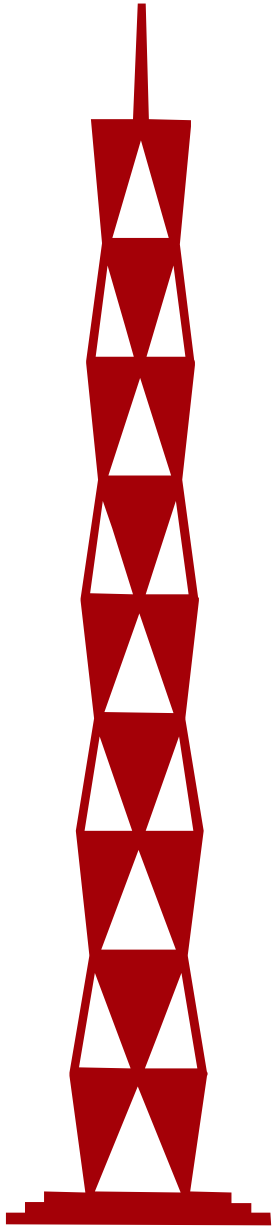


SI100B Introduction to Information Science and Technology (Part 3: Electrical Engineering)

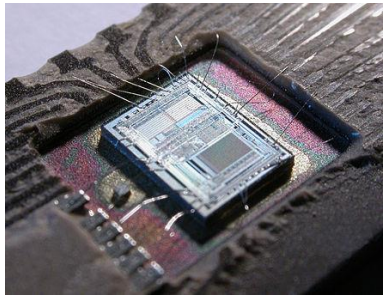
Lecture #5 (Digital) Sequential Logic Circuits #2

Instructor: Haoyu Wang(王浩宇)

Apr 18th, 2023

The Theme Story

Devices (1)



Circuits (3/5)

Systems (1)

(Figures from Internet)

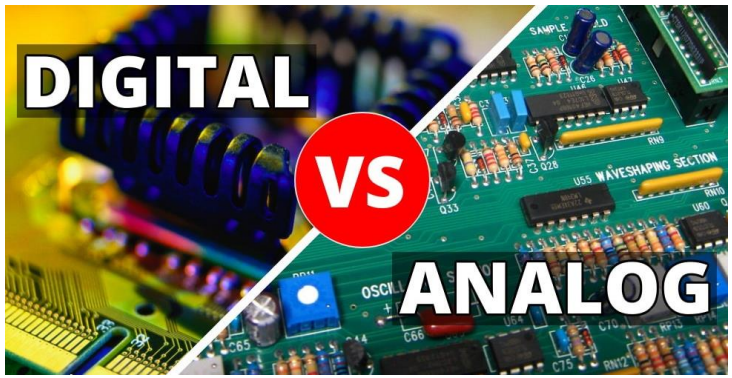
Study Purpose of Lecture #5

- 哲学三问
  - Who are you?
  - Where are you from?
  - Where are you going?

To answer those questions throughout your life

(Figures from Internet)

- In this lecture, we ask
  - What is synchronous sequential circuit?
  - How do we realize memory array?
  - How to use finite state machine to build useful applications?

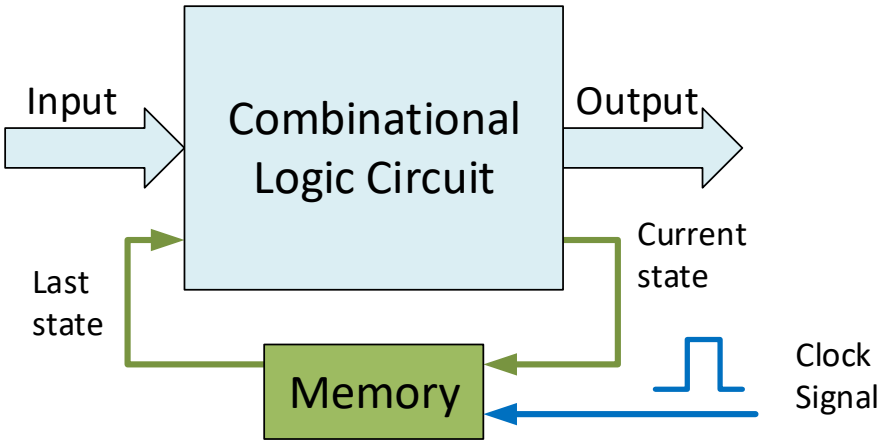


Lecture Outline

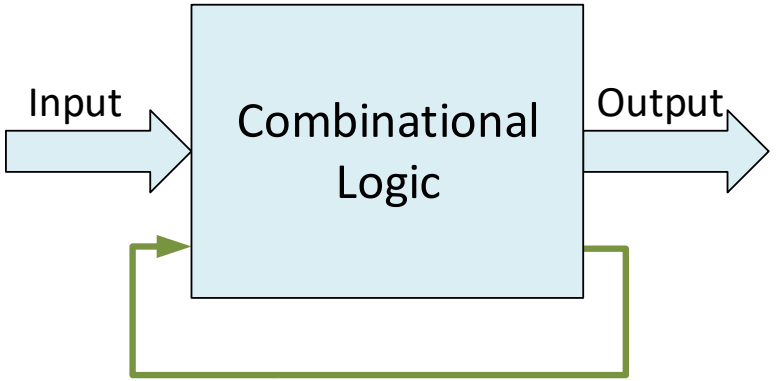
- Synchronous sequential circuit
  - The evolution of flip-flops
  - Synchronous and asynchronous circuits
  - Example: 4 x 3 memory
- Memory Array
- Finite state machines (FSM)
  - Traffic light example

Synchronous and asynchronous circuits

- Synchronous circuit

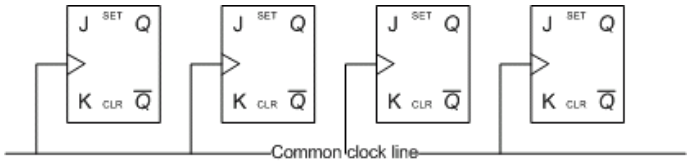


- Asynchronous circuit or self-timed circuit



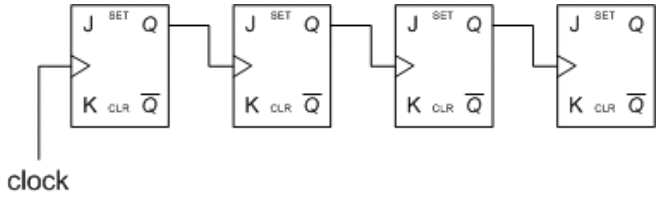
Synchronous and asynchronous circuits

- Synchronous circuit



  - Common clock signal
  - Output only change at the edge of clock pulse
  - clock signal should be long enough so that the critical path can settle before next clock edge
  - Easy design

