EE1005 – Digital Logic Design

- **Lecture Slides**
- **Week 2**

Course Instructor:

Dr. Arslan Ahmed Amin

FAST National University of Computer and Emerging Sciences CFD Campus

Lecture 1: Number Systems (Chapter 1)

Commonly used number systems are;

Decimal:	0 to 9	Base = 10
Binary:	0 and 1	Base = 2
Octal:	0 to 7	Base = 8
Hexadecimal:	0 to F (0 to 9, A to F)	Base = 16

Base of number system is also called as Radix.

Examples:

Let's see how numbers are represented in different radix systems.

$$(7392)_{10} = 7 \times 10^3 + 3 \times 10^2 + 9 \times 10^1 + 2 \times 10^0$$

$$(4536.89)_{10} = 4 \times 10^3 + 5 \times 10^2 + 3 \times 10^1 + 6 \times 10^0 + 8 \times 10^{-1} + 9 \times 10^{-2}$$

$$(11010.11)_2 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$$

$$(11010.11)_2 = (26.75)_{10}$$

$$(127.4)_8 = 1 \times 8^2 + 2 \times 8^1 + 7 \times 8^0 + 4 \times 8^{-1} = (87.5)_{10}$$

$$(B65F)_{16} = 11 \times 16^3 + 6 \times 16^2 + 5 \times 16^1 + 15 \times 16^0 = (46687)_{10}$$

In general, a number expressed in a base-r system has coefficients multiplied by powers of r;

$$a_n \times r^n + a_{n-1} \times r^{n-1} + \cdots \ldots + a_1 \times r^1 + a_0 \times r^0 + a_{-1} \times r^{-1} + \cdots .. + a_{-m} \times r^{-m}$$

The coefficients a_i range in value from 0 to $r - 1$.

Number Base Conversion

Converting a Decimal Number to a Number in Base *r* (without fraction part):

Ex 1: $(41)_{10} = (\ ?\ )_2$

	Quotient	Remainder
41/2	20	1
20/2	10	0
10/2	5	0
5/2	2	1
2/2	1	0

From above table, we can write; $(41)_{10} = (101001)_2$

Ex 2: $(153)_{10} = (\ ?\ )_8$

	Quotient	Remainder
153/8	19	1
19/8	2	3

From above table, we can write; $(153)_{10} = (231)_8$

Ex 3: $(956)_{10} = (\ ?\ )_{16}$

	Quotient	Remainder
956/16	59	12 = C
59/16	3	11 = B

From above table, we can write; $(956)_{10} = (3BC)_{16}$

Conversion of Decimal Fraction Part to a Number in Base r:

Ex 4: $(0.6875)_{10} = (\ ?\ )_2$

	Integer	Fraction
0.6875×2	1	0.375
0.375×2	0	0.75
0.75×2	1	0.5
0.5×2	1	0

From above table, we can write; $(0.6875)_{10} = (0.1011)_2$

Using Ex 1 and Ex 4; $\Rightarrow$ $(41.6875)_{10} = (1010010.1011)_2$

Ex 5: $(0.513)_{10} = (\ ?\ )_8$

	Integer	Fraction
0.513×8	4	0.104
0.104×8	0	0.832
0.832×8	6	0.656
0.656×8	5	0.248
0.248×8	1	0.984
0.984×8	7	0.872

From above table, we can write; $(0.513)_{10} = (0.406517\ldots.)_8$

Using Ex 2 and Ex 5; $\Rightarrow$ $(153.513)_{10} = (231.406517\ldots.)_8$

Ex 6: $(0.786)_{10} = (\ ?\ )_{16}$

	Integer	Fraction
0.786×16	$12 = C$	0.576
0.576×16	9	0.216
0.216×16	3	0.456
0.456×16	7	0.296
0.296×16	4	0.736

From above table, we can write; $(0.786)_{10} = (0.C9374\ldots.)_{16}$

Using Ex 3 and Ex 6; $\Rightarrow$ $(956.786)_{10} = (3BC.C9374\ldots.)_{16}$

Octal and Hexadecimal Numbers:

