Digital Electronic Circuits

Cyber Security Department

Lecturer 1

Major concepts in digital electronic circuits

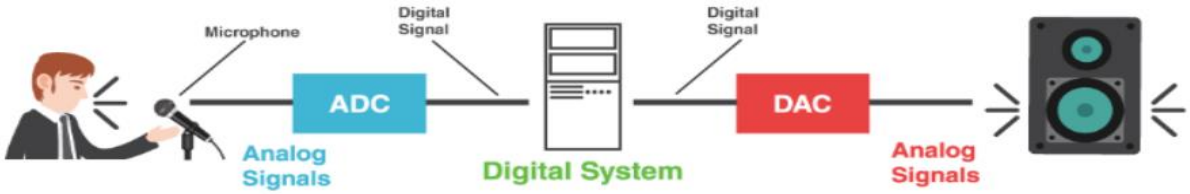
- ❖ Digital and Analog Quantities
  - An Analog System
  - A System Using Digital and Analog Methods
- ❖ Binary Digits, Logic Levels, and Digital Waveforms
  - Binary Digits
  - Logic Levels
  - Digital Waveforms

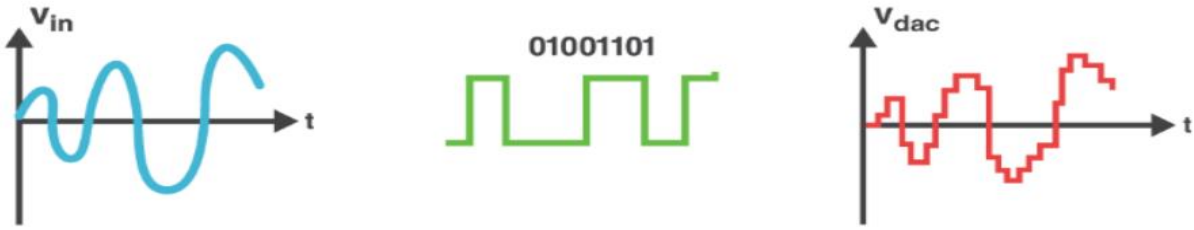
❖ Digital and Analog Quantities

Electronic circuits can be divided into two broad categories digital and analog.

Digital electronics involves quantities with discrete values.

analog electronics involves quantities with continuous values.

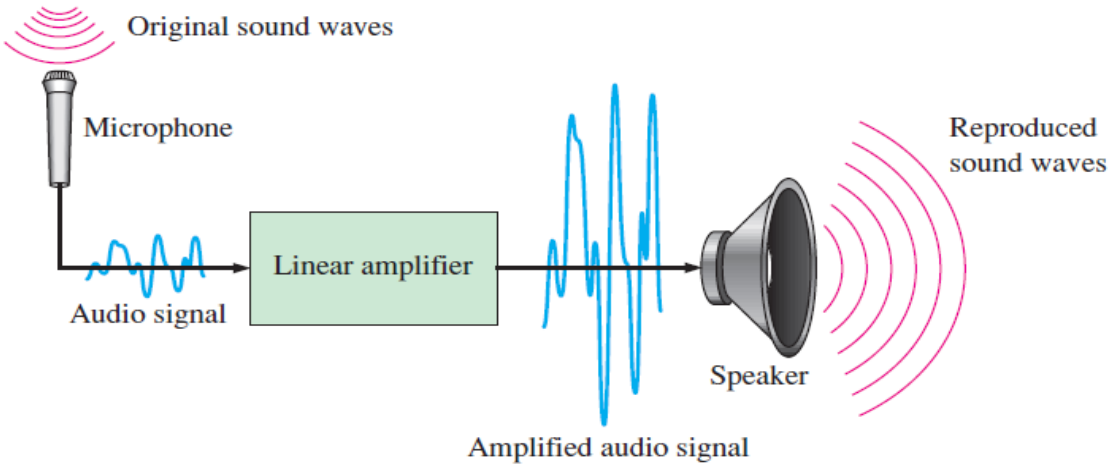




An Analog System

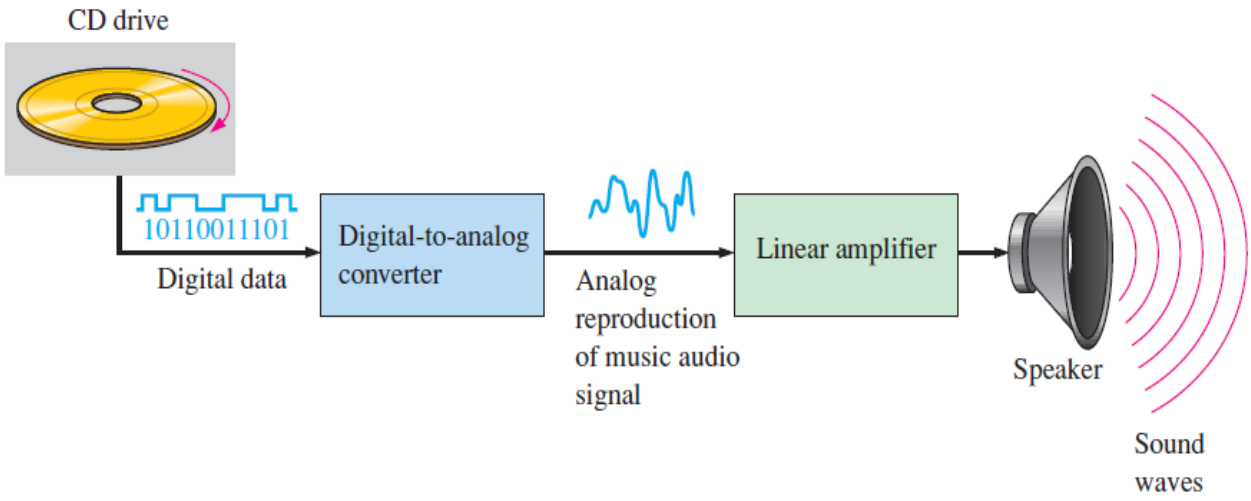
one simple example of an application of analog electronics is sound waves, which are analog in nature, are picked up by a microphone and converted to a small analog voltage called the audio signal. This voltage varies continuously as the volume and frequency of the sound changes and is applied to the input of a linear amplifier.

