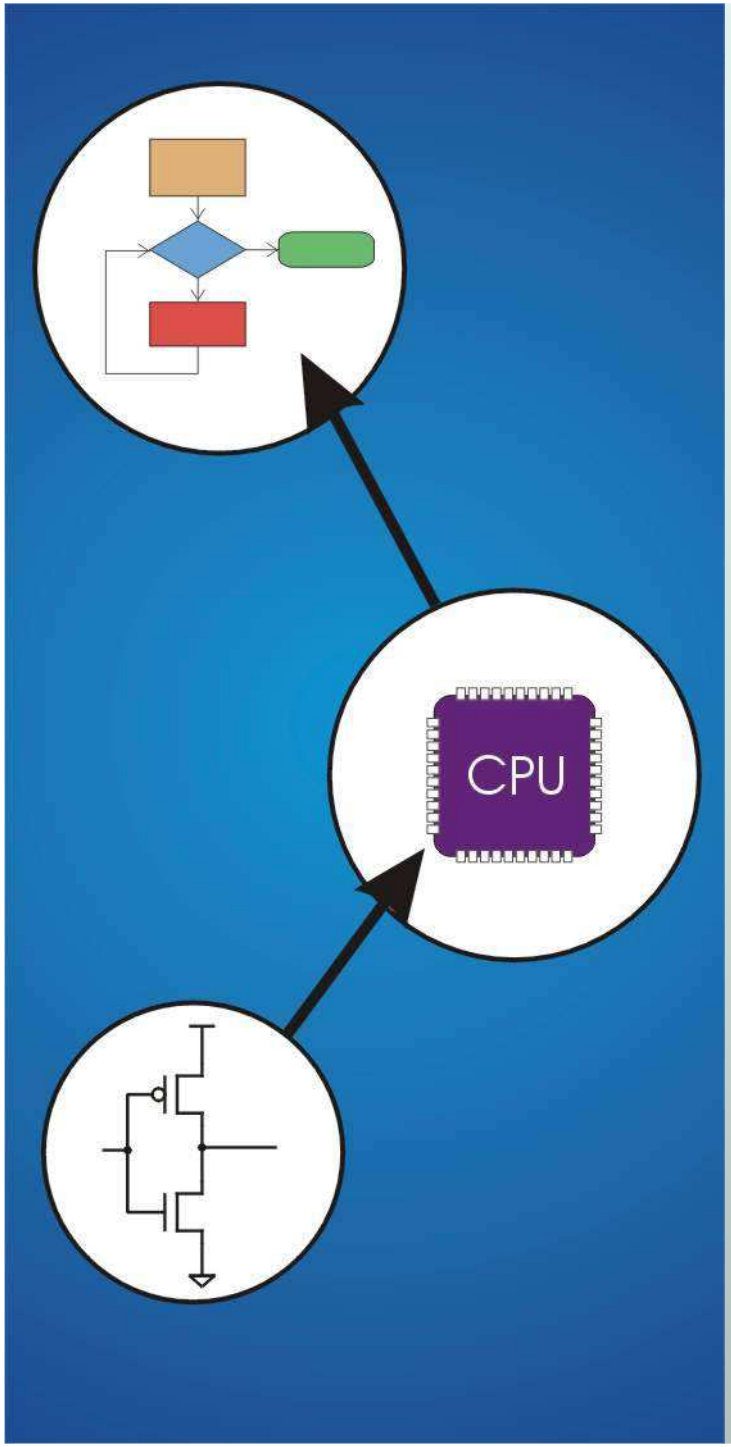
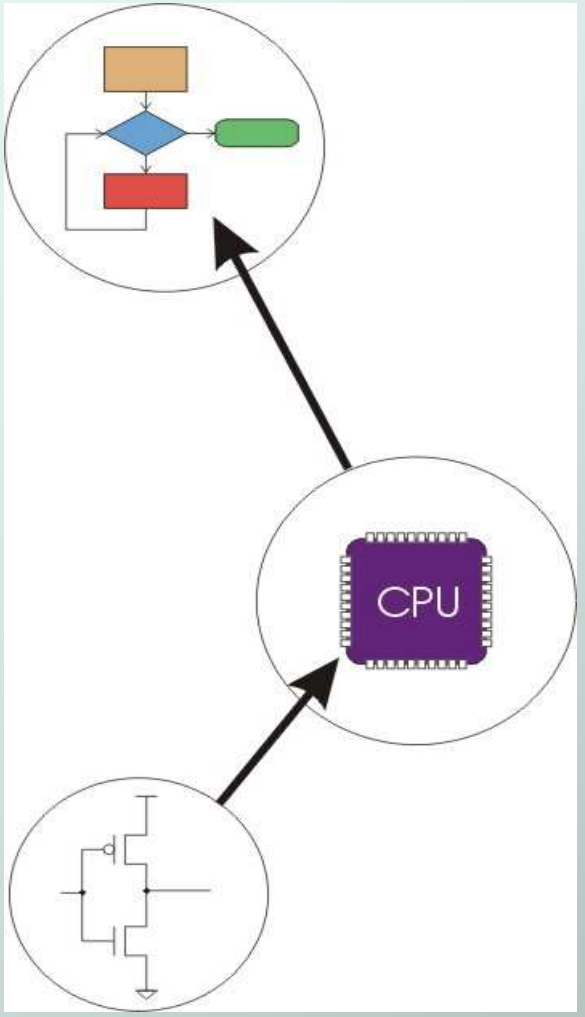