Numbers with Different Bases

Decimal (base 10)	Binary (base 2)	Octal (base 8)	Hexadecimal (base 16)
00	0000	00	0
01	0001	01	1
02	0010	02	2
03	0011	03	3
04	0100	04	4
05	0101	05	5
06	0110	06	6
07	0111	07	7
08	1000	10	8
09	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Ex 7: $(10110001101011.111100000110)_2 = (\ ?\ )_8$

Partitioning the binary number into groups of three digits each, starting from the binary point and proceeding to the left and to the right.

$$\begin{array}{cccccccccccc} (10 & 110 & 001 & 101 & 011 & \cdot & 111 & 100 & 000 & 110)_2 = (26153.7406)_8 \\ 2 & 6 & 1 & 5 & 3 & & 7 & 4 & 0 & 6 \end{array}$$

Ex 8: $(10110001101011.111110010)_2 = (\ ?\ )_{16}$

Conversion from binary to hexadecimal is similar, except that the binary number is divided into groups of four digits;

$$(10\ \ 1100\ \ 0110\ \ 1011\ \cdot\ 1111\ \ 1001)_2 = (2C6B.F9)_{16}$$
$$2\quad C\quad 6\quad B\qquad F\quad 9$$

Ex 9: $(673.124)_8 = (\ ?\ )_{16}$

First convert octal number into binary number

$$(673.124)_8 = (110\ \ 111\ \ 011\ \cdot\ 001\ \ 010\ \ 100)_2$$
$$6\quad 7\quad 3\qquad 1\quad 2\quad 4$$

Now, convert binary number to hexadecimal number

$$(110111011.001010100)_2 = (1\ \ 1011\ \ 1011.0010\ \ 1010\ \ 0)_2 = (1BB.2A0)_{16}$$

So, now we can write;

$$(673.124)_8 = (1BB.2A0)_{16}$$

Ex 10: $(A6D.5F)_{16} = (\ ?\ )_8$

First convert hexadecimal number into binary number

$$(A6D.5F)_{16} = (1010 \quad 0110 \quad 1101.0101 \quad 1111)_2$$

Now, convert binary number to octal number

$$(101001101101.01011111)_2 = (101 \quad 001 \quad 101 \quad 101.010 \quad 111 \quad 110)_2$$

$$= (5155.276)_8$$

So, now we can write;

$$(A6D.5F)_{16} = (5155.276)_8$$

Lecture 2: Complements of Numbers

There are two types of complements for each base-r system:

1. Radix Complement: r's complement
2. Diminished Radix Complement: $(r-1)$'s complement

Given a number N in base-r having n digits, it's r's and $(r-1)$'s complements can be found as;

r's complement: $r^n - N$

$(r-1)$'s complement: $(r^n - 1) - N$

Examples:

Ex 1: Let given number $N = (7392)_{10}$, Find r's and $(r-1)$'s complements?

10's complement of $(7392)_{10} = 10^4 - (7392)_{10} = 10000 - (7392)_{10} = (2608)_{10}$

9's complement of $(7392)_{10} = 10^4 - 1 - (7392)_{10} = 9999 - (7392)_{10} = (2607)_{10}$

Ex 2: Let given number $N = (1011\ 1000)_2$, Find 2's and 1's complement?

2's complement of $N = 2^8 - N = (1\ 0000\ 0000)_2 - (1011\ 1000)_2 = (0100\ 1000)_2$

1's complement of $N = 2^8 - 1 - N = (1111\ 1111)_2 - (1011\ 1000)_2 = (0100\ 0111)_2$

Ex 3: Let given number $N = (4573)_8$, Find 8's and 7's complement?

8's complement of $N = 8^4 - N = (10000)_8 - (4573)_8 = (3205)_8$

7's complement of $N = 8^4 - 1 - N = (7777)_8 - (4573)_8 = (3204)_8$

Ex 4: Let given number $N = (8AD1)_{16}$, Find 16's and 15's complement?