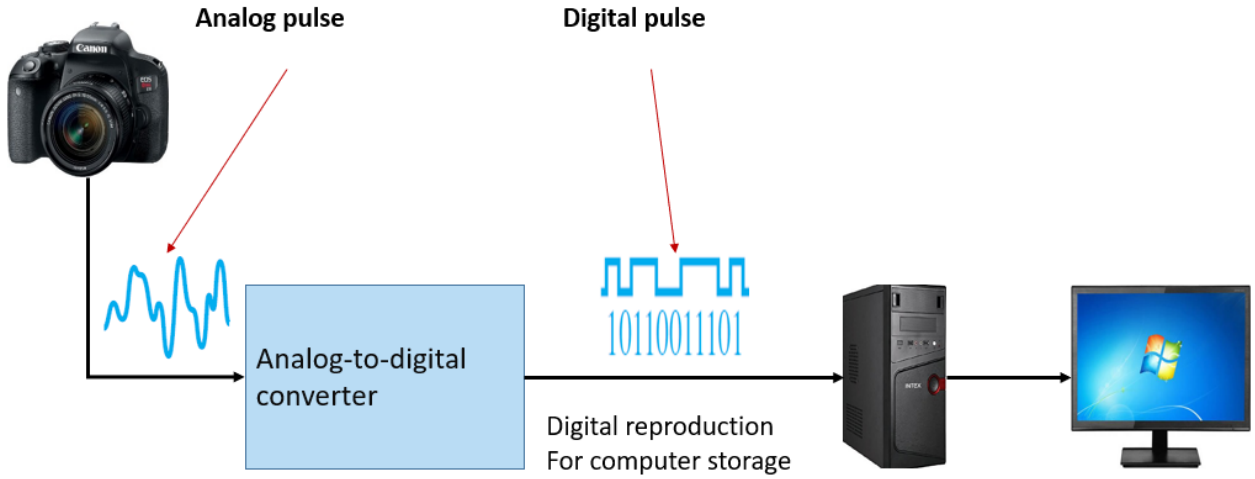
The output of the amplifier goes to the speaker(s). The speaker changes the amplified audio signal back to sound waves that have a much greater volume than the original sound waves picked up by the microphone.



A System Using Digital and Analog Methods

The compact disk (CD) player is an example of a system in which both digital and analog circuits are used. Music in digital form is stored on the compact disk. A laser diode optical system picks up the digital data from the rotating disk and transfers it to the **digital-to-analog converter (DAC)**. The DAC changes the digital data into an analog signal that is an electrical reproduction of the original music. This signal is amplified and sent to the speaker for you to enjoy. When the music was originally recorded on the CD, a process, essentially the reverse of the one described here, using an **analog-to-digital converter (ADC)** was used.





❖ Binary Digits, Logic Levels, and Digital Waveforms

Binary Digits

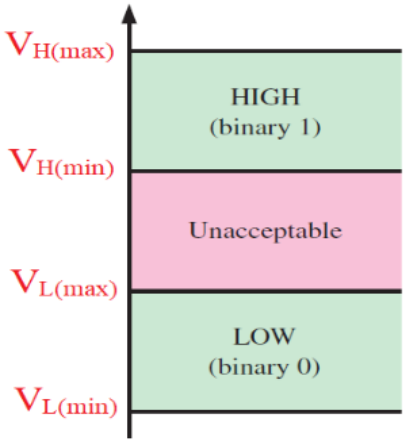
Each of the two digits in the **binary** system, 1 and 0, is called a **bit**, which is a contraction of the words *binary digit*. In digital circuits, two different voltage levels are used to represent the two bits. Generally, 1 is represented by the higher voltage, which we will refer to as a HIGH, and a 0 is represented by the lower voltage level, which we will refer to as a LOW.

HIGH _ 1 and LOW _ 0

Logic Levels

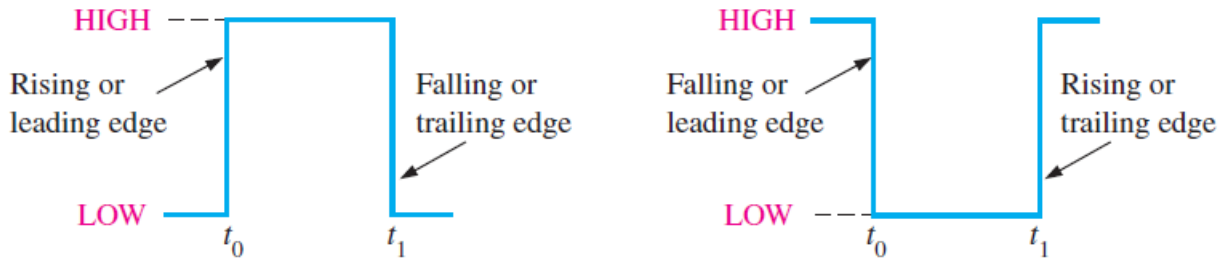
The voltages used to represent a 1 and a 0 are called *logic levels*. Ideally, one voltage level represents a HIGH and another voltage level represents a LOW. In a practical digital circuit, however, a HIGH can be any voltage between a specified minimum value and a specified maximum value. Likewise, a LOW can be any voltage between a specified minimum and a specified maximum. There can be no overlap between the accepted range of HIGH levels and the accepted range of LOW levels

Logic Level	Voltage	True/False	On/Off	0/1
HIGH	5 volts	True	On	1
LOW	0 volts	False	Off	0



Digital Waveforms

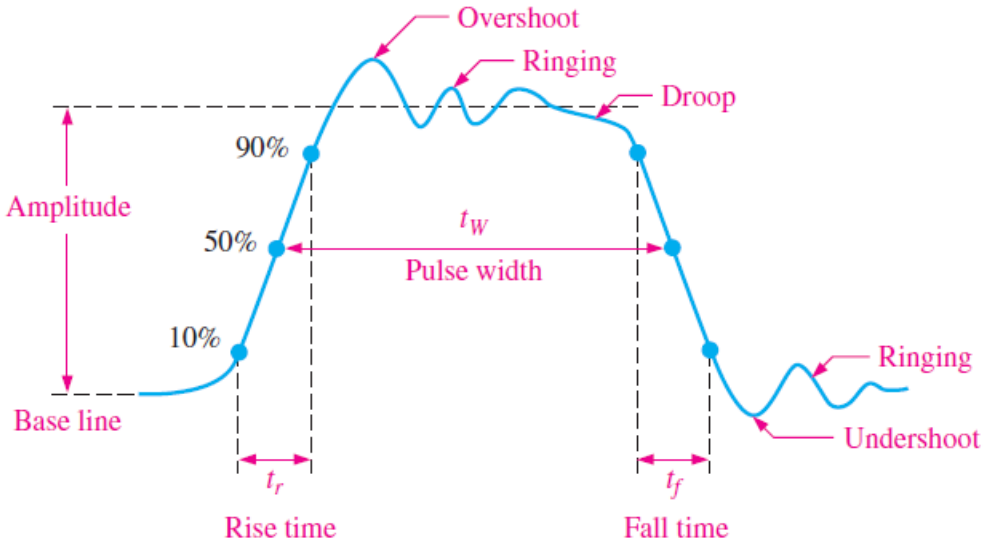
Digital waveforms consist of voltage levels that are changing back and forth between the HIGH and LOW levels or states. Figure (a) shows that a single positive-going **pulse** is generated when the voltage (or current) goes from its normally LOW level to its HIGH level and then back to its LOW level. The negative-going pulse in Figure (b) is generated when the voltage goes from its normally HIGH level to its LOW level and back to its HIGH level. A digital waveform is made up of a series of pulses.