Chapter 3
Digital Logic Structures

Original slides from Gregory Byrd, North Carolina State University

Modified by C. Wilcox, M. Strout, Y. Malaiya
Colorado State University

Computing Layers



Problems

Algorithms

Language

Instruction Set Architecture

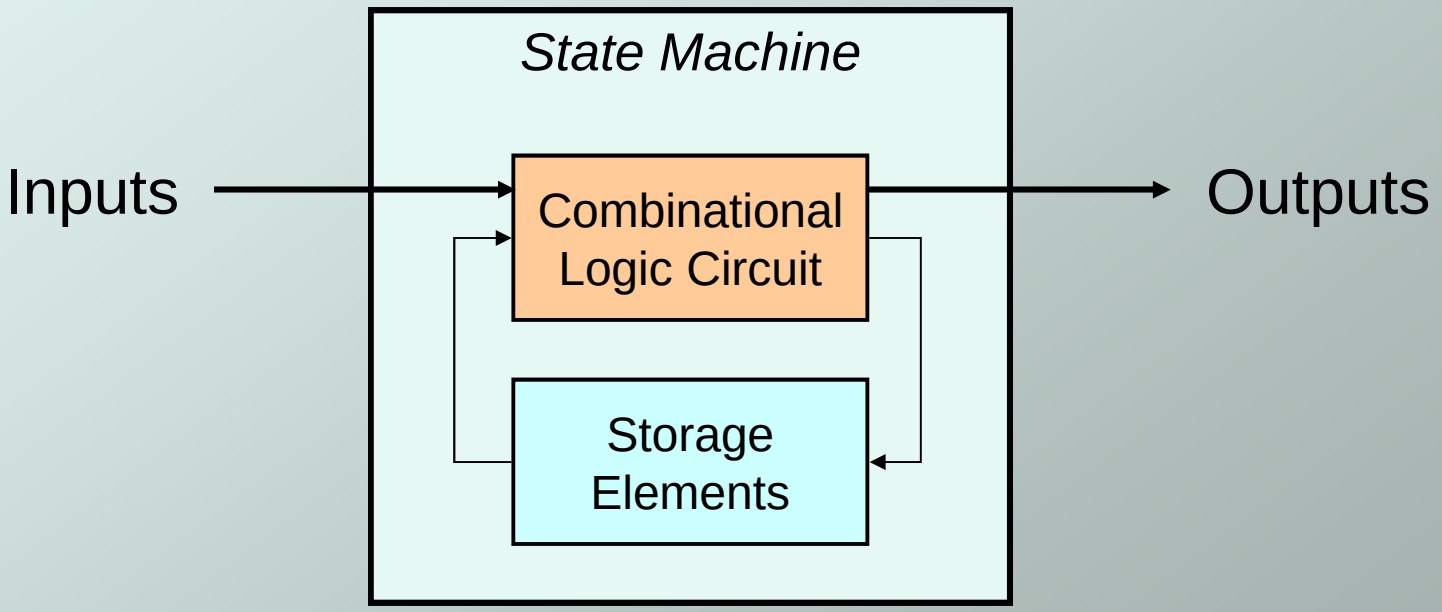
Microarchitecture

Circuits

Devices

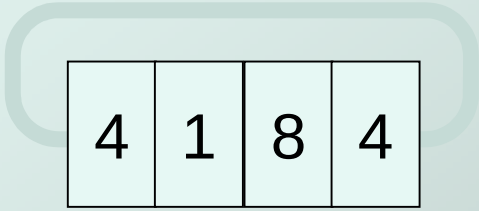
State Machine

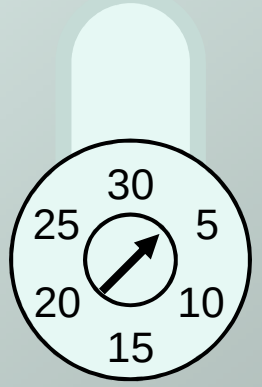
- Another type of sequential circuit
  - Combines combinational logic with storage
  - "Remembers" state, and changes output (and state) based on inputs and current state



Combinational vs. Sequential

- Two types of “combination” locks





Combinational
Success depends only on the values, not the order in which they are set.

Sequential
Success depends on the sequence of values (e.g, R-13, L-22, R-3).

State

- The state of a system is a snapshot of all the relevant elements of the system at the moment the snapshot is taken.
  Examples:
  - The state of a basketball game can be represented by the scoreboard: number of points, time remaining, possession, etc.
  - The state of a tic-tac-toe game can be represented by the placement of X's and O's on the board.

State of Sequential Lock

Our lock example has four different states, labelled A-D:

A: The lock is not open, and no relevant operations have been performed.

