DIGITL LOGIC DESIGN

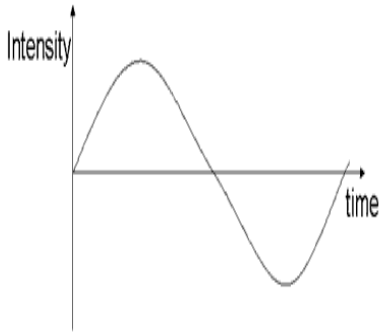
COURSE FILE

UNIT I

Digital and Analog Signals

Signals carry information and are defined as any physical quantity that varies with time, space, or any other independent variable. For example, a sine wave whose amplitude varies with respect to time or the motion of a particle with respect to space can be considered as signals. A system can be defined as a physical device that performs a no operation on a signal. For example, an amplifier is use do to amplify the input signal amplitude. In this case, the amplifier performs some operation(s)on the signal, which has the effect of increasing the amplitude of the desired information-bearing signal.

Signal scan be categorized in various ways; for example discrete and continuous time domains. Discrete-time signals are defined only on a discrete set of times. Continuous-time signals are often referred to as continuous signal seven when the signal functions are not continuous; an example is a square-wave signal.



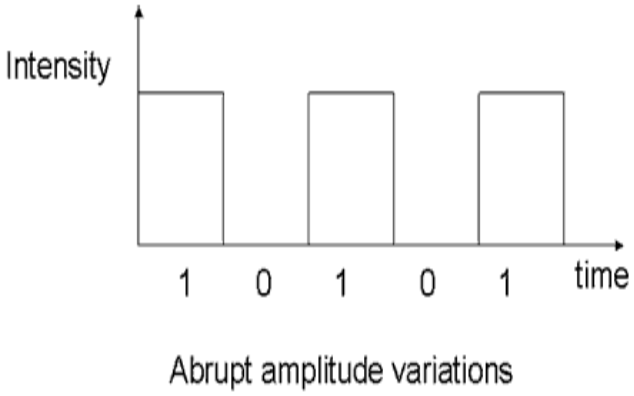


Figure1a:AnalogSignal

Figure1b: DigitalSignal

Digital is the method of storing, processing and transmitting information through the use of distinct electronic pulses that represent the binary digits 0 and 1. Examples of discrete sets are the 10 decimal digits, the 26 letters of the alphabet etc. A digital system would be to flick the lights witch on and off. There's no' in between' values.

Advantagesofdigitalsignals

The usual advantages of digital circuits when compared to analog circuits are:

Noise Margin (resistance to noise/robustness): Digital circuits are less affected by noise. If the noise is below a certain level (the noise margin),a digital circuit be haves as if there was no noise at all. The stream of bits can be reconstructed into a perfect replica of the original source. However, if the noise exceeds this level, the digital circuit cannot give correct results.

Error Correction and Detection: Digital signals can be regenerated to achieve lossless data transmission, within certain limits. Analog signal transmission and processing, by contrast, always introduces noise.

EasilyProgrammable: Digital systems interface well with computer and are easy to control with software. It is often possible to add new features to a digital system with out changing hardware, and to do this remotely, just by uploading new software. Design errors or bugs can be worked-around with a software

upgrade, after the product is in customer hands. A digital system is often preferred because of (re-)programmability and ease of upgrading without requiring hardware changes.

Cheap Electronic Circuits: More digital circuitry can be fabricated per square millimeter of integrated-circuit material. Information storage can be much easier in digital systems than in analog ones. In particular, the great noise-immunity of digital systems makes it possible to store data and retrieve it later with out degradation. In analog system, aging and wear and tear will degrade the information in storage, but in a digital system, as long as the wear and tear is below a certain level, the information can be recovered perfectly. Theoretically, there is no data-loss when copying digital data. This is a great advantage over analog systems, which faithfully reproduce every bit of noise that makes its way into the signal.

