# Lecture 9: Digital Electronics

## Introduction:

- We can classify the building blocks of a circuit or system as being either analog or digital in nature.
  - If we focus on voltage as the circuit parameter of interest:
    - Analog: The voltage can take on a range of voltages, e.g. any value between 0.1 and 2 Volts.
    - Digital: The voltage can have only two values, e.g. 0 or 5 Volts
      - We say the voltage is either on or off
      - ☞ Digital circuits are useful when we don't need a continuous range of voltage or current.
      - Examples: Representing numbers, binary logic, counting circuits.
      - Example: Represent base 10 numbers using the binary system:

        $2|_{10} = 10|_2 = 1 \times 2^1 + 0 \times 2^0$

        $10|_{10} = 1010|_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

- Digital circuits use standard voltages (or currents) to denote <u>ON</u> (high, 1) or <u>OFF</u> (low, 0).
  - These standards are called "Logic Families" and there are several families.
  - Two of the most popular families are:
    - TTL (**T**ransistor-**T**ransistor-**L**ogic): ON = 5 Volts, OFF = 0 Volts
    - ECL (**E**mitter-**C**oupled-**L**ogic): ON = -1 Volt, OFF = -1.6 Volts
  - ☞ For practical reasons both ON and OFF are given by a range of voltages or currents.
  - ON for an input to a circuit might have slightly different voltage than ON for an output to a circuit.

- A description of several logic families is given in the table below:

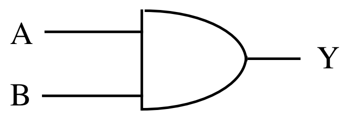
| | Delay (ns) | Max. FF Rate (MHz) | Power/Gate (mW) | High (V) | Low (V) |
|---|---|---|---|---|---|
| Standard TTL (7400) | 10 | 35 | 15 | 3.5 | 0.2 |
| Low-power Schottky (74LS00) | 9.0 | 33 | 2 | 3.5 | 0.2 |
| Fast TTL (74F00) | 3.5 | 125 | 5.5 | 2.7 | 0.5 |
| CMOS (74C00) | 25 @ 10 V<br>50 @ 25 V | 10 @ 10 V<br>3.5 @ 5 V | 0.01 | 5-15 | 0 |
| High-speed CMOS (74HC00) | 8.0 | 40 | 0.01 | 2-6 | 0.1 |
| ECL | 2 | 250 | 25 | -0.9 | -1.8 |
| 100k ECL | 0.75 | 500 | 40 | -1.0 | -1.7 |

  - Advantages of Digital:
    - only deal with two voltage levels (either ON or OFF)
    - voltages (or currents) are standardized
    - do not deal with individual transistors…
  - Disadvantages of Digital:
    - too many "black" boxes
    - need good power supplies, clocks etc. for circuits to work properly

## Logic Gates:

- We want to make decisions based on digital information.
  - For now consider the basic building blocks with one or two inputs and one output.
- The basic logic units (gates) are: AND, OR, NOT.
  - These functions are defined by their truth tables.

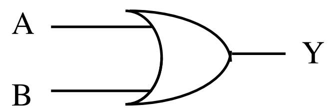
AND




| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

$A \bullet B = Y$

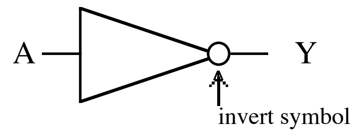
OR




| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

$A + B = Y$

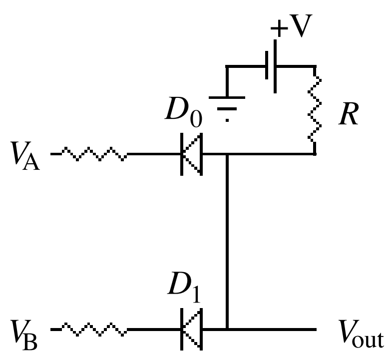
NOT (inverter)



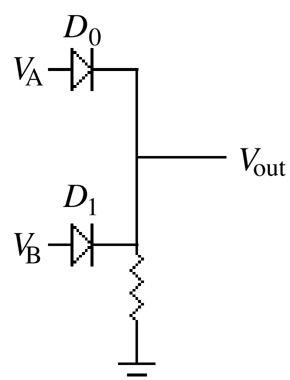

| A | Y |
|---|---|
| 0 | 1 |
| 1 | 0 |

$\overline{A} = Y$

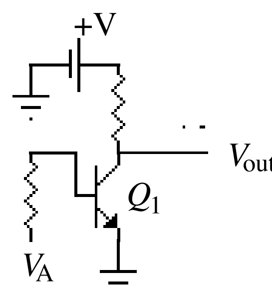
A and B stand for the inputs
Y stands for the output
0: low input or output
1: high input or output




