# Example: Implementation of a Finite-State Machine
# Supplemental Notes for EE 306

J. A. Abraham

September 19, 2018

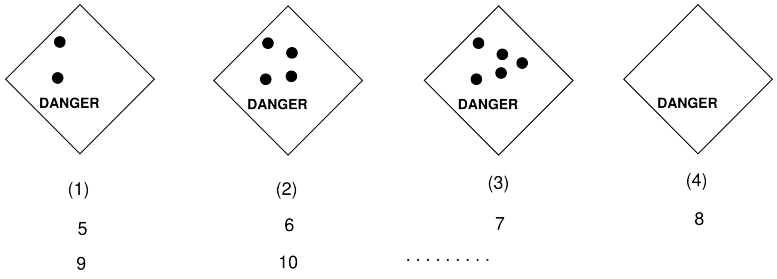
Acknowledgement to Dr. Yale Patt who developed this material for his class.

## 1. Introduction

The concept of designing sequential circuits is a difficult one. This example takes you through the design of a sequential circuit using Master-Slave Flip-Flops.

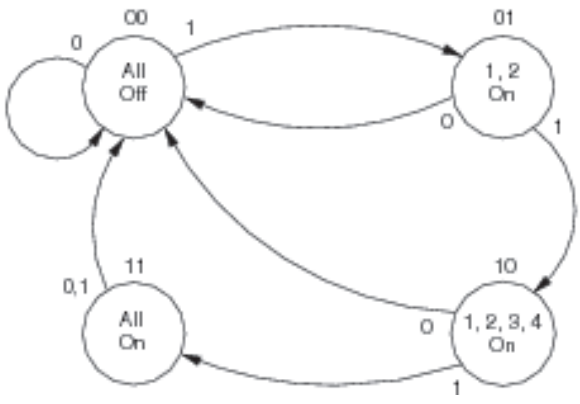
## 2. Problem Statement

Design a controller for a warning sign that operates as follows.
When the input switch is in the ON position, the lights should flash as shown below: (1) two ON, then (2) four ON, then (3) all five ON (completing the arrow), then (4) all OFF, then go back to (1) and repeat the process.
When the input switch is OFF, all lights remain OFF.




## 3. Finite-State Description

We can represent the behavior of the controller with a finite-state machine (FSM) *State Diagram* (with four states, labeled 00, 01, 10, 11 or alternately A, B, C, D):



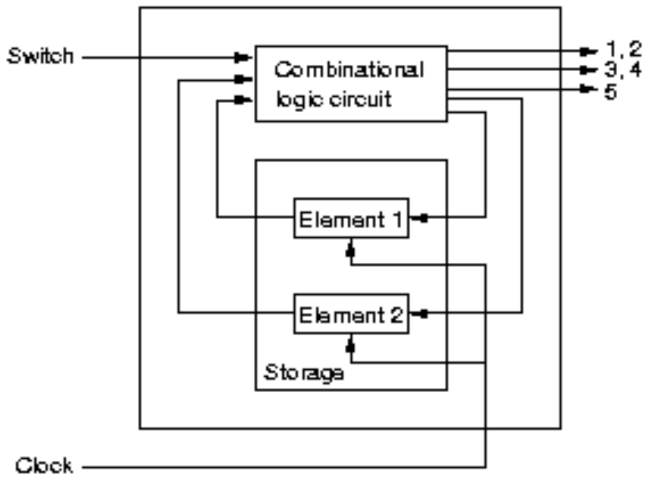
Note that there is one input (the Switch). When it is ON, we cycle through the states. When it is OFF, we go to the state A and remain in that state. Also note that we associate the outputs with the States *ONLY*.

We can also represnt the FSM as a *State Table* and assign a **unique** code to each state:

| $S_1S_0$ | | SW OFF | SW ON | 1,2 | 3,4 | 5 |
|---|---|---|---|---|---|---|
| (00) | A | A | B | 0 | 0 | 0 |
| (01) | B | A | C | 1 | 0 | 0 |
| (10) | C | A | D | 1 | 1 | 0 |
| (11) | D | A | A | 1 | 1 | 1 |

## 4. Implementing the State Table

Our next job is to design the logic to implement this state table. First an overall block diagram:

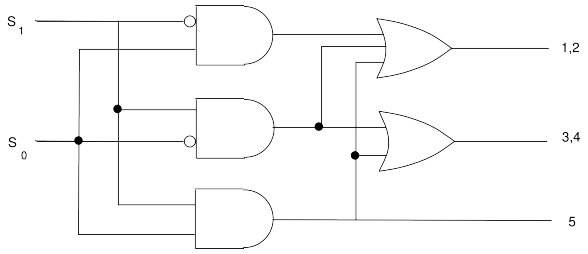



We have three tasks:

- Combinational logic for the Lights
- Combinational logic for the "Next State"
- Design of the Flip-Flops

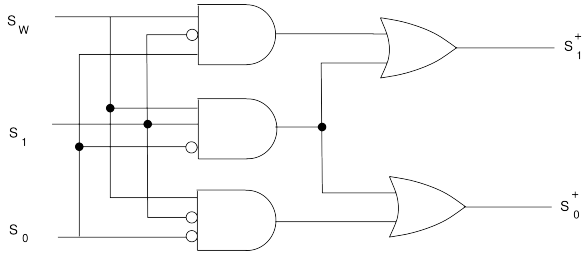
The State Diagram for the Output Lights and the Combinational Logic implementing it:

| | State | 1,2 | 3,4 | 5 |
|---|---|---|---|---|
| A | 00 | 0 | 0 | 0 |
| B | 01 | 1 | 0 | 0 |
| C | 10 | 1 | 1 | 0 |
| D | 11 | 1 | 1 | 1 |

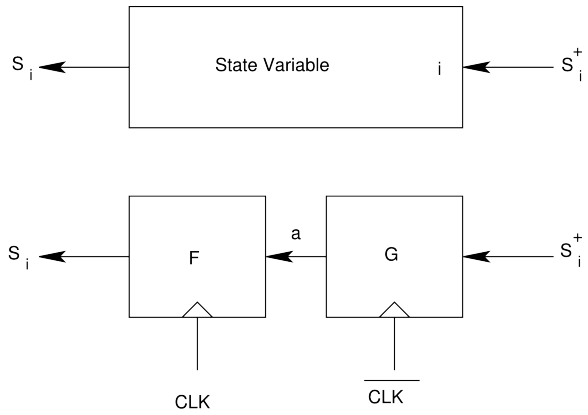


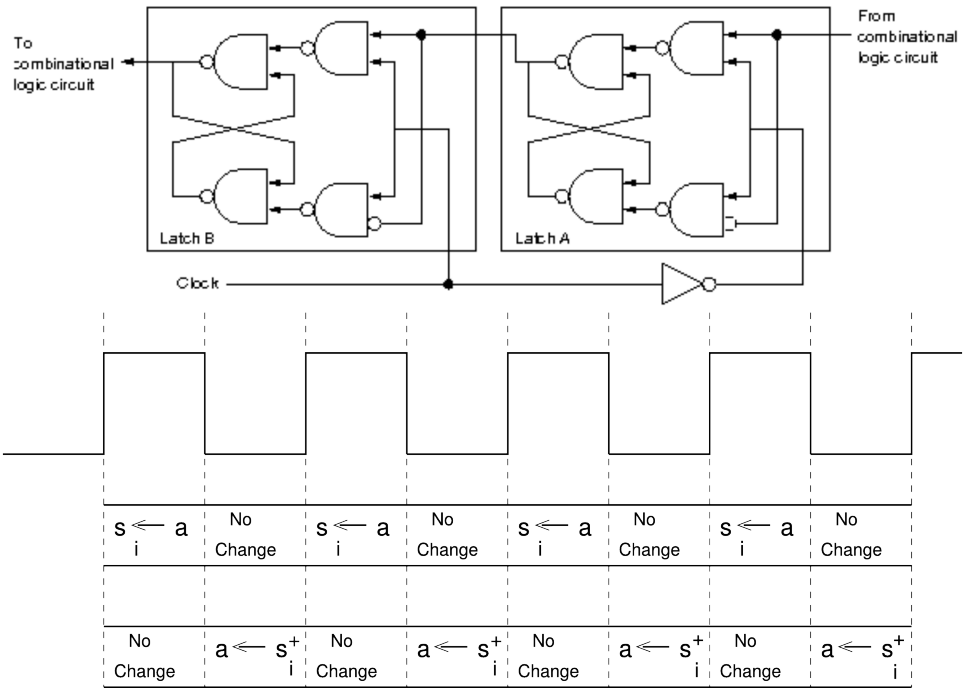

The State diagram for the Next State Function and the Combinational Logic implementing it:

| $S_W$ | State $(S_1\ S_0)$ | Next State $(S_1^+\ S_2^+)$ |
|---|---|---|
| 0 | 0 0 | 0 0 |
| 0 | 0 1 | 0 0 |
| 0 | 1 0 | 0 0 |
| 0 | 1 1 | 0 0 |
| 1 | 0 0 | 0 1 |
| 1 | 0 1 | 1 0 |
| 1 | 1 0 | 1 1 |
| 1 | 1 1 | 0 0 |