(a) Positive–going pulse (b) Negative–going pulse

- The Pulse

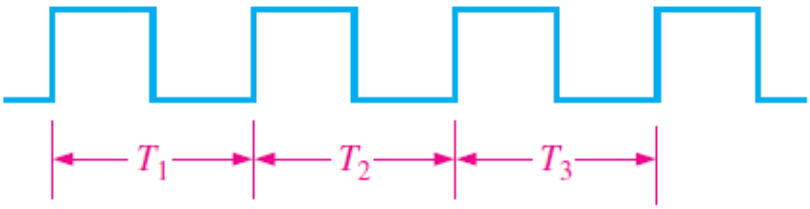
A pulse has two edges: a **leading edge** that occurs first at time *t*0 and a **trailing edge** that occurs last at time *t*1. For a positive-going pulse, the leading edge is a rising edge, and the trailing edge is a falling edge. The pulses are ideal because the rising and falling edges are assumed to change in zero time (instantaneously).



The time required for a pulse to go from its LOW level to its HIGH level is called the **rise time (*tr*)**, and the time required for the transition from the HIGH level to the LOW level is called the **fall time (*tf*)**. In practice, it is common to measure rise time from 10% of the pulse **amplitude** (height from baseline) to 90% of the pulse amplitude and to measure the fall time from 90% to 10% of the pulse amplitude. The **pulse width (*tW*)** is a measure of the duration of the pulse and is often defined as the time interval between the 50% points on the rising and falling edges.

Waveform Characteristics

Most waveforms encountered in digital systems are composed of series of pulses, sometimes called *pulse trains*, and can be classified as either periodic or nonperiodic. A **periodic** pulse waveform is one that repeats itself at a fixed interval, called a **period (*T*)**. The **frequency(*f*)** is the rate at which it repeats itself and is measured in hertz (Hz). A nonperiodic pulse waveform, of course, does not repeat itself at fixed intervals and may be composed of pulses of randomly differing pulse widths and/or randomly differing time intervals between the pulses.



Period $= T_1 = T_2 = T_3 = \ldots = T_n$

Frequency $= \frac{1}{T}$

(a) Periodic (square wave)

(b) Nonperiodic

The frequency (f) of a pulse (digital) waveform is the reciprocal of the period. The relationship between frequency and period is expressed as follows:

$$f = \frac{1}{T} \qquad T = \frac{1}{f}$$

An important characteristic of a periodic digital waveform is its **duty cycle**, which is the ratio of the pulse width (tW) to the period (T).

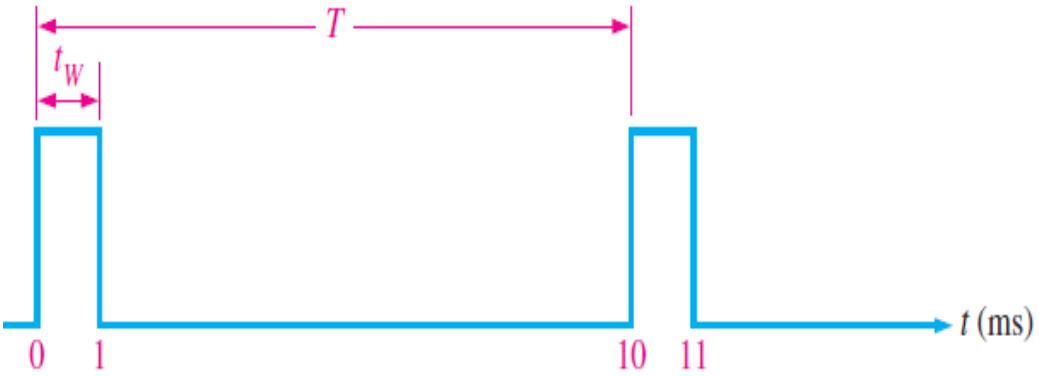
$$\text{Duty cycle} = \left(\frac{t_W}{T}\right)100\%$$

Example:

A portion of a periodic digital waveform is shown in Figure. The measurements are in milliseconds.

Determine the following:

(a) period **(b)** frequency **(c)** duty cycle



Solution

(a) The period (T) is measured from the edge of one pulse to the corresponding edge of the next pulse. In this case T is measured from leading edge to leading edge, as indicated. T equals **10 ms**.

(b) $$f = \frac{1}{T} = \frac{1}{10\text{ ms}} = \mathbf{100\ Hz}$$

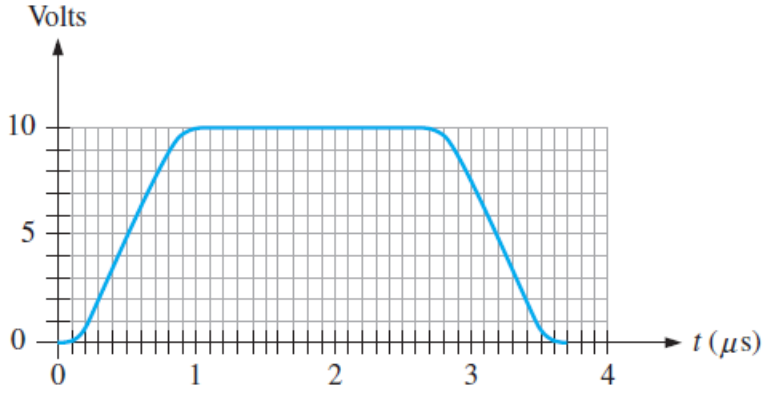
(c) $$\text{Duty cycle} = \left(\frac{t_W}{T}\right)100\% = \left(\frac{1\text{ ms}}{10\text{ ms}}\right)100\% = \mathbf{10\%}$$

Homework

1- A periodic digital waveform has a pulse width of 25 ms and a period of 150 ms. Determine the frequency and the duty cycle.

2- For the pulse shown in Figure, graphically determine the following:

(a) rise time **(b)** fall time **(c)** pulse width **(d)** amplitude



Summary

- An analog quantity has continuous values.
- A digital quantity has a discrete set of values.
- A binary digit is called a bit.
- A pulse is characterized by rise time, fall time, pulse width, and amplitude.
- The frequency of a periodic waveform is the reciprocal of the period. The formulas relating

frequency and period are

$$f = \frac{1}{T} \quad \text{and} \quad T = \frac{1}{f}$$

- The duty cycle of a pulse waveform is the ratio of the pulse width to the period, expressed by

the following formula as a percentage:

$$\text{Duty cycle} = \left(\frac{t_W}{T}\right)100\%$$

Digital Electronic Circuits

Cyber Security Department

Lecturer 2

❖ Basic Logic Functions

LOGIC GATES:

- NOT GATE (INVERTER)
- AND GATE
- OR GATE
- NAND GATE
- EXCLUSIVE – OR (X-OR) GATE
- EXCLUSIVE – NOR (X-NOR) GATE

❖ Basic Logic Functions

In its basic form, logic is the realm of human reasoning that tells you a certain proposition (declarative statement) is true if certain conditions are true. Propositions can be classified as true or false.

