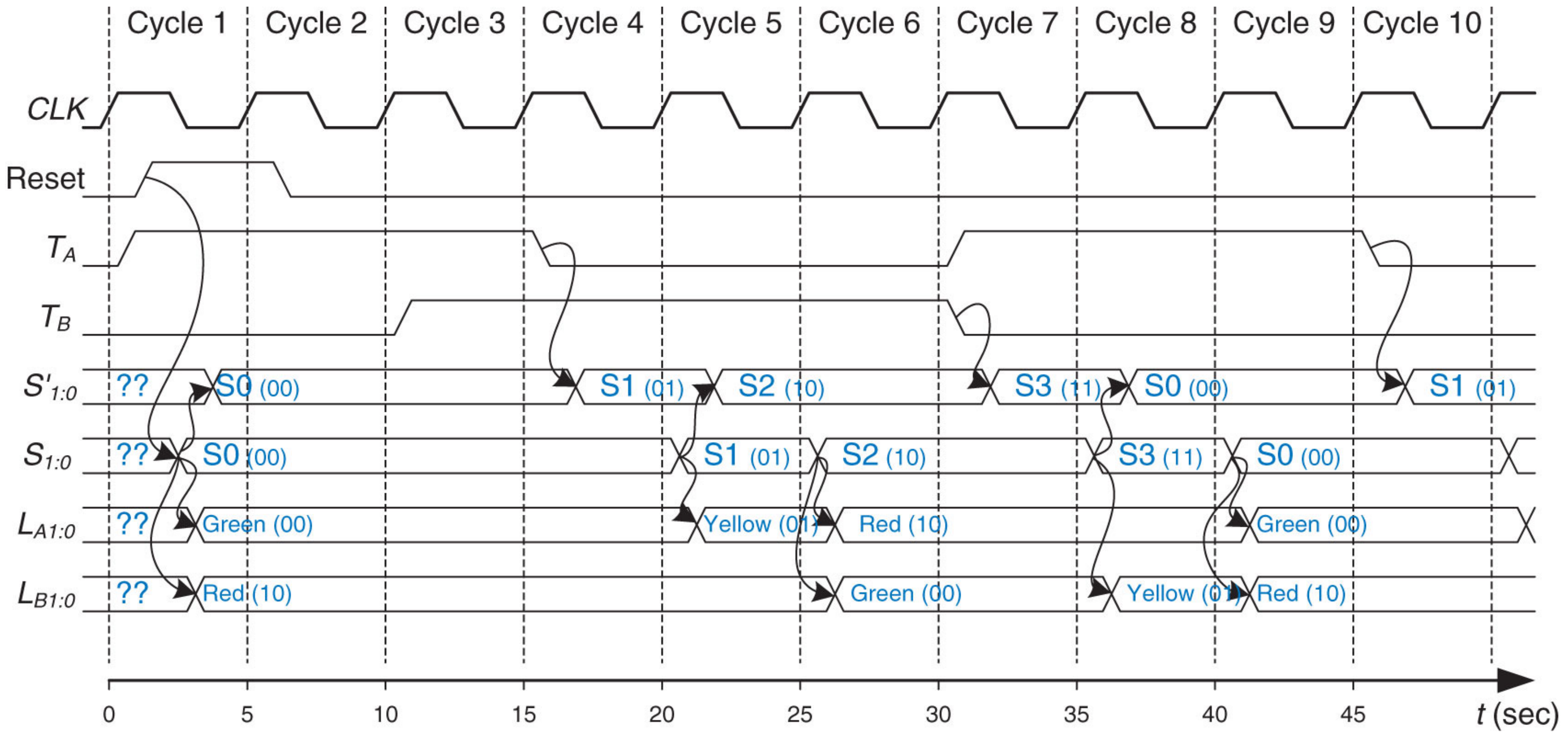
- The final step is to design the next step logic and the output logic circuits using the basic logic gate



- *Registers*
  - $S'_1 = \bar{S}_1 S_0 + S_1 \bar{S}_0 = S_1 \oplus S_0$
  - $S'_0 = \bar{S}_1 \bar{S}_0 \bar{T}_A + S_1 \bar{S}_0 \bar{T}_B$
- *Outputs*
  - $L_{A1} = S1$
  - $L_{A0} = \bar{S}_1 S_0$
  - $L_{B1} = \bar{S}_1$
  - $L_{B0} = S_1 S_0$

Traffic Light: Waveform Analysis

- The following is how the waveform looks like for the traffic circuit



State and Output Encoding

- The state as well as output encodings were chosen randomly and could be selected differently
- Different encoding will result in different circuit design.
- Challenge would then be which encoding can produce the best circuit design (least number of gates, least propagation delay, etc)
- Therefore, choosing good encodings is crucial in the circuit design
- There are two types of encoding:
  - `Binary encoding`: each state or output is represented as a binary number
  - `One-hot-encoding`: each state is represented by its own memory bit.
- `One-hot-encoding` usually means simpler circuit, but more memory bits

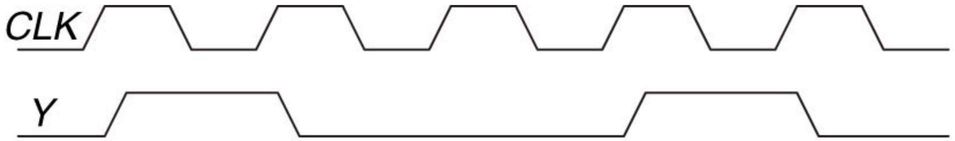
Output	Encoding $L_{1:0}$	
Green	1	1
Yellow	1	0
Red	0	0

Output	Encoding $L_{1:0}$	
Green	1	0
Yellow	1	1
Red	0	1

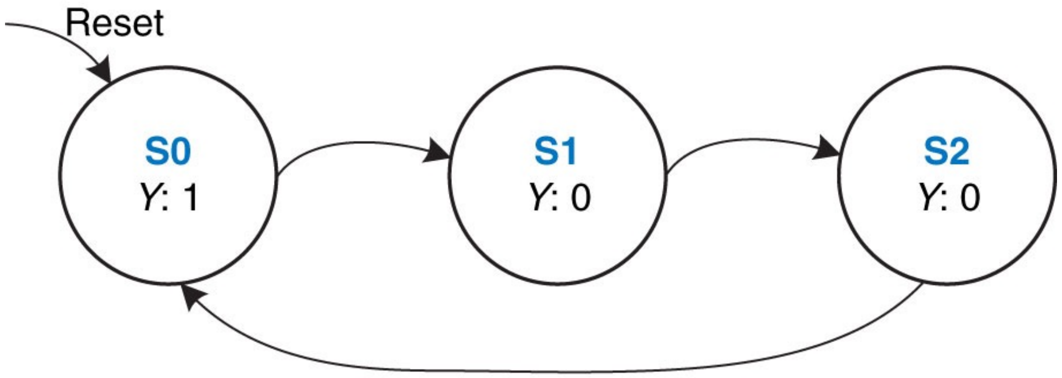
Output	Encoding $L_{1:0}$	
Green	0	1
Yellow	0	0
Red	1	1

Example: Divide-by-N Counter

- This special circuit has no inputs and one output.
- The output Y is High (1) for one clock cycle out of every N



(a)



(b)

- Design the circuit using `binary` and `one-hot` encodings.

To Do List

- Review lecture notes
- Continue working on assignment 2