NUMBER SYSTEMS

Introduction

Number systems provide the basis for all operations in information processing systems. In a number system the information is divided into a group of symbols; for example, 26 English letters, 10 decimal digits etc. In conventional arithmetic, a number system based upon ten units (0to9) is used. However, arithmetic and logic circuits used in computers and other digital systems operate with only 0's and 1's because it is very difficult to design circuits that require ten distinct states. The number system with the basic symbols 0 and 1 is called binary. ie. A binary system uses just two discrete values. The binary digit (either0 or 1) is called a bit.

A group of bits which is used to represent the discrete elements of information is a symbol. The mapping of symbols to a binary value is known a binary code. This mapping must be unique. For example, the decimal digits 0 through 9 are represent digital system with a code of four bits. Thus a digital system is a system that manipulates discrete elements of information that is represented internally in binary form.

Decimal Numbers

The invention of decimal number system has been the most important factor in the development of science and technology. The decimal number system uses positional number representation, which means that the value of each digit is determined by its position in a number.

The base, also called the radix of a number system is the number of symbols that the system contains. The decimal system has ten symbols: 0,1,2,3,4,5,6,7,8,9. In other words, it has a base of 10. Each position in the decimal system is 10 times more significant than the previous position. The numeric value of a decimal number is determined by multiplying each digit of the number by the value of the position in which the digit appears and then adding the products. Thus the number 2734 is interpreted as

$$2 \times 1000 + 7 \times 100 + 3 \times 10 + 4 \times 1 = 2000 + 700 + 30 + 4$$ + 3 x 10 + 4 x 1 = 2000 + 700 + 30 + 4

Here 4 is the least significant digit (LSD) and 2 is the most significant digit (MSD).

In general in a number system with a base or radix r, the digits used are from 0 to r-1 and the num ber can be represented as

$N = a_n r^n + a_{n-1} r^{n-1} + \ldots\ldots\ldots\ldots + a_1 r^1 + a_0 r^0$ where, for n = 0,1,2,3,....(1)

r = base or radix of the system.

a = number of digits having values between 0 and r-1

Equation (1) is for all integers and for the fractions (numbers between 0 and 1), the following equation holds.

$$N = a_{-1} r^{-1} + a_{-2} r^{-2} + \ldots\ldots\ldots + a_{-n+1} r^{-n+1} + a_{-n} r^{-n}$$

9's &10'sComplements:

It is the Subtraction of decimal no.s can be accomplished by the 9's & 10's compliment methods similar to the 1's & 2's compliment methods of binary. The 9'scompliment of a decimal no. is obtained by subtracting each digit of that decimal no. from9. The 10's compliment of a decimal no is obtained by adding a 1 to its 9's compliment.

Example:9's complimentof3465 and782.54is

```
9999                          999.99
-3465                         -782.54
----------                    -----------
6534                          217.45
------------------            ----------------
```

10's complementof4069is

```
  9999
-4069
----------
  5930
    +1
----------
 5931
-----------
```

9's compliment method of subtraction:

To perform this, obtainthe9'scomplimentofthesubtrahendandittotheminuendnow call this no. the intermediate result. If there is a carry to the LSD of this result together answer called **end around carry**. If there is no carry ,it indicates that the answer is negative & the intermediate resultisits 9's compliment.

Example: Subtractusing9's comp

(1)745.81-436.62

```
745.81
-436.62
----------
309.19
-----------
```

(2)436.62-745.82

```
 436.62
-745.81
----------
 -309.19
---------
```

745.81 436.62

+563.37 9‘s complimentof436.62 +254.18

---------- ----------

1309.18 Intermediate result 690.80

+1 end around carry

309.19

If there is on carry indicating that answer is negative. So take 9‘s complement of intermediate Result & put minus sign(-) result should be-309.19

If carry indicates that the answer is positive +309.19

10’s compliment method of subtraction:

Toper form this, obtain the 10‘s compliment of the subtrahend & addit to the minuend. If there is a carrying it. The presence of the carry indicates that the answer is positive, the result is the answer. If there is no carry, it indicates that the answer is negative & there sultisits 10‘s compliment. Obtain the 10‘s compliment of there sult & place negative sign infront to get the answer.

