

Digital Logic

III. *Sequential Circuits*

Jalal Kawash

A series of horizontal lines in teal, light blue, and white, stacked and offset to the right, creating a modern, layered effect.

Outline

1. Introduction
 - a. Types of Sequential Circuits
 - b. Clocks
2. Basic Memory Circuits
 - a. Latches
 - b. Flip-Flops
3. Synchronous Sequential Circuits
 - a. Finite State Machines

Digital Logic III

Sequential Circuits

Section 1

Introduction

Digital Logic III

Sequential Circuits

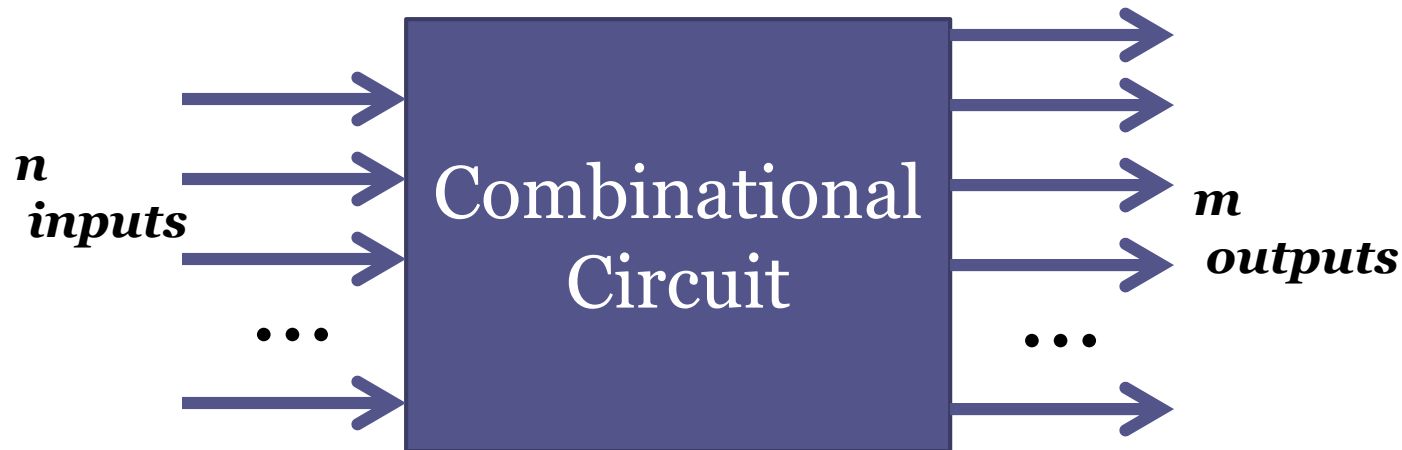
Section 1 Objectives

At the end of this section you will

1. Understand the two types of sequential circuits
2. Understand clock signals

Combinational Circuits

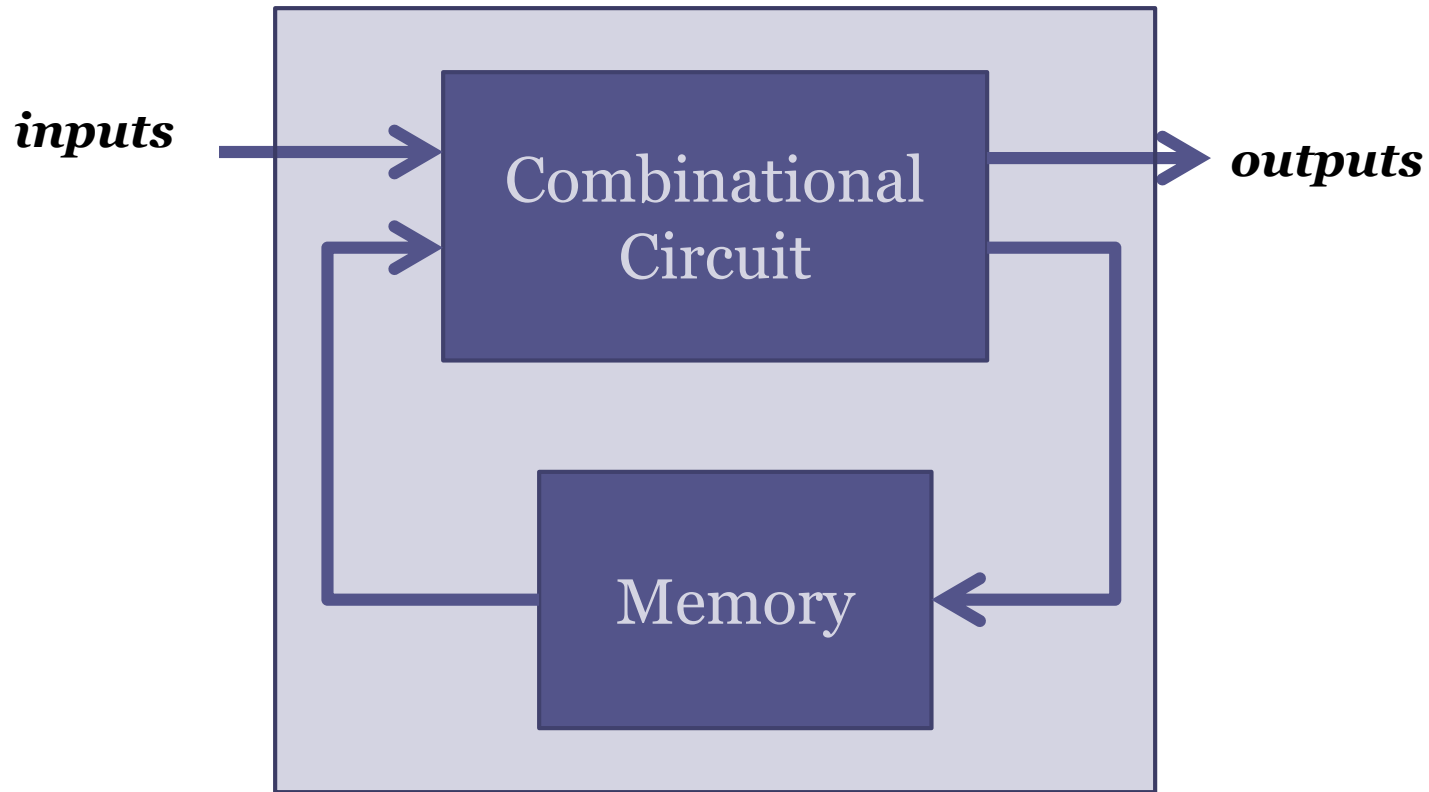
- A combinational circuit is a logic circuit whose output is solely determined by the inputs



Sequential Circuits

- Unlike a combinational circuit, the output of a sequential circuit at time t can affect the circuits output at time $t+1$

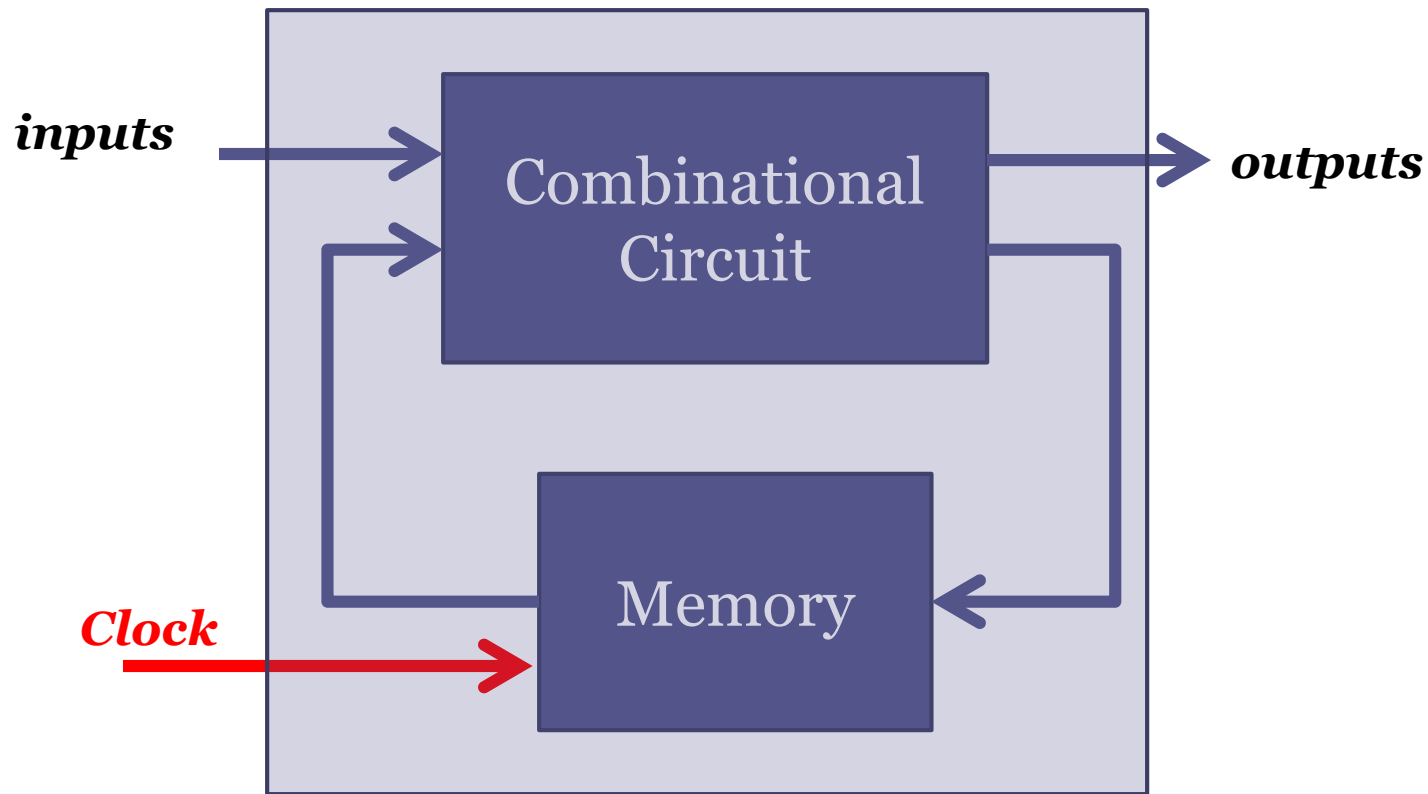
Sequential Circuits



Types of Sequential Circuits

- Synchronous: Employ clock signals
 - Clock signals affect memory at discrete time
 - Also called “clocked”
- Asynchronous: Do not employ clocks
 - Rely on signal propagation delay to affect memory
 - Harder to design
- In the course, I will only talk about synchronous circuits

Synchronous Sequential Circuits



Clocks

- A clock is a device that pulses at regular intervals



- Synchronous sequential Circuits can be:
 - level-triggered: Memory is activated during a level
 - E.g: during the high level
 - edge-triggered: Memory is activated when signal changes
 - E.g: from 0 to 1 (positive edge) or 1 to 0 (negative edge)
 - Used more