LOGIC GATES:

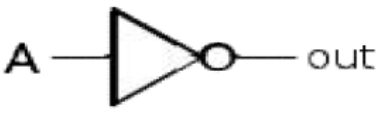
- Logic gates are the fundamental building blocks of digital systems.
- There are 3 basic types of gates AND, OR and NOT.
- Logic gates are electronic circuits because they are made up of a number of electronic devices and components.
- Inputs and outputs of logic gates can occur only in 2 levels. These two levels are termed HIGH and LOW, or TRUE and FALSE, or ON and OFF or simply 1 and 0.

DIFFERENT TYPES OF LOGIC GATES

NOT GATE (INVERTER):

- A NOT gate, also called and inverter, has only one input and one output.
- It is a device whose output is always the complement of its input.
- The output of a NOT gate is the logic 1 state when its input is in logic 0 state and the logic 0 state when its inputs is in logic 1 state.

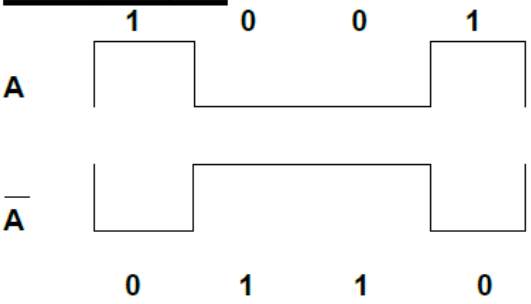
Logic Symbol



Truth table

INPUT A	OUTPUT $\overline{A}$
0	1
1	0

Timing Diagram



AND GATE:

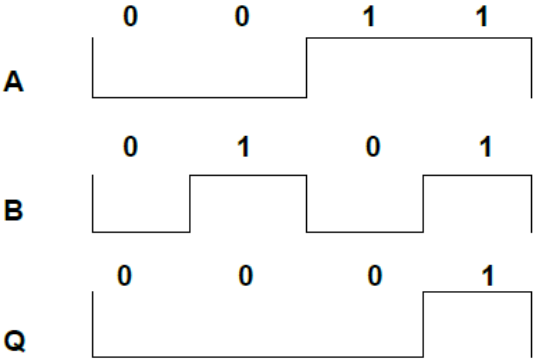
- An AND gate has two or more inputs but only one output.
- The output is logic 1 state only when each one of its inputs is at logic 1 state.
- The output is logic 0 state even if one of its inputs is at logic 0 state.

Logic Symbol

Truth Table

		OUTPUT
A	B	Q=A . B
0	0	0
0	1	0
1	0	0
1	1	1

Timing Diagram



Note

The total number of possible combinations of binary inputs to a gate is determined by the following formula:

$$N = 2^n$$

Where **N** is the number of possible input combinations and **n** is the number of input variables. To illustrate,

For two input variables: $N = 2^2 = 4$ combinations
For three input variables: $N = 2^3 = 8$ combinations
For four input variables: $N = 2^4 = 16$ combinations

OR GATE:

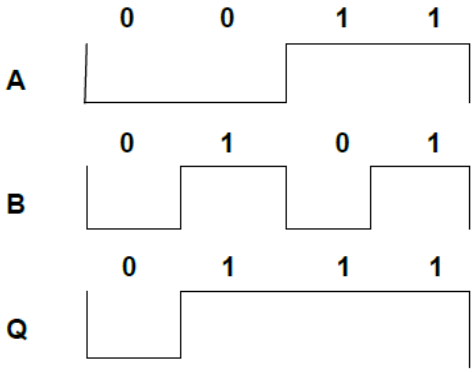
- An OR gate may have two or more inputs but only one output.
- The output is logic 1 state, even if on
- The output is logic 0 state, only when each one of its inputs is in logic state.

Logic Symbol

Truth Table

INPUT		OUTPUT
A	B	Q=A + B
0	0	0
0	1	1
1	0	1
1	1	1

Timing Diagram



NAND GATE:

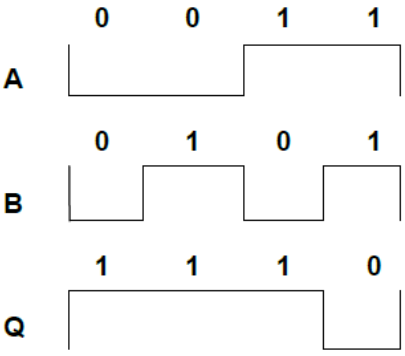
- NAND gate is a combination of an AND gate and a NOT gate.
- The output is logic 0 when each of the input is logic 1 and for any other combination of inputs, the output is logic 1.

Logic Symbol

Truth Table

INPUT		OUTPUT
A	B	Q= $\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Timing Diagram



NOR GATE:

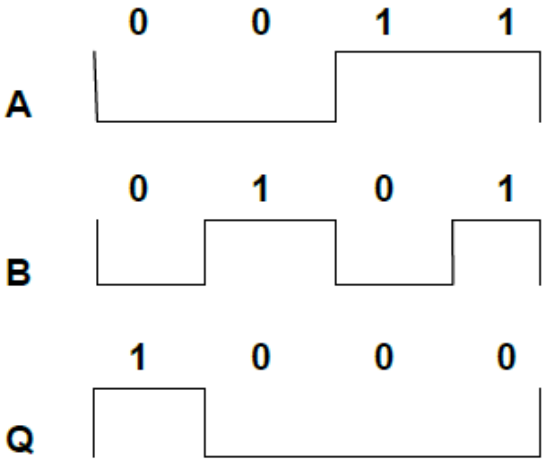
- NOR gate is a combination of an OR gate and a NOT gate.
- The output is logic 1, only when each one of its input is logic 0 and for any other combination of inputs, the output is a logic 0 level.

Logic Symbol

Truth Table

INPUT		OUTPUT
A	B	$Q= \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

Timing Diagram



EXCLUSIVE – OR (X-OR) GATE:

- An X-OR gate is a two input, one output logic circuit.
- The output is logic 1 when one and only one of its two inputs is logic 1. When both the inputs is logic 0 or when both the inputs is logic 1, the output is logic 0.