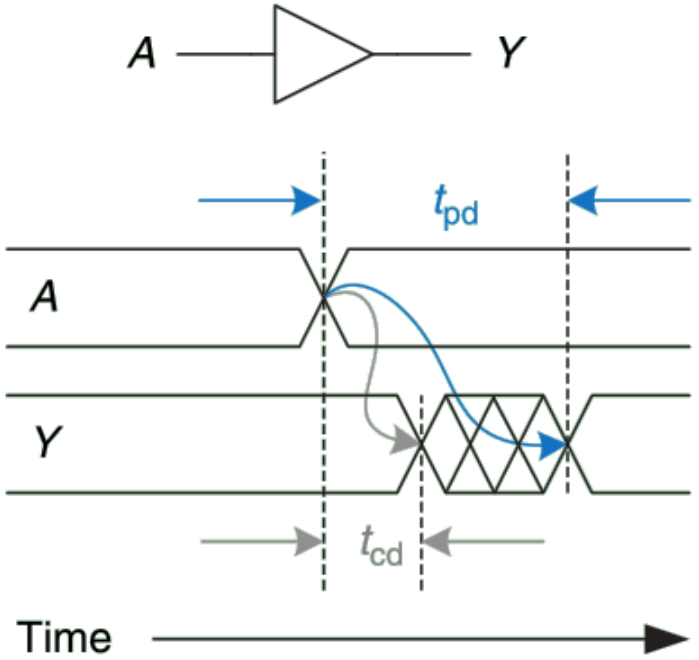
- Asynchronous circuit or self-timed circuit



  - Not governed by global clock
  - Resulting state can be sensitive to the relative arrival times of inputs at gates, the race condition

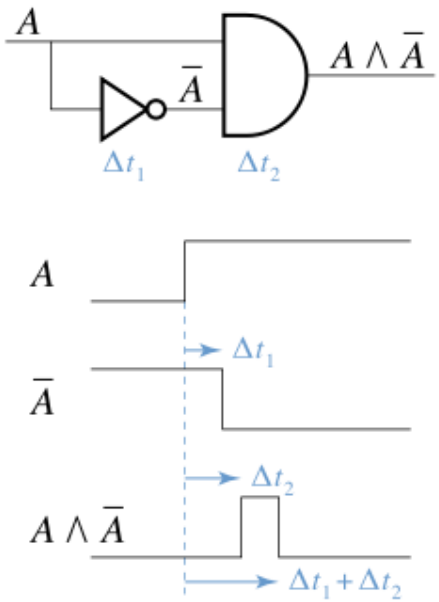
Delay in combinational circuit

- contamination delay t_{cd}
  - Y (output) starts to change after the change of A (input)
- propagation delay t_{pd}
  - Y (output) definitely settles in new value



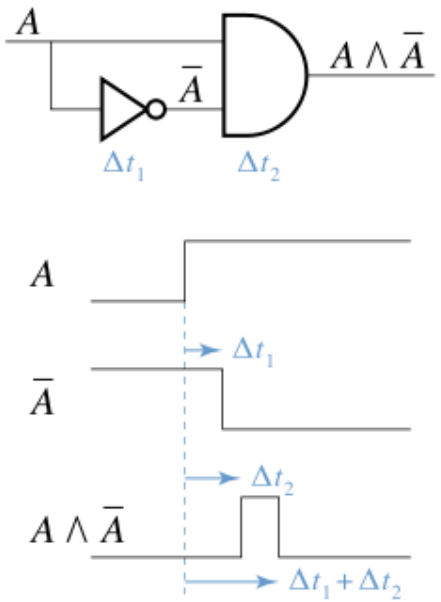
Racing condition

- In combinational circuit



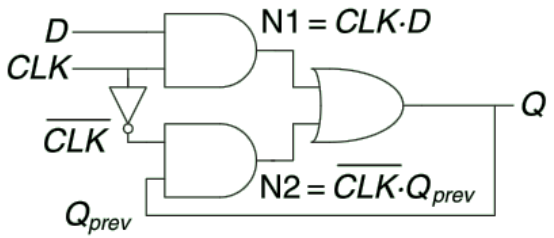
Racing condition

- In combinational circuit

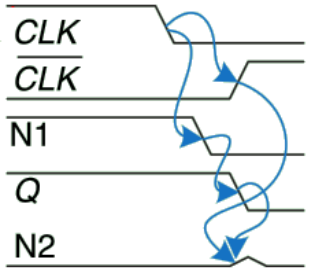


- In asynchronous sequential circuit

$Q = CLK \cdot D + \overline{CLK} \cdot Q_{prev}$



- D = 1
  CLK = 1; $\overline{CLK}$ = 0
  Q_{prev} = 1*1 + 0*X = 1
- CLK = 1 → 0 ; $\overline{CLK}$ = 1
  Q = 1*0 + 1*1 = 1
- Suppose the delay through the inverter from CLK to CLK' is rather long compared to the delays of the AND and OR gates
- eventually Q=0 because of the race condition



Rules of synchronous sequential circuit

- Every circuit element is either a **register** or a **combinational** circuit
- At least one circuit element is a **register**
- All registers receive the **same clock signal**
- Every cyclic path contains **at least one register**

Example: 2 x 2 memory

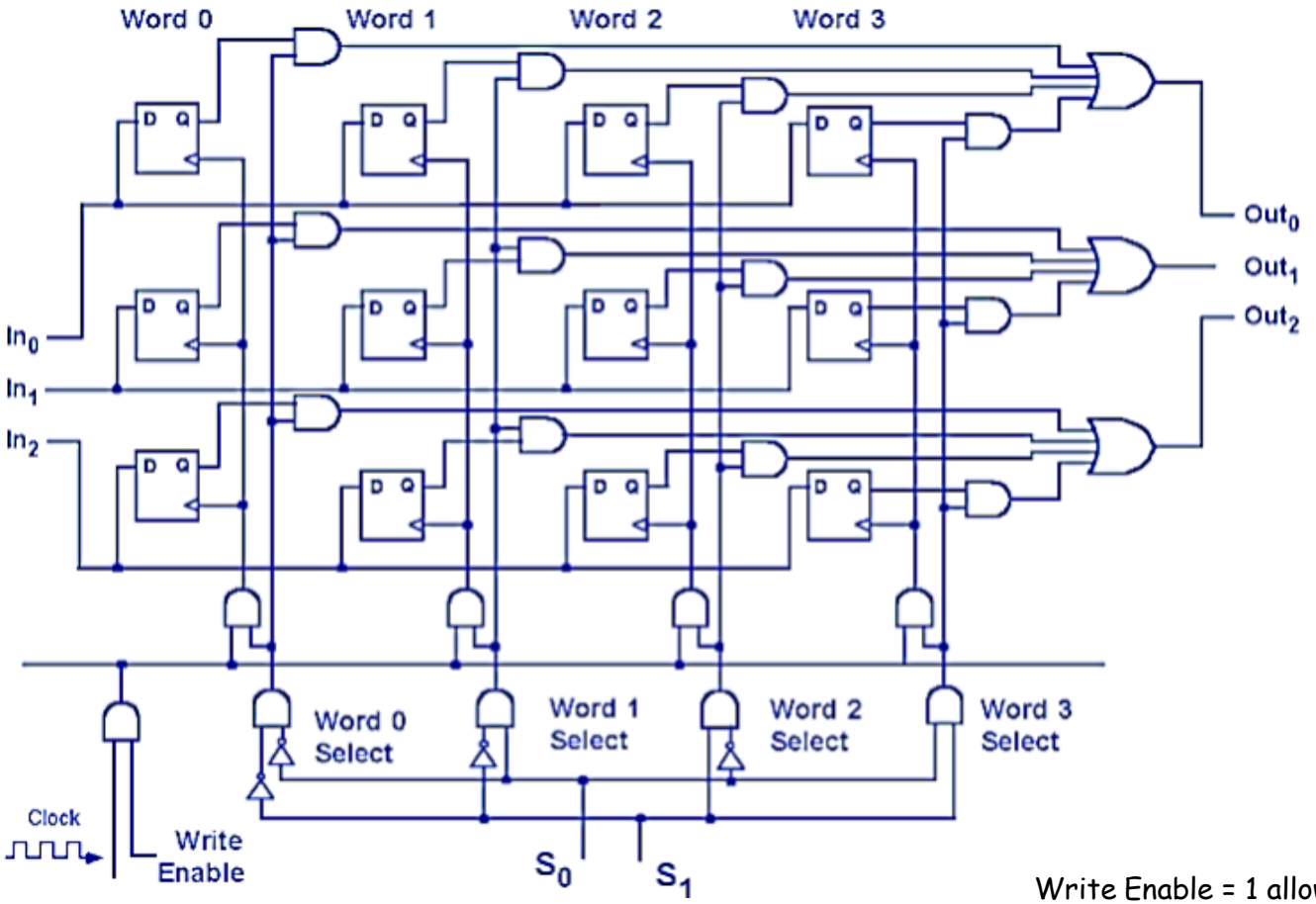
- **2 words** and each can store **2 bits** of information
- To represent **2 words** need **1-bits** for address
  - Address decoder performs the address decoding
- To store information we use **D Flip-Flop** for each bit (total 4, as each location as **2 bits** and we have **2 total locations**)
- Need select variable to chose if we want to read or write data and that is combined with clock signal (using some combinational logic)
- Also need some other combinational logic…