Digital Logic III

Sequential Circuits

Section 2

Basic Memory Circuits

Section 2 Objectives

At the end of this section you will

1. Build 1-bit memory circuits
2. Understand latches
3. Realize problems with latches
4. Build flip-flops from latches
5. Construct higher-capacity registers from flip-flops

Digital Logic III

Sequential Circuits

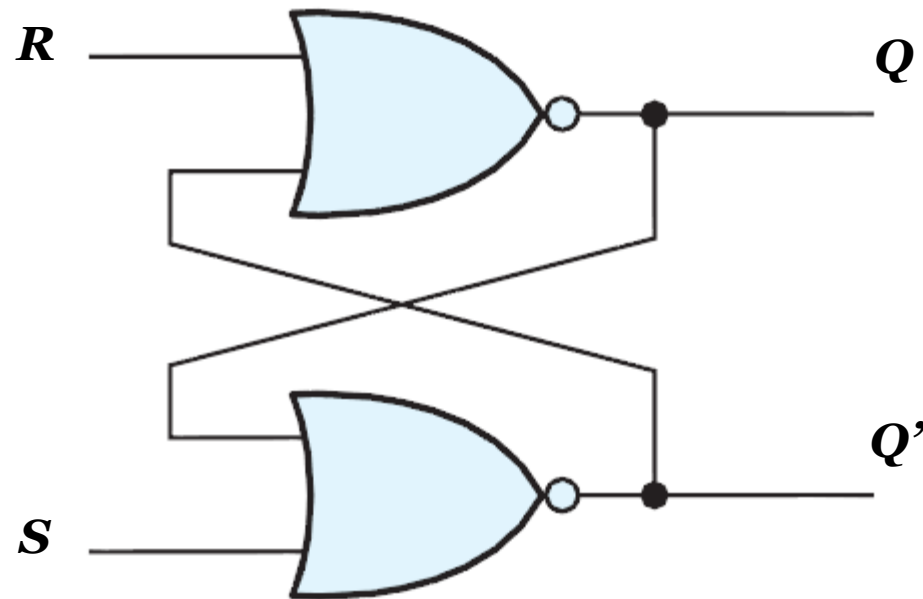
Section 2

Basic Memory Circuits

Latches

Latches

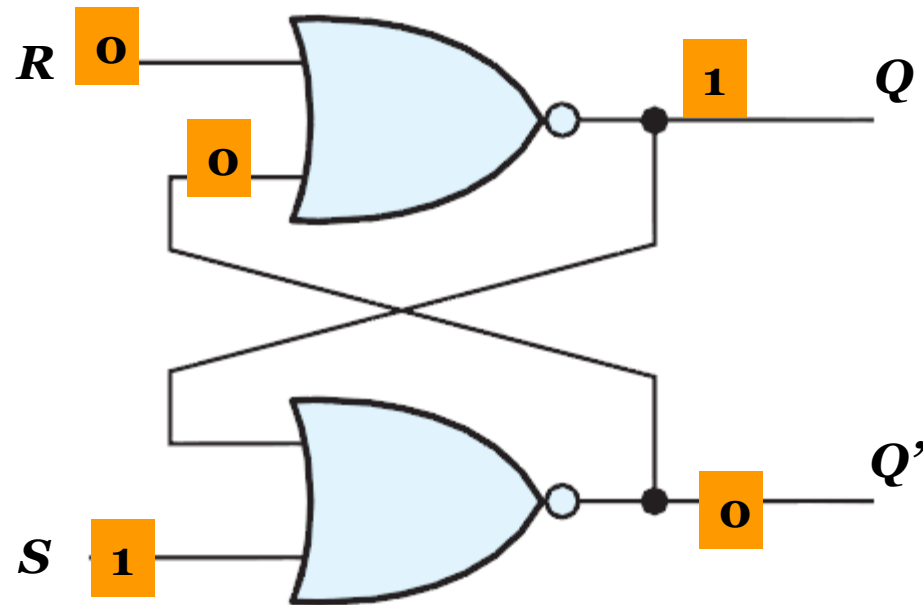
- A latch is a level-triggered, single-bit memory
- The Set-Rest (SR) latch:



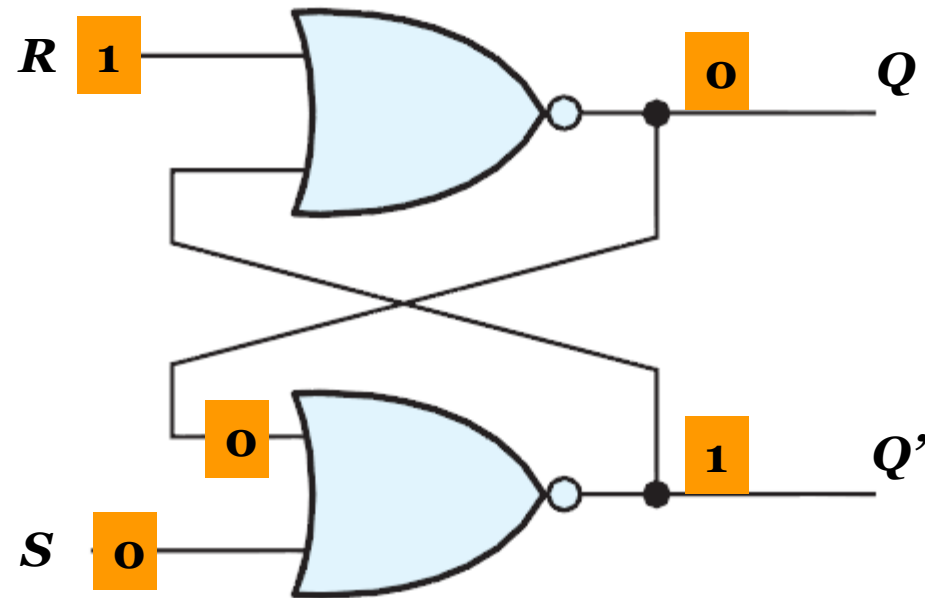
NOR Truth Table

x	y	$x+y$	$(x+y)'$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

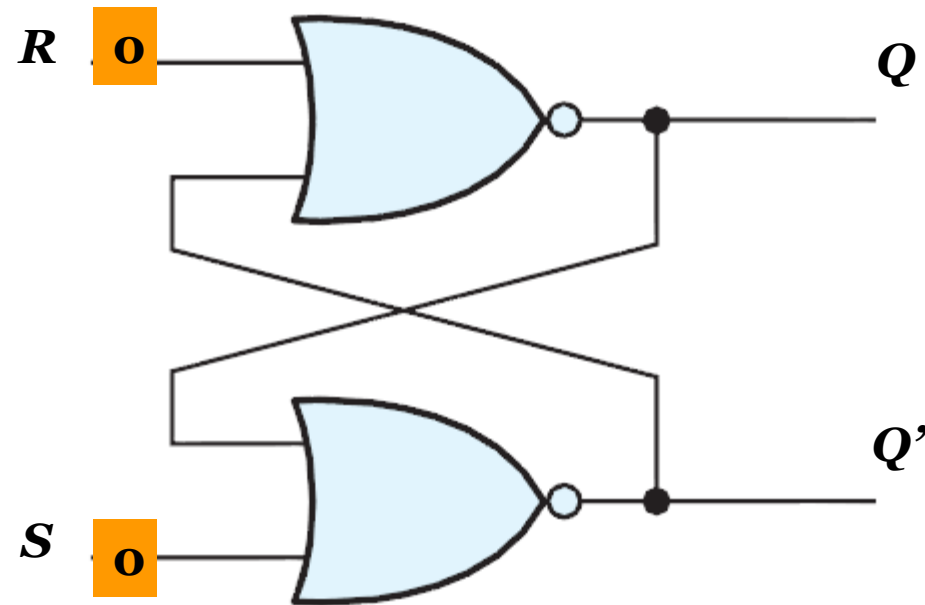
Setting the latch, $S = 1$, $R = 0$
(memory = $Q = 1$)



Resetting the latch, $S = 0$, $R = 1$
(memory = $Q = 0$)