16's complement of $N = 16^4 - N = (10000)_{16} - (8AD1)_{16} = (752F)_{16}$

15's complement of $N = 16^4 - 1 - N = (FFFF)_{16} - (8AD1)_{16} = (752E)_{16}$

Subtraction using r's Complements:

The subtraction of two n-digit unsigned numbers $M - N$ in base-r can be performed as;

1. Add the M to the r's complement of the N. Mathematically, it can be written as;
   $M +$ r's complement of $N = M + (r^n - N) = M - N + r^n$
2. If $M \geq N$, the sum will produce an end carry, which can be discarded and what is left is the result i.e. $M - N$
3. If $M < N$, the sum will not produce an end carry. To obtain the answer in this case; take the r's complement of the sum and place a negative sign in front.

Ex 5: Using 10's complement, subtract $(72532)_{10} - (3250)_{10}$

$$M = (72532)_{10}$$
$$\text{10's complement of } N = (96750)_{10}$$
$$\text{Sum} = (169282)_{10}$$
$$\text{Discard the end carry, gives Answer} = (69282)_{10}$$

Ex 6: Using 10's complement, subtract $(3250)_{10} - (72532)_{10}$

$$M = (03250)_{10}$$
$$\text{10's complement of } N = (27468)_{10}$$
$$\text{Sum} = (30718)_{10}$$

Take 10's complement of Sum and place negative sign in front

$$\text{Answer} = -(69282)_{10}$$

Ex 7: Using 2's complement, subtract $(1010100)_2 - (1000011)_2$

$$M = (1010100)_2$$
$$\text{2's complement of } N = (0111101)_2$$
$$\text{Sum} = (10010001)_2$$
$$\text{Discard the end carry, gives Answer} = (0010001)_2$$

Ex 8: Using 2's complement, subtract $(1000011)_2 - (1010100)_2$

$$M = (1000011)_2$$
$$\text{2's complement of } N = (0101100)_2$$
$$\text{Sum} = (1101111)_2$$

Take 2's complement of Sum and place negative sign in front

$$\text{Answer} = -(0010001)_2$$

Ex 9: Using 16's complement, subtract $(AB45)_{16} - (78EF)_{16}$

$$M = (AB45)_{16}$$
$$\text{16's complement of } N = (8711)_{16}$$
$$\text{Sum} = (13256)_{16}$$
$$\text{Discard the end carry, gives Answer} = (3256)_{16}$$

Ex 10: Using 16's complement, subtract $(78EF)_{16} - (AB45)_{16}$

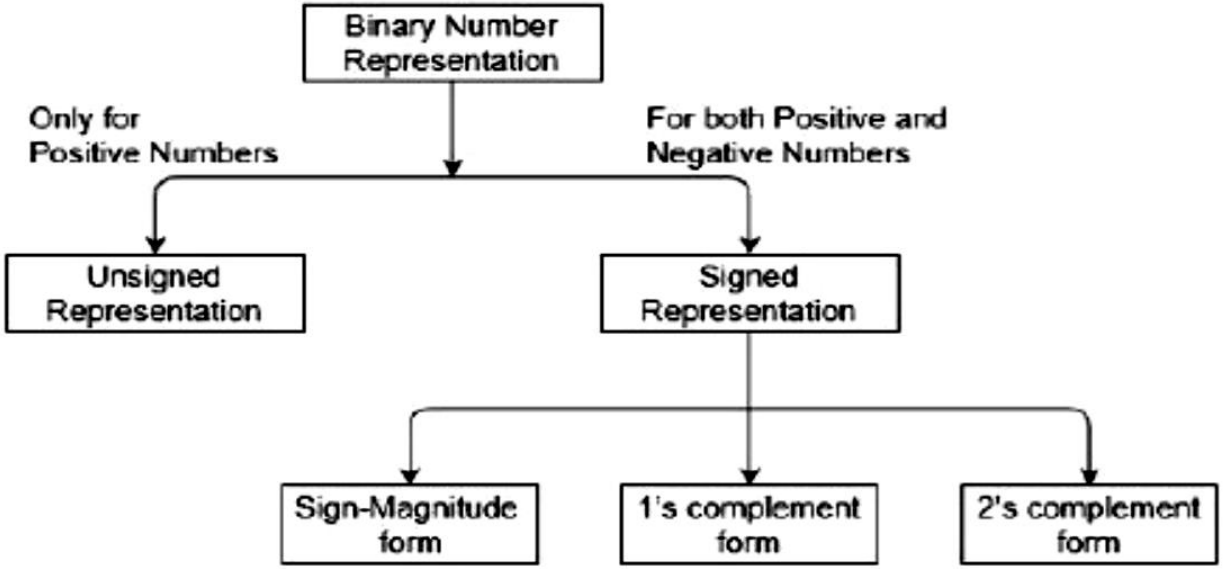
$$M = (78EF)_{16}$$
$$\text{16's complement of } N = (54BB)_{16}$$
$$\text{Sum} = (CDAA)_{16}$$