Example: 2 x 2 memory

Example: 4 x 3 memory

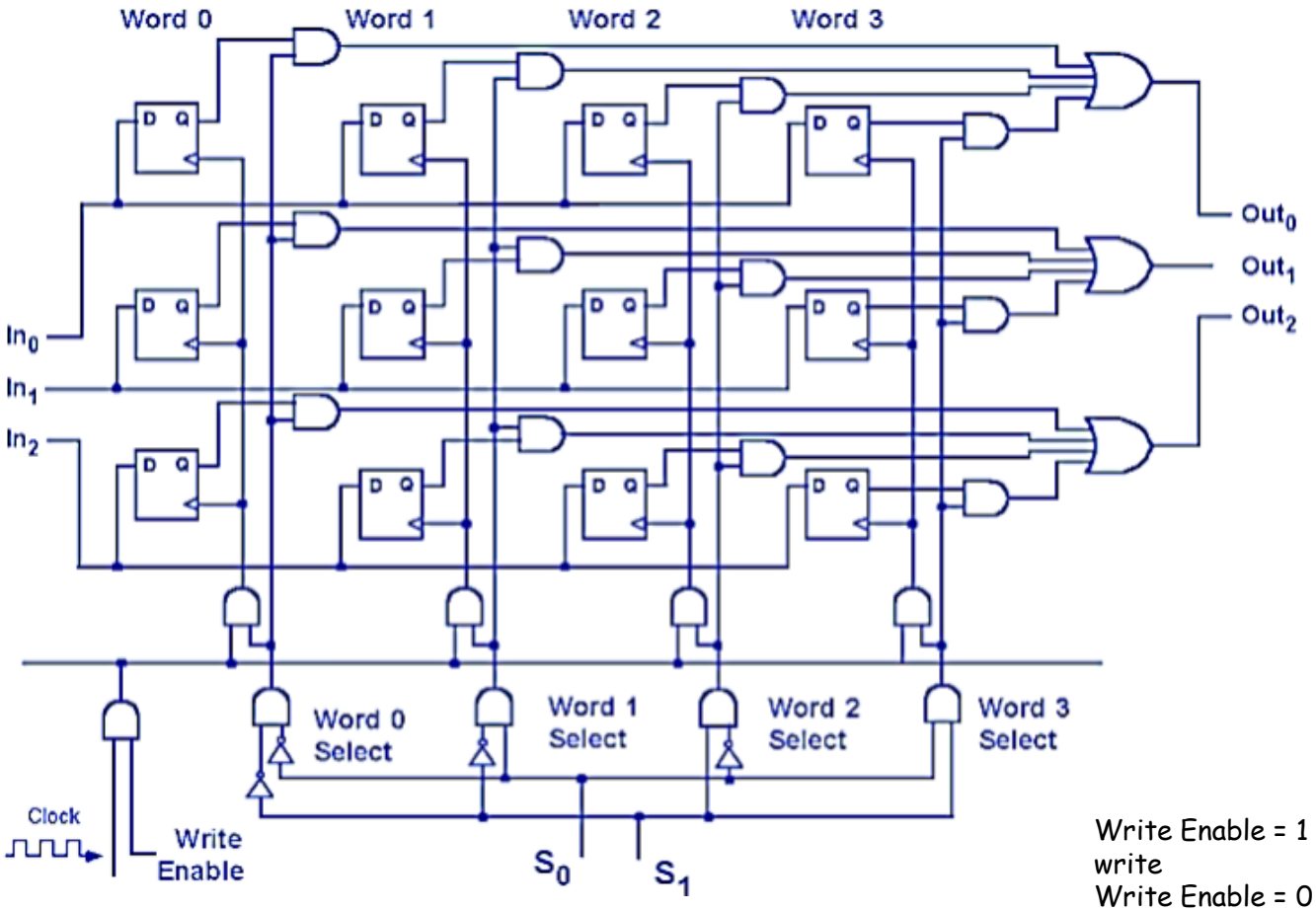
- **4 words** and each can store **3 bits** of information
- To represent **4 words** need **2-bits** for address
  - Address decoder performs the address decoding
- To store information we use **D Flip-Flop** for each bit (total 12, as each location as 3 bits and we have 4 total locations)
- Need select variable to chose if we want to read or write data and that is combined with clock signal (using some combinational logic)
- Also need some other combinational logic…

