

## Electronic Digital Logic Circuits

### 1- Electronic Digital Logic

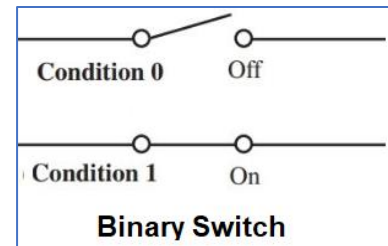
In digital computers, binary information is represented using electrical signals.

These signals voltage has two states only **high** or **low**.

Each state corresponds to a binary value: either **1** or **0**.

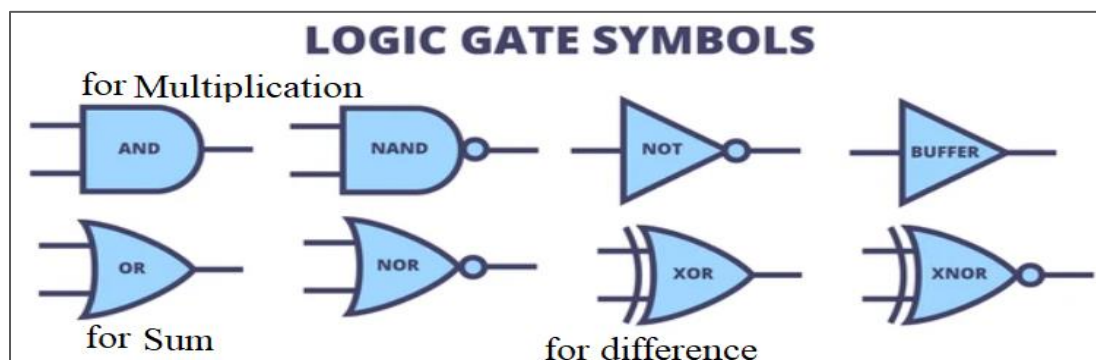
The binary data is processed using logic gates which is a **circuit that perform operations. Digital circuits used**

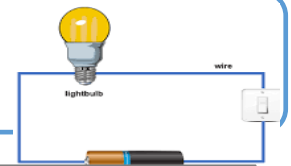
**in: Microprocessors, memory units, digital communication systems, and embedded devices.**



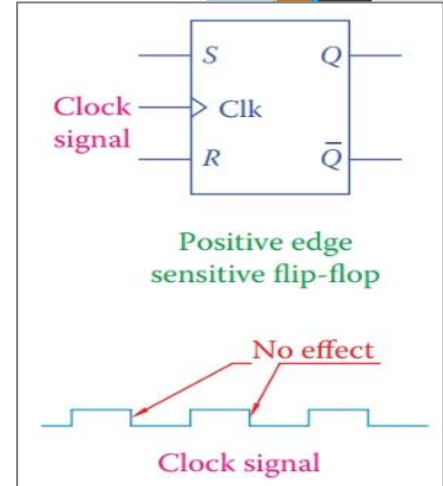
**1-1 A flip-flop** (FFs): Flip-flops are bistable electronic circuit that store one bit of data (0 or 1) used as a fundamental memory element in digital circuits. Flip-flops are **sequential circuits** built from combinations of logic gates (e.g. NAND, NOR) **A flip-flop** are edge-Triggered and only change state on a transition (rising or falling edge) of the clock signal: the **rising** edge (when clock signal transitions from low to **high**) or the **falling** edge (**high** to low). The flip-flop switches states when control inputs are activated  like SET or RESET otherwise it keeps current state without change. FFs used in CPUs, internal registers, counters, cache, and control logic.