Example:(a)2928.54-41673 (b)416.73-2928.54

2928.54 0416.73

-0416.73 -2928.54

---------- ----------

2511.81 -2511.81

---------- ---------

2928.54 0416.73

+9583.27 10‘s complimentof436.62 +7071.46

---------- ----------

12511.81 Ignore the carry 7488.19

Decimal	Binary	Octal	Hexadecimal
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	10	8
9	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

Binary to Decimal Conversion:

It is by the positional weights method. In this method, each binary digit of the no. is multiplied by its position weight. The product terms are added to obtain the decimal no.

Example: convert **10101**$_2$ to decimal

Positional weights $2^4 2^3 2^2 \ 2^1 \ 2^0$

Binaryno.**10101**$_2$=$(1x2^4)+(0x2^3)+(1x2^2)+(0x2^1)+(1x2^0)$

=16+0+4+0+1

=21_{10}

Example: convert **11011.101**$_2$ to decimal

Positional weights $2^4 2^3 2^2 \ 2^1 \ 2^0 2^{-1} 2^{-2} \ 2^{-3}$

=16+8+0+2+1+.5+0+.125

=27.625_{10}

* Left bit MSB , multipliy this bit by 2 & add the provided to next bit to the right

*Multiply the result obtained in the previous step by 2 & add the product to then extra bit to the right.

Example:convert 52_{10} to binary using double-dabble method

Divide the given decimal no successively by2 & read the remainders upwards to get the equivalent binary no.

Successive division remainder

2| **52**
|_____
2| 26 --- 0
|_____
2| 13 ---0
|_____
2| 6 --- 1
|_____
2| 3 --- 0 ↓
|↓ = 110100_2
2| 1 --- 1 ↓
|_____
2 |0 --- 1
|_____

Binary Addition:

Rules:

0+0=0

0+1=1

1+0=1

1+1=10 i.e, 0 with a carry of 1.

Example: add binary no.s 1101.101 & 111.011

8421 $2^{-1}2^{-2}2^{-3}$

1101.101

111.011

_____

10101.000

The binary numbering system works just like the decimal numbering system, with two exceptions: binary only allows the digits 0 and 1 (rather than 0–9), and binary uses powers of two rather than powers of ten. Therefore, it is very easy to convert a binary number to decimal. For each “1” in the binary string, add 2n where “n” is the bit position in the binary string (0 to *n*–1 for *n* bit binary string).