Example: 4 x 3 memory



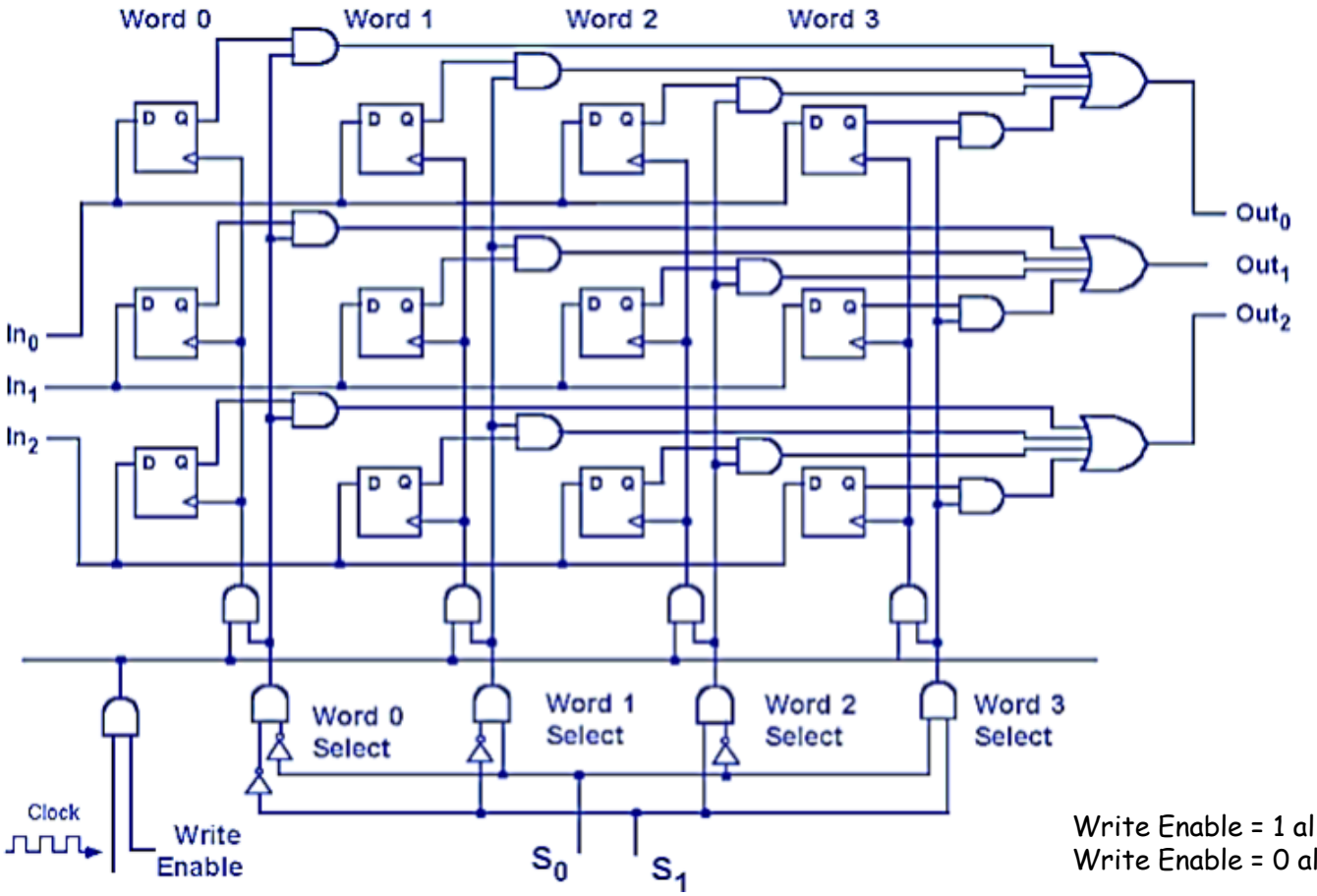
Write Enable = 1 allows memory write
Write Enable = 0 allows memory read

Write operation to word 1



Write Enable = 1 allows memory write
Write Enable = 0 allows memory read

Read operation from word 3

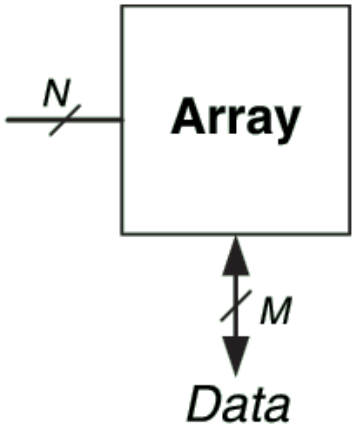


Write Enable = 1 allows memory write
Write Enable = 0 allows memory read

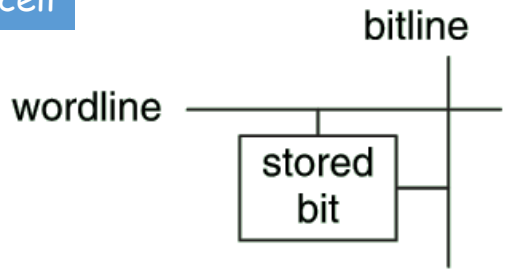
Memory array

- Two-dimensional array of memory cells
- The row is specified as *Address*
- Each row of data is called a *Word*
- The array contains 2^N *M-bit* words
- *Depth* = 2^N
- *Width* = 2^M
- Types:
  - Dynamic random access memory (DRAM)
  - Static random access memory (SRAM)
  - Read only memory (ROM)
  - etc.