The next figure show the Symbols and functions of Logic Gates





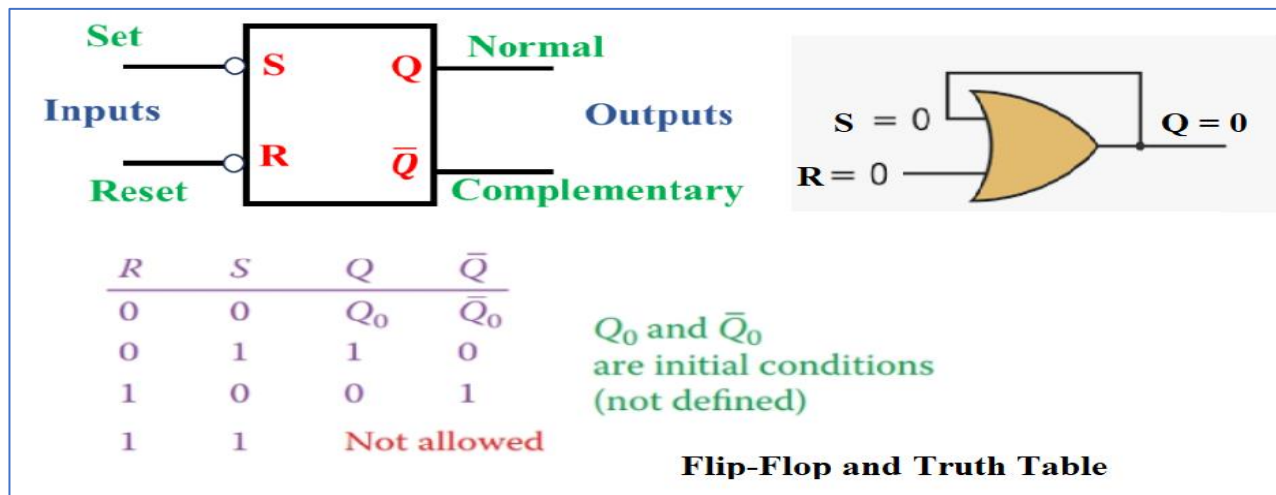
- The most basic flip-flop is called the **S-R (Set-Reset)**, it has two inputs used to **Set (S)** or **Reset (R)** the device. The two outputs are denoted by  $Q$  and  $\bar{Q}$ . they reflect the state of the **flip-flop 0 or 1**. If  $Q = 1$ , then  $\bar{Q} = 0$ , and if  $\bar{Q} = 0$ , then  $Q = 1$ . The SR- **flip-flop** used in simple control circuits e.g.(enable/disable) signal, latch, which is **simplest circuit that can store 1 bit (0 or 1)**.

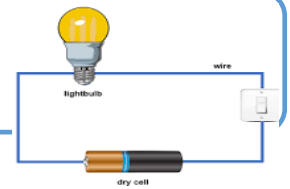


- While The Clocked controlled RS flip-flop**, add clock to the input, to updates stored bit only at the clock **rising** pulse, it is in synchronism with all other flip-flops operations in the electronic circuit. Therefore, used for microprocessor-controlled systems.
- D-flip-flop (Data or Delay)** used for registers and memory to stores the input data at the clock edge.
- T (Toggle) flip-flop** Used for counting and frequency division.
- J-K flip-flop** used in complex logic design.

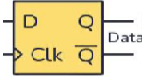
The differences in flip-flop types are how the inputs and clock signal interact to determine the output.

The input-output relationship of binary variables for each gate is represented in **Truth-Table**. The next **Figure** show the S-R-flip-flop circuit and the truth table for the states of a flip-flop.





Where the flip-flops are used in a CPU ?



- Registers:

A register is a group of flip-flops, with each flip-flop capable of storing one bit. Registers within the CPU, such as the program counter or instruction register, use flip-flops to store data and addresses that the CPU needs to access and process.

- Cache memory:

The CPU contain on-chip fast **cache** memory, particularly a Static RAM (SRAM), is built using flip-flops to store data. This allows rapid access to frequently used information, and improv the CPU performance.