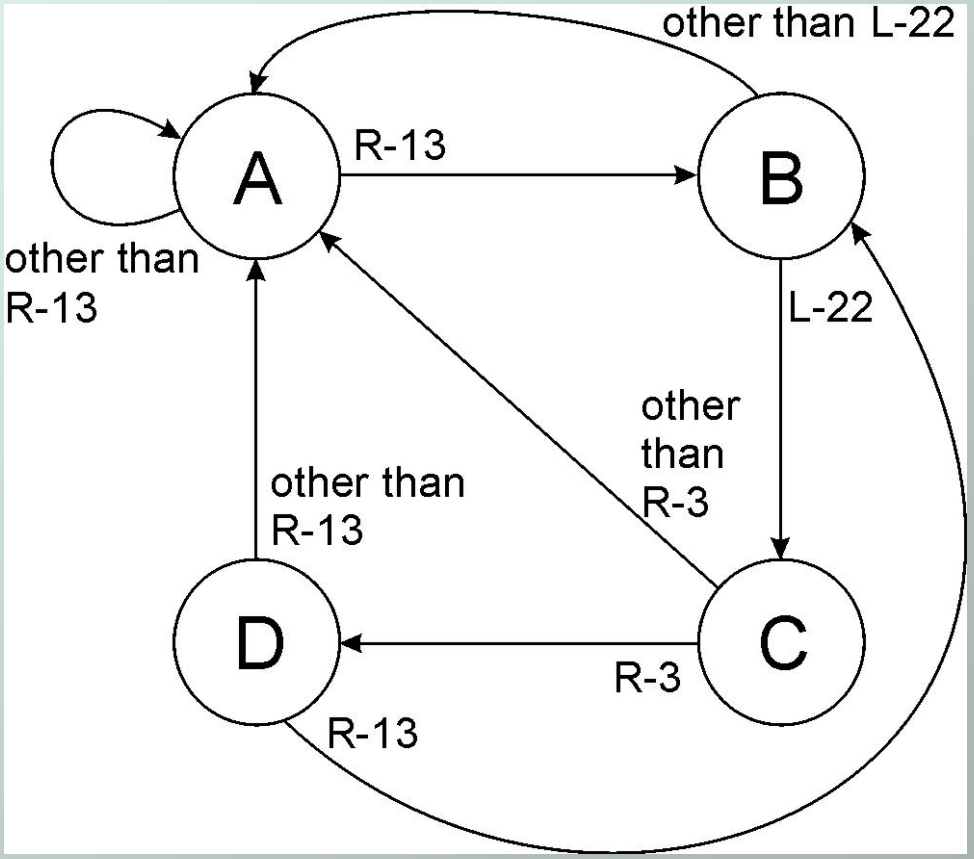
B: The lock is not open, and the user has completed the R-13 operation.

C: The lock is not open, and the user has completed R-13, followed by L-22.

D: The lock is open.

State Diagram

- Shows states and actions that cause a transition between states.



Finite State Machine

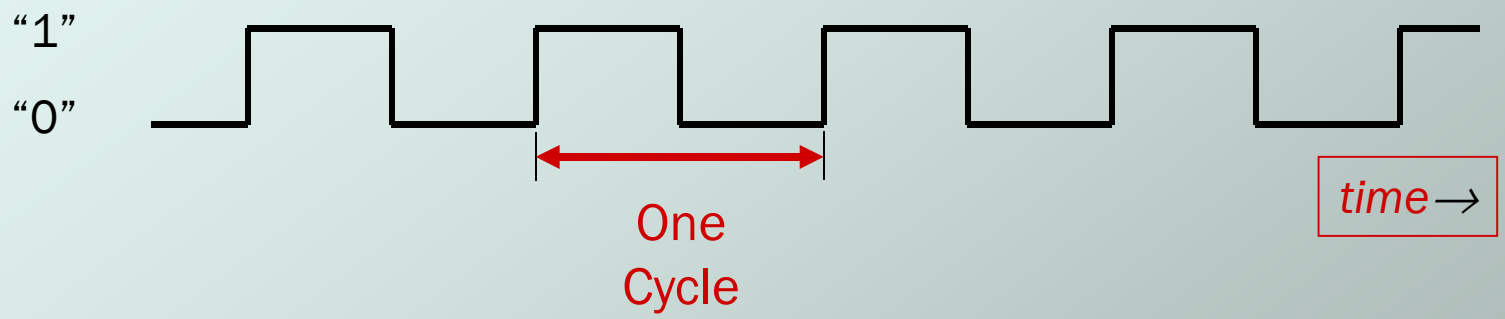
- A system with the following components:

1. A finite number of states
2. A finite number of external inputs
3. A finite number of external outputs
4. An explicit specification of all state transitions
5. An explicit specification of what determines each external output value

- Often described by a state diagram.
  - Inputs trigger state transitions.
  - Outputs are associated with each state (or with each transition).