Logic Symbol

INPUTS are **A** and **B**

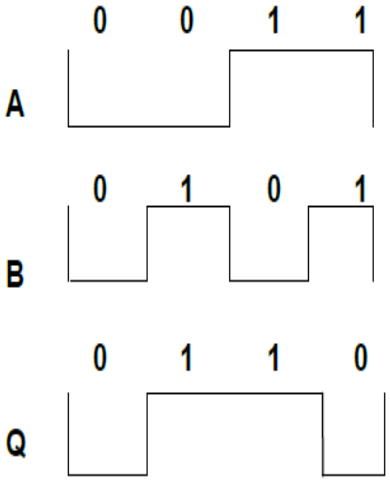
OUTPUT is **Q** = A ⊕ B

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

Truth Table

INPUT		OUTPUT
A	B	Q = A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

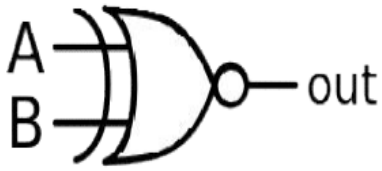
Timing Diagram



EXCLUSIVE – NOR (X-NOR) GATE:

- An X-NOR gate is the combination of an X
- An X-NOR gate is a two input, one output logic circuit.
- The output is logic 1 only when both the inputs
- The output is logic 0 when one of the inputs is logic 0 and other is 1.

Logic Symbol

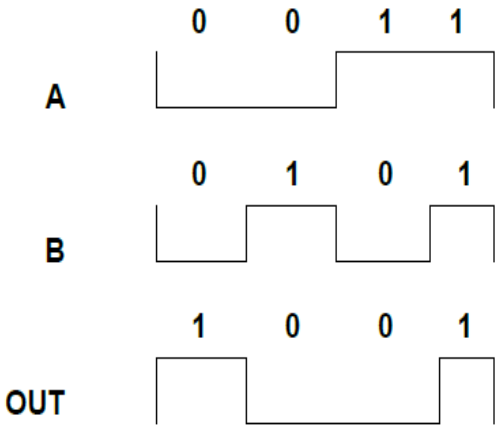


INPUT		OUTPUT
A	B	OUT =A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

OUT =A B + $\overline{A}$ $\overline{B}$

= A XNOR B

Timing Diagram



Example:

(a) Develop the truth table for a 3-input AND gate.

(b) Determine the total number of possible input combinations for a 4-input AND gate.

Solution

(a) There are eight possible input combinations ($2^3 = 8$) for a 3-input AND gate. The input side of the truth table shows all eight combinations of three bits. The output side is all 0s except when all three input bits are 1s.

Inputs			Output
A	*B*	*C*	*X*
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

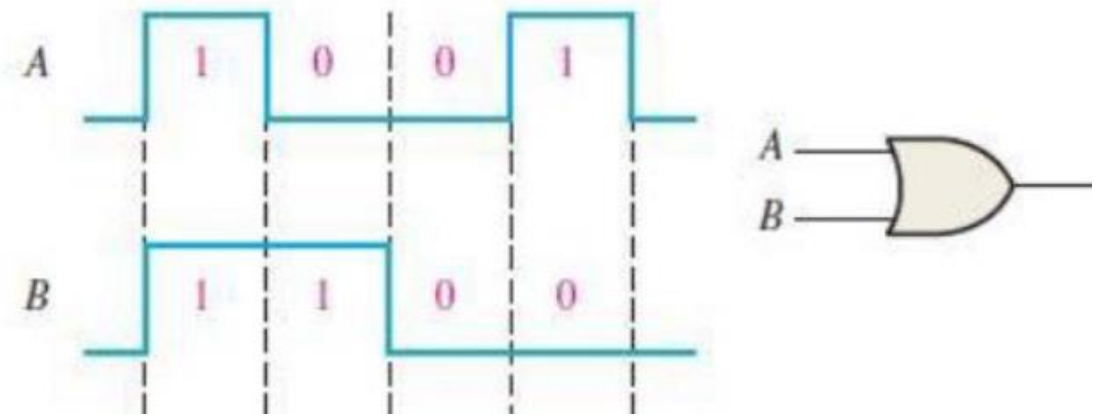
(b) $N = 2^4 = 16$. There are 16 possible combinations of input bits for a 4-input AND gate.

Related Problem

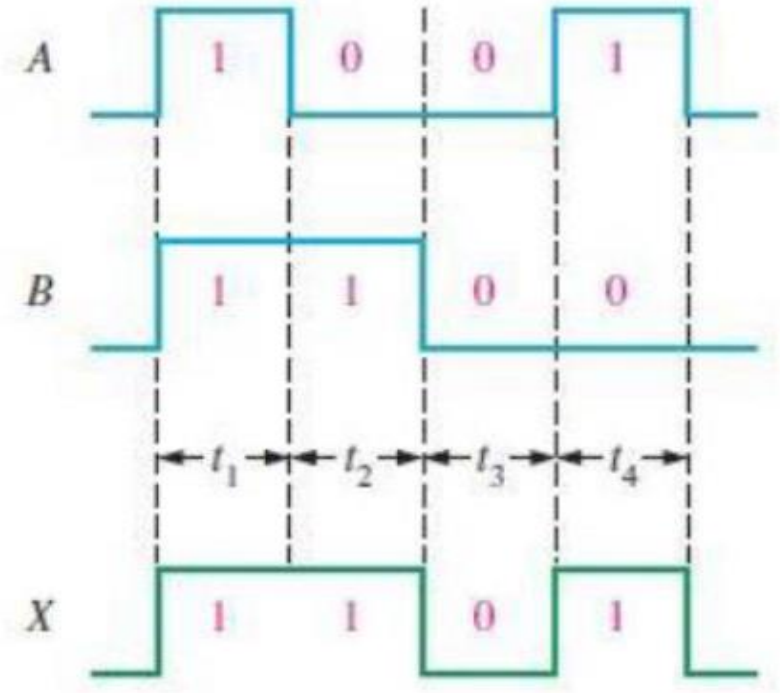
Develop the truth table for a 4-input AND gate.

Example:

For the two input waveforms A , B figure below shows the output waveform with its proper relation to the inputs.

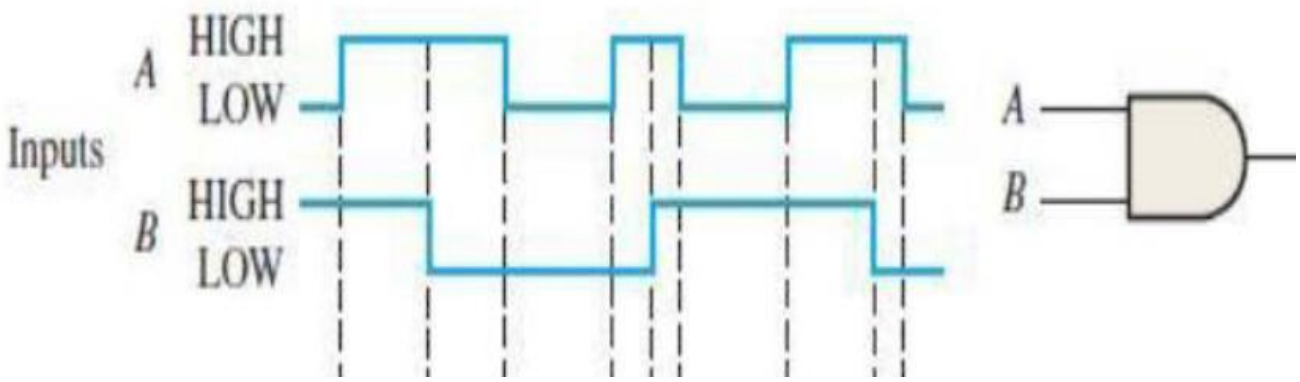


Solution:

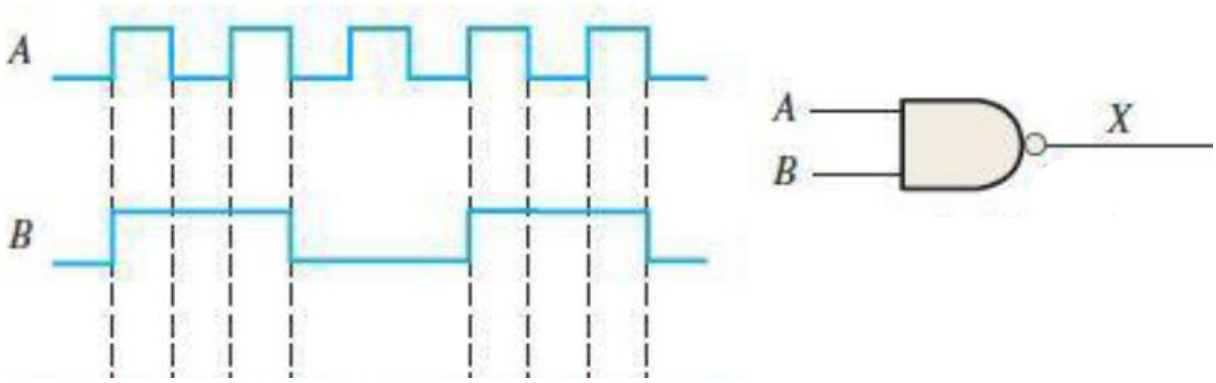


Homework:

1- For the two input waveforms A , B figure below shows the output waveform with its proper relation to the inputs.



2- For the two input waveforms A , B figure below shows the output waveform with its proper relation to the inputs.



3- Describe the truth table for a 3-input OR gate.

Digital Electronic Circuits

Cyber Security Department

Lecturer 3

Boolean Algebra and Logic Simplification

- ❖ Laws of Boolean Algebra
- ❖ Rules of Boolean Algebra
- ❖ Boolean Analysis of Logic Circuits

Variable, complement, and *literal* are terms used in Boolean algebra. A **variable** is a symbol (usually an italic uppercase letter or word) used to represent an action, a condition, or data. Any single variable can have only a 1 or a 0 value. The **complement** is the inverse of a variable and is indicated by a bar over the variable (overbar). For example, the complement of the variable A is $\overline{A}$. If $A = 1$, then $\overline{A} = 0$. If $A = 0$, then $\overline{A} = 1$. A **literal** is a variable or the complement of a variable.

❖ Laws of Boolean Algebra

✓ Commutative Laws

$$A + B = B + A$$

$$AB = BA$$

✓ Associative Laws

$$A + (B + C) = (A + B) + C$$

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

✓ Distributive Law

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

❖ Rules of Boolean Algebra

1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

Rule 10: $A + AB = A$

This rule can be proved by applying the distributive law, rule 2, and rule 4 as follows:

$$\begin{aligned} A + AB &= A \cdot 1 + AB = A(1 + B) && \text{Factoring (distributive law)} \\ &= A \cdot 1 && \text{Rule 2: } (1 + B) = 1 \\ &= A && \text{Rule 4: } A \cdot 1 = A \end{aligned}$$

Rule 11: $A + \bar{A}B = A + B$

This rule can be proved as follows:

$$\begin{aligned} A + \bar{A}B &= (A + AB) + \bar{A}B && \text{Rule 10: } A = A + AB \\ &= (AA + AB) + \bar{A}B && \text{Rule 7: } A = AA \\ &= AA + AB + A\bar{A} + \bar{A}B && \text{Rule 8: adding } A\bar{A} = 0 \\ &= (A + \bar{A})(A + B) && \text{Factoring} \\ &= 1 \cdot (A + B) && \text{Rule 6: } A + \bar{A} = 1 \\ &= A + B && \text{Rule 4: drop the 1} \end{aligned}$$

Rule 12: $(A + B)(A + C) = A + BC$

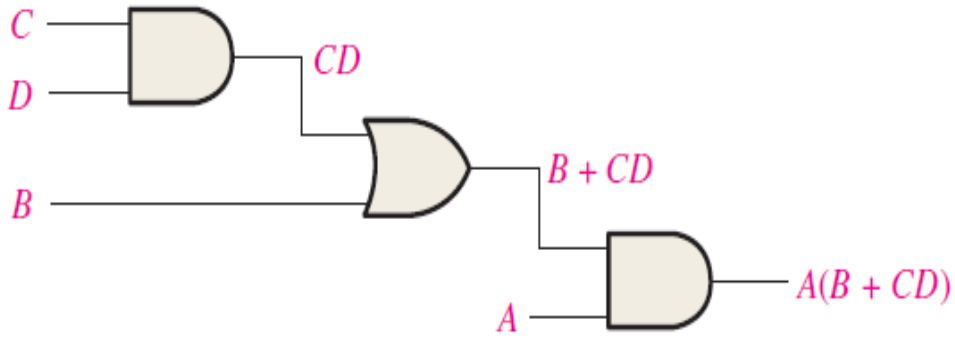
This rule can be proved as follows:

$$\begin{aligned} (A + B)(A + C) &= AA + AC + AB + BC && \text{Distributive law} \\ &= A + AC + AB + BC && \text{Rule 7: } AA = A \\ &= A(1 + C) + AB + BC && \text{Factoring (distributive law)} \\ &= A \cdot 1 + AB + BC && \text{Rule 2: } 1 + C = 1 \\ &= A(1 + B) + BC && \text{Factoring (distributive law)} \\ &= A \cdot 1 + BC && \text{Rule 2: } 1 + B = 1 \\ &= A + BC && \text{Rule 4: } A \cdot 1 = A \end{aligned}$$

❖ Boolean Analysis of Logic Circuits

Boolean algebra provides a concise way to express the operation of a logic circuit formed by a combination of logic gates so that the output can be determined for various combinations of input values.

Boolean Expression for a Logic Circuit



To derive the Boolean expression for a given combinational logic circuit, begin at the left-most inputs and work toward the final output, writing the expression for each gate. For the example circuit in, the Boolean expression is determined in the following three steps:

1. The expression for the left-most AND gate with inputs *C* and *D* is *CD.*

2. The output of the left-most AND gate is one of the inputs to the OR gate and *B* is the other input. Therefore, the expression for the OR gate is *B* + *CD.*

3. The output of the OR gate is one of the inputs to the right-most AND gate and *A* is the other input. Therefore, the expression for this AND gate is *A*(*B* + *CD*), which is the final output expression for the entire circuit.