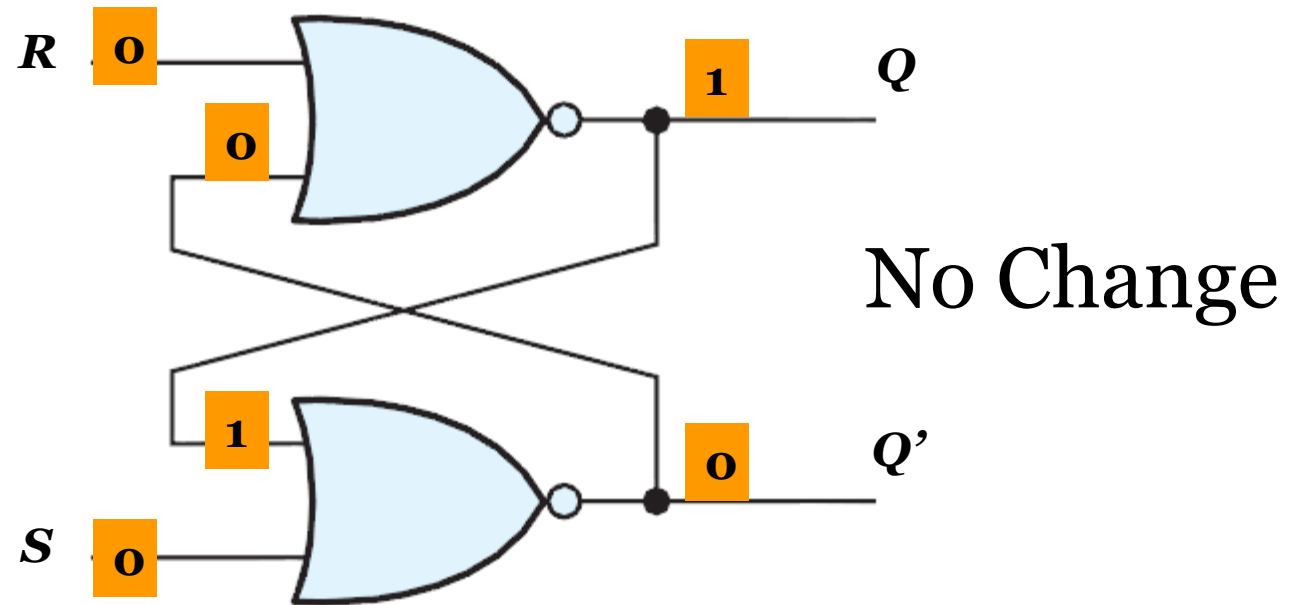


$S = R = 0?$

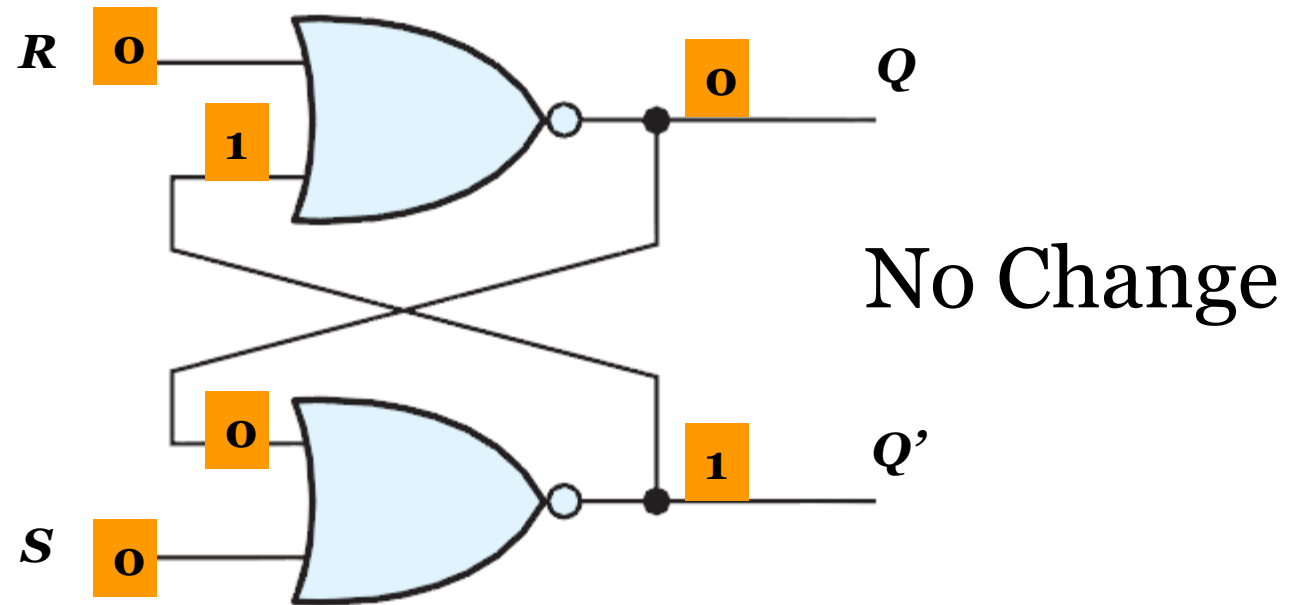


Behaviour depends on previous state, Q

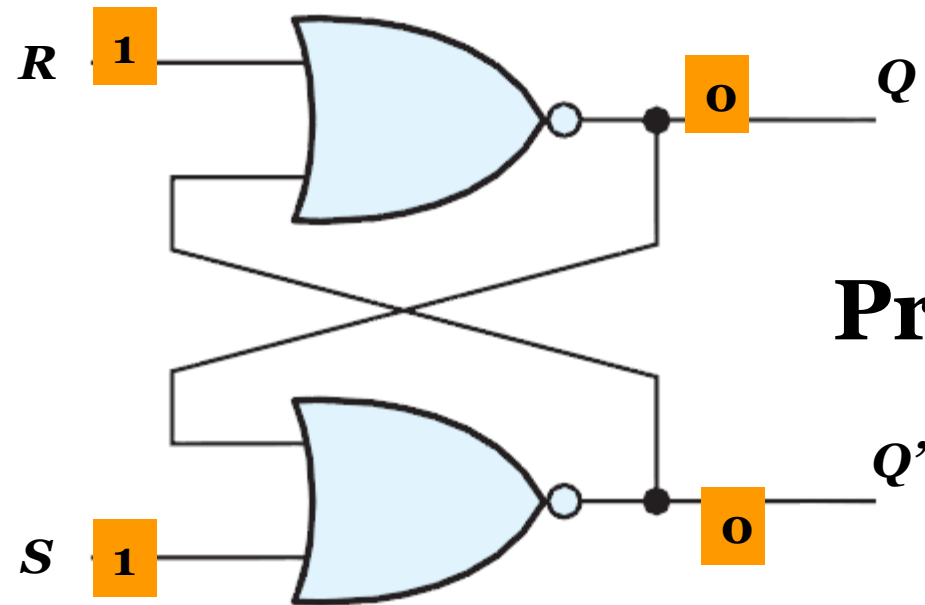
When $Q = 1$



When $Q = 0$



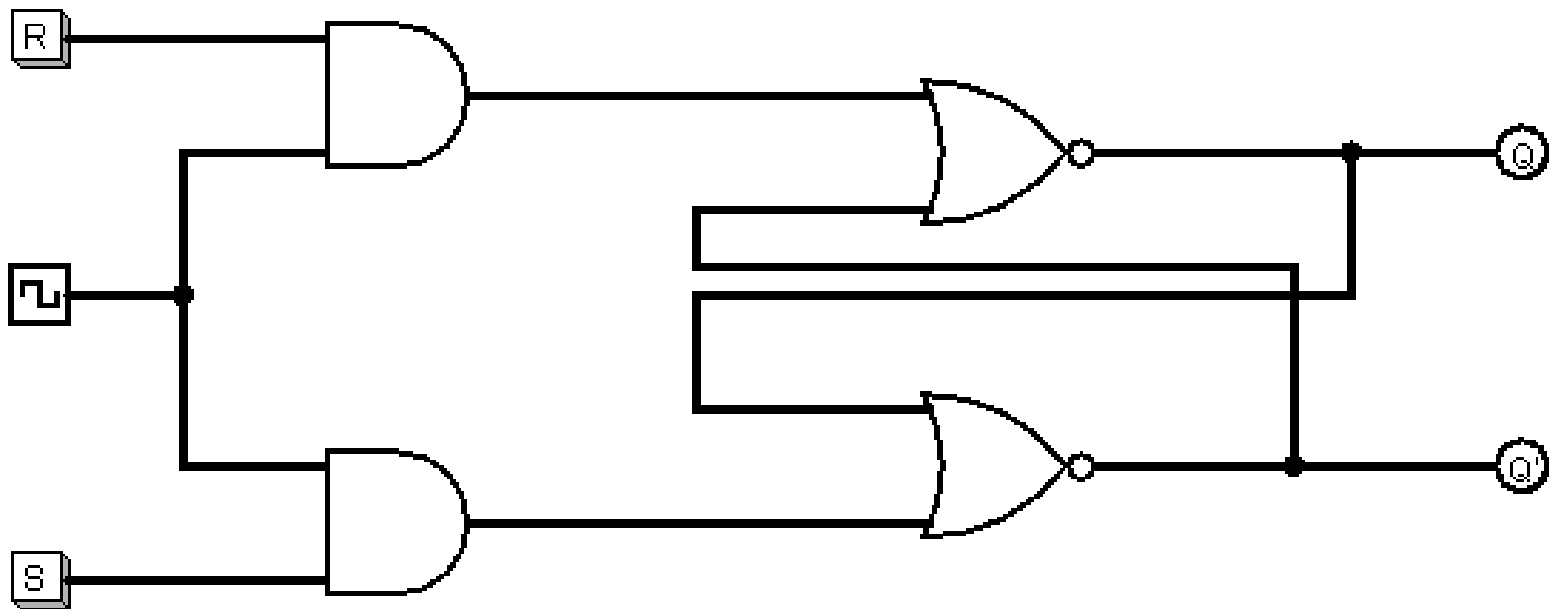
$S = R = 1?$



Prohibited

Contradicts requirement that Q and Q' are opposites

Clocked SR

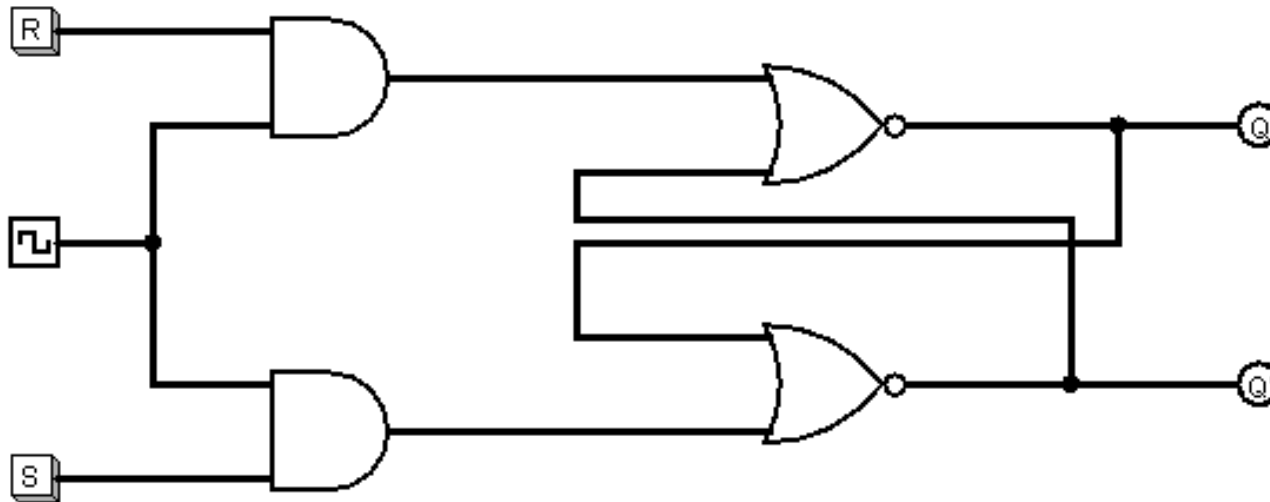


Source: DL3.circ (Clocked-SR)

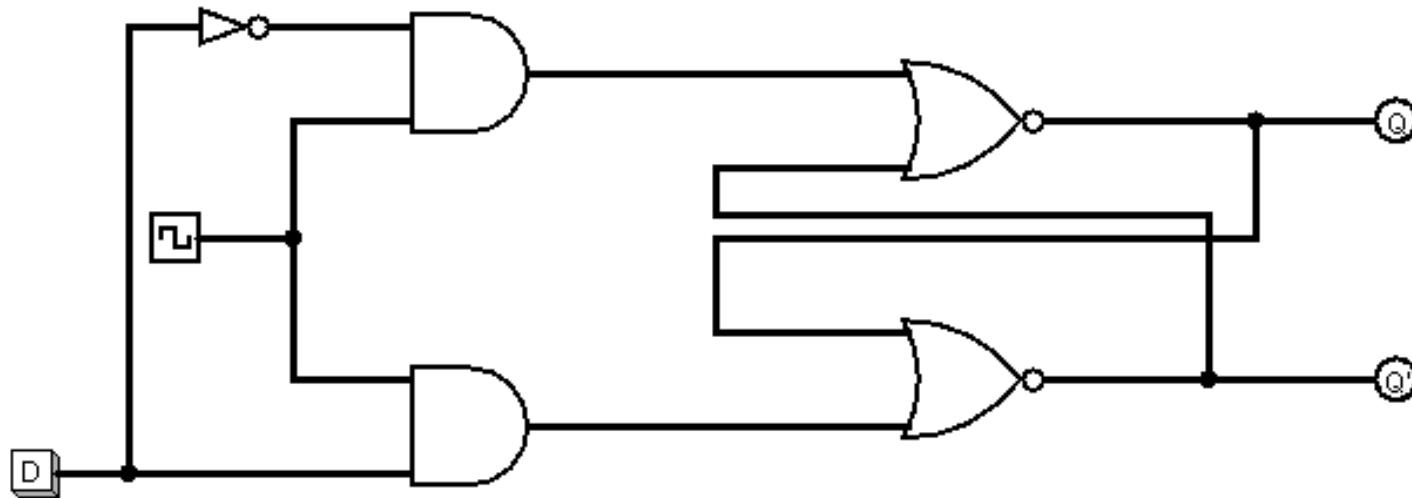
Demo

Clocked SR

- This is a positive-level triggered latch
- When Clock is 0, R & S have no effect