$V_{out}$ = +V if both diodes off
= 0 if one or more diodes conduct




$V_{out} = V_A$ or $V_B$ if either diodes conduct
= 0 if both diodes are off




$V_A > 0.6$ V, $Q_1$ is on, $V_{out} \sim 0$
$V_A < 0.6$ V, $Q_1$ is off, $V_{out} = +V$

## Boolean Algebra or the Algebra of 1's and 0's

- Circuits consisting of logic gates are described by Boolean algebra.
  - Use of this algebra can greatly simplify circuit design, e.g. minimize the number of components.
- The following theorems can be proved using a truth table and the definition of OR, AND, and NOT.

1) $A + A = A,\ A + 1 = 1,\ A + 0 = A$

2) $AA = A$

3) $AB = BA$

4) $ABC = (AB)C = A(BC)$

5) $A(B + C) = AB + AC$

6) $\overline{1} = 0,\ \overline{0} = 1$

7) $A + \overline{A} = 1,\ A\overline{A} = 0,\ A \cdot 1 = A$

8) $\overline{\overline{A}} = A$

$$\left.\begin{array}{l} 9)\ \overline{A + B} = \overline{A} \cdot \overline{B} \\ 10)\ \overline{AB} = \overline{A} + \overline{B} \end{array}\right\} \text{DeMorgan's Theorem}$$

See Simpson page 540 for more theorems.

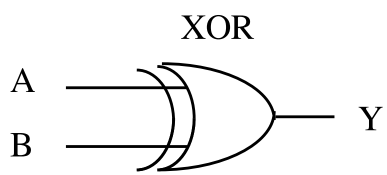
For clarity "·" (AND) is not shown in some theorems.

- Example using Boolean algebra:
  - **Prove:** $X + YZ = (X + Y)(X + Z)$

$$\begin{aligned} (X + Y)(X + Z) &= XX + XZ + YX + YZ && \text{by 5)} \\ &= X + X(Z + Y) + YZ && \text{by 2) and 5)} \\ &= X(1 + Z + Y) + YZ && \text{by 5)} \\ &= X + YZ && \text{by 1)} \end{aligned}$$

- We could also have proven the above using a truth table.
    - There are 8 ($2^3$) possible combinations of X, Y, Z.
    - For a large number of inputs using a truth table becomes unwieldy.
    - Example, if there are 10 inputs
        - $2^{10} = 1024$ possible combinations!

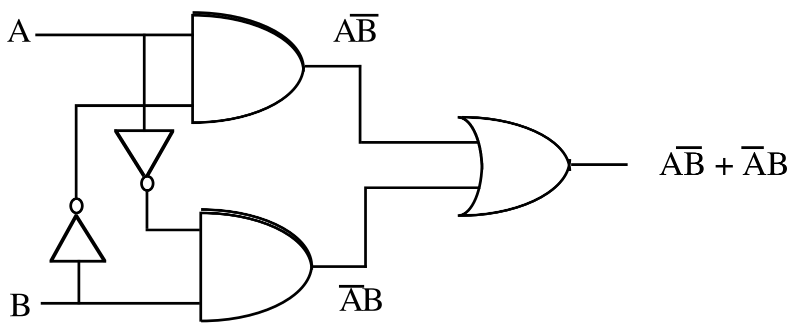
- Example: Exclusive OR = XOR = $A \oplus B$.
    - Output is high if inputs are different.




| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

$$A\bar{B} + \bar{A}B = Y$$

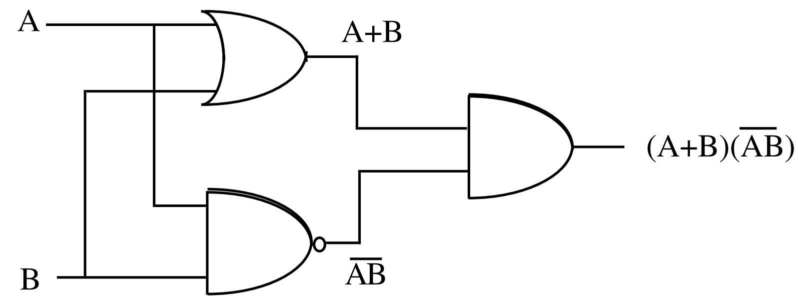
- How do we make an exclusive OR with AND, OR, and NOT gates?