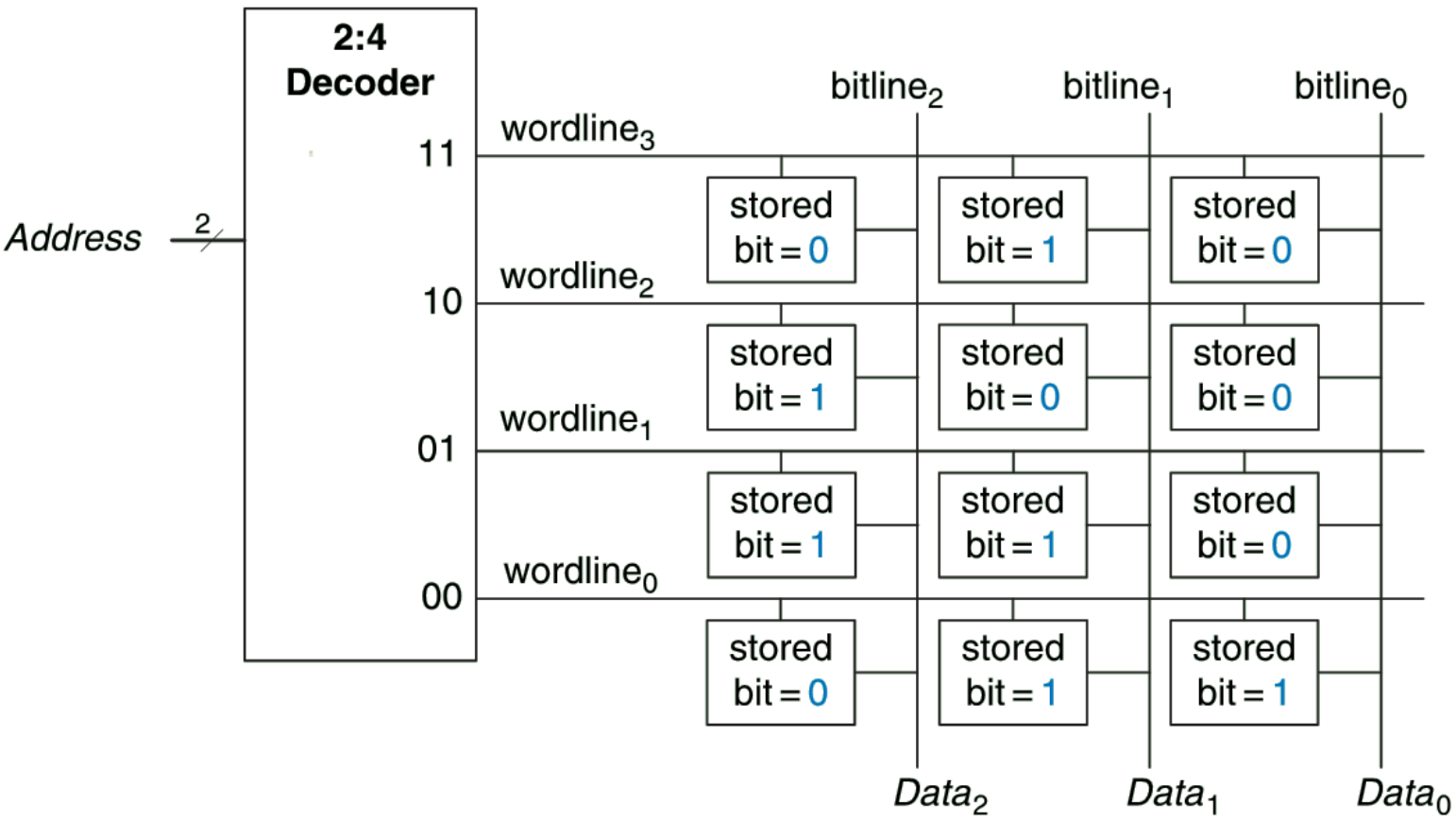
General symbol



Bit cell

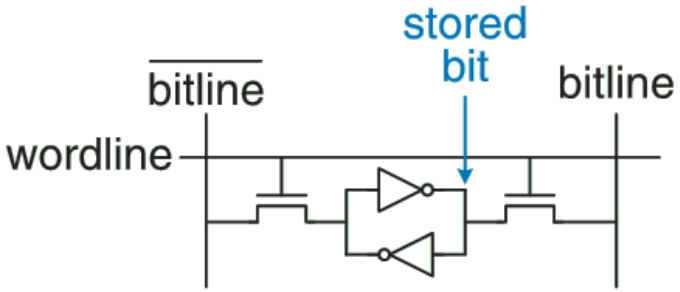


Example: 4 × 3 Memory Array

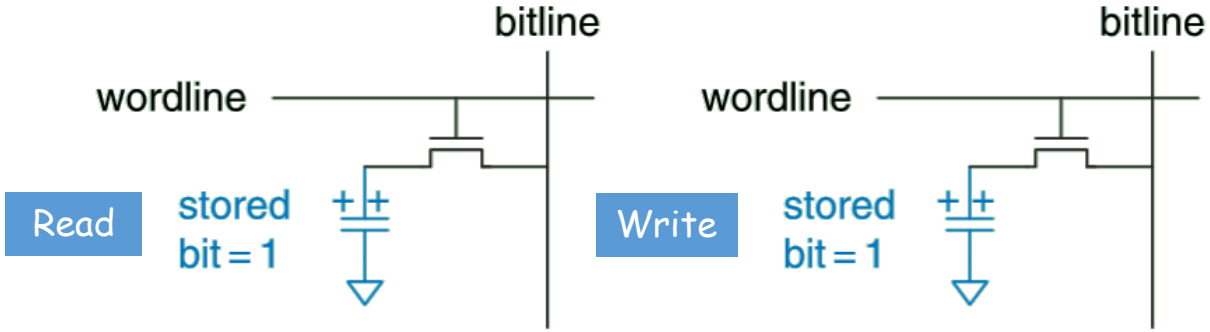


Volatile memory

- Static Random Access Memory (SRAM)

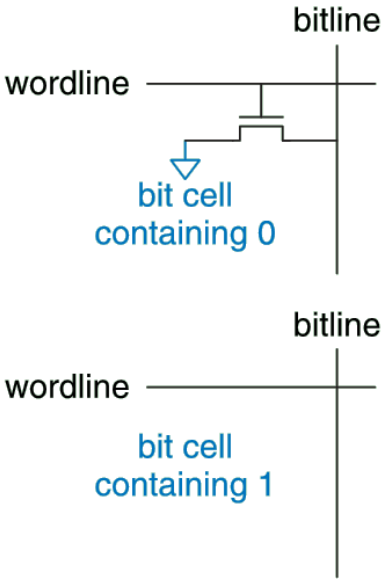


- Dynamic Random Access Memory (DRAM)

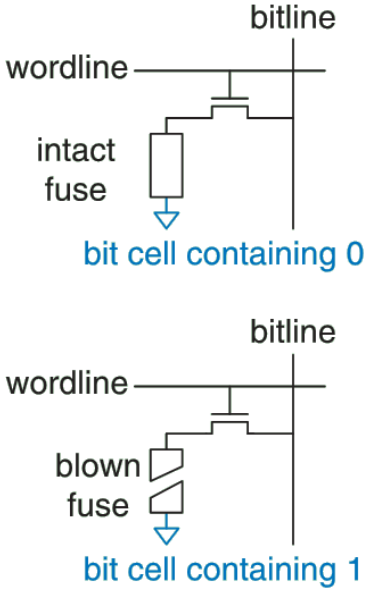


Nonvolatile memory

- Read only memory (ROM)



ROM bit cells



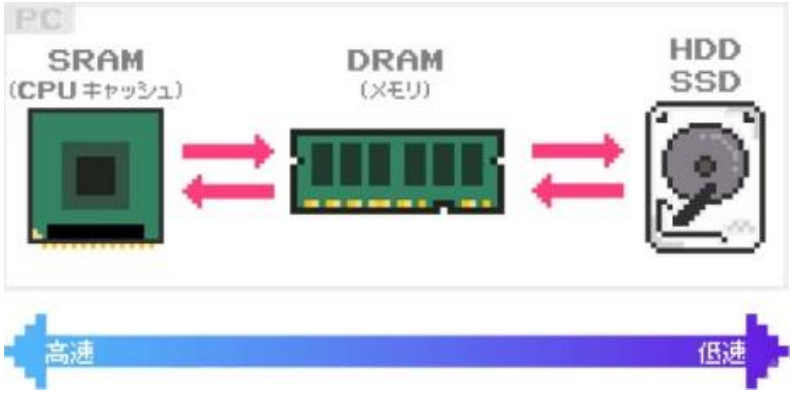
Programmable ROM

- Modern ROMs can be programmed (written) as well. For example flash memory
- Generally, ROMs take a longer time to write than RAMs, but are nonvolatile.