- Control logic:

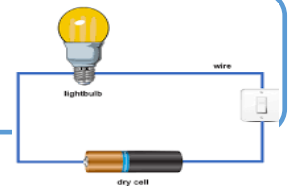
The control unit of a CPU uses flip-flops and other logic gates to manage the sequence of operations. These circuits interpret instructions and control signals, ensuring that the CPU performs tasks in the correct order.

**Tip:** Flip-flops are built from gates, and both are combined in various ways to create functional blocks inside the microprocessor.

## 2-CPU Digital Circuits:

Integrated circuits in the CPU contain both **digital circuits** that include digital logic and sequential logic components for fast data processing:

- **Digital logic:** electronic circuits built from basic digital logic gates like AND, OR, and NOT, it processes binary signals **0s and 1s**. this circuits used to perform fast binary operations for arithmetic and decision making, like calculations, compare values, and execute instructions.
- **Sequential logic:** in this circuit, the output depends on the current input values and the past sequence of input values. The Control Unit (CU), consist of **sequential logic** to manage the instruction **sequencing** and data storage. The D-Type Flip-Flop



 are the main components for all sequential logic circuits, it works as memory to store binary data depend on the sequence of previous inputs to determine their current output. The RAM **memory cells** are sequential circuits.

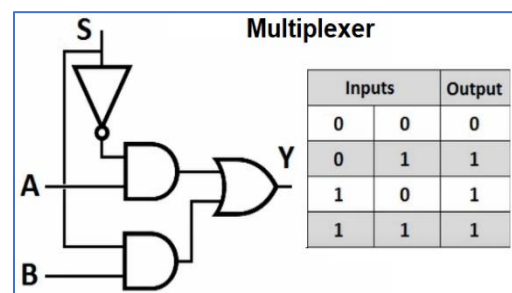
- A **combinational logic circuit**: a digital electronic circuit whose output is determined by current inputs only. It does not have a memory element in its circuit that stores previous inputs and outputs. Combinational circuits are very easy to implement since it does not need memory. This circuits perform mathematical tasks like arithmetic and logical operations, it is the main component of the ALU

### Some common types of combinational circuits:

#### **Binary Adders, Binary Subtractors, Multiplexers (MUX) Demultiplexers (DEMUX), Encoders, Decoders, Comparators**

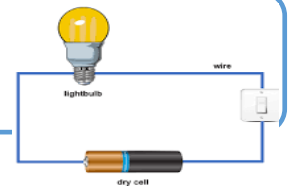
- **Binary Adders**: Circuits perform the addition of two binary numbers of multiple bits, including carry-in. Adder is a core component of ALUs for processing binary data
- **Binary Subtractors**: Digital circuits designed to calculate the difference between two binary numbers. Implement using adders by taking the two's complement of the subtrahend and performing addition.

- **Multiplexers (MUX)**: Select one of several input lines, routes the data to a single output line. Used to select between different data sources or instructions in microprocessors and for multiplexing data for transmission



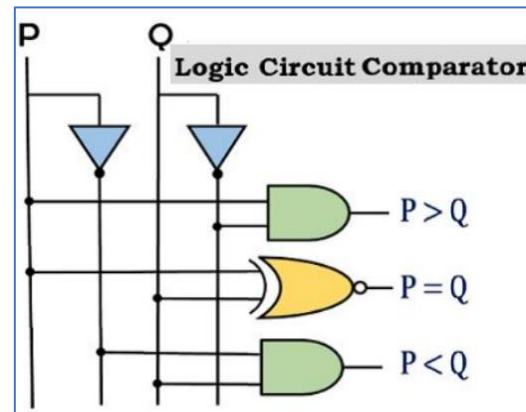
over a single channel. Multiplexers are also known as "N-to-1 selectors, or  $2^n$  input lines, n selection lines are needed, its **many to one** circuit.

**Note:** DEMUX is **not** a multiplier circuit, it can be part of the signal **routing** and **control in arithmetic circuits**.