Take 16's complement of Sum and place negative sign in front

$$\text{Answer} = -(3256)_{16}$$

Lecture 3: Binary Numbers Representation and Binary Codes

Binary numbers can be represented in signed and unsigned way. Unsigned binary numbers do not have sign bit, whereas signed binary numbers uses signed bit as well or these can be distinguishable between positive and negative numbers.



Unsigned Numbers: Unsigned numbers don't have any sign, these contain only magnitude of the number. So, unsigned binary numbers can represent only positive numbers. For n-bit binary number, the range of unsigned binary number is from 0 to $2^n - 1$.

Ex 1: Represent decimal number 92 in unsigned binary number.

Simply convert it into binary number, it contains only magnitude of the given number.

$$(92)_{10} = (1011100)_2$$

It's 7 bit binary magnitude of the decimal number 92.

Ex 2: Find range of 5 bit unsigned binary numbers. Also, find minimum and maximum value in this range.

Range of 5 bit unsigned binary number is from 0 to $(2^5 - 1)$ which is equal from minimum value 0 (i.e., 00000) to maximum value 31 (i.e., 11111).

Signed Numbers:

Signed numbers are used specifically to take care of +ve or –ve signs of numbers. Signed numbers are used to represent both positive and negative numbers. There are three types of representations for signed binary numbers. These are: Sign-Magnitude form, 1's complement form, and 2's complement form.

Sign-Magnitude Form:

For n-bit binary number, 1 bit is reserved for sign symbol. If the value of sign bit is 0, then the given number will be positive, else if the value of sign bit is 1, then the given number will be negative. Remaining $(n - 1)$ bits represent magnitude of the number.

Ex 3: Represent +9 and −9 using Sign-Magnitude Form with 8-bits

We know that: $(9)_{10} = (1001)_2 \quad \Rightarrow \quad (9)_{10} = (0000\ 1001)_2$

$+9 = (0000\ 1001)_2$ | $-9 = (1000\ 1001)_2$

1's complement Form:

In 1's complement form, positive numbers are represented in simple binary form while negative numbers are represented by taking their 1's complement.

Ex 4: Represent +9 and −9 using 1's Complement Form with 8-bits

We know that: $(9)_{10} = (1001)_2 \quad \Rightarrow \quad (9)_{10} = (0000\ 1001)_2$

$+9 = (0000\ 1001)_2 \quad | \quad -9 = (1111\ 0110)_2$

2's complement Form:

In 2's complement form, positive numbers are represented in simple binary form while negative numbers are represented by taking their 2's complement.