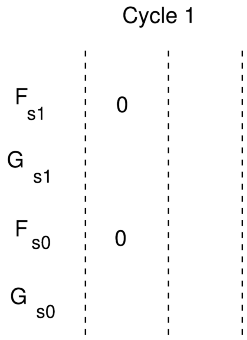



## 5. The Master-Slave Flip-Flop



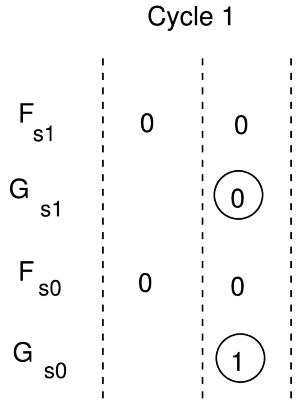



Example: We start in Cycle 1 in State A (0 0) with the Switch ON.

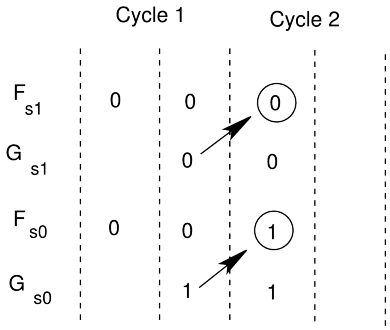



In the first half of Cycle 1, $\overline{CLK}$ prevents $G_i$ from Latching $S_i^+$

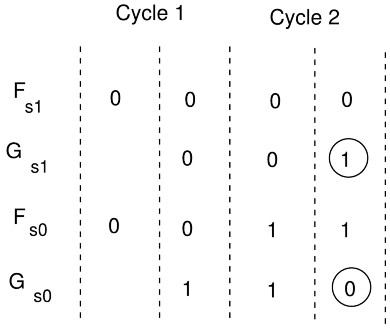
In the second half of Cycle 1, $\overline{CLK} = 1$, so $S_i^+$ can be latched in $G_i$. But, since CLK = 0, the output of $G_i$ (which is $a_i$) cannot be latched in $F_i$. The result:



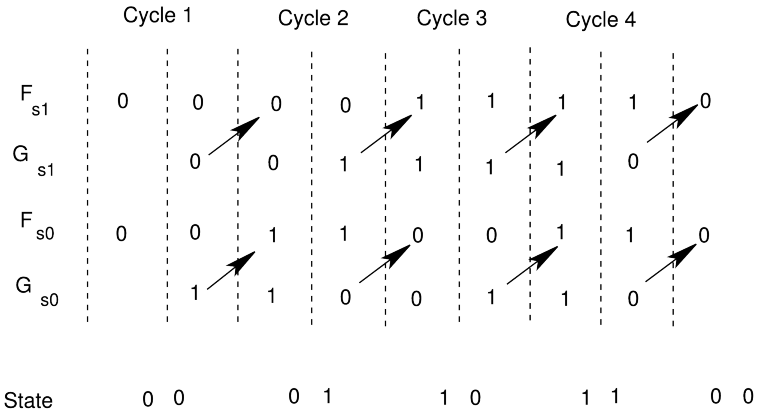
At the start of Cycle 2, $CLK = 1$, so $F_i$ can latch $a_i$ (the output of $G_i$). Since $\overline{CLK} = 0$, the subsequent generation of the **new** $S_i^+$ cannot be latched in $G_i$. The result:




In the second half of Cycle 2, $\overline{CLK} = 1$, so the next $S_i^+$ can be latched in $G_i$. But, since $CLK = 0$, the output of $G_i$ cannot be latched in $F_i$.




We can continue on to Cycle 3, Cycle 4, etc. The result is as follows.




$F_{s1}$, $F_{s0}$ Specify the state

$G_{s1}$, $G_{s0}$ Latch the next state in the second half of each cycle