D-Latch

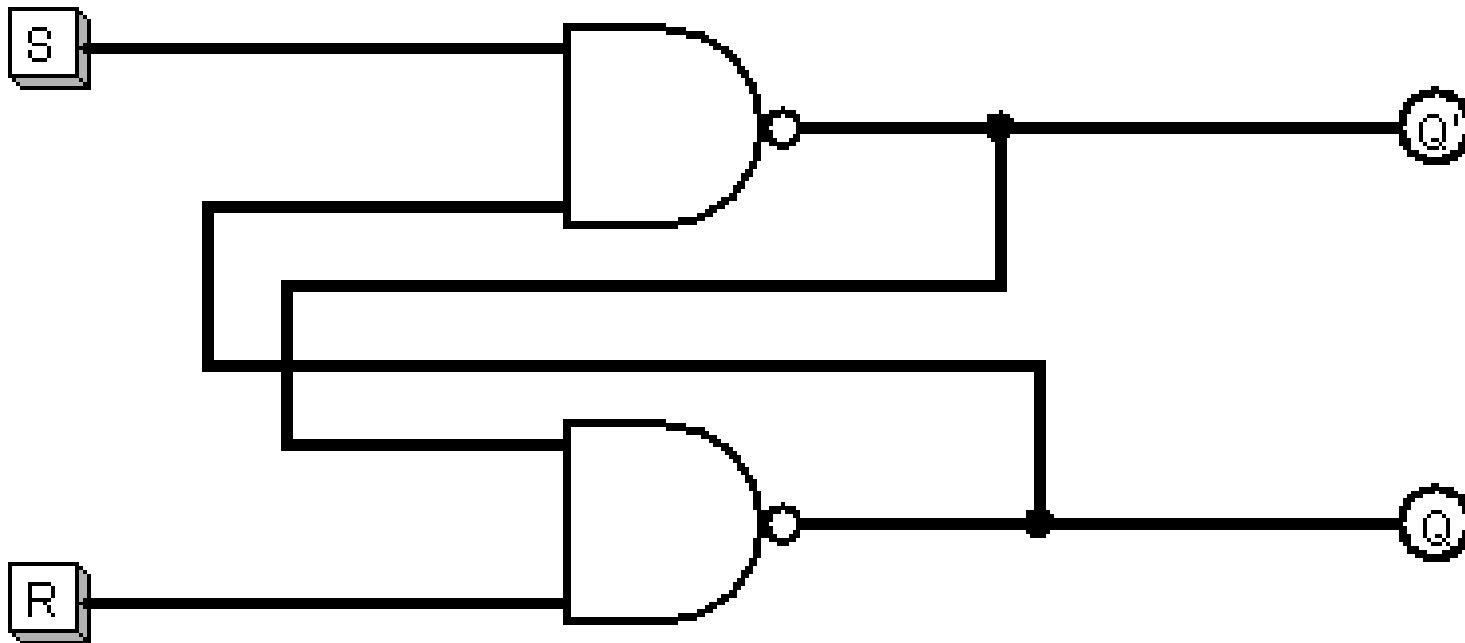


- There is no $S=R=1$ state

D-latch demo

Source: DL3.circ (D-latch)

SR Latch with NAND Gates



Source: DL3.circ (NAND SR-Latch)

SR-Latch Comparison

NOR

- $R=S=0$ has no effect
- $R=S=1$ prohibited

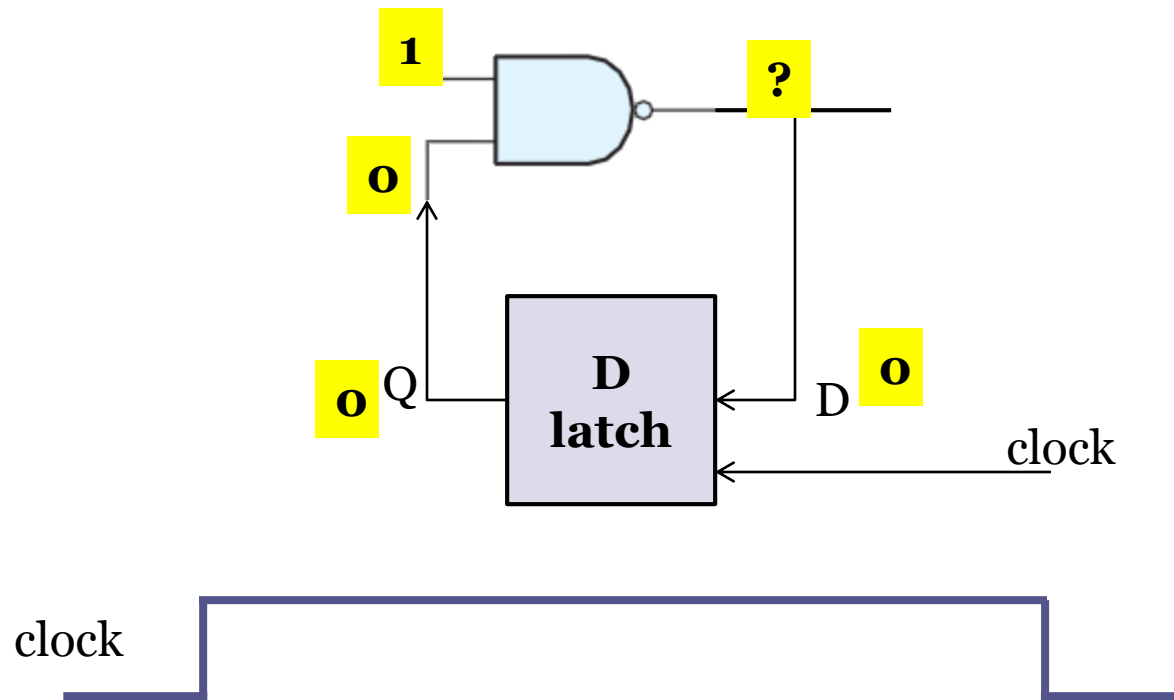
NAND

- $R=S=1$ has no effect
- $R=S=0$ prohibited

Problem With Latches

- Latches are level-triggered, so circuits can become unreliable

Why unreliable?



Digital Logic III

Sequential Circuits

Section 2

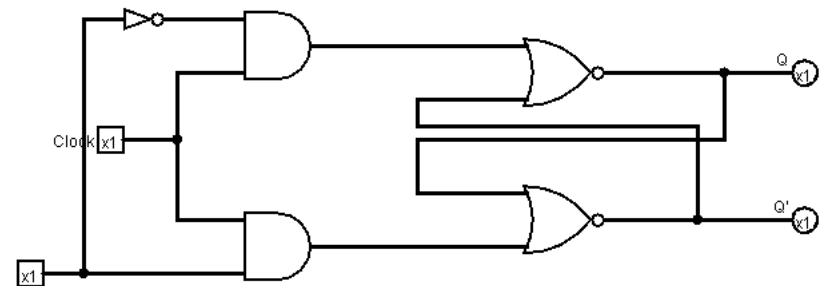
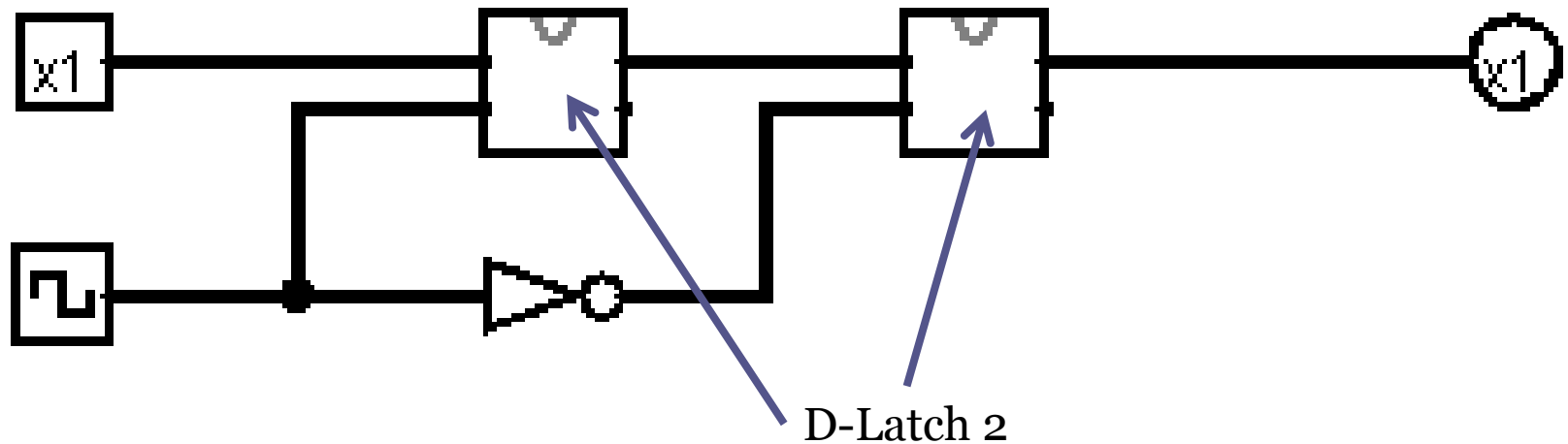
Basic Memory Circuits

Flip-Flops

Flip-Flops

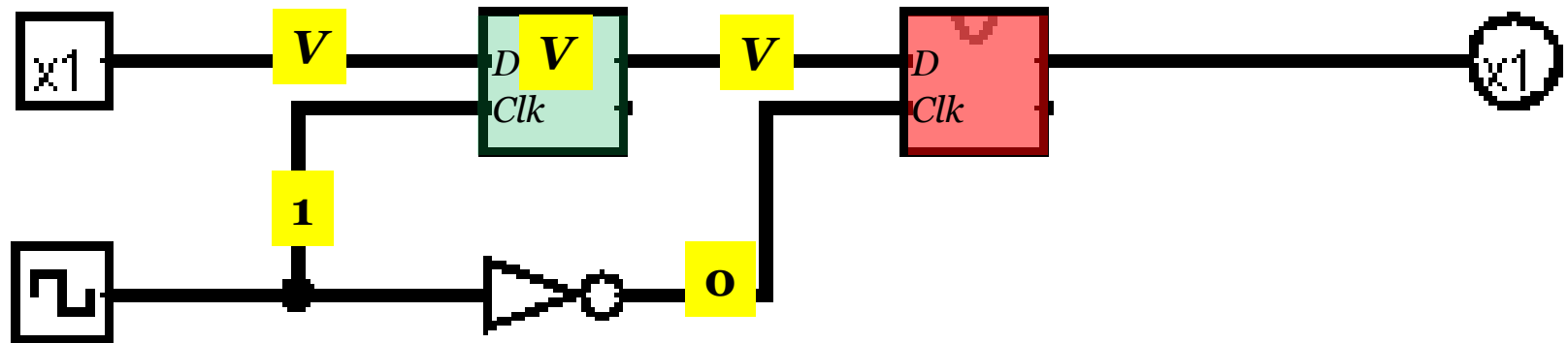
- Flip-flops are also 1-bit memory circuits
- They are edge triggered
- They solve the unreliability issue with sequential circuits that employ latches
- Flip-flops are the basis for registers

