EITF35: Introduction to Structured VLSI Design

Part 1.2.1: Finite State Machines

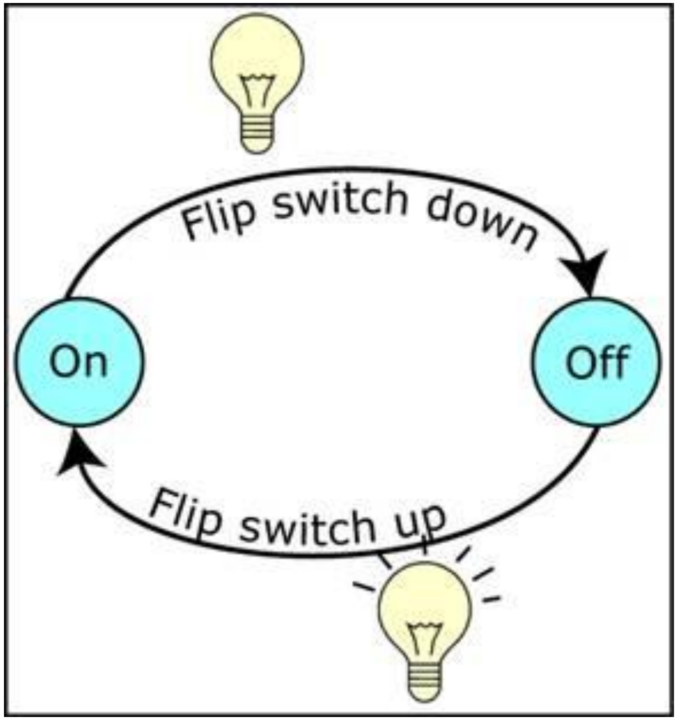
Liang Liu
liang.liu@eit.lth.se

Outline

- FSM Overview
- FSM Representation
  - examples
- Moore vs. Mealy Machine
  - from circuits perspective

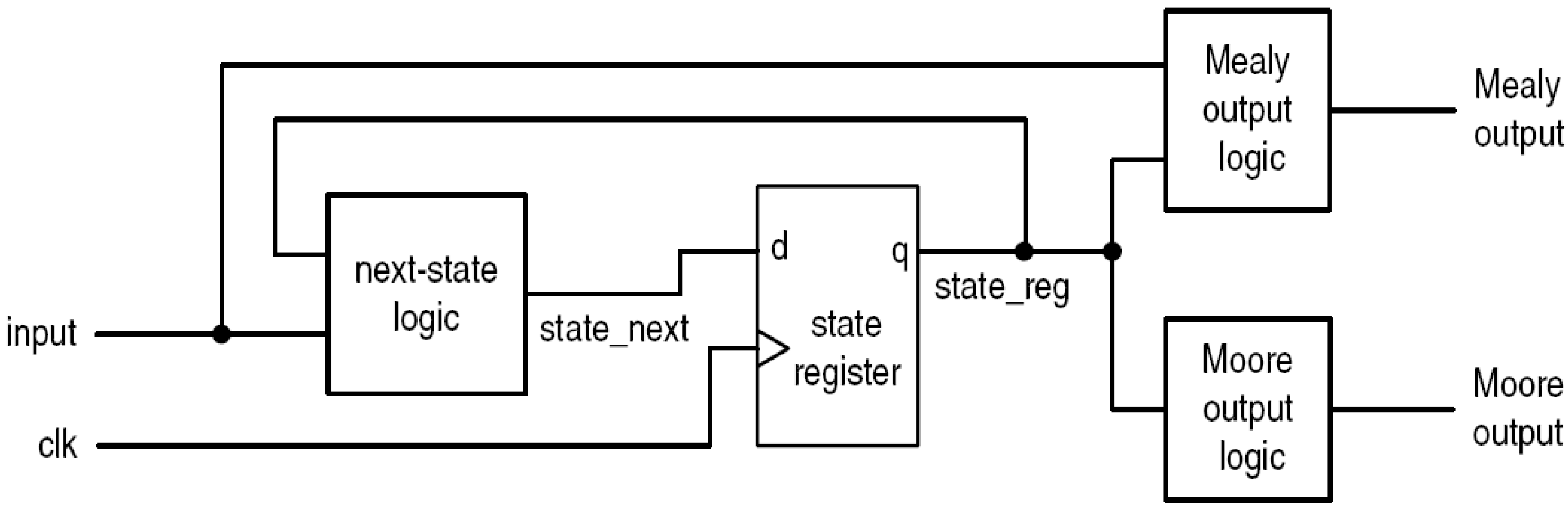
FSM Overview

- It has at most a finite number of states
- Models for representing sequential circuits
- Used mainly as a controller in a large system
- Moore vs. Mealy machines



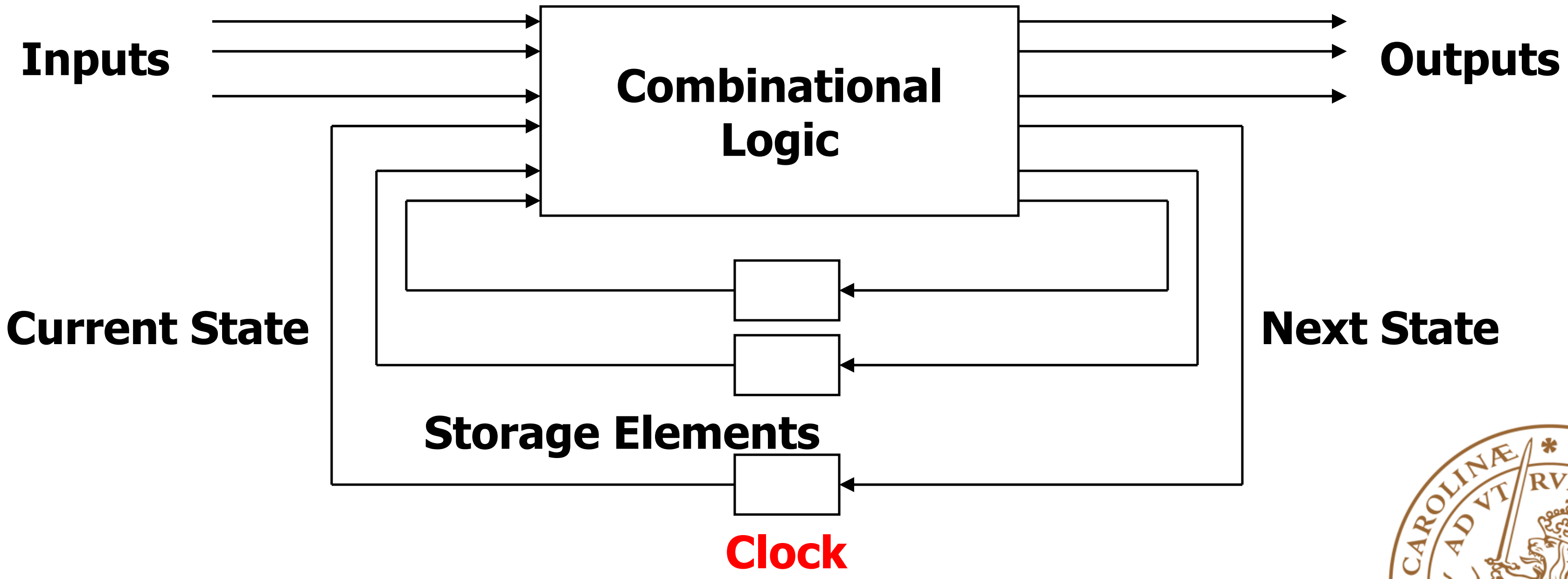
Abstraction of state elements

- A FSM consists of several **states**. **Inputs** into the machine are combined with the ***current state*** of the machine to determine the new state or ***next state*** of the machine.
- Depending on the state of the machine, **outputs** are generated based on either the state or the state and inputs of the machine.



Abstraction of state elements

- **A FSM consists of several states. Inputs into the machine are combined with the *current state* of the machine to determine the new state or *next state* of the machine.**
- **Depending on the state of the machine, outputs are generated based on either the state or the state and inputs of the machine.**
- **Divide circuit into combinational logic and state**



Outline

- FSM Overview
- FSM Representation
- Moore vs. Mealy Outputs
- Exercise

FSM Representation

- **Can be represented using a state transition table which shows the *current state*, *input*, any *outputs*, and the *next state*.**

Input / Current State	$Input_0$	$Input_1$	....	$Input_n$
$State_0$	Next State / Output		....	Next State / Output
$State_1$	....	....		....
....	....	....		....
$State_n$	....	....		....

FSM Representation

- It can also be represented using a ***state diagram*** which has the same information as the state transition table.

- **Mealy Output**

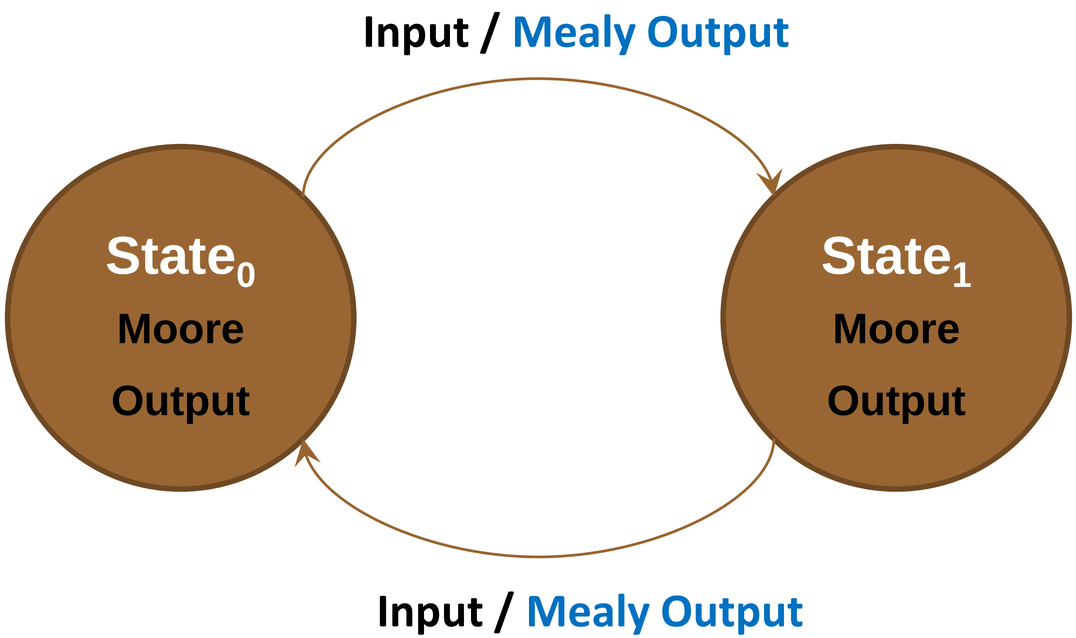
Outputs =F(*Inputs*, *Current state*)

Next state = F(*Inputs*, *Current state*)

- **Moore Output**

Outputs = F(*Current state*)

Next state = F(*Inputs*, *current state*)



Suggestion: do NOT mix Mealy and Moore in one design

Example 1: A Mod-4 Synchronous Counter

- **Function: Counts from 0 to 3 and then repeats; Reset signal reset the counter to 0.**
- **It has a clock (*CLK*) and a *RESET* input.**
- **Outputs appear as a sequence of values of 2 bits (q1 q0)**
- **As the outputs are generated, a new state (s1 s0) is generated which takes on values of 00, 01, 10, and 11.**

State Transition Table of Mod-4 Counter

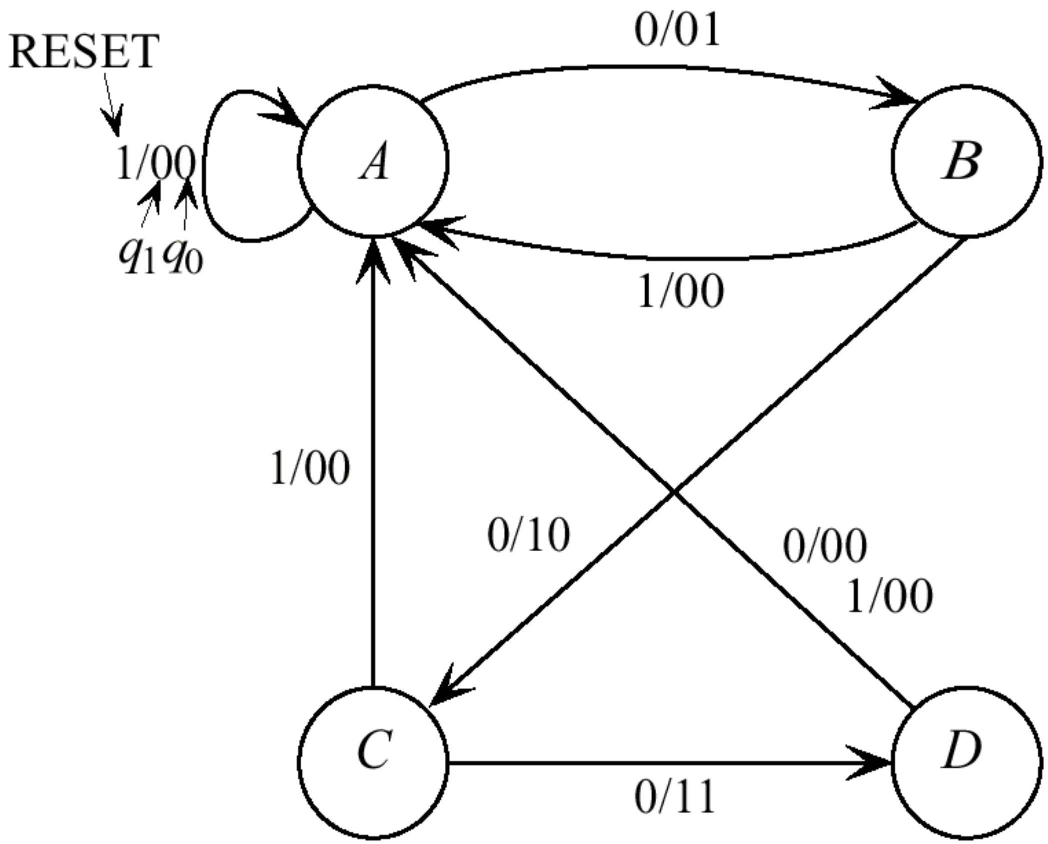
Present state (S_t) \ Input	*RESET* 0	*RESET* 1
A:00	01/01	00/00
B:01	10/10	00/00
C:10	11/11	00/00
D:11	00/00	00/00

Next State

Output

One input is missing!

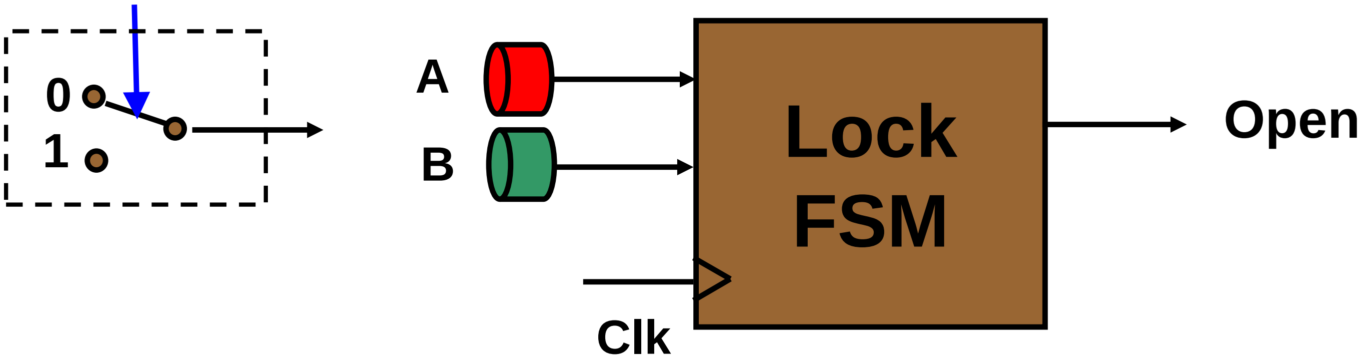
State Transition Diagram for the Mod-4 Counter



Use meaningful names for states

Example 2: Lock

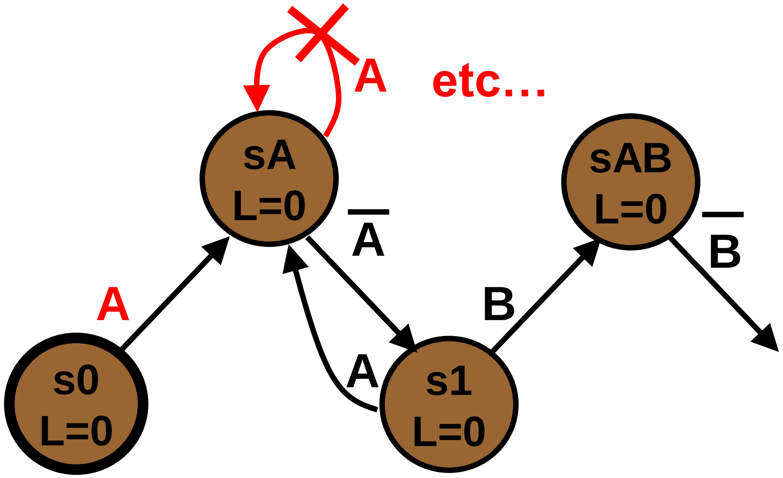
- **Pushing: * { A; B; B; A } => Open**
  - **A & B never push at the same time**
  - **Have to release the button before next pushing**



State Diagram for lock-FSM

{A;B;B;A}

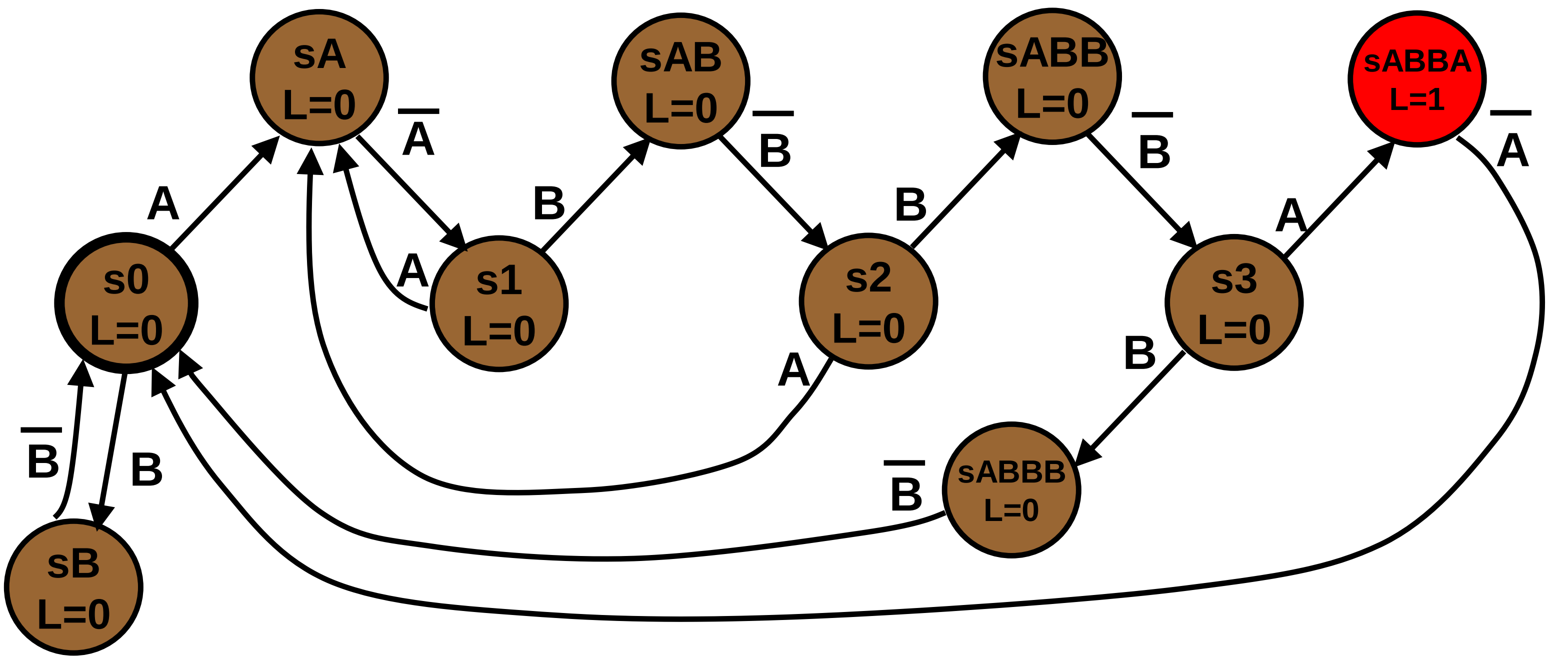
- A and B are never pressed at the same time …
- Debounce before next pushing



Finish the state graph for the Lock-FSM (5min)

State Diagram for lock-FSM

- A and B are never pressed at the same time …
- Debounce before next pushing

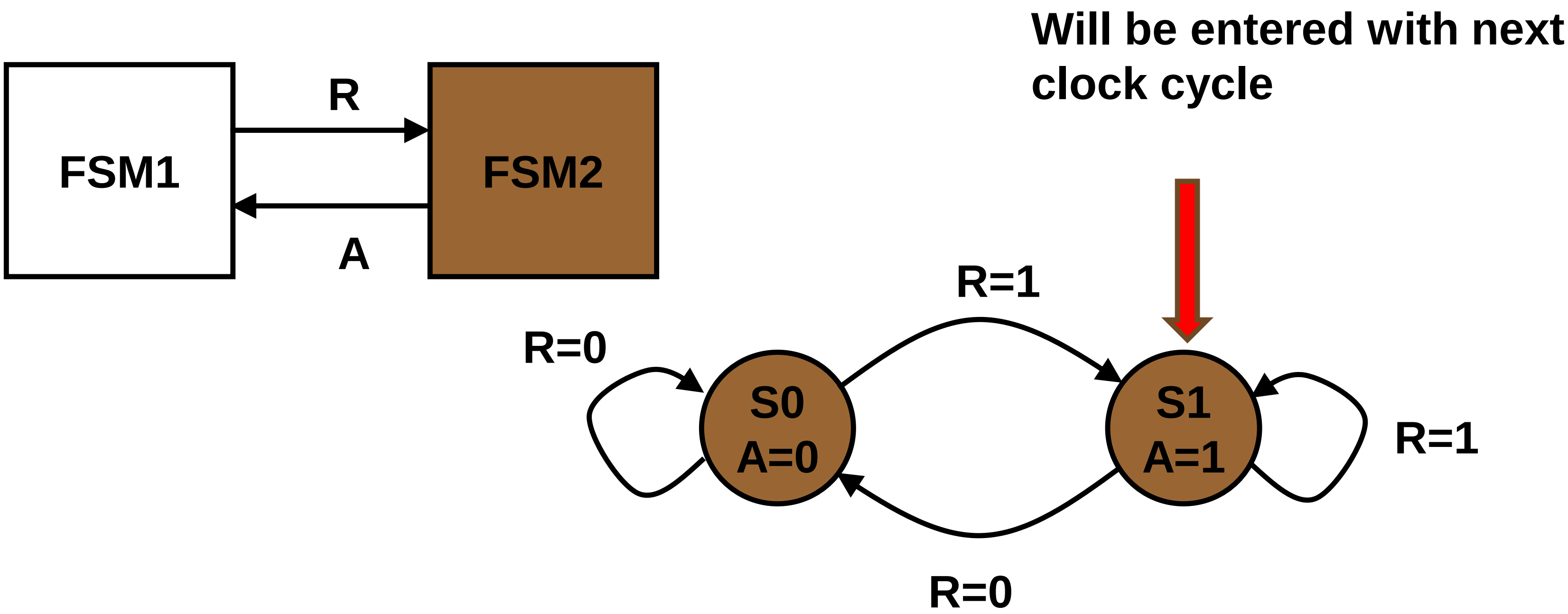


Consider all the input possiblities at each state

Outline

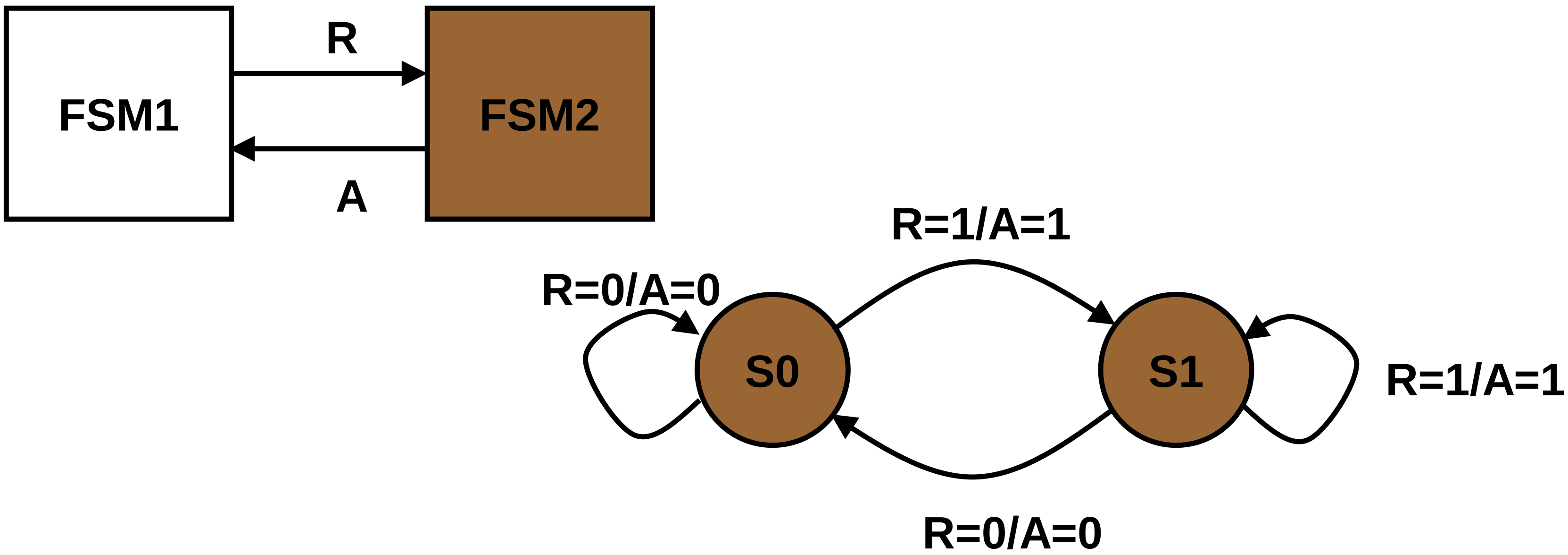
- FSM Overview
- FSM Representation
- **Moore vs. Mealy Outputs**
- Exercise

Output Timing: Moore



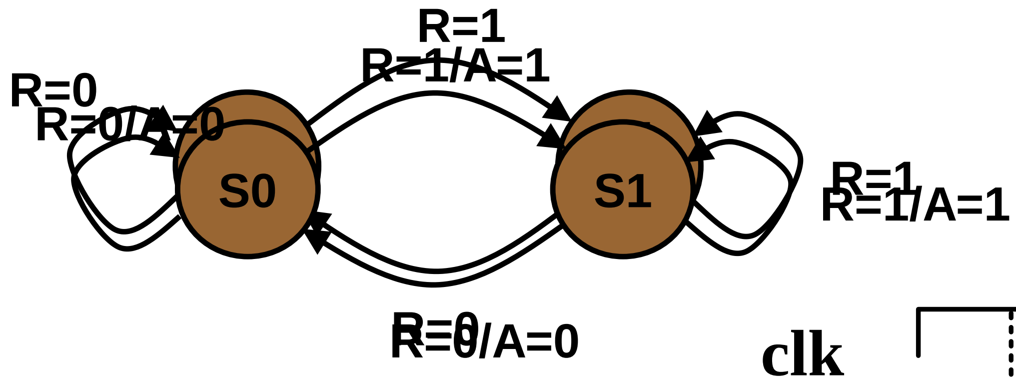
- … a Moore machine is not able to produce A->1 until the **next clock** when it enters s1

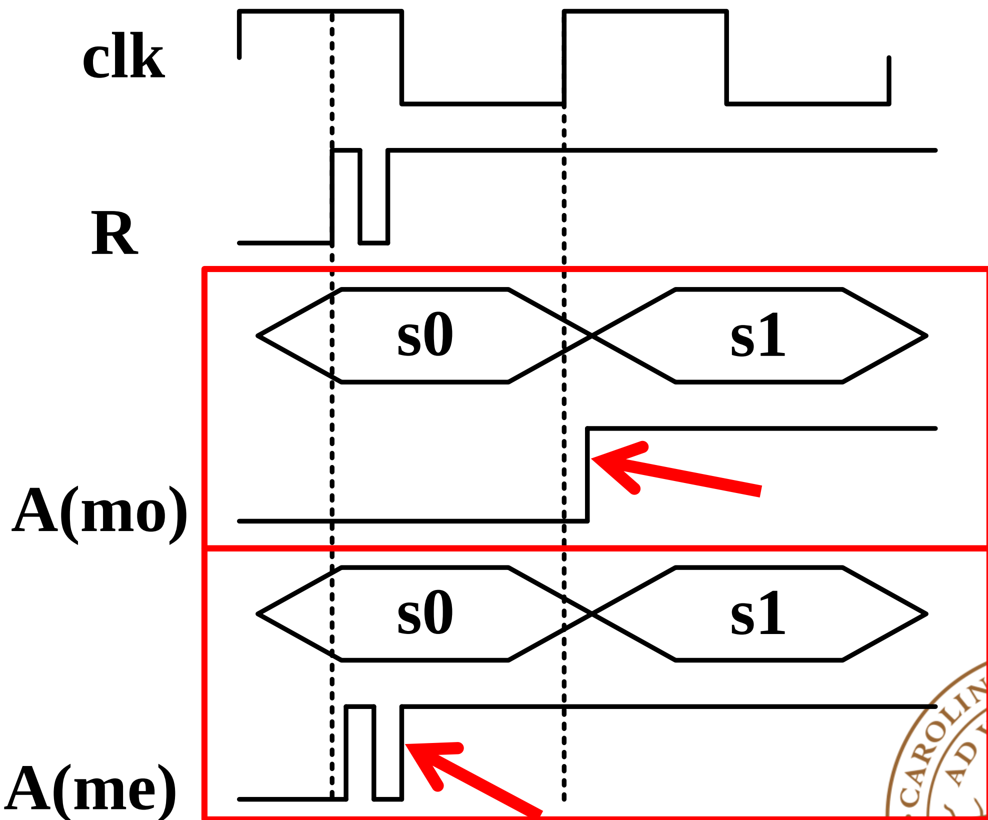
Output Timing: Mealy



- When in s0, a Mealy machine may produce A->1 immediately in response to R->1

Output Timing: Moore and Mealy





Moore vs. Meally (summary)

- A Moore machine produces glitch free outputs.
- A Moore machine produces outputs depending only on states, and this may allow using a higher-frequency clock.
- A Mealy machine can be specified using less states because it is capable of producing different outputs in a given state.
- A Mealy machine can be faster because an output may be produced immediately instead of at the next clock tick.
- Which one is better?
  - Edge sensitive control
    - ***E.g., enable signal of counter***
    - ***Both can be used but Mealy is faster***
  - Level sensitive control
    - ***E.g., write enable signal of SRAM***
    - ***Moore is preferred for glitch free***

?