Ex 5: Represent +9 and −9 using 2's Complement Form with 8-bits

We know that: $(9)_{10} = (1001)_2 \quad \Rightarrow \quad (9)_{10} = (0000\ 1001)_2$

$+9 = (0000\ 1001)_2 \quad | \quad -9 = (1111\ 0111)_2$

Range of Numbers

- For a specific number of bits only a certain range of decimal numbers can be represented.
- The maximum and minimum number can be found with the following relation

- For unsigned numbers, the range is

$$0 \text{ to } [(2^{\text{No.of Bits}}) - 1]$$

  - For example, we can represent the number from 0 to 255 by using 8 bits
- For signed numbers, the range is

$$-\left[\frac{2^{No.of\ bits}}{2}\right] \ to + \left[\left(\frac{2^{No.of\ bits}}{2}\right) - 1\right]$$

  - For example, we can represent the numbers from –128 to +127 by using 8 bits

Representation of 4-bit signed binary numbers using different representations is given as;

Signed Binary Numbers

Decimal	Signed-2's Complement	Signed-1's Complement	Signed Magnitude
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
−0	—	1111	1000
−1	1111	1110	1001
−2	1110	1101	1010
−3	1101	1100	1011
−4	1100	1011	1100
−5	1011	1010	1101
−6	1010	1001	1110
−7	1001	1000	1111
−8	1000	—	—

Note that in computers positive and negative numbers (i.e. signed numbers) are represented using 2's complement form.

Binary Codes:

Different coding schemes are available to represent the decimal or binary data.

Binary Coded Decimal Code (BCD Code):

BCD codes are used to represent the decimal numbers specifically. In BCD code, each of the decimal digit is represented by 4 bit binary code, as per table given below;

Binary-Coded Decimal (BCD)

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

Note that binary combinations from 1010 to 1111 are not used and have no meaning in BCD.

Ex 6: Find BCD and binary value of $(185)_{10}$

To find BCD value, replace each decimal digit with it's 4-bit BCD value

$$(185)_{10} = (0001\ 1000\ 0101)_{BCD}$$

To find binary value, convert $(185)_{10}$ to binary by repeatedly dividing with 2 (usual procedure)

$$(185)_{10} = (1011\ 1001)_2$$

Decimal number to BCD number Conversion

To convert a decimal number to the BCD number, we need to replace every digit of the decimal number by its 4 bit unsigned binary equivalent.

Example: Convert the decimal number 9673 to BCD.

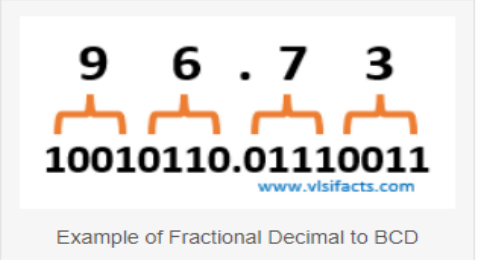
Solution:

Example of Decimal to BCD Conversion

$9673_{10} = (1001\_0110\_0111\_0011)_{BCD}$

Example: Convert the decimal number 96.73 to BCD.

Solution:



Example of Fractional Decimal to BCD

Binary to BCD code conversion

← Prev Next →

BCD code plays an important role in digital circuits. The BCD stands for Binary Coded Decimal Number. In BCD code, each digit of the decimal number is represented as its equivalent binary number. So, the LSB and MSB of the decimal numbers are represented as its binary numbers. There are the following steps to convert the binary number to BCD:

1. First, we will convert the binary number into decimal.
2. We will convert the decimal number into BCD.

Let's take an example to understand the process of converting a binary number into BCD

Example 1: $(11110)_2$

1. First, convert the given binary number into a decimal number.

Binary Number: $(11110)_2$

Finding Decimal Equivalent of the number:

Steps	Binary Number	Decimal Number
1)	$(11110)_2$	$((1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0))_{10}$
2)	$(11110)_2$	$(16 + 8 + 4 + 2 + 0)_{10}$
3)	$(11110)_2$	$(30)_{10}$

Decimal number of the Binary number $(11110)_2$ is $(30)_{10}$

2. Now, we convert the decimal to the BCD

We convert each digit of the decimal number into groups of the four-bit binary number.