D Flip-Flop (Master-Slave)



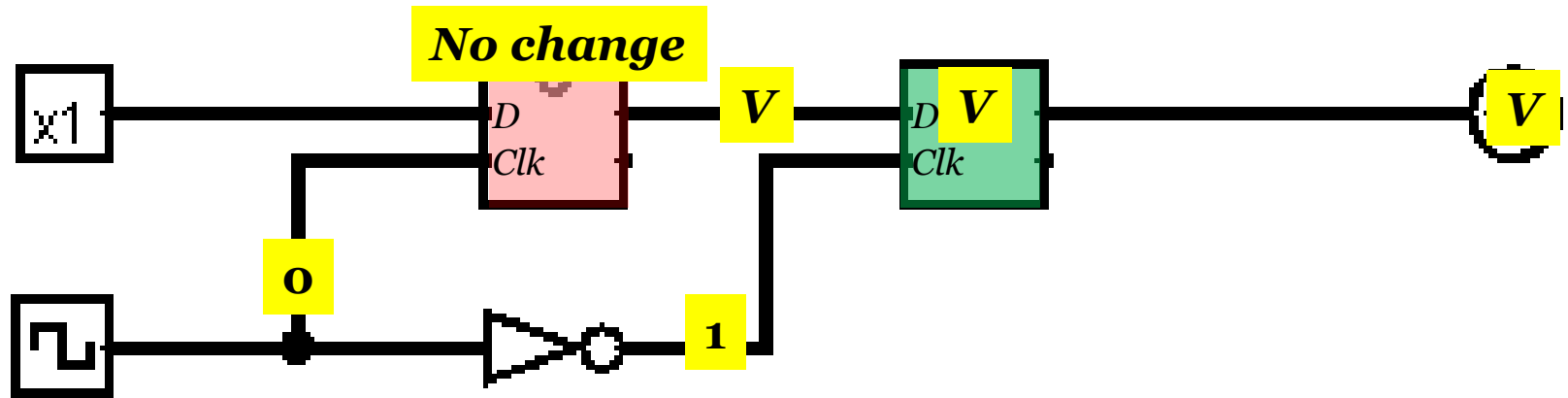
Source: DL3.circ (D Flip Flop, D-Latch 2

D Flip-Flop (Master-Slave)

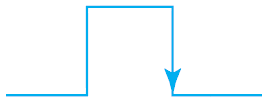


Slave latch is not affected
Q does not change

D Flip-Flop (Master-Slave)

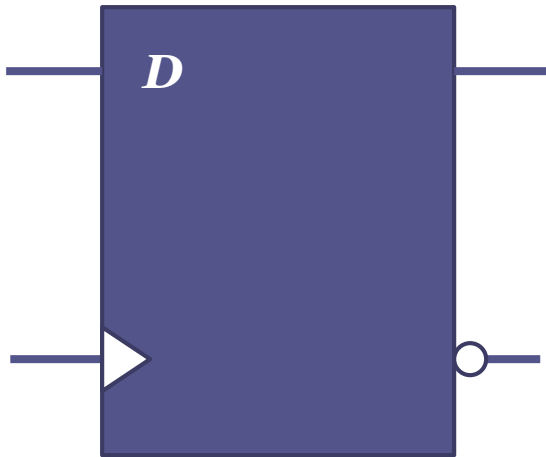


Master latch is not affected
Q changes to V

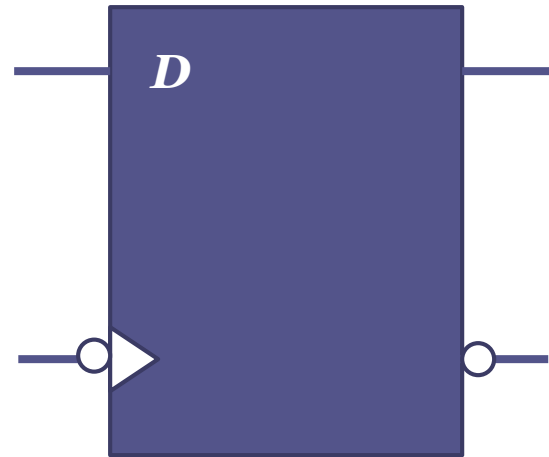


This is a negative-edge triggered flip-flop

Symbols for D-Flip-Flops

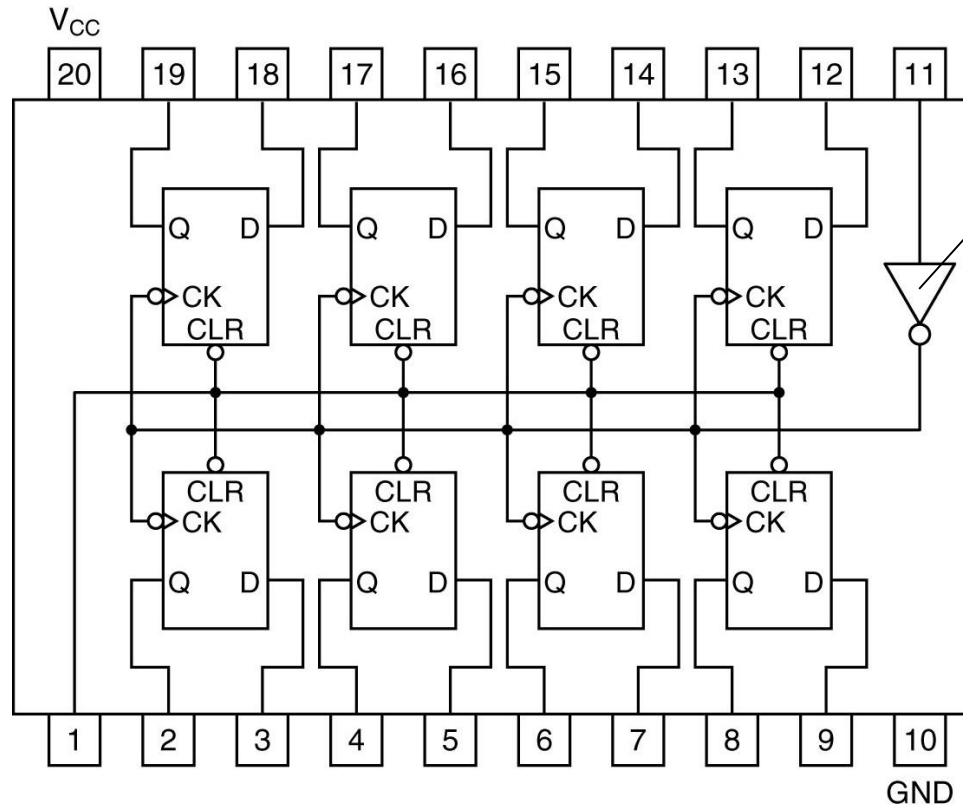


Positive-edge



Negative-edge

8-bit Register



Works as
a signal
amplifier

Octal flip-flop

Memory Organization

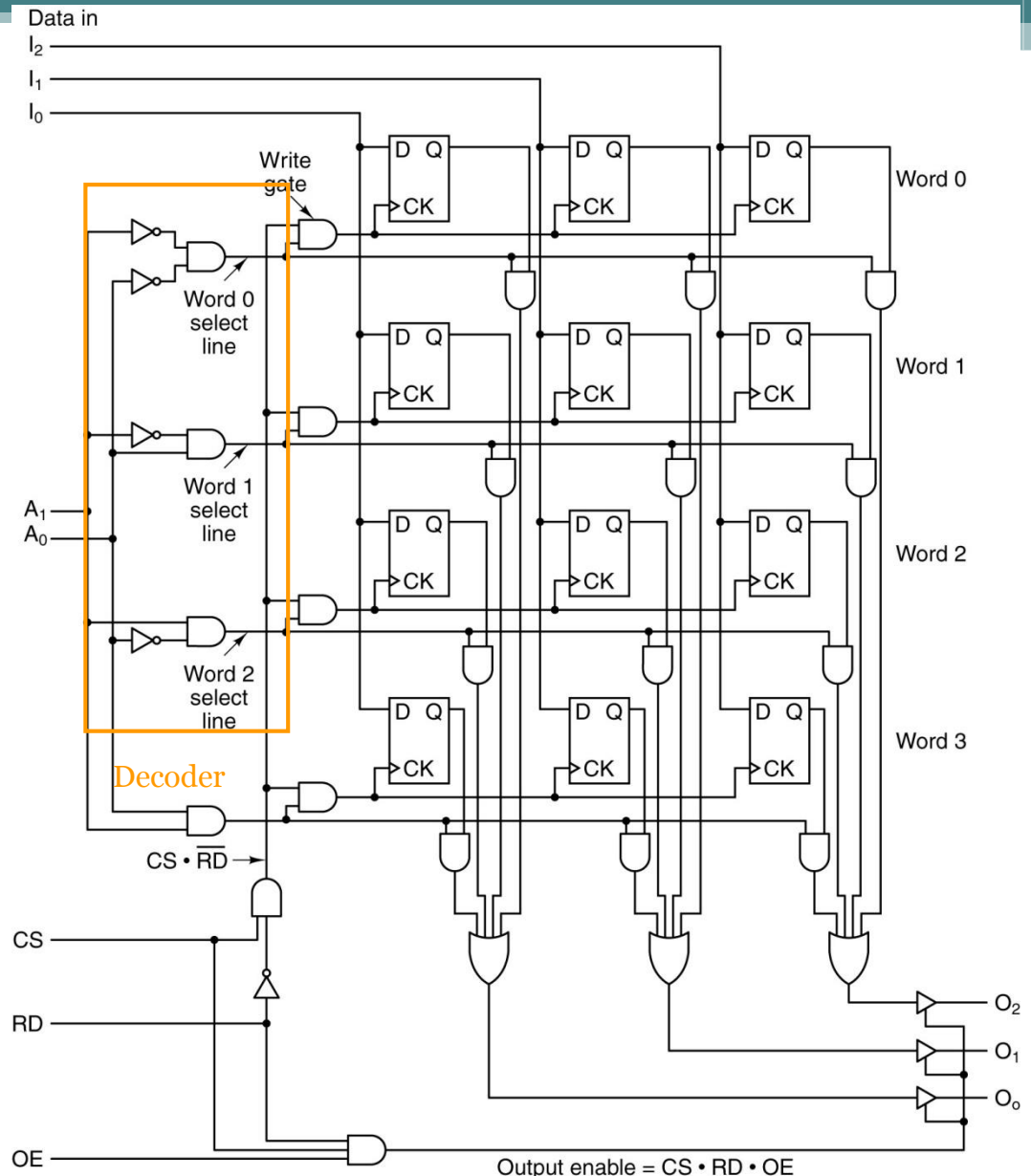
Logic diagram for a 4 x 3 memory.

Each row is one of the four 3-bit words.