Memory comparison

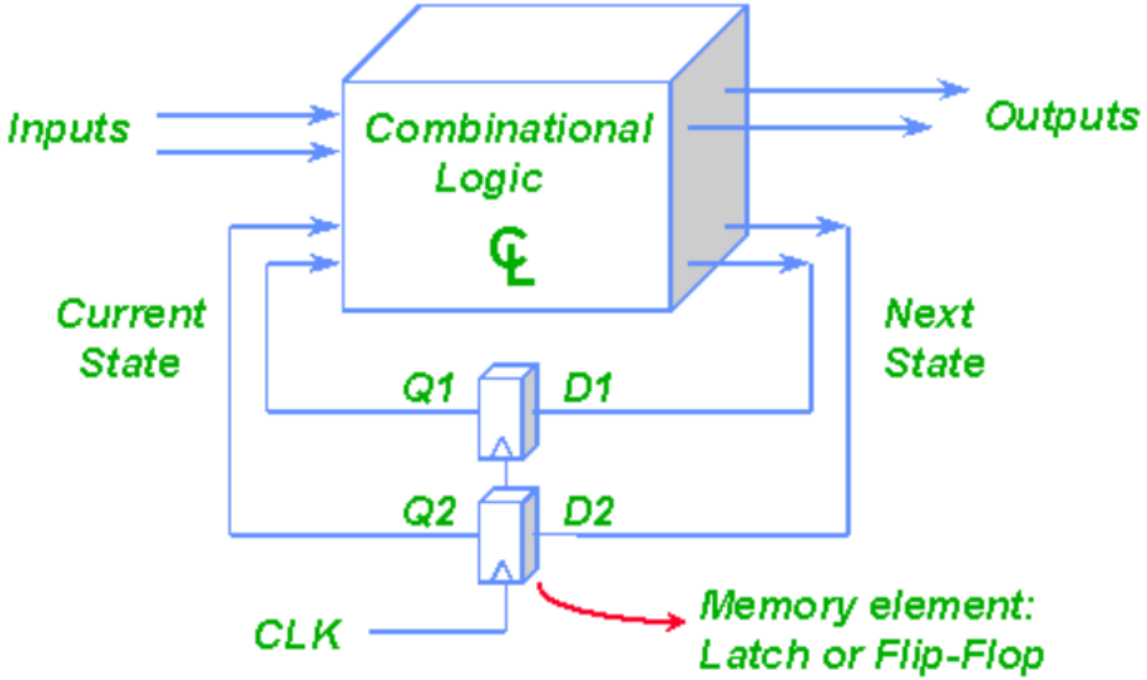
- The best memory type for a particular design depends on the *speed*, *cost*, and *power constraints*.

Memory Type	Transistors per Bit Cell	Latency
flip-flop	~20	fast
SRAM (register, cache)	6	medium
DRAM (main memory)	1	slow

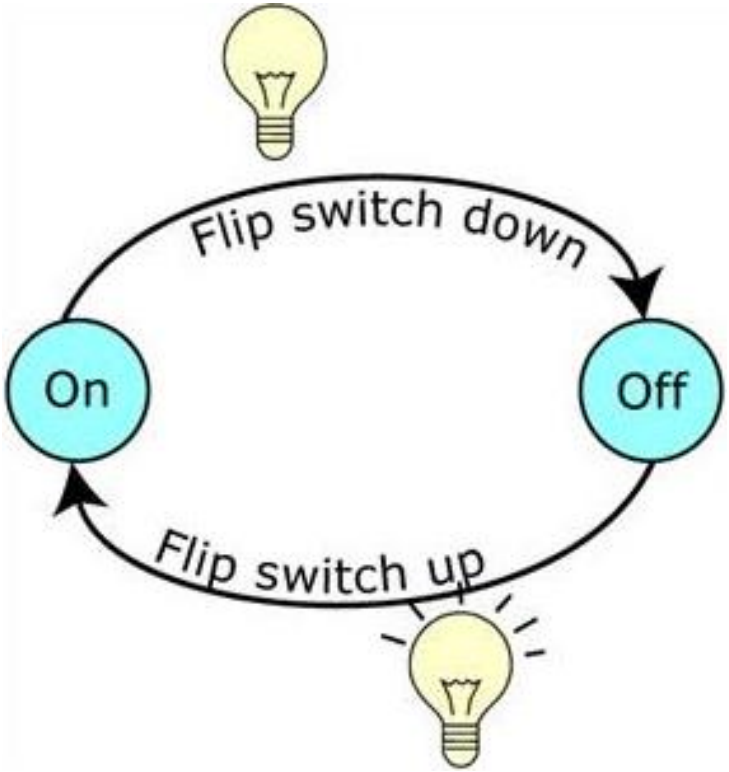


Finite-state machine

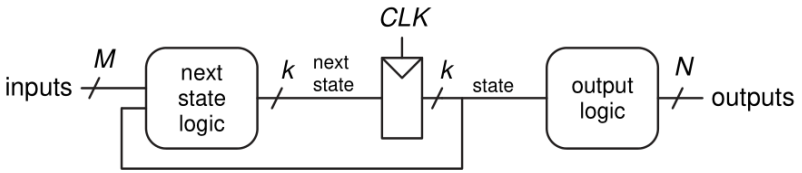
- FSM - a mathematical model of computation used to design both computer programs and sequential logic circuits