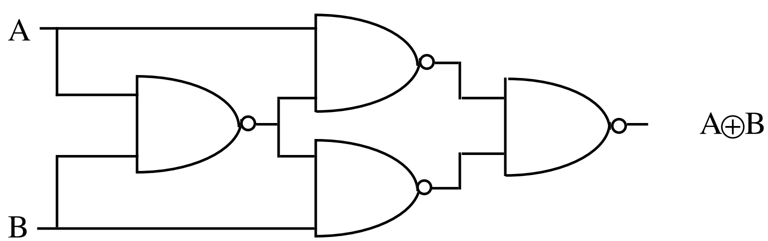

Brute force method

- Can we simplify this circuit with the use of less parts?
  - Use logical theorems:

$$A \oplus B = A\overline{B} + \overline{A}B$$
$$= A\overline{A} + A\overline{B} + \overline{A}B + B\overline{B} \quad 7) \text{ and } 1)$$
$$= A(\overline{A} + \overline{B}) + B(\overline{A} + \overline{B}) \quad 5)$$
$$= A(\overline{AB}) + B(\overline{AB}) \quad 10)$$
$$= (A + B)(\overline{AB}) \quad 5)$$

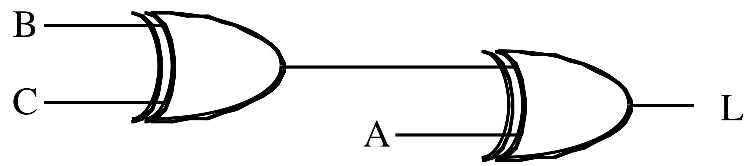



  - The circuit uses only 3 parts (OR, NAND, AND), but each of them is different!
  - Usually there are many ways to synthesize the same function (circuit).
  - Must decide if you want to minimize:
    - number of components
    - types of components
    - number of connections
    - power consumption
  - For example we can make an XOR using only 4 NAND gates:



- Final example: Suppose you have a light controlled by 3 switches.
  - You want the light to be on if any one of the 3 switches is on or if all 3 switches are on.

| A | B | C | Light |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 |




1 = ON

0 = OFF

$$
\begin{aligned}
L &= ABC + A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C \\
&= A(BC + \bar{B}\bar{C}) + \bar{A}(B\bar{C} + \bar{B}C) \\
&= A(B + \bar{C})(\bar{B} + C) + \bar{A}(B \oplus C) \\
&= A\overline{\bar{B}C}\,\overline{B\bar{C}} + \bar{A}(B \oplus C) \\
&= A\overline{\bar{B}C + B\bar{C}} + \bar{A}(B \oplus C) \\
&= A\overline{B \oplus C} + \bar{A}(B \oplus C) \\
&= A \oplus (B \oplus C)
\end{aligned}
$$

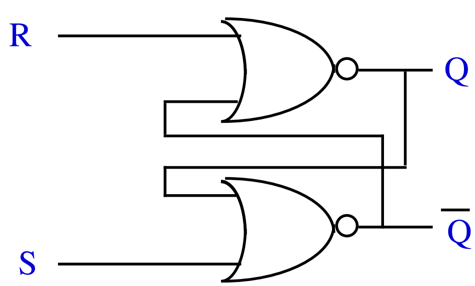
$$\overline{BC} = \bar{B} + \bar{C}$$

$$\overline{B + C} = \bar{B}\bar{C}$$

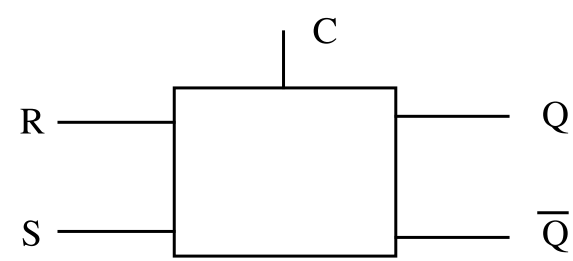
## Flip-Flops:

- Basic counting unit in computer:
  - counters
  - shift registers
  - memory
- Circuit whose output depends on the history of its inputs.
- Can make a flip-flop with just 2 transistors (or 2 vacuum tubes 1919!).
- Lots of different types of flip-flops (e.g. RS, JK, T, D).