Chip select

Read (1) –
Write (0)

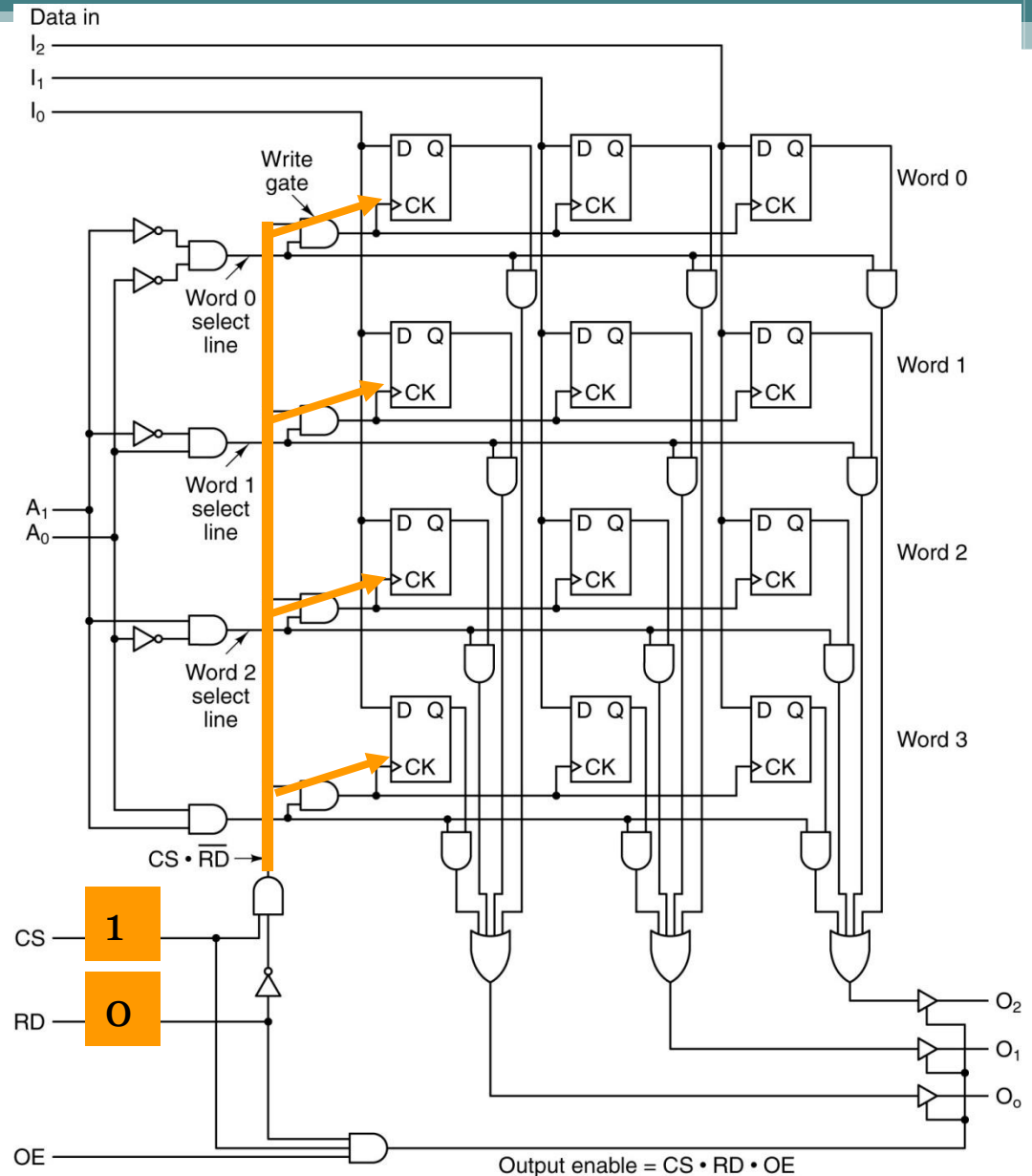
Output
Enabled



Writing

Logic diagram for a 4 x 3 memory.

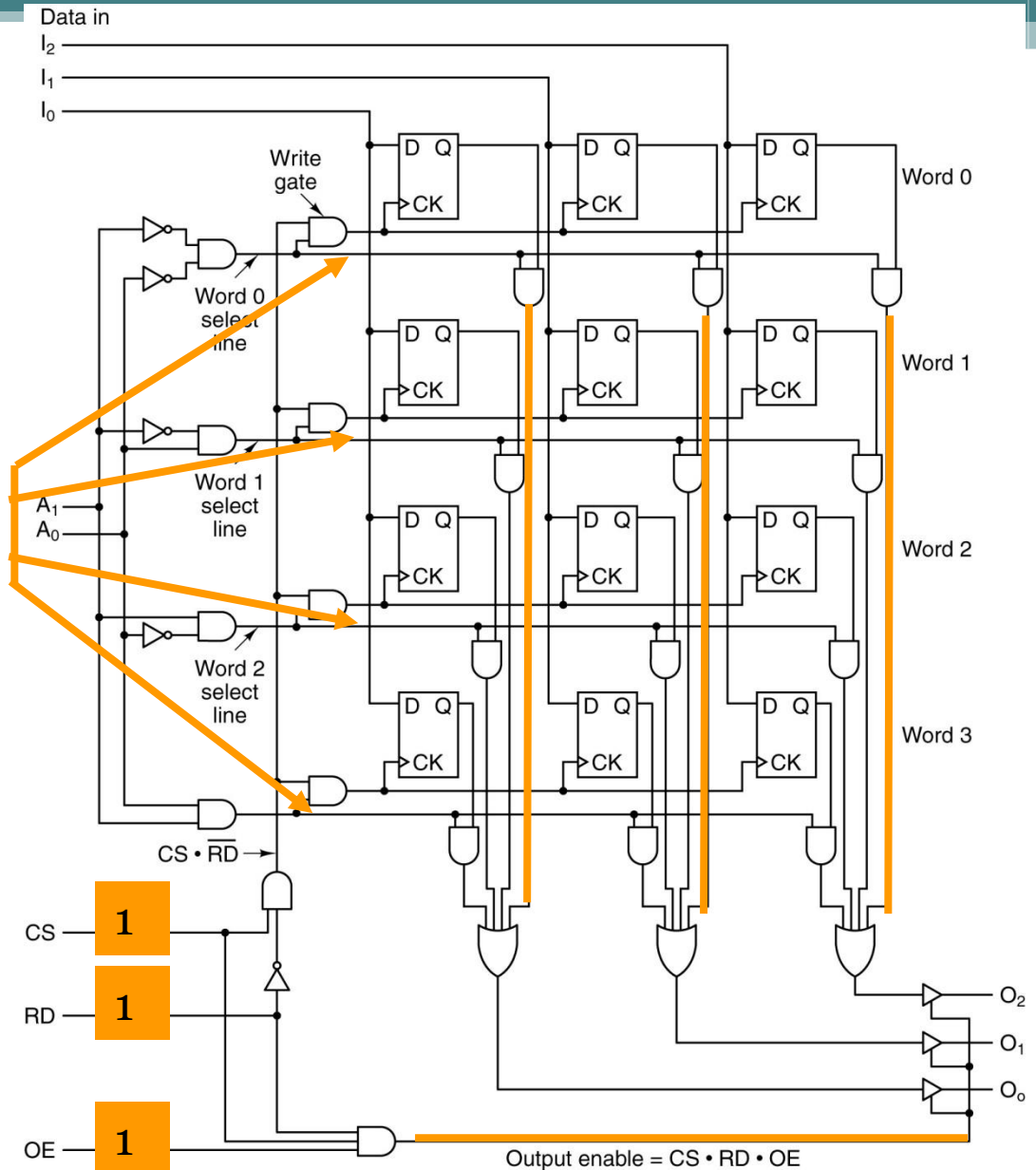
Each row is one of
the four 3-bit
words.



Reading

Logic diagram for a 4 x 3 memory.

Each row is one of the four 3-bit words.