- **Demultiplexers (DEMUX)** The opposite of a multiplexer; it takes data from a single input line and directs it to **one to- many output** lines.
- **Encoders:** Converts multiple input binary codes into a smaller, encoded output. Used for code conversion, such as converting binary to BCD (Binary-Coded Decimal)
- **Decoders:** Converts coded binary information from n input signals into a **maximum of  $2^n$**  unique output signals. Used for instruction decoding, address decoding for memory
- **Comparators:** A device that compares two analog voltages or currents and outputs a digital signal indicating their relationship (e.g., one is greater than the other). **Types of Comparator Circuits:**

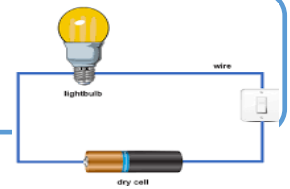
- **Identity Comparator:** Has one output that signals if the two inputs are equal.
- **Magnitude Comparator:** Has three outputs indicating if one input is greater than, less than, or equal to the other.



**Tip:** - AND gate used for binary multiplication.  
- XOR, AND, NOT gates used for addition and subtraction.  
- Division requires a combination of gates in a more complex circuit.

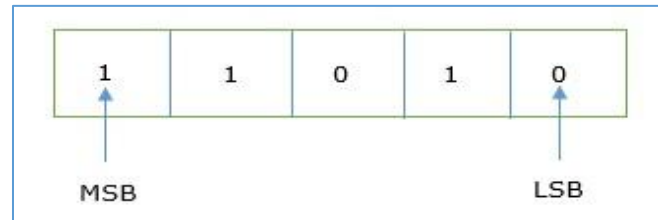
### 3-Number System

The technique to represent and work with numbers is called **number system**. The most common number system is **Decimal number system** is base 10 number system having 10 digits from 0 to 9. Other popular number systems include **binary number system**, **octal number system**, **hexadecimal number system**.



- **Binary numbers:** are numbers represented in a base-2 numeral system, using only the digits 0 and 1, called bits. These numbers form the fundamental language of computers, representing data through electrical signals that are either "on" (1) or "off" (0). Each bit corresponds to a power of two.

In binary number, the rightmost digit is called **least significant bit** (LSB) and leftmost digit is called **most significant bit** (MSB).



**Note:** In a binary number, bits are arranged from left to right. **MSB**  $\longrightarrow$  **LSB**

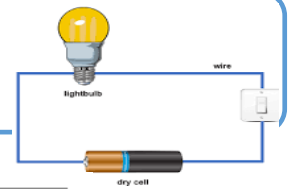
The binary equivalent for some decimal numbers are given below.

|         |   |   |    |    |     |     |     |     |      |      |      |      |
|---------|---|---|----|----|-----|-----|-----|-----|------|------|------|------|
| Decimal | 0 | 1 | 2  | 3  | 4   | 5   | 6   | 7   | 8    | 9    | 10   | 11   |
| Binary  | 0 | 1 | 10 | 11 | 100 | 101 | 110 | 111 | 1000 | 1001 | 1010 | 1011 |

- **Octal Number System:** has eight digits 0, 1, 2, 3, 4, 5, 6 and 7. Then Number 8<sub>d</sub> is represented in as "1000<sub>b</sub>, =10<sub>8</sub> 9=1001<sub>b</sub> =11<sub>8</sub>..etc Octal number system is also a positional value system, each digit has its value expressed in powers of 8 (e.g., 8<sup>0</sup>, 8<sup>1</sup>, 8<sup>2</sup> etc.) Example the number 351 is=(3×8<sup>2</sup>)+ (5×8<sup>1</sup>)+ (1×8<sup>0</sup>). It is used in computer programming as a compact representation of binary numbers, where each octal digit corresponds to three binary digits.

|                          |     |     |     |     |     |     |     |     |
|--------------------------|-----|-----|-----|-----|-----|-----|-----|-----|
| <b>Octal Digit</b>       | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   |
| <b>Binary Equivalent</b> | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