Steps	Decimal Number	Conversion
Step 1	30_{10}	$(0011)_2\ (0000)_2$
Step 2	30_{10}	$(00110000)_{BCD}$

Result:

$(11110)_2 = (00110000)_{BCD}$

BCD Addition:

Rule for addition of two BCD numbers is given as;

1. When two BCD digits are added, if their sum is greater than or equal to 9 $(1001)_2$, add 6 $(0110)_2$ to sum else not.

Examples:

4 0100 +5 +0101 9 1001	8 1000 +9 1001 17 10001 ← Binary code for 17 +0110 10111 ← BCD code for 17

Ex 6: Perform the addition of $(184)_{10} + (576)_{10}$ in BCD

BCD codes for $(184)_{10}$ and $(576)_{10}$ are;

$(184)_{10} = (0001\ \ 1000\ \ 0100)_{BCD}$

$(576)_{10} = (0101\ \ 0111\ \ 0110)_{BCD}$

BCD addition is performed as;

	1	1		
	0001	1000	0100	184
	+0101	0111	0110	+576
Binary sum	0111	10000	1010	
Add 6		0110	0110	
BCD sum	0111	0110	0000	760

Other Decimal Codes:

Like BCD, there are some other codes which are used to represent the decimal digits from 0 to 9, given as follows;

Four Different Binary Codes for the Decimal Digits

Decimal Digit	BCD 8421	2421	Excess-3	8, 4, −2, −1
0	0000	0000	0011	0000
1	0001	0001	0100	0111
2	0010	0010	0101	0110
3	0011	0011	0110	0101
4	0100	0100	0111	0100
5	0101	1011	1000	1011
6	0110	1100	1001	1010
7	0111	1101	1010	1001
8	1000	1110	1011	1000
9	1001	1111	1100	1111
Unused bit combi-nations	1010	0101	0000	0001
	1011	0110	0001	0010
	1100	0111	0010	0011
	1101	1000	1101	1100
	1110	1001	1110	1101
	1111	1010	1111	1110

Gray Code:

Gray code is also used to represent the binary numbers, however, their main property is this that only 1-bit is changed going from given digit to next digit. 4-bit gray code is;

Gray Code

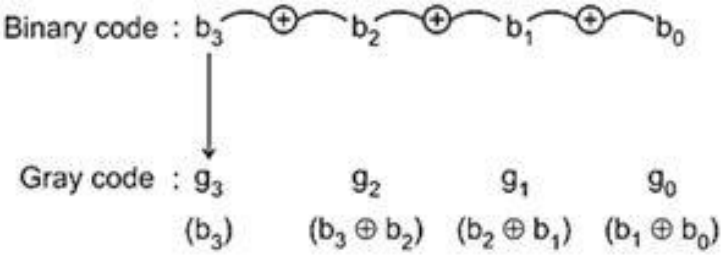
Gray Code	Decimal Equivalent
0000	0
0001	1
0011	2
0010	3
0110	4
0111	5
0101	6
0100	7
1100	8
1101	9
1111	10
1110	11
1010	12
1011	13
1001	14
1000	15

Gray Code Values

Decimal Number	Binary Code	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Generating Gray Codes from Binary

- Start from Left
  - If it is MSB then place it as it is
  - Otherwise, Add the bit to the previous bit and place the sum in GRAY, ignoring any carry
  - Repeat step 2 till end



- Write the gray code for the binary sequence 1010
  - The answer is 1111

ASCII Code:

ASCII is a 7-bit character set containing 128 characters. It contains the numbers from 0-9, the upper and lower case English letters from A to Z, and some special characters. The character sets used in modern computers, in HTML, and on the Internet, are all based on ASCII. ASCII codes are shown below;

Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value
00	NUL	10	DLE	20	SP	30	0	40	@	50	P	60	`	70	p
01	SOH	11	DC1	21	!	31	1	41	A	51	Q	61	a	71	q
02	STX	12	DC2	22	"	32	2	42	B	52	R	62	b	72	r
03	ETX	13	DC3	23	#	33	3	43	C	53	S	63	c	73	s
04	EOT	14	DC4	24	$	34	4	44	D	54	T	64	d	74	t
05	ENQ	15	NAK	25	%	35	5	45	E	55	U	65	e	75	u
06	ACK	16	SYN	26	&	36	6	46	F	56	V	66	f	76	v
07	BEL	17	ETB	27	'	37	7	47	G	57	W	67	g	77	w
08	BS	18	CAN	28	(	38	8	48	H	58	X	68	h	78	x
09	HT	19	EM	29	)	39	9	49	I	59	Y	69	i	79	y
0A	LF	1A	SUB	2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
0B	VT	1B	ESC	2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
0C	FF	1C	FS	2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
0D	CR	1D	GS	2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
0E	SO	1E	RS	2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
0F	SI	1F	US	2F	/	3F	?	4F	O	5F	_	6F	o	7F	DEL

Error Detecting Codes:

Error-detecting codes are a sequence of numbers generated by specific procedures for detecting errors in data that has been transmitted over computer networks.

ASCII character consists of 7-bits, when ASCII character bits are transmitted over the computer network, they are subject to get corrupted due to interference and network problems.

To detect errors in data communication and processing, an eighth bit is sometimes added to the ASCII character to indicate its parity. A parity bit is an extra bit included with a message to make the total number of 1's either even or odd.

Consider the following two characters and their even and odd parity:

	With even parity	With odd parity
ASCII A = 1000001	01000001	11000001
ASCII T = 1010100	11010100	01010100

Parity bits added to ACII's are underlined.

Suppose ASCII character is transmitted with even parity, then the parity of each character is checked at the receiving end. If the parity of the received character is not even, then at least one bit has changed value during the transmission. This method detects one, three, or any odd combination of errors in each character that is transmitted.

In general, one or the other parity is adopted, with even parity being more common.

Binary Logic:

Binary logic deals with variables that take on two discrete values and with operations that assume logical meaning. The two values the variables assume may be called by different names (true and false, yes and no, 1 and 0 etc.).

Binary logic consists of binary variables and a set of logical operations. The variables are designated by letters of the alphabet, such as A, B, C, x, y, z, etc., with each variable having two and only two distinct possible values: 1 and 0. There are three basic logical operations: AND, OR, and NOT.

Truth Tables of Logical Operations

AND

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT

x	x'
0	1
1	0

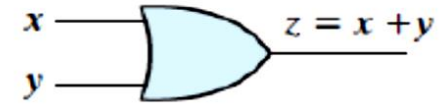
For each combination of the values of x and y, there is a value specified by the definition of the logical operation. Definitions of logical operations may be listed in a compact form called **truth tables**.

Logic Gates:

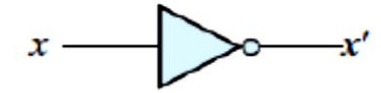
Logic gates are electronic circuits that operate on one or more input signals to produce an output signal.

The input terminals of digital circuits accept binary signals within the allowable voltage range and respond at the output terminals with binary signals that fall within the specified voltage range.