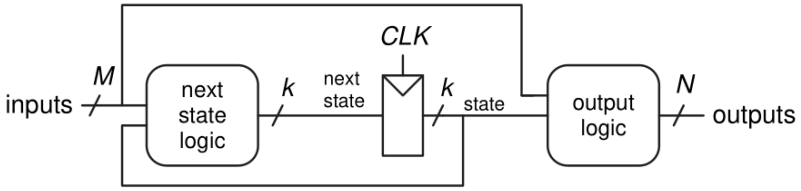
The most seen FSM



Moore machine

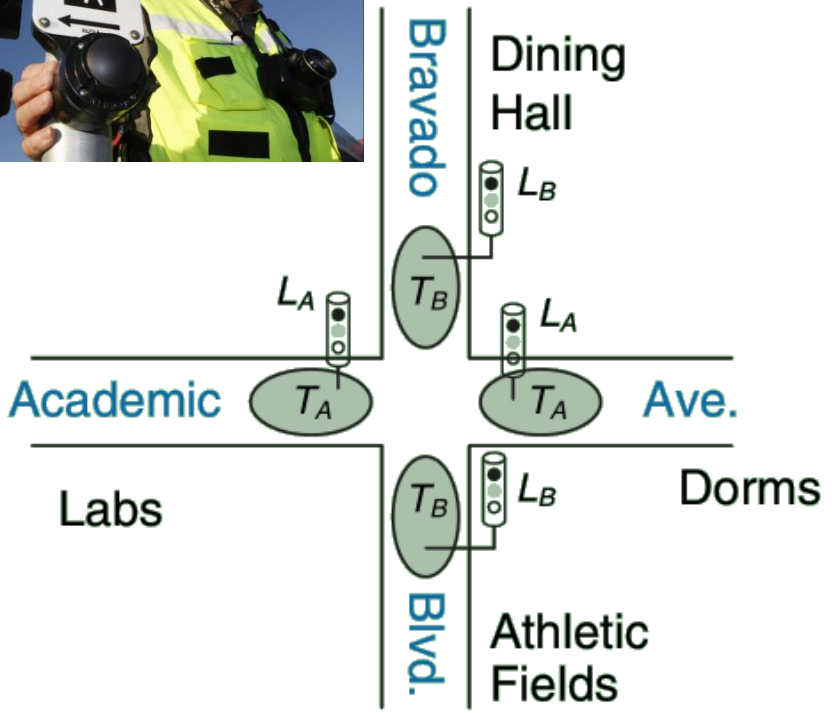


Mealy machine

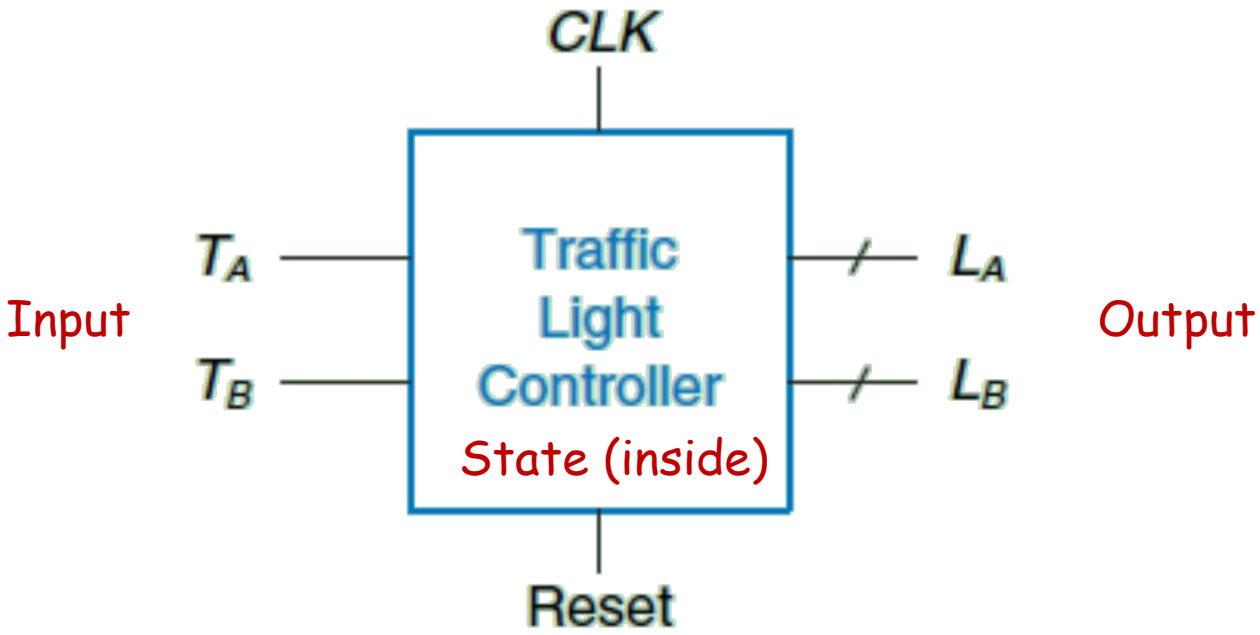


Example: traffic light controller

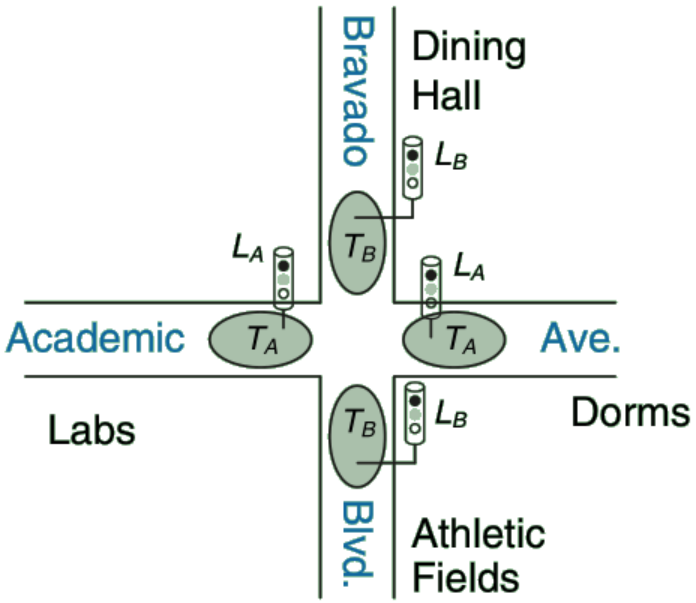
- Traffic sensors T_A and T_B ("1" busy, "0" empty)
- Traffic lights L_A and L_B (each light have red, yellow, and green)
- 5-second clock, at each rising edge, lights may change based on the sensors
- Reset button

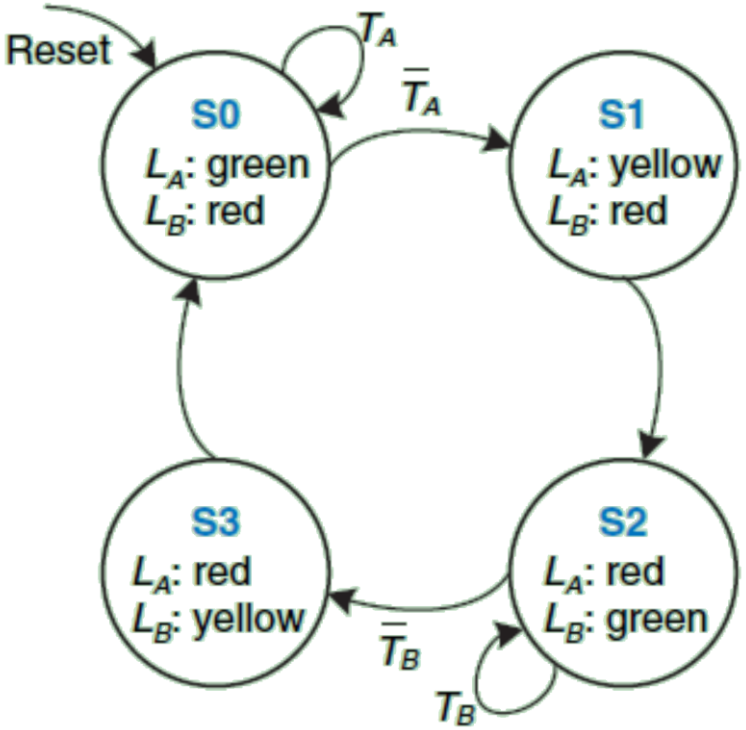


Black box view



State transition diagram

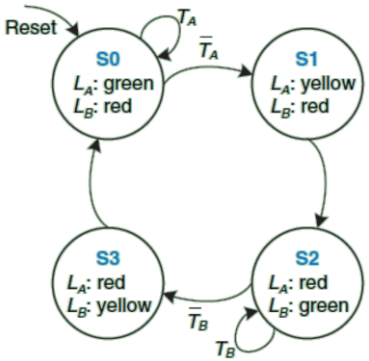




State table

State	Encoding $S_{1:0}$
S0	00
S1	01
S2	10
S3	11

State encoding



State transition table

Current State S	Inputs T_A	Inputs 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

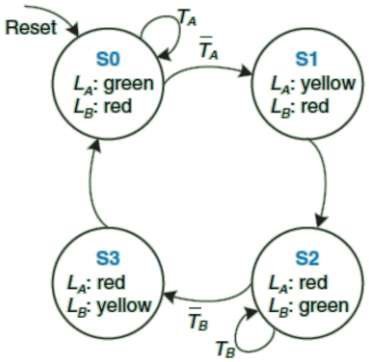
Binary encoded state transition table

Current State S_1	Current State S_0	Inputs T_A	Inputs T_B	Next State S'_1	Next State 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