- Example: RS flip-flop or <u>R</u>eset-<u>S</u>et flip-flop
  - Flip-flops, like logic gates are defined by their truth table.
  - Flip-flops are controlled by an external clock pulse.
  - All inputs and outputs are logic levels (e.g. TTL, ECL).
  - Can make an RSFF out of NOR gates:

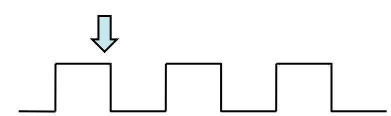



| R | S | $Q_{n+1}$ |
|---|---|---|
| 0 | 0 | $Q_n$ |
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 1 | undefined |

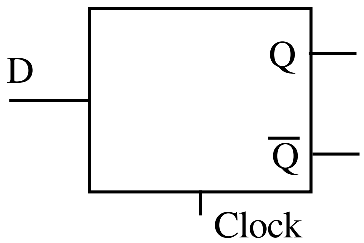



- $Q_n$ is the present state of the FF.
- $Q_{n+1}$ will be the output after the clock enables the FF to look at its inputs (R and S).
- Many FF change state ($Q_n \rightarrow Q_{n+1}$) on the trailing edge of the clock.

*The state with* R = S =1 *is undefined. The output is not predictable!*



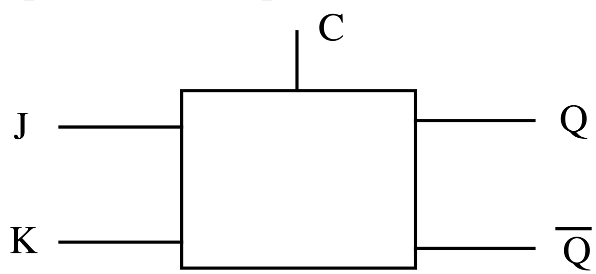
- Example: D flip-flop (Like RS but only one input)




| D | Q | $Q_{next}$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

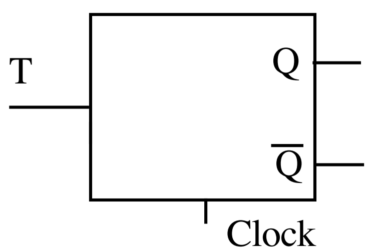
- Example: JK flip-flop
  - JKFF is like the RSFF except that both inputs (J and K) can be high (1).

| J | K | $Q_{n+1}$ |
|---|---|---|
| 0 | 0 | $Q_n$ |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 1 | $\overline{Q}_n$ |




  - Most JKFF' s have a connection for forcing Q = 0 (clear) or forcing Q = 1 (preset).

- Example: T (*Toggle*) flip-flop
  - T flip-flop is like the JKFF with both inputs (J and K) tied to each other.




| T | Q | $Q_{next}$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

- Flip-Flops are a class of circuits called "multivibrators"
  - Multivibrators are circuits with one or more stable states.
    - Monostable multivibrators (one shot) have one stable state.
      - If the circuit is forced out of its stable state (e.g. by an input pulse)
        - it eventually returns back to the stable state by itself.
    - Bistable multivibrators have two stable states.
      - Transitions between states occur only by an external action (e.g. voltage pulse for flip-flops).
      - Transition voltages can be different for the two states (e.g. Schmitt trigger).
    - Astable multivibrators are two-state devices which switch on their own accord.
      - Commonly used as oscillators.

- Example: Bistable Multivibrator.
  - This circuit has two stable states.
  - When either transistor conducts there is 4 mA flowing (by design) in the collector ($I_{C1}$ or $I_{C2}$).
  - State 1: transistor 1 off, transistor 2 on
    - $V_{C1} \approx 11$ V, $V_{B1} \approx 2$ V, $V_{E1} \approx 4$ V
    - $V_{C2} \approx 4$ V, $V_{B2} \approx 5.5$ V, $V_{E2} \approx 4$ V — $T_2$ is saturated ($V_{CE} \approx 0$)
  - Transition from state 1 to state 2:
    - Input pulse forces $T_1$ to conduct, $T_1$ conducting means that $V_{C1}$ drops.
    - $V_{C1}$ causes $V_{B2}$ to drop to the point where $T_2$ is not conducting.
  - State 2: transistor 1 on, transistor 2 off
    - $V_{C1} \approx 4$ V, $V_{B1} \approx 5.5$ V, $V_{E1} \approx 4$ V — $T_1$ is saturated ($V_{CE} \approx 0$)
    - $V_{C2} \approx 11$ V, $V_{B2} \approx 2$ V, $V_{E2} \approx 4$ V