(a) Two-input AND gate

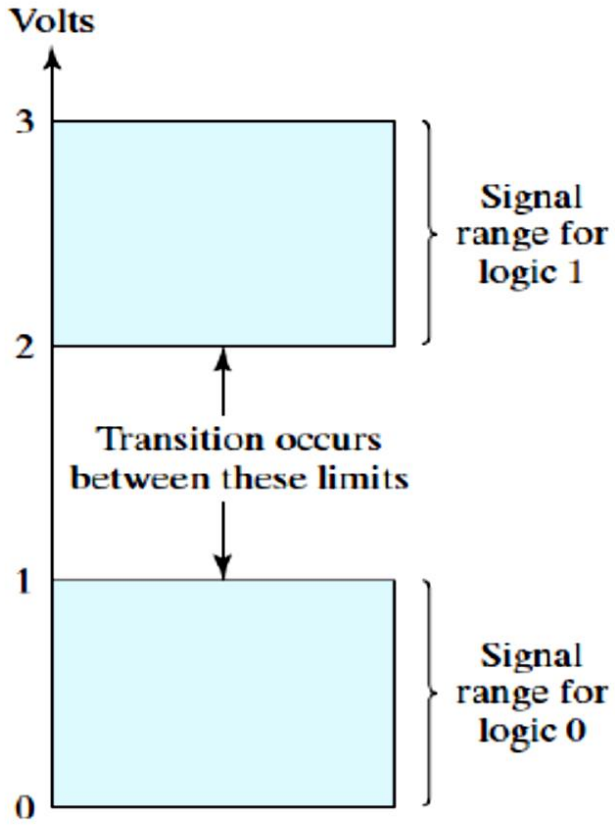


(b) Two-input OR gate



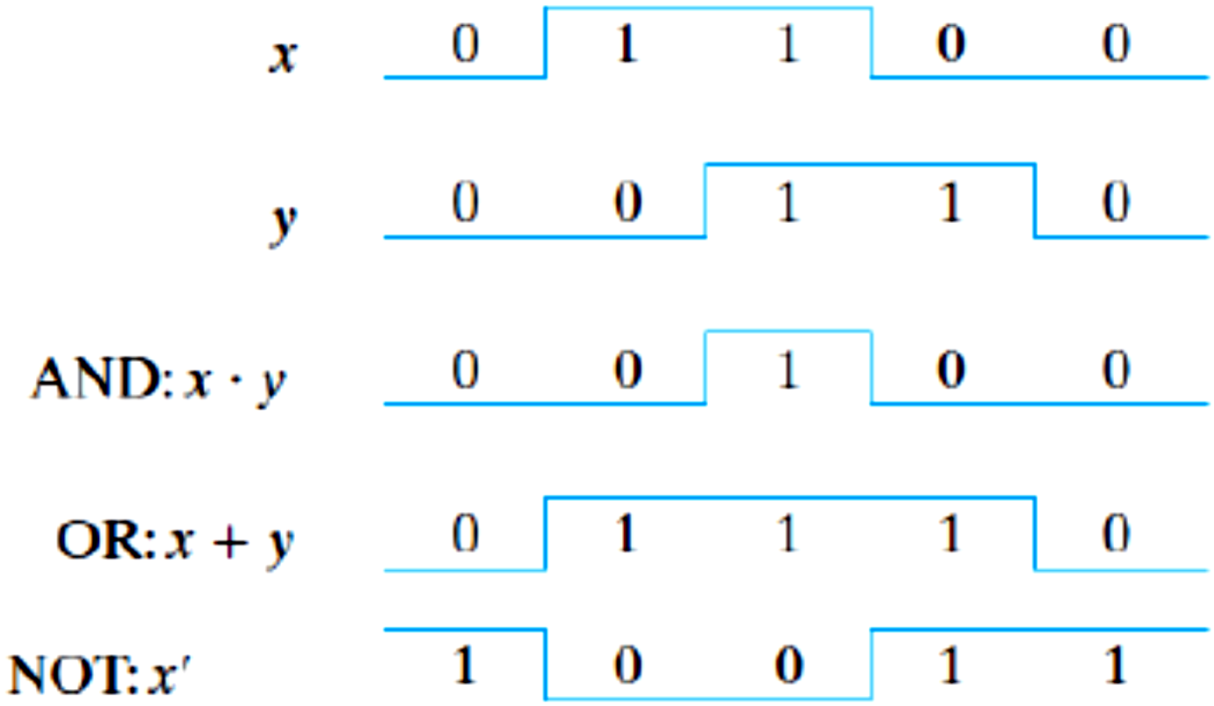
(c) NOT gate or inverter

Symbols for digital logic circuits



Signal levels for binary logic values

Truth tables of each of the gate can be represented by the following waves, it is known as timing diagram representation of gates.


x
0 1 1 0 0
y
0 0 1 1 0
AND: x · y
0 0 1 0 0
OR: x + y
0 1 1 1 0
NOT: x'
1 0 0 1 1