The Clock

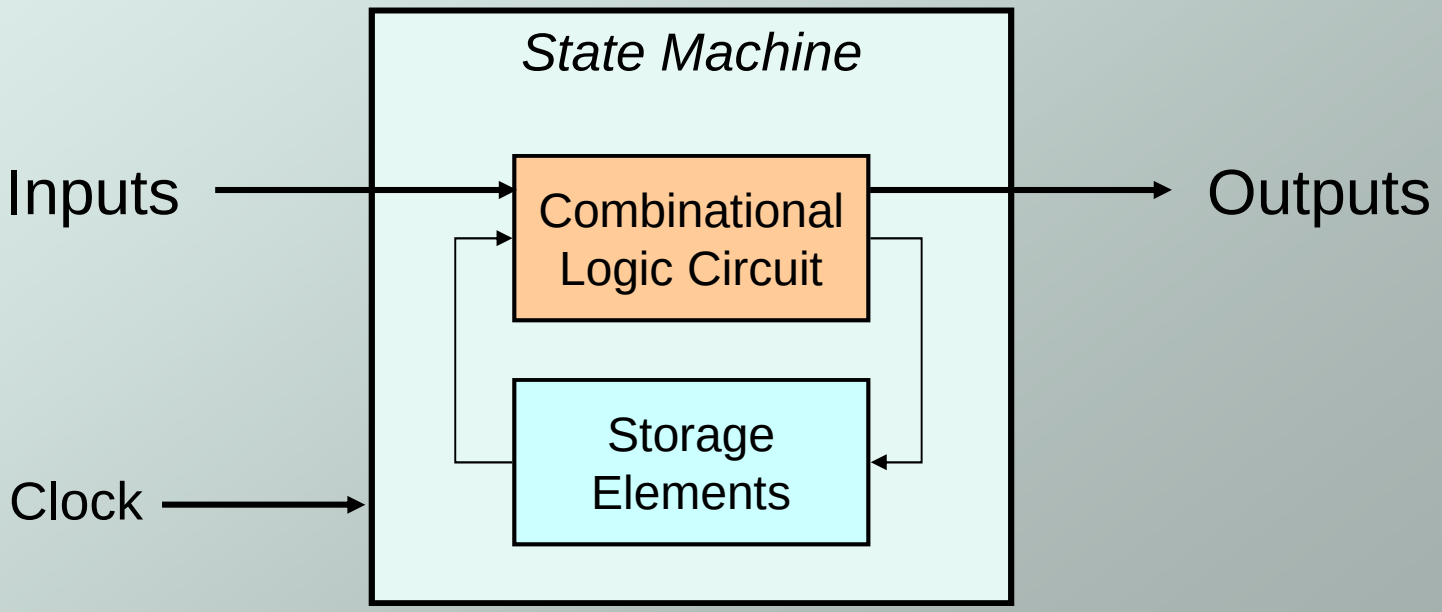
- Frequently, a clock circuit triggers transition from one state to the next.



- At the beginning of each clock cycle, state machine makes a transition, based on the current state and the external inputs.
  - **Not always required. In lock example, the input itself triggers a transition.**

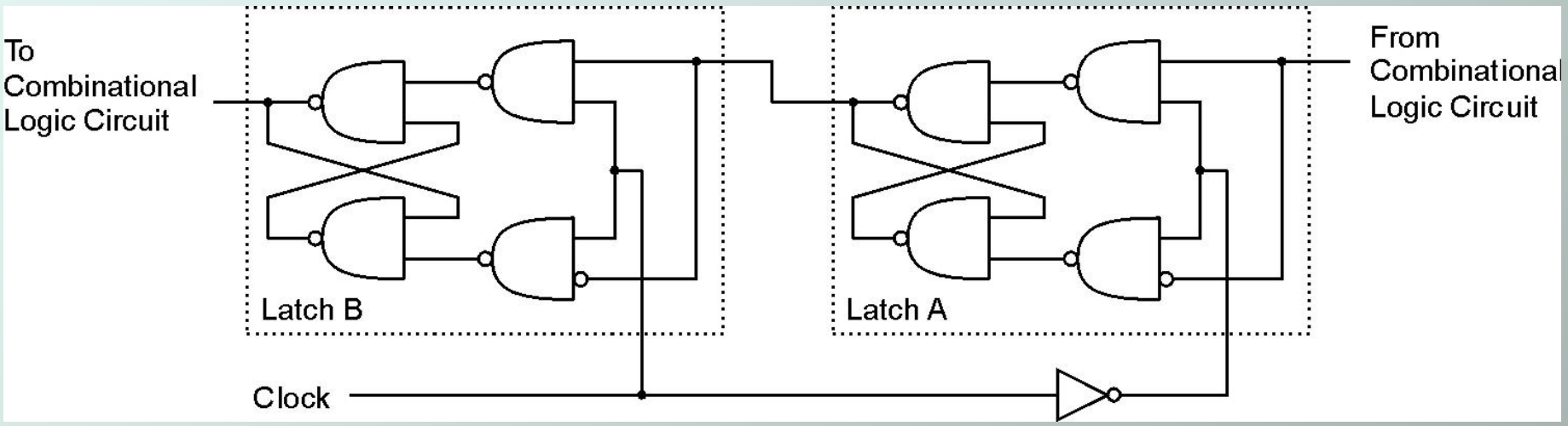
Implementing a Finite State Machine

- Combinational logic
  - Determine outputs and next state.
- Storage elements
  - Maintain state representation.



Storage: Master-Slave Flipflop

- A pair of gated D-latches,
  to isolate *next* state from *current* state.



During 1st phase (clock=1), previously-computed state becomes *current* state and is sent to the logic circuit.

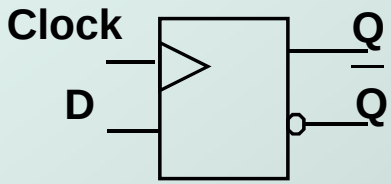
During 2nd phase (clock=0), *next* state, computed by logic circuit, is stored in Latch A.