Digital Logic III

Sequential Circuits

Section 2

Finite State Machines

Digital Logic III

Sequential Circuits

Section 3 Objective

At the end of this section you will

1. Design sequential circuits as Finite State Machines

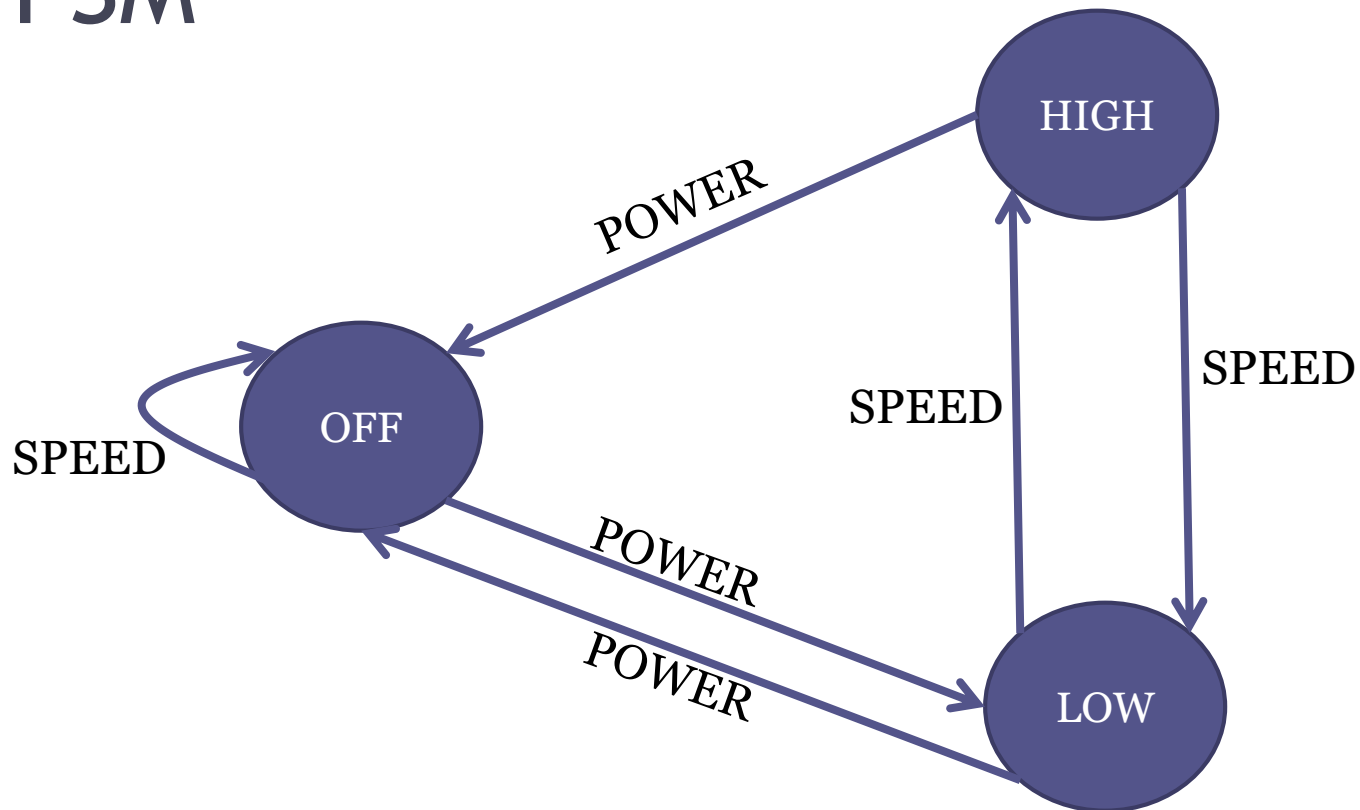
Finite State Machines

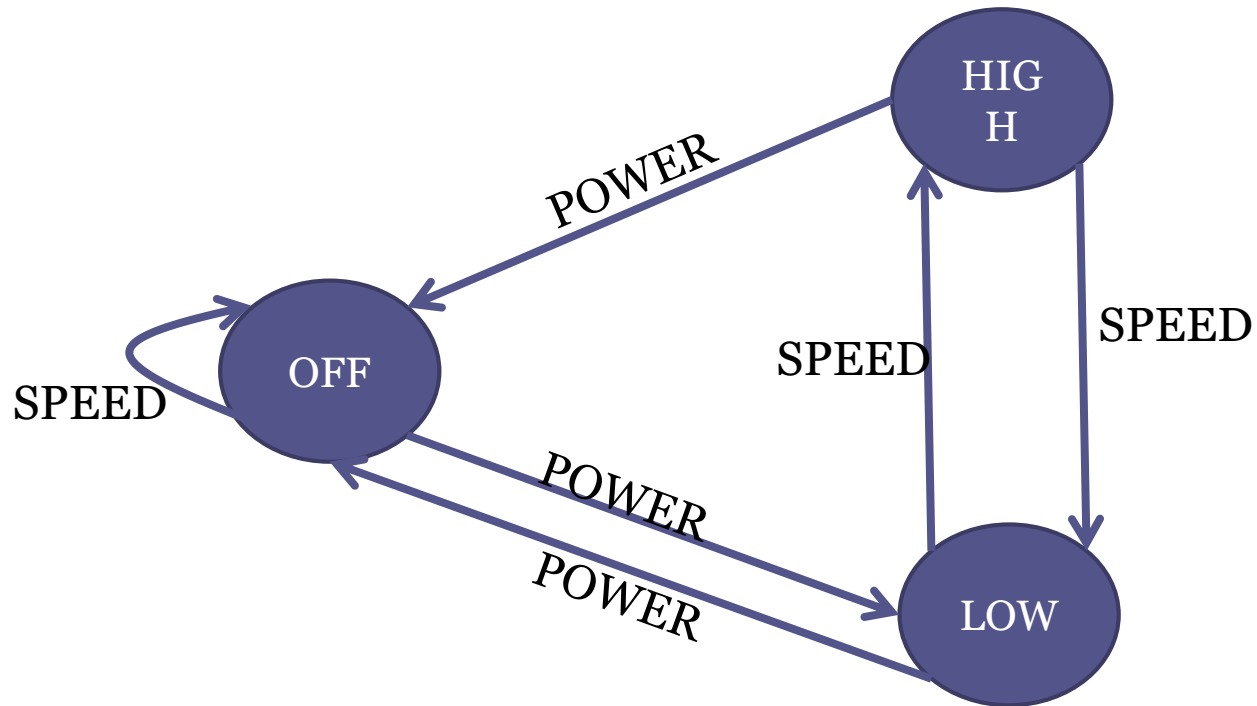
- Hardware is typically designed as an FSM
- An FSM is equivalent to an synchronous DL circuit
- FSM
 - A finite set of states
 - A set of transitions
- Example: A simple Air-conditioning system

Simple Air-Conditioning System

- Can be only controlled by a remote device
 - Device has two buttons: POWER and SPEED
- AC has two fan speeds: LOW and HIGH
- When turned on it always starts in LOW
- POWER switches it ON and OFF
- SPEED toggles between both speeds
 - When on LOW, goes to HIGH
 - When on HIGH goes to LOW

FSM





POWER	SPEED	Current State	Next State
1	0	OFF	LOW
1	0	LOW	OFF
1	0	HIGH	OFF
0	1	OFF	OFF
0	1	LOW	HIGH
0	1	HIGH	LOW

POWER	SPEED	Current State	Next State
1	0	OFF	LOW
1	0	LOW	OFF
1	0	HIGH	OFF
0	1	OFF	OFF
0	1	LOW	HIGH
0	1	HIGH	LOW
0	0	OFF	OFF
0	0	LOW	LOW
0	0	HIGH	HIGH

POWER	SPEED	Current State	Next State
1	0	OFF	LOW
1	0	LOW	OFF
1	0	HIGH	OFF
0	1	OFF	OFF
0	1	LOW	HIGH
0	1	HIGH	LOW
0	0	OFF	OFF
0	0	LOW	LOW
0	0	HIGH	HIGH

- Replace:
- **OFF by 00**
- LOW by 01
- HIGH by 10

POWER	SPEED	Current State	Next State
1	0	00	LOW
1	0	LOW	00
1	0	HIGH	00
0	1	00	00
0	1	LOW	HIGH
0	1	HIGH	LOW
0	0	00	00
0	0	LOW	LOW
0	0	HIGH	HIGH

- Replace:
- **OFF by 00**
- LOW by 01
- HIGH by 10

POWER	SPEED	Current State	Next State
1	0	00	LOW
1	0	LOW	00
1	0	HIGH	00
0	1	00	00
0	1	LOW	HIGH
0	1	HIGH	LOW
0	0	00	00
0	0	LOW	LOW
0	0	HIGH	HIGH

- Replace:
- OFF by 00
- **LOW by 01**
- HIGH by 10

POWER	SPEED	Current State	Next State
1	0	00	01
1	0	01	00
1	0	HIGH	00
0	1	00	00
0	1	01	HIGH
0	1	HIGH	01
0	0	00	00
0	0	01	01
0	0	HIGH	HIGH

- Replace:
- OFF by 00
- **LOW by 01**
- HIGH by 10

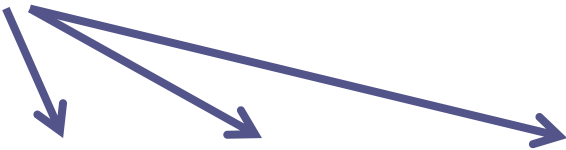
POWER	SPEED	Current State	Next State
1	0	00	01
1	0	01	00
1	0	HIGH	00
0	1	00	00
0	1	01	HIGH
0	1	HIGH	01
0	0	00	00
0	0	01	01
0	0	HIGH	HIGH

- Replace:
- OFF by 00
- LOW by 01
- **HIGH by 10**

POWER	SPEED	Current State	Next State
1	0	00	01
1	0	01	00
1	0	10	00
0	1	00	00
0	1	01	10
0	1	10	01
0	0	00	00
0	0	01	01
0	0	10	10

- Replace:
- OFF by 00
- LOW by 01
- **HIGH by 10**

Three inputs: two bits (P&S) from the device
And two (B&C) bits representing the state the FSM is in



P	S	AB	XY
1	0	00	01
1	0	01	00
1	0	10	00
0	1	00	00
0	1	01	10
0	1	10	01
0	0	00	00
0	0	01	01
0	0	10	10

Two output bits (XY) determining the next state



Boolean Sum of Products for X

- $X = P'SA'B + P'S'AB'$
- $= P'(SA'B + S'AB')$

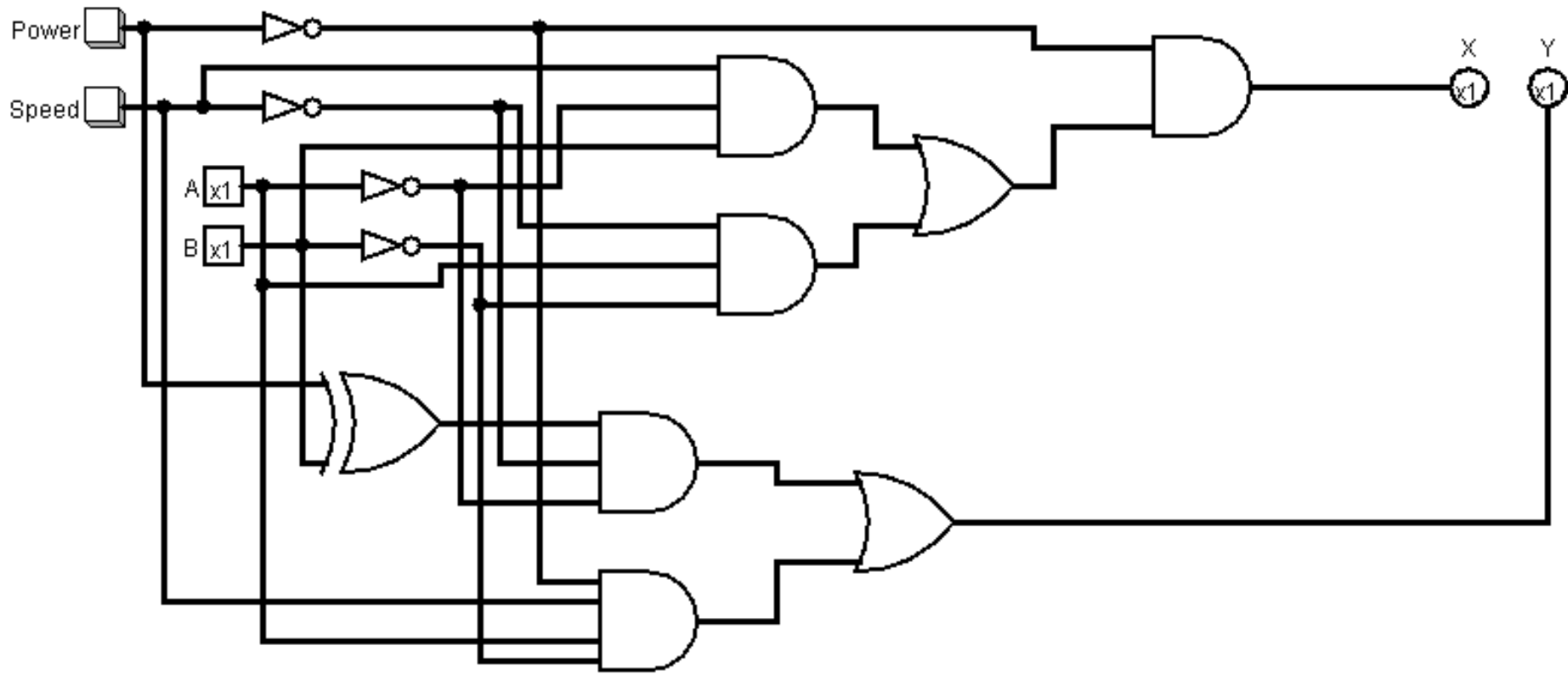
P	S	AB	XY
1	0	00	01
1	0	01	00
1	0	10	00
0	1	00	00
0	1	01	10
0	1	10	01
0	0	00	00
0	0	01	01
0	0	10	10