- The **hexadecimal number system:** is a type of number system, that has a base value equal to 16. It is also pronounced sometimes as 'hex'. Hexadecimal numbers are represented by only 16 symbols. These symbols or values are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E and F. Each digit represents a decimal value. For example, D is equal to base-10 13.



| decimal digit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---------------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| hex digit     | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A  | B  | C  | D  | E  | F  |

- **Why Hexadecimal used in computer system:**

- Hex expressing large binary values with fewer digits, reducing errors.
- Each hex digit corresponds directly to 4 bits, so the memory addresses that would be long in binary are shorter and easier to read in hex.
- Machine code, assembly language addresses, and error codes often use hex to simplify reading and debugging.
- Hexadecimal codes define colors on web pages, **like #FF5733 in RGB**

- **BCD (Binary-Coded Decimal):** The BCD code converts each decimal digit to binary individually encoded by its 4-bit binary equivalent; it represents decimal digit (0-9). Example The number **10<sub>d</sub>** is **0001 0000**. This system is used in **calculators, cash registers, and accounting systems.**

| Decimal Digit | 0    | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    | 9    |
|---------------|------|------|------|------|------|------|------|------|------|------|
| BCD           | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 |

**Note: No Unused Codes:** In standard BCD, binary codes (10<sub>d</sub>)1010 through 1111 (15) are not used.

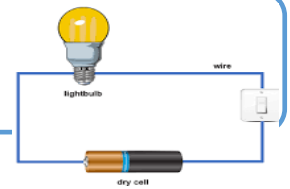
**Note:** The BCD number is not the same as a binary number. The binary code represents the complete decimal number in binary.

**Eexample:** convert the number 137 to binary and BCD codes.

$$137_{(10)} = 10001001_{(2)} \text{ (Binary)} , 137_{(10)} = 001 \mathbf{110} \mathbf{111} \text{ (BCD)}$$

**Test:** convert number 10 to binary, octal, hex, BCD

$$\text{Sol.: } 10_d = 1010_2, 12_8, A_{16}, \mathbf{10000}_{\text{BCD}}$$



#### 4-Odd and Even number in computer system

In computer systems, odd and even numbers are represented in binary, and their parity (odd or even) is determined by the value of the least significant bit (LSB), which is the rightmost bit in binary, representing the smallest value ( $2^0=1$ ).

- Even numbers always have their least significant bit (LSB) as 0. **Example:** Decimal 4  $\rightarrow$  Binary 100, LSB = 0  $\rightarrow$  Even.
- Odd numbers always have their least significant bit (LSB) as 1. **Example:** Decimal :5  $\rightarrow$  Binary 101, LSB = 1  $\rightarrow$  Odd.

**5- Signed numbers:** signed numbers are a way of representing both positive and negative integers using the binary system. **Two methods are used to represent the negative number:**

- **One's complement** is bitwise NOT operation. change every 0 to 1 and every 1 to 0. To represent the negative of a binary number
- **Two's complement:** represent signed integers in modern computers. 2's Complement Invert all bits and then add 1 to the **LSB** convert a **positive** number into its **negative** equivalent. In two's complement, the most significant bit (**LSB**) is the **sign** bit. If the bit is 0 the number is positive, else its negative.

**Note:** 2's complement also used in computer hardware for both addition and subtraction. The system is designed so that a single binary addition circuit can perform both operations, because it simplifies the hardware needed for arithmetic operation of the Arithmetic Logic Unit (ALU)

**Test** Convert 5d (0000 0101) to 2' complement

**Solution:** **Step1:** Take 1's complement: 0000 0101  $\rightarrow$  1111 1010

**Step 2:** Add 1: 1111 1010 + 1 = 1111 1011

= **251** which is  $(1 \times 2^7) + (1 \times 2^6) + (1 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$   
= 128 + 64 + 32 + 16 + 8 + 0 + 2 + 1 = 251