Constructing a Truth Table for a Logic Circuit

In the case of the circuit in Figure, there are four input variables (A, B, C, and D) and therefore sixteen ($2^4 = 16$) combinations of values are possible.

To evaluate the expression $A(B + CD)$, first find the values of the variables that make the expression equal to 1, using the rules for Boolean addition and multiplication. In this case, the expression equals 1 only if $A = 1$ and $B + CD = 1$ because $A(B + CD) = 1 \cdot 1 = 1$

Now determine when the $B + CD$ term equals 1.

The term $B + CD = 1$ if either $B = 1$ or $CD = 1$ or if both B and CD equal 1 because

$$B + CD = 1 + 0 = 1$$
$$B + CD = 0 + 1 = 1$$
$$B + CD = 1 + 1 = 1$$

The term $CD = 1$ only if $C = 1$ and $D = 1$.

To summarize, the expression $A(B + CD) = 1$ when $A = 1$ and $B = 1$ regardless of the values of C and D or when $A = 1$ and $C = 1$ and $D = 1$ regardless of the value of B.

Putting the Results in Truth Table Format

Inputs				Output
A	B	C	D	$A(B + CD)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Example 1:

Using Boolean algebra techniques, simplify this expression:

$$AB + A(B + C) + B(B + C)$$

Solution

The following is not necessarily the only approach.

Step 1: Apply the distributive law to the second and third terms in the expression, as follows:

$$AB + AB + AC + BB + BC$$

Step 2: Apply rule 7 ($BB = B$) to the fourth term.

$$AB + AB + AC + B + BC$$

Step 3: Apply rule 5 ($AB + AB = AB$) to the first two terms.

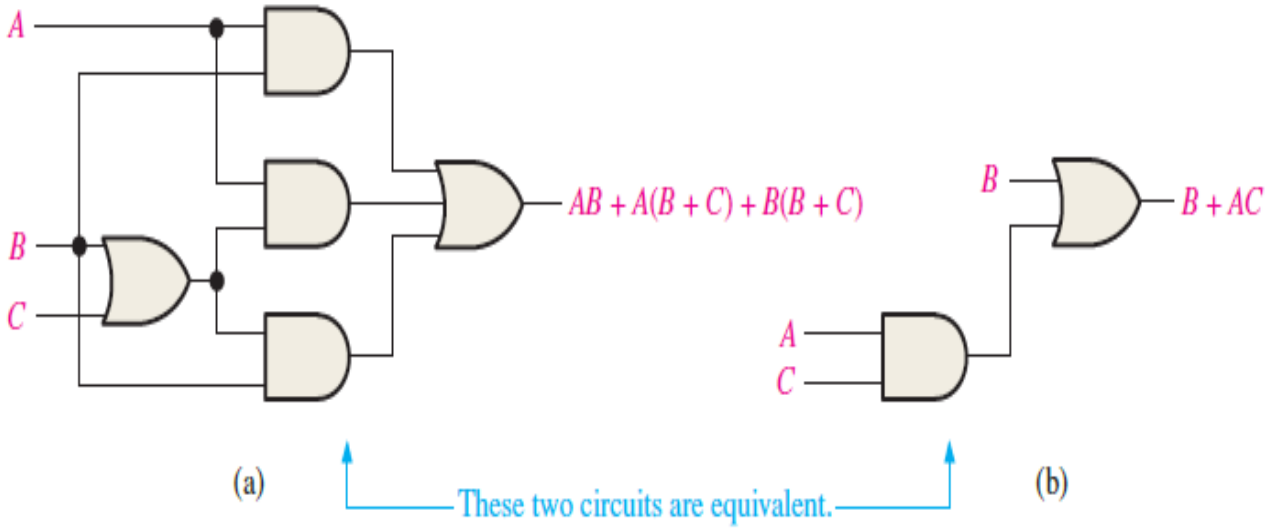
$$AB + AC + B + BC$$

Step 4: Apply rule 10 ($B + BC = B$) to the last two terms.

$$AB + AC + B$$

Step 5: Apply rule 10 ($AB + B = B$) to the first and third terms.

$$B + AC$$



Homework

Simplify the following Boolean expression:

$$[A\overline{B}(C + BD) + \overline{A}\,\overline{B}]C$$

Digital Electronic Circuits

Cyber Security Department

Lecturer 4

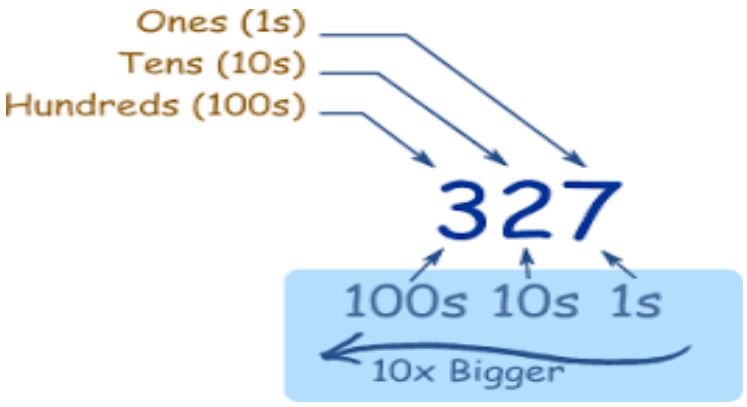
Number Systems, Operations, and Codes

- DECIMAL NUMBER SYSTEM
- BINARY NUMBER SYSTEM
- OCTAL NUMBER SYSTEM
- HEXADECIMAL NUMBER SYSTEM
- The Code Conversion Function

✓ The Encoding Function

✓ The Decoding Function

DECIMAL NUMBER SYSTEM:



- The decimal number system contains ten unique symbols 0,1,2,3,4,5,6,7,8 and 9.
- In decimal system 10 symbols are involved, so the base is 10.
- The value attached to the symbol depends on its location with respect to the decimal point.

Example:

Convert 132.654_{10} to Decimal:

	1	3	2	. 6	5	4_{10}
Position (p) →	2	1	0	-1	-2	-3
Weight →	$(10)^2$	$(10)^1$	$(10)^0$	$(10)^{-1}$	$(10)^{-2}$	$(10)^{-3}$

$$\text{Value} = \sum_{i=-3}^{2} d_i \cdot 10^{\,i}$$

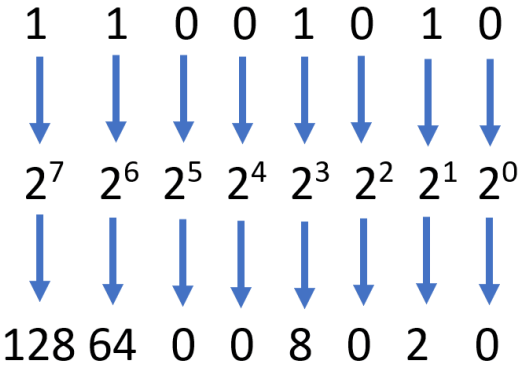
$$\text{Value} = 1\cdot10^2 + 3\cdot10^1 + 2\cdot10^0 + 6\cdot10^{-1} + 5\cdot10^{-2} + 4\cdot10^{-3}$$

$$\text{Value} = 1\cdot(100) + 3\cdot(10) + 2\cdot(1) + 6\cdot(^1/_{10}) + 5\cdot(^1/_{100}) + 4\cdot(^1/_{1000})$$

$$\text{Value} = 100 + 30 + 2 + 0.6 + 0.05 + 0.004$$

$$\text{Value} = 132.654_{10}$$

BINARY NUMBER SYSTEM:



- The base of this number system is 2.
- It has two independent symbols.
- The symbols used are 0 and 1.
- A binary digit is called a bit.