Boolean Sum of Products for Y

- $Y = PS'A'B' + P'SAB' + P'S'A'B$
- $= S'A' (PB' + P'B) + P'SAB'$
- $= S'A' (P \oplus B) + P'SAB'$

P	S	AB	XY
1	0	00	01
1	0	01	00
1	0	10	00
0	1	00	00
0	1	01	10
0	1	10	01
0	0	00	00
0	0	01	01
0	0	10	10

The Combinational Circuit

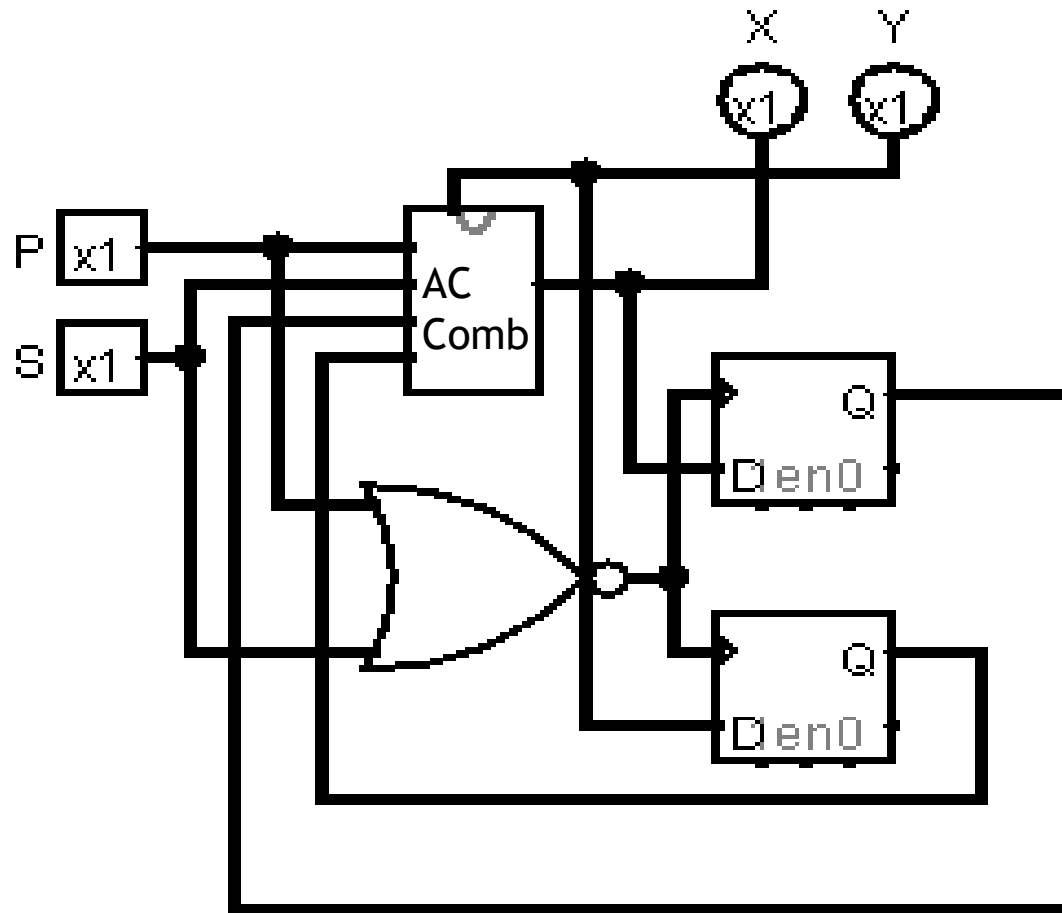


Source:DL3.circ (AC Combinational)

A Sequential Circuit?

- Not yet
- The FSM needs to ‘remember’ current state to be used to determine next state
- Store X and feed it back to B
- Store Y and feed it back to C

The Sequential Circuit



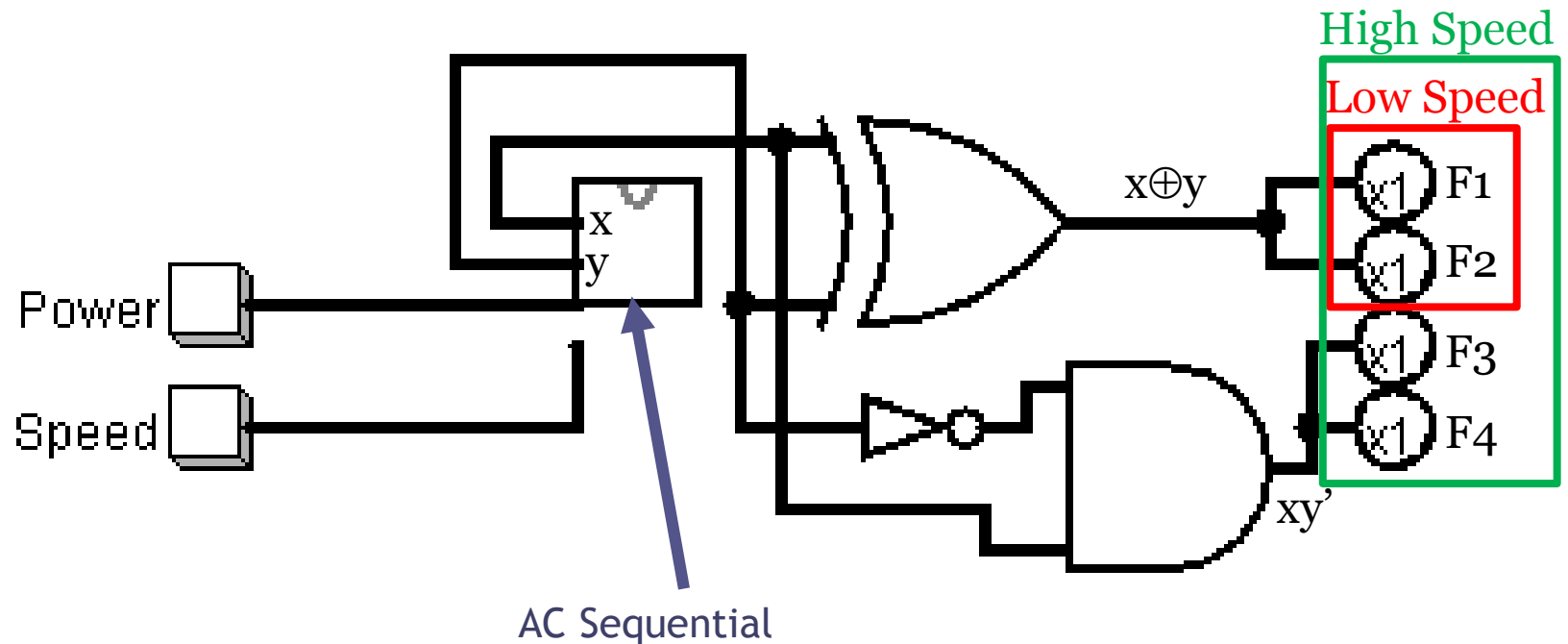
Source: LD3.circ (AC Sequential)

The System

x	y	F1	F2	F3	F4
0	1	1	1	0	0
1	0	1	1	1	1

$$F1 = F2 = x \oplus y$$

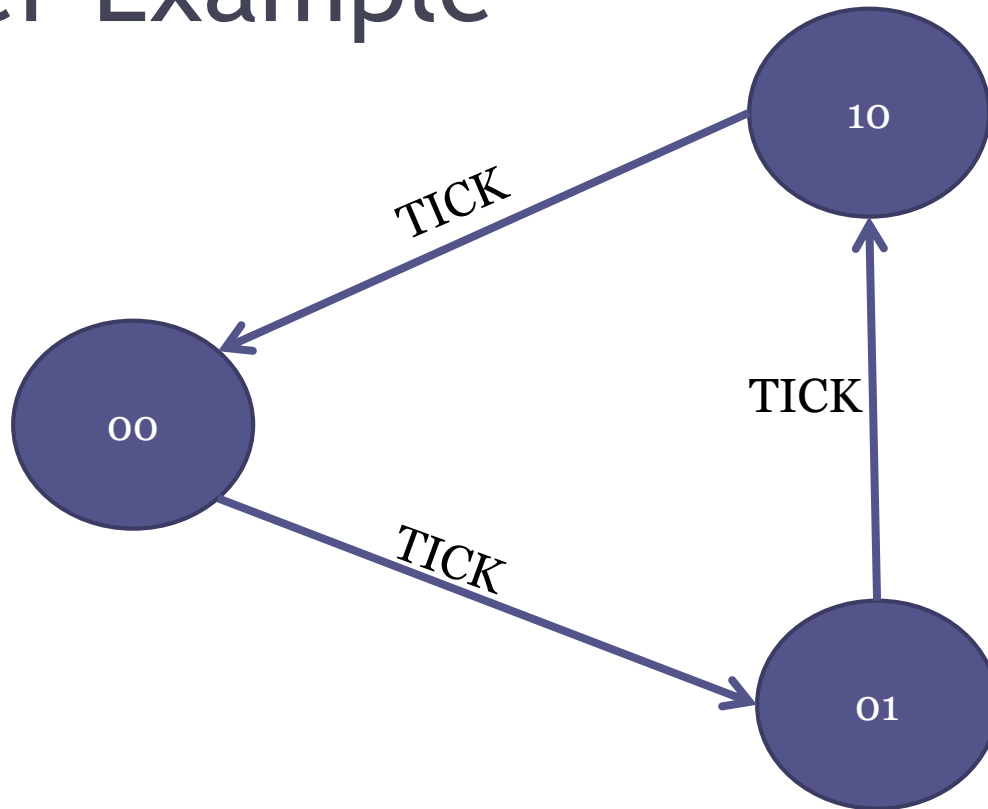
$$F3 = F4 = xy'$$

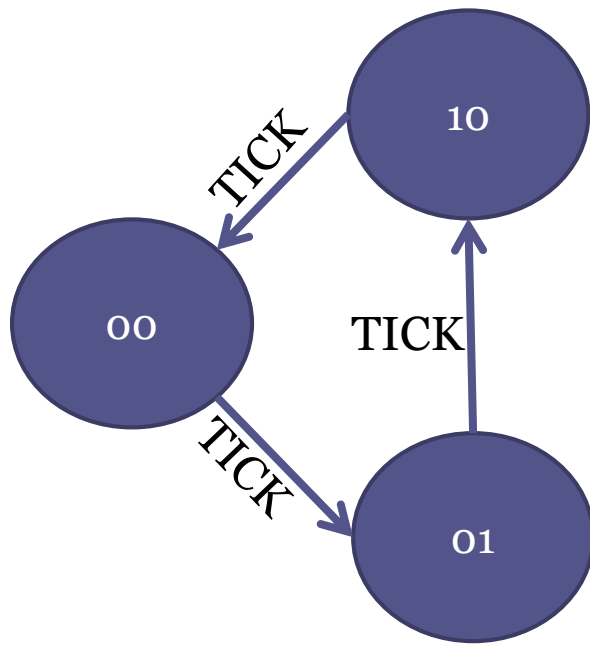


Demo

Source: LD3.circ (AC System)

Timer Example





A	B	Tick	A _{new}	B _{new}
0	0	1	0	1
0	1	1	1	0
1	0	1	0	0
0	0	0	0	0
0	1	0	0	1
1	0	0	1	0

$$\begin{aligned}
 A_{\text{new}} &= A'BT + AB'T \\
 B_{\text{new}} &= A'B'T + A'BT' \\
 &= A'(B \oplus T)
 \end{aligned}$$