Output table

Output encoding

Output	Encoding $L_{1:0}$
green	00
yellow	01
red	10



Output table

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

Sum-of-product form

State table

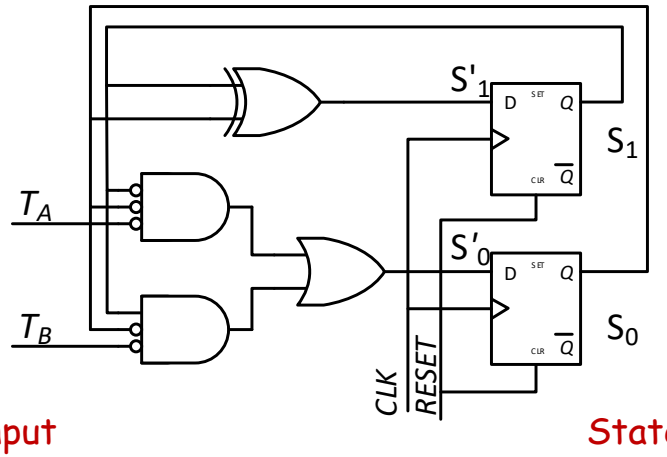
Current State S_1	Current State S_0	Inputs T_A	Inputs T_B	Next State S'_1	Next State S'_0
0	0	0	X	0	1
0	0	1	X	0	0
0	1	X	X	1	0
1	0	X	0	1	1
1	0	X	1	1	0
1	1	X	X	0	0

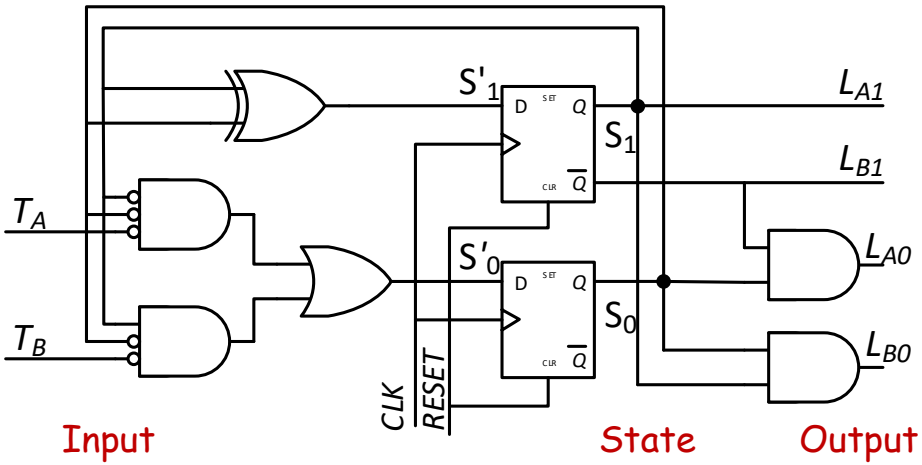
Output table

Current State S_1	Current State S_0	Outputs L_{A1}	Outputs L_{A0}	Outputs L_{B1}	Outputs 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

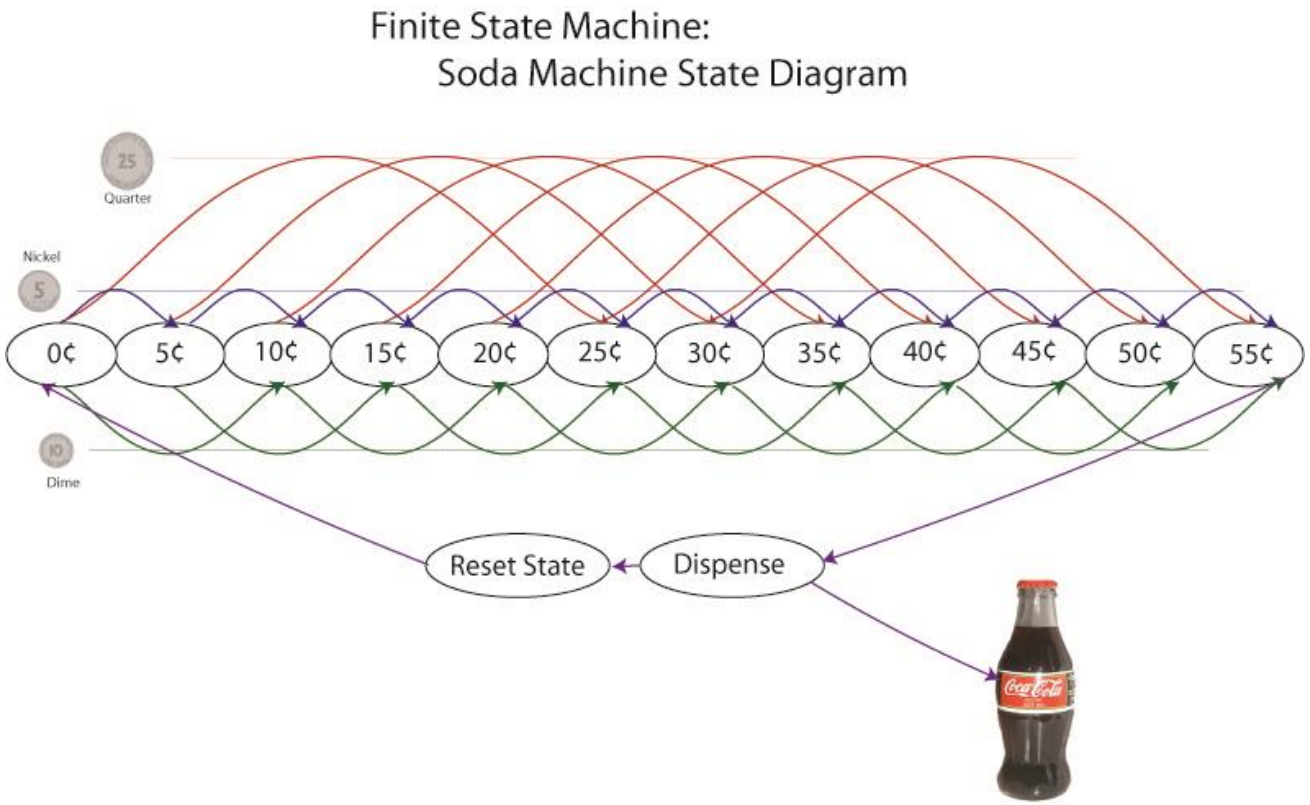
State & output logic

- State logic (sequential)
- Output Logic (combinational)



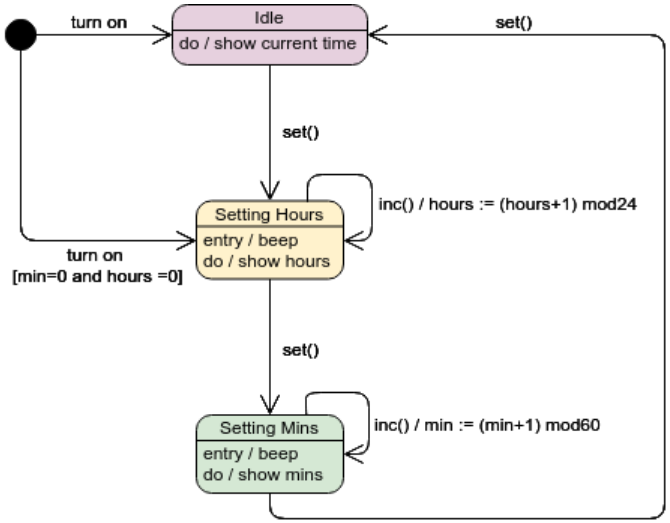


FSM with more states ……



FSM with more states ……

- Digital clock



- Microwave oven