Example:

Convert 101.11_2 to Decimal:

$$1\ 0\ 1\ .\ 1\ 1_2$$

Position (p) → 2 1 0 -1 -2

Weight → $(2)^2$ $(2)^1$ $(2)^0$ $(2)^{-1}$ $(2)^{-2}$

$$\text{Value} = \sum_{i=-2}^{2} d_i \cdot 2^i$$

$$\text{Value} = 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 + 1\cdot2^{-1} + 1\cdot2^{-2}$$

$$\text{Value} = 1\cdot(4) + 0\cdot(2) + 1\cdot(1) + 1\cdot(1/2) + 1\cdot(1/4)$$

$$\text{Value} = 4 + 0 + 1 + 0.5 + 0.25$$

$$\text{Value} = 5.75_{10}$$

OCTAL NUMBER SYSTEM:

- Its base is 8.
- It has 8 independent symbols 0,1,2,3,4,5,6 and 7.
- Its base 8 = 2^3 , every 3- bit group of binary can be represented by an octal digit.

Example:

Convert 17.17_8 to Decimal:

	1	7 .	1	7_8
Position (p) →	1	0	-1	-2
Weight →	$(8)^1$	$(8)^0$	$(8)^{-1}$	$(8)^{-2}$

$$\text{Value} = \sum_{i=-2}^{1} d_i \cdot 8^i$$

$$\text{Value} = 1\cdot8^1 + 7\cdot8^0 + 1\cdot8^{-1} + 7\cdot8^{-2}$$

$$\text{Value} = 1\cdot(8) + 7\cdot(1) + 1\cdot(^1/_8) + 7\cdot(^1/_{64})$$

$$\text{Value} = 8 + 7 + 0.125 + 0.109375$$

$$\text{Value} = 15.234375_{10}$$

HEXADECIMAL NUMBER SYSTEM:

- The base or radix of this number system is 16.
- The symbols used are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E and F
- The base 16 = 2^4 , every 4 – bit group of binary can be represented by an hexadecimal digit.

Example:

Convert 1AB.EF$_{16}$ to Decimal:

	1	A	B	.	E	F$_{16}$
Position (p) →	2	1	0		-1	-2
Weight →	$(16)^2$	$(16)^1$	$(16)^0$		$(16)^{-1}$	$(16)^{-2}$

$$\text{Value} = \sum_{i=-2}^{2} d_i \cdot 16^i$$

$$\text{Value} = 1\cdot16^2 + A\cdot16^1 + B\cdot16^0 + E\cdot16^{-1} + F\cdot16^{-2}$$

$$\text{Value} = 1\cdot(256) + 10\cdot(16) + 11\cdot(1) + 14\cdot(1/16) + 15\cdot(1/256)$$

$$\text{Value} = 256 + 160 + 11 + 0.875 + 0.05859375$$

$$\text{Value} = 427.93359375_{10}$$

Table of Number Systems

DECIMAL	BINARY	HEXADECIMAL	OCTAL
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

❖ Decimal to Binary

Example: Convert 11.375_{10} to Binary:

$$11.375_{10}$$

Part 1: Converting the whole number portion:

			Quotient	Remainder	
Step 1:	2	11	5	1	LSB
Step 2:	2	5	2	1	Next highest order bit
Step 3:	2	2	1	0	Next highest order bit
Step 4:	2	1	0	1	MSB
			Done	Converted Whole Number = 1011_2	

Part 2: Converting the fractional number portion:

		Product	Whole Number	
Step 1:	2 · (0.375)	0.75	0	MSB
Step 2:	2 · (0.75)	1.50	1	Next lower order bit
Step 3:	2 · (0.5)	1.00	1	LSB
		Done	Converted Fractional Number = $.011_2$	

Part 3: Combine the two components to form the new number:

$$1011.011_2$$

- The Code Conversion Function

A **code** is a set of bits arranged in a unique pattern and used to represent specified information.

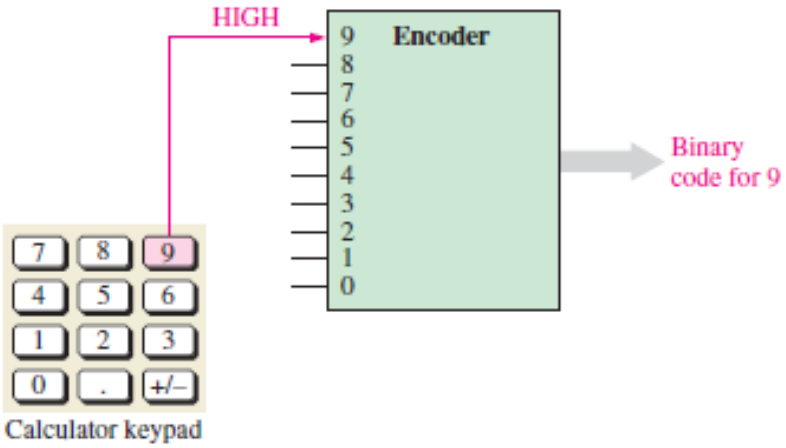
A code converter changes one form of coded information into another coded form. Examples are conversion between binary and other codes such as the binary coded decimal (BCD)

✓ The Encoding Function

The encoding function is performed by a logic circuit called an encoder. The encoder converts information, such as a decimal number or an alphabetic character, into some coded form. For example, one certain type of encoder converts each of the decimal digits, 0 through 9, to a binary code. A HIGH level on the input

corresponding to a specific decimal digit produces logic levels that represent the proper binary code on the output lines.

A simple illustration of an encoder used to convert (encode) a calculator keystroke into a binary code that can be processed by the calculator circuits.



✓ The Decoding Function

The decoding function is performed by a logic circuit called a decoder. The decoder converts coded information, such as a binary number, into a noncoded form, such as a decimal form.

A simple illustration of one type of decoder that is used to activate a 7-segment display. Each of the seven segments of the display is connected to an output line from the decoder. When a particular binary code appears on the decoder inputs, the appropriate output lines are activated and light the proper segments to display the decimal digit corresponding to the binary code.