Storage

- Each master-slave flipflop stores one state bit.
- The number of storage elements (flipflops) needed is determined by the number of states (and the representation of each state).
- Examples:
  - Sequential lock
    - Four states – two bits
  - Basketball scoreboard
    - 7 bits for each score, 5 bits for minutes, 6 bits for seconds,1 bit for possession arrow, 1 bit for half, …

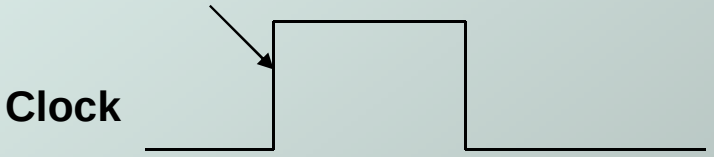
Flip-flops

- D Flip-flop: a storage element, can be edge-triggered (available in logisim)

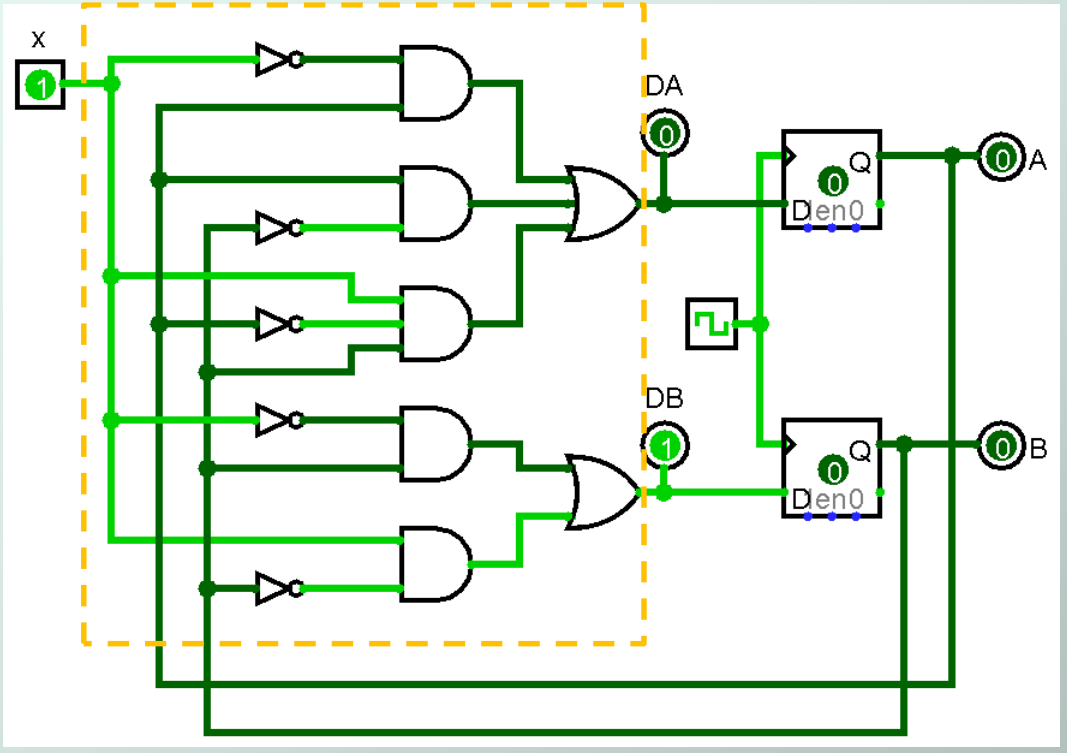


D	Next Q
0	0
1	1

Rising edge: input sampled



Analyze this FSM



Input: *x*
State: *A,B*
Output: *A, B*

Combinational block
In: *x, A, B* Out: *DA, DB*

$$DA = \overline{x}A + A\overline{B} + x\overline{A}B$$

$$DB = \overline{x}B + x\overline{B}$$

Analyze this FSM

$$DA = \overline{x}A + A\overline{B} + x\overline{A}B$$

$$DB = \overline{x}B + x\overline{B}$$

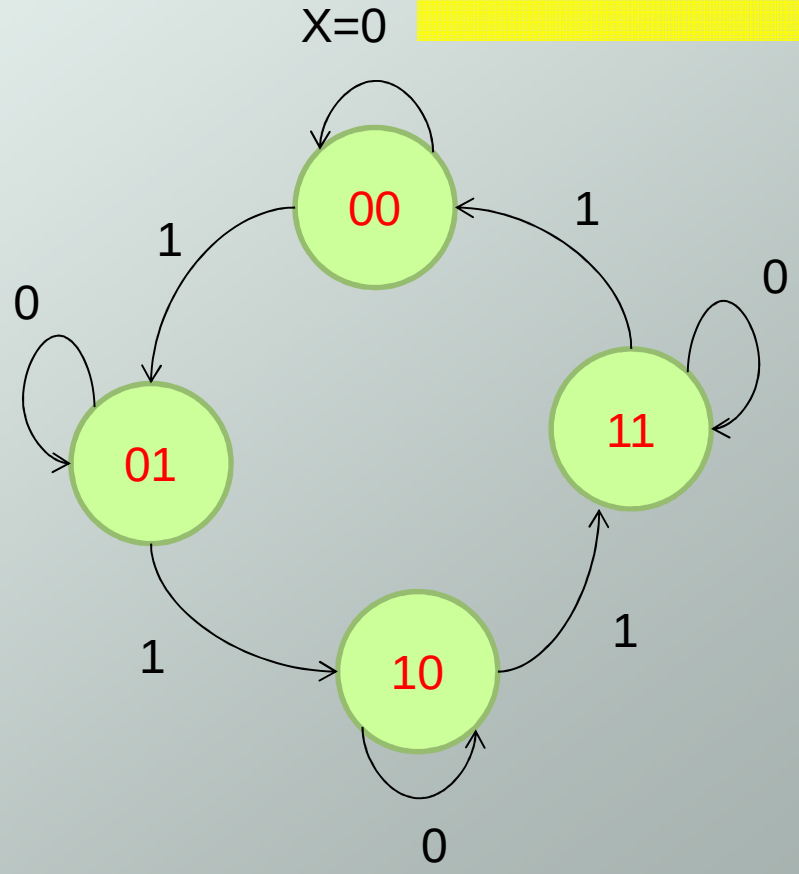
Input	Present State		Next State	
X	A	B	A	B
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