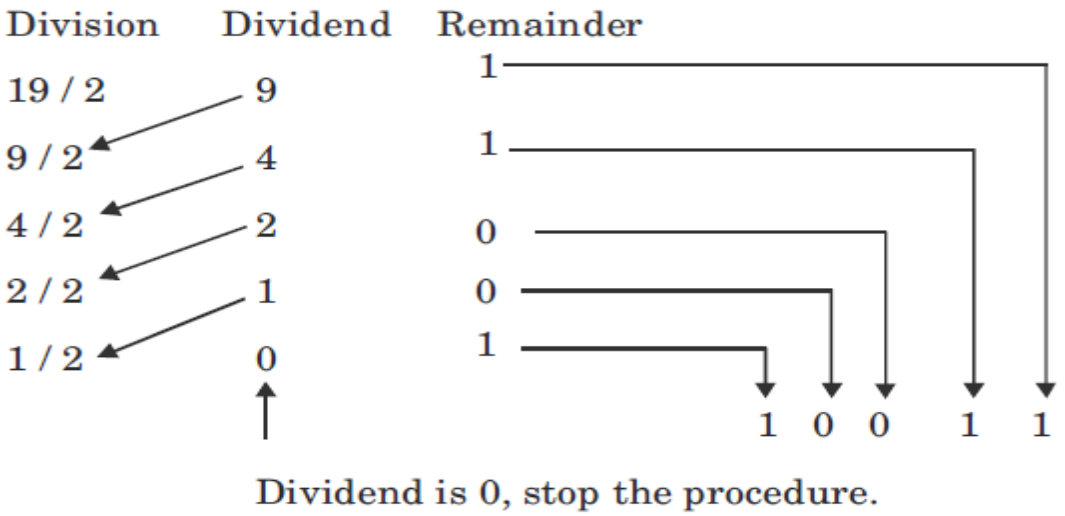
Table 1

Binary No.	1	0	1	0
Bit Position (n)	3rd	2nd	1st	0th
Weight Factor (2^n)	2^3	2^2	2^1	2^0
bit * 2^n	$1*2^3$	$0*2^2$	$1*2^1$	$0*2^0$
Decimal Value	8	0	2	0
Decimal Number	8 + 0 + 2 + 0 = 10			

All the steps in above procedure can be summarized in short as
1*23 + 0*22 + 1*21 + 0*20 = 8 + 0 + 2 + 0 = 1010
i.e.,
1. Multiply each digit of the binary number by its positional weight and then add up the result.
2. If any digit is 0, its positional weight is not to be taken into account.

1.2.2 Decimal to Binary Conversion

The inverse problem would be to find out the binary equivalent of given decimal number for instance let us find out binary of 1910 (decimal 19)



1.6 BINARY ARITHMETIC

The binary arithmetic operations such as addition, subtraction, multiplication and division are similar to the decimal number system. Binary arithmetics are simpler than decimal because they involve only two digits (bits) 1 and 0.

Binary Addition

Rules for binary addition are summarized in the table shown in Fig. 1.8.

Augend	*Addend*	*Sum*	*Carry*	*Result*
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	0	1	10

Fig. 1.8 Rules for binary addition

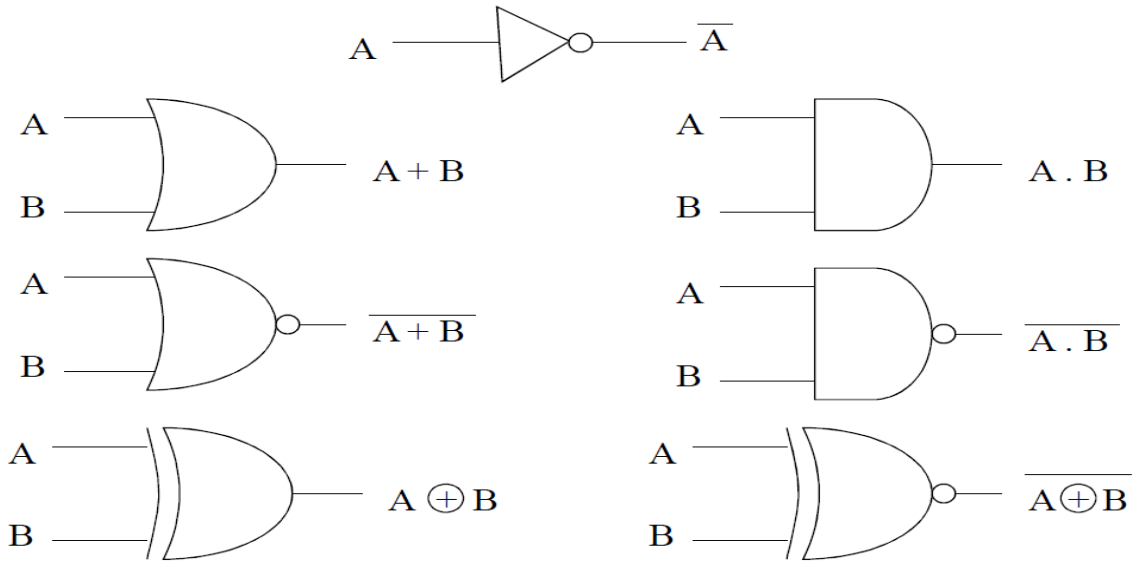
As shown in 4th row adding 1 to 1 gives 9 carry which, is given to next binary position, similar to decimal system. This is explained in examples below:

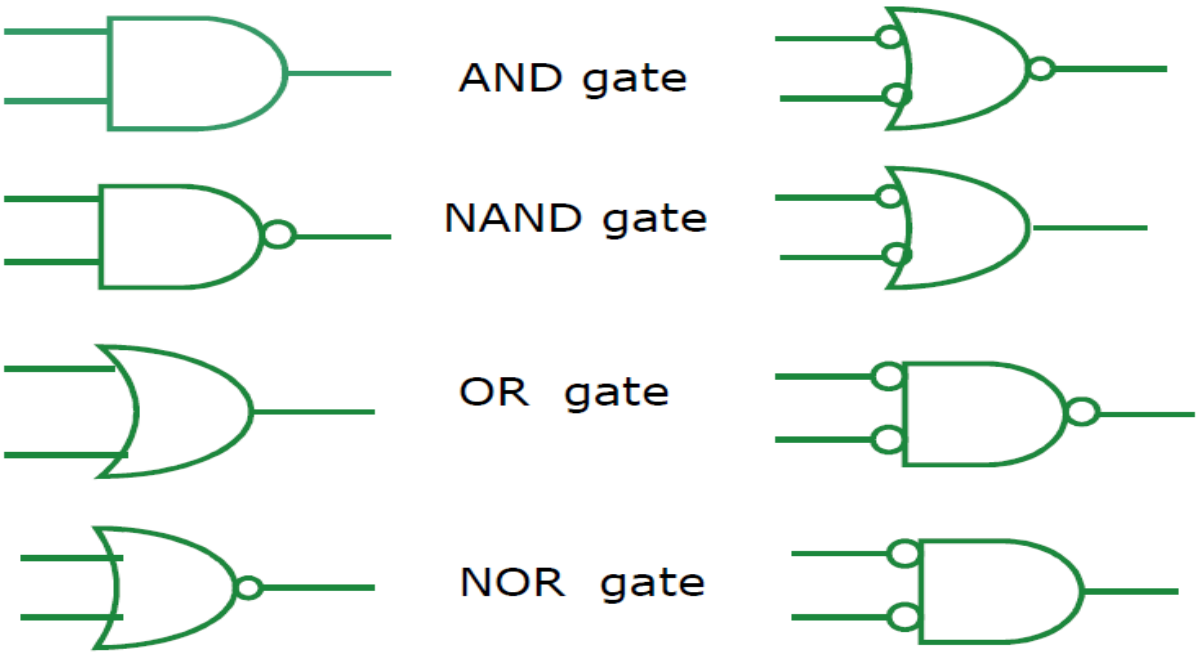
Example 1. (*i*) *Add 1010 and 0011 (ii) Add 0101 and 1111*

Solution.

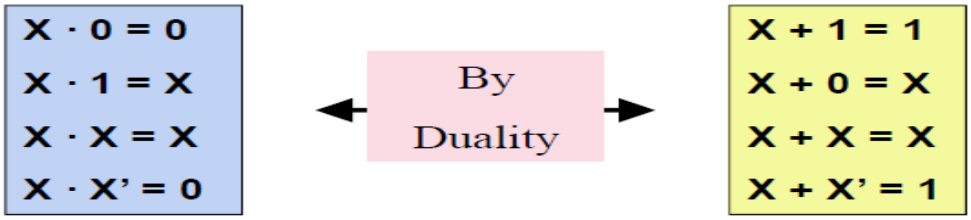
$$\begin{array}{r} {\scriptstyle 1} \\ 1\,0\,1\,0 \\ +\;0\,0\,1\,1 \\ \hline 1\,1\,0\,1 \end{array} \leftarrow \text{Carry} \qquad \begin{array}{r} {\scriptstyle 1\,1\,1} \\ 0\,1\,0\,1 \\ +\;1\,1\,1\,1 \\ \hline 1\,0\,1\,0\,0 \end{array} \leftarrow \text{Carry}$$

Carry





Duality:The dual of a Boolean function is obtained by **INTERCHANGING** OR & AND operations and **REPLACING** 1's by 0's and 0's by 1's.



Logic Minimization:

A major step in digital design
This was critical in olden days
Cost per logic gate
Size of the product
Power consumption

Types of Logic Minimization

Minimization Using Theorems
Minimization Using Karnaugh Map.

Example : Simplify & Describe logic circuit for

$Y = A` B` C` + A` B C` + A B` C` + A B C`$

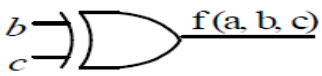
$Y = (A` B` + A` B + A B` + A B) C`$
$= [A`( B` + B) + A (B` + B ) ] C`$
$= [A' (1) + A (1)] C'$
$= (A` + A ) C`$
$Y = C`$

Simplify the output function and draw the circuit

$$\begin{aligned}\bar{f}(a,b,c) &= (b \oplus \bar{c}) + \overline{ab} \cdot \overline{\bar{a} + c} \\ &= bc + \bar{b}\bar{c} + \overline{ab} \cdot \overline{\bar{a} + c} \\ &= bc + \bar{b}\bar{c} + (\bar{a} + \bar{b})a\bar{c} \\ &= bc + \bar{b}\bar{c} + a\bar{b}\bar{c} \\ &= bc + \bar{b}\bar{c}\end{aligned}$$

$$\bar{f}(a,b,c) = b \odot c$$

Therefore, $f(a,b,c) = (b \odot c)' = b \oplus c$



Laws of BooleanAlgebra

Postulate2 :

(a)$0 + A = A$ (b)$1.A = A$

(a)$A + A' = 1$ (b)$A.\ A' = 0$

(a)$A + A = A$ (b)$A\ A = A$

(a)$1 + A = 1$ (b)$0.A = 0$

$A'' = A$

Postulate3 :CommutativeLaw

(a)$A + B = B + A$ (b)$A\ B = B\ A$

Theorem4:AssociateLaw

(a)*(A + B)+ C= A + (B +C)* (b)*(A B)C= A (B C)*

Postulate4:DistributiveLaw

(a)*A (B + C)= A B + A C* (b)*A + (B C)= (A + B)(A + C)*

Theorem5 :DeMorgan'sTheorem

(a) (A+B)'= A'B' (b)(AB)'= A'+ B'

Theorem6 :Absorption

(a)*A+ A B = A* (b)*A (A + B)= A*

ProveTheorem1 : (a) X+X=X

x+x=(X+X).1 bypostulate2b

=(x+x)(x+x') 5a

=x+xx' 4b

=x+0 5b

=x 2a

ProveTheorem1 : (b) X.X=X

xx=(X.X)+0 bypostulate2a

=x.x+x.x' 5b

=x(x+x') 4a

=x.1 5a

=x 2b

ProveTheorem2 : (a) X+1=X

x+1=1.(X+1) bypostulate2b

(x+x')=(x+1) 5a

=x+x'.1 4b

=x+ x' 2b

=1 5a

ProveTheorem2 : (b) X.0=0

x.0=0+(X.0) bypostulate2a

(x.x')=(x.0) 5b

=x.x'+0 4a

=x.x' 2a

=0 5b

ProveTheorem6 : (a) X+xy=X

x+xy=x.1+xy bypostulate2b

=x(1+y) 4b

=x(y+1) 3a

=x.1 2b

=x 2b

ProveTheorem6 : (b) X(x+y)=X

X(x+y)=(x+0).(x+y) bypostulate2a

=x+0.y 4a

=x +0 2a

=x 2a

Using the laws given above, complicated expressions can be simplified.

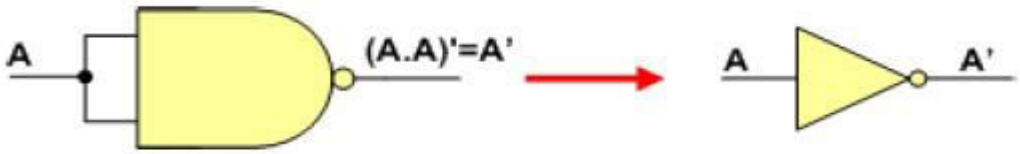
NAND Gate is a Universal Gate

To prove that any Boolean function can be implemented using only NAND gates, we will show that the AND, OR, and NOT operations can be performed using only these gates. A universal gate is a gate which can implement any Boolean function without need to use any other gate type.

Implementing an Inverter Using only NAND Gate

The figure shows two ways in which a NAND gate can be used a san inverter (NOT gate).

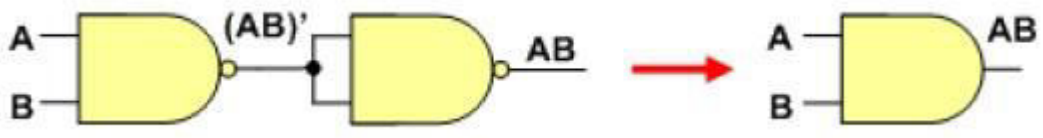
1. All NAND input pins connect to the input signal A gives an output A'.



2. One NAND input pin is connected to the input signal A while all other input pins are connected to logic1. The output will be A'.

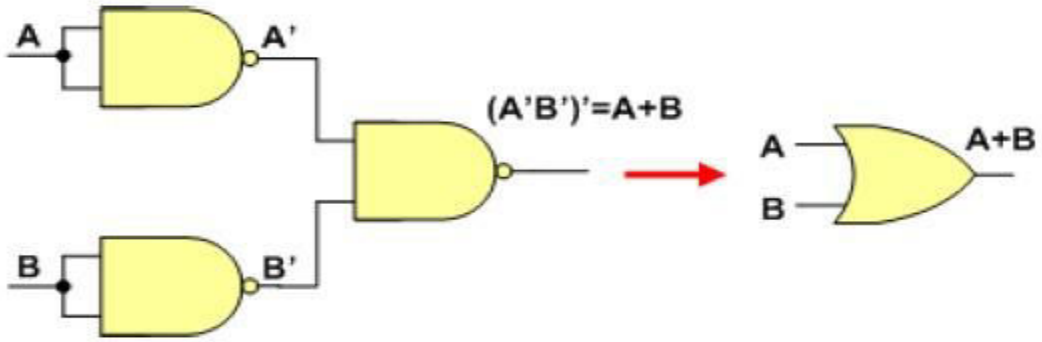
Implementing AND Using only NAND Gates

An AND gate can be replaced by NAND gates as shown in the figure (The AND is replaced by a NAND gate with its output complemented by a NAND gate inverter).



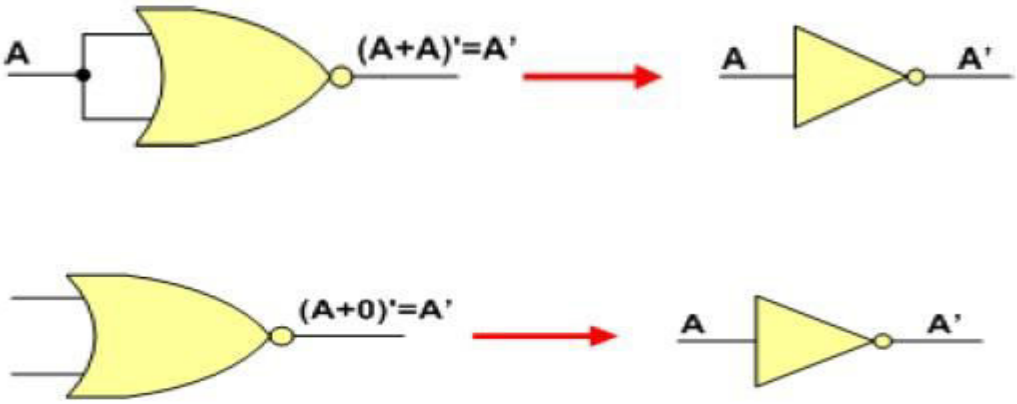
Implementing