CSSE2010/CSSE7201
Lecture 8

Sequential Circuits 3
State Machines

School of Information Technology and Electrical Engineering
The University of Queensland

Outline

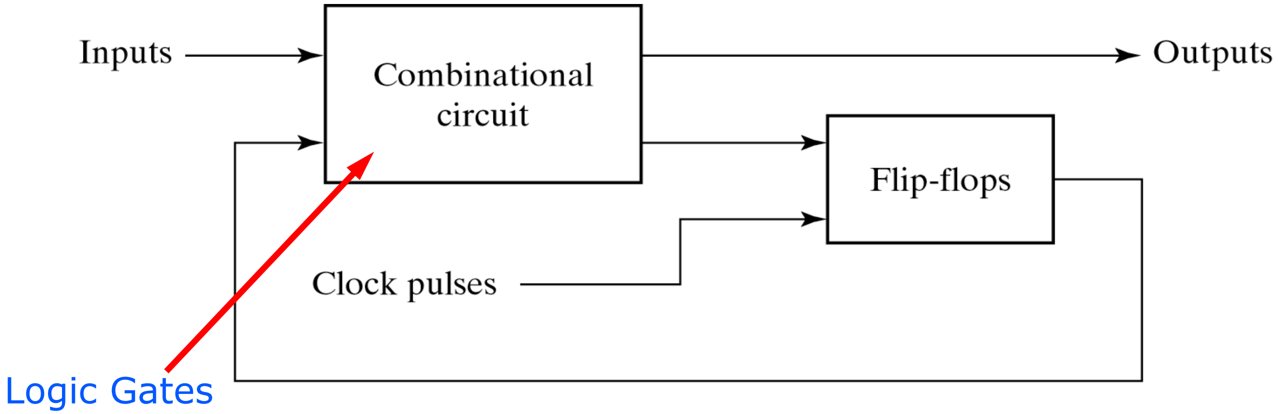
- Admin
- State machines
  - State diagrams
  - State tables
  - State encoding

Admin

- Quiz 3 is due on Friday 4pm this week

Sequential Circuits

- **State** = value stored in flip-flops
- Output depends on input and state
- Next state depends on inputs and state



[Figure from Mano, "Digital Design", 3rd edition]

State Machines

- Sequential circuits can also be called
  - state machines
  - finite state machines (FSMs)
- State machine has
  - Finite number of possible states
  - Only one **current state**
  - Can **transition** to other states based on inputs and current state

State Machines

- The states can be defined based on the problem:

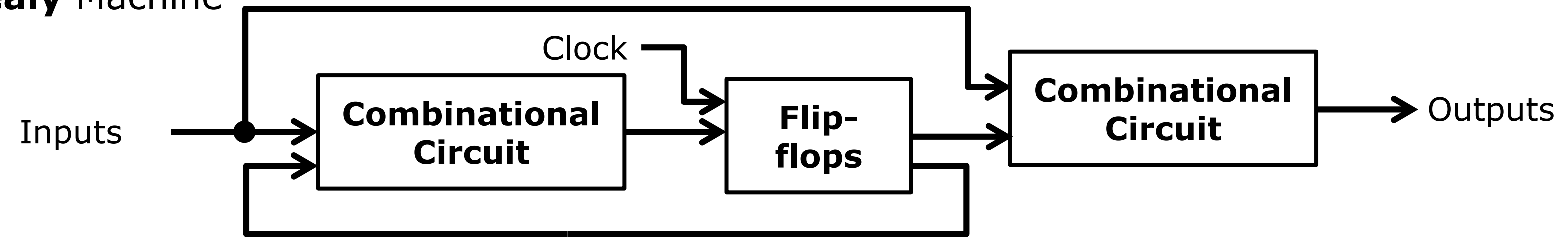
E.g. a vending machine accepting 5 cents and 10 cents coins to dispense a candy when it receives 15 cents in total. What can be the different states?

Types of State Machines

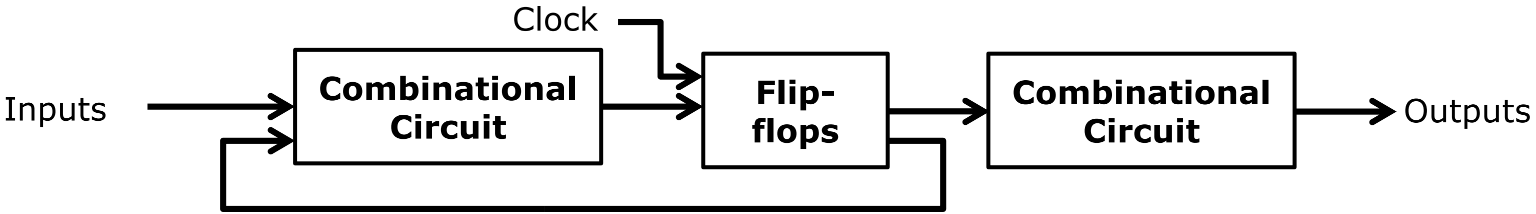
- Two types
  - Mealy machines
    - Outputs depend on current state and inputs
  - Moore machines
    - Outputs depend only on current state (flip-flop values)
      - Outputs can only change when state changes

Moore vs Mealy Machine

- **Mealy** Machine



- **Moore** Machine – special case of a Mealy Machine



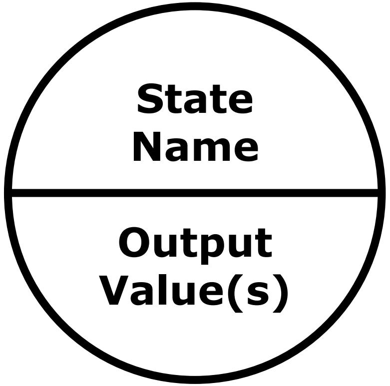
- We'll stick to Moore machines in this course

State Diagram

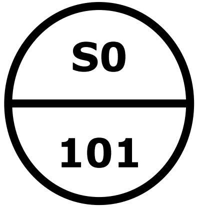
- To be illustrated in class

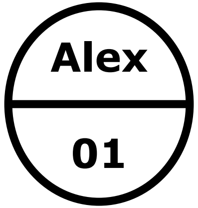
State Diagram (cont.)

- Notation
  - State



Examples:





  - Transition $\xrightarrow{\textbf{Logic expression}}$ or $\boxed{\xrightarrow{0} \quad \xrightarrow{1}}$ **Special case for only one input**
    - If expression is true, state transition is made
    - No label means "true" – i.e. always transition
    - Examples $\xrightarrow{Z}$ $\xrightarrow{\overline{RESET}}$ $\xrightarrow{A+B.\overline{C}}$ $\xrightarrow{\quad}$

Example – to be worked out in class

- Binary counter with 2-bit output

Example

- Binary up/down counter with 2-bit output
  - One input: U
    - 1 means count up
    - 0 means count down

Example

- Binary up/down counter with 2-bit output
  - Two Inputs: U,D
    - U=1, D=0 means count up
    - U=0, D=1 means count down
    - U=1, D=1 means set to 11
    - U=0, D=0 means reset to 00

- Binary up/down counter with 2-bit output
  - U=1, D=0 means count up
  - U=0, D=1 means count down
  - U=1, D=1 means set to 11
  - U=0, D=0 means reset to 00

Completeness

- Each possible combination of inputs should be addressed exactly once for each state
  - i.e. transition arrows from each state must encompass all possibilities (exactly once)
- Example:

State Table

- State diagrams can also be represented in a state table
- Example - binary up/down counter with 2-bit output and single input U (1 means up, 0 means down)
- One dimensional state table:

Current State	Input U	Next State	Outputs	
			Q1	Q0

Every combination of state and inputs

State Table (cont.)

- Two-dimensional state table
- Same example

Every combination of inputs

Current State	Next State		Outputs	
	$\overline{U}$	U	Q1	Q0

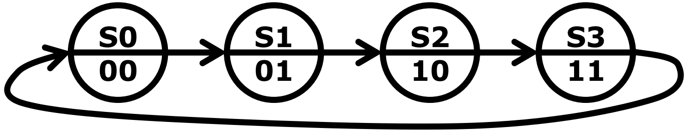
One row per state

State encoding

- Must encode each state into flip-flop values
- Choose
  - Number of flip-flops
  - Bit patterns that represent each state
- Ideally
  - Choose state encoding to make combinational logic simple, for both
    - Output logic
    - Next state logic

Example

- Binary counter with 2-bit output



One-hot coding

- Use one flip-flop per state
- Only one flip-flop has 1 at any time
- Example – binary counter with 2-bit output

Sequence Detector Example

- Design a state machine which detects the pattern 101 in an incoming bit stream and outputs a 1 whenever it detects this pattern
    - Output is 1 for one clock cycle after the third digit is clocked in
- Example:
    - Input: 0010111010100
    - Output: 00001000010100

A pre-recorded example will be provided on Blackboard