Analyze this FSM

State Table

Input	Present State		Next State	
X	A	B	A	B
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

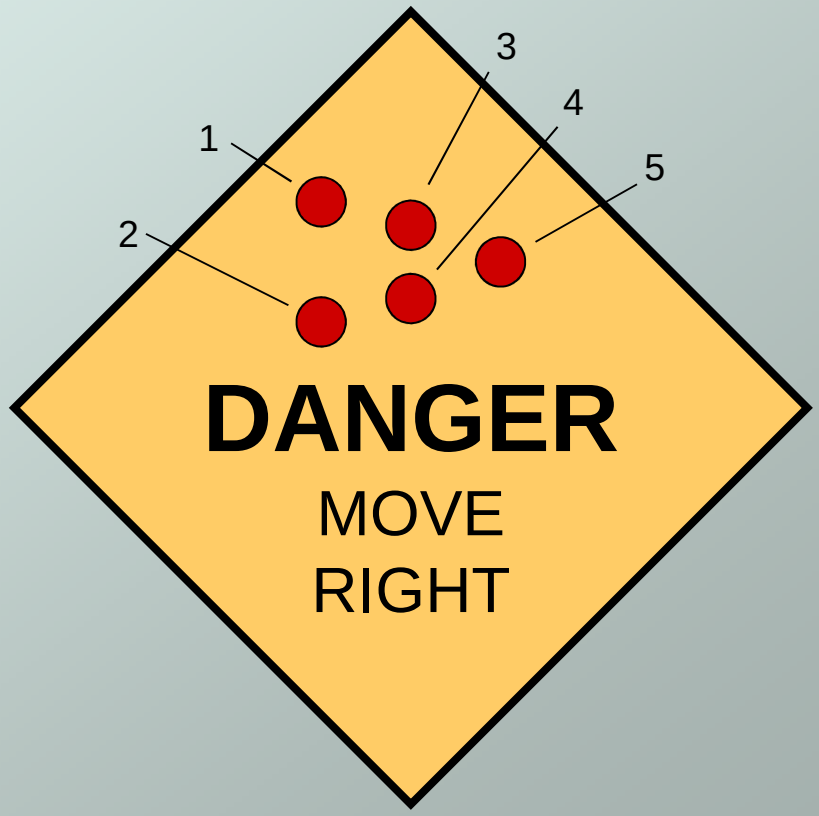
State Diagram



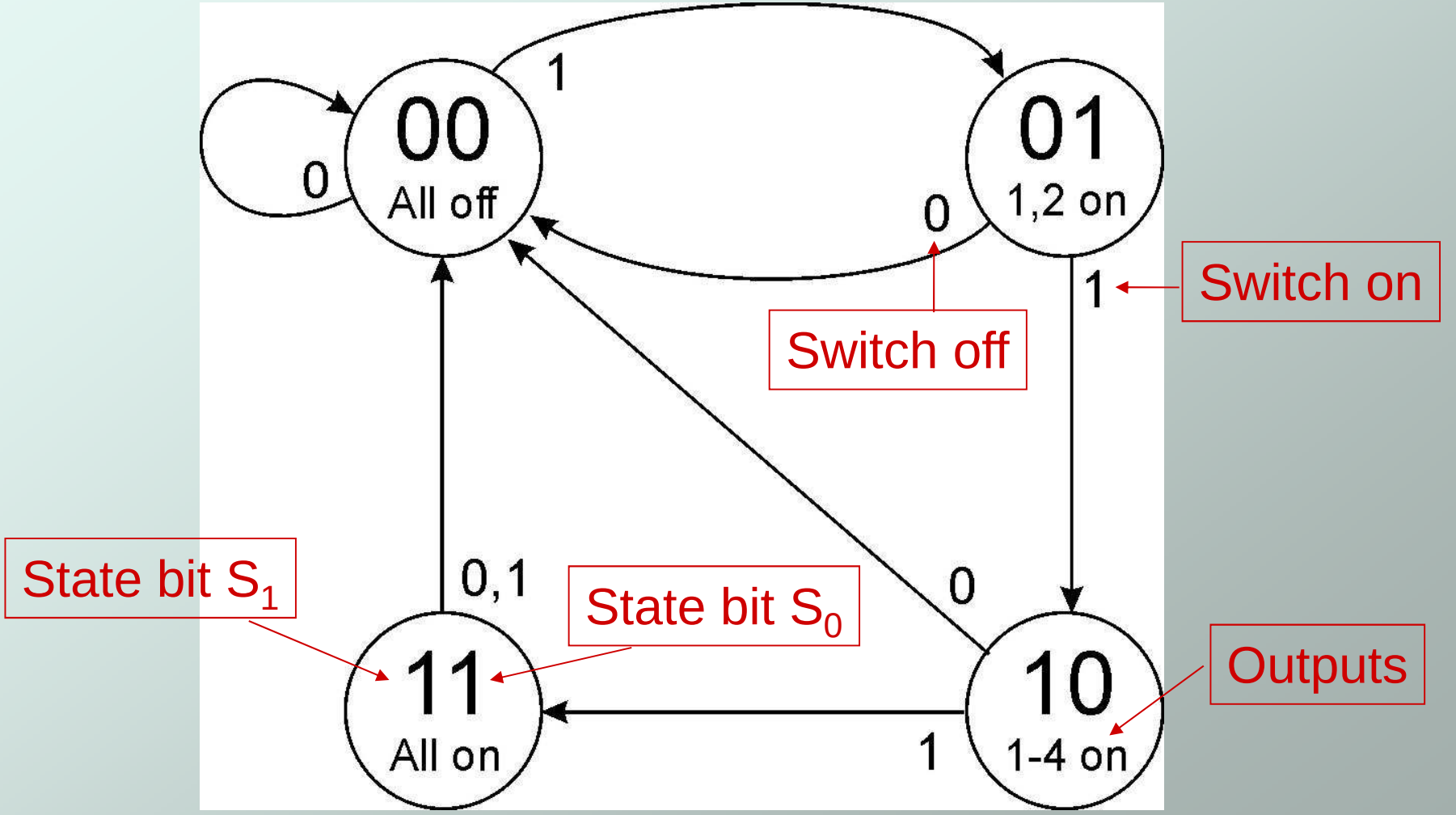
It is an up counter

Complete Example

- A blinking traffic sign
  - No lights on
  - 1 & 2 on
  - 1, 2, 3, & 4 on
  - 1, 2, 3, 4, & 5 on
  - (repeat as long as switch is turned on)



Traffic Sign State Diagram



Transition on each clock cycle.

Traffic Sign Truth Tables

Outputs (depend only on state: S_1S_0)

Lights 1 and 2
Lights 3 and 4
Light 5

S_1	S_0	Z	Y	X
0	0	0	0	0
0	1	1	0	0
1	0	1	1	0
1	1	1	1	1

Next State: $S_1'S_0'$ (depend on state and input)

Switch

In	S_1	S_0	S_1'	S_0'
0	X	X	0	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

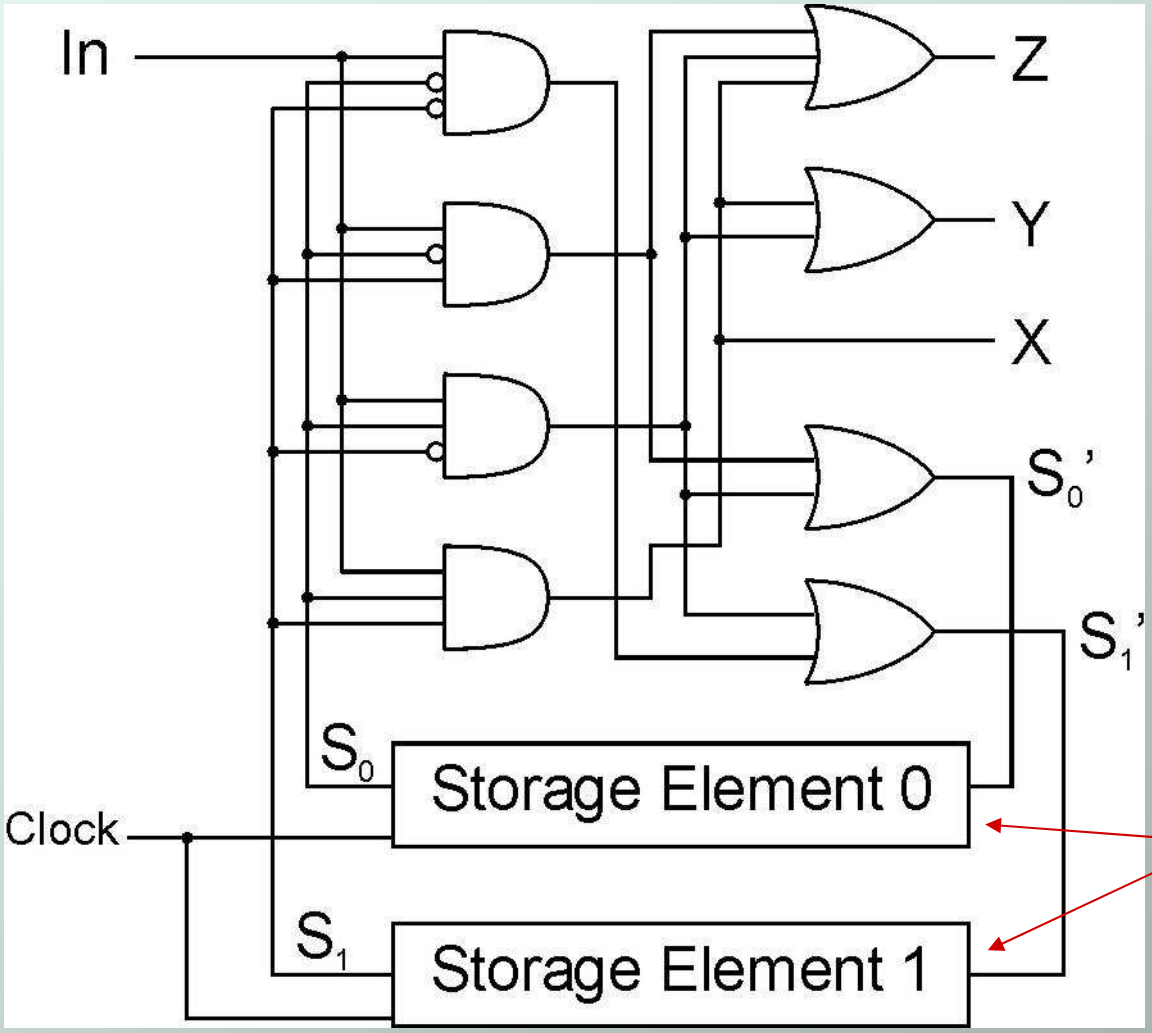
Whenever In=0, next state

State Table

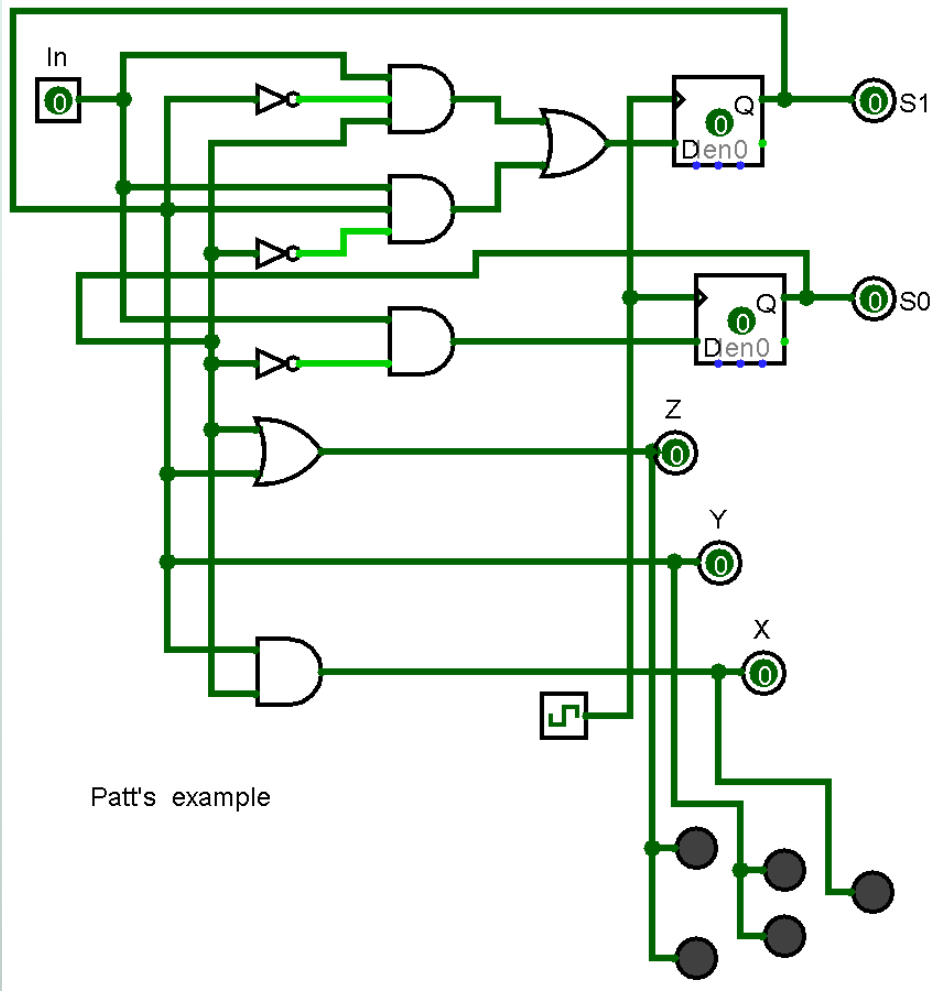
In	Pr State		Nx State		Outputs		
In	S1	S0	S1	S0	Z	Y	X
0	0	0	0	0	0	0	0
0	0	1	0	1	1	0	0
0	1	0	1	0	1	1	0
0	1	1	1	1	1	1	1
1	0	0	0	1	0	0	0
1	0	1	1	0	1	0	0
1	1	0	1	1	1	1	0
1	1	1	0	0	1	1	1

Comb Ckt:
Input: In
Output: S1', S0'
Z, Y, X

Traffic Sign Logic



Traffic Sign Logic: Optimal Design



From Logic to Data Path

- The data path of a computer is all the logic used to process information.
  - See the data path of the LC-3 on next slide.
- Combinational Logic
  - Decoders -- convert instructions into control signals
  - Multiplexers -- select inputs and outputs
  - ALU (Arithmetic and Logic Unit) -- operations on data
- Sequential Logic
  - State machine -- coordinate control signals and data movement
  - Registers and latches -- storage elements

LC-3 Data Path

Combinational Logic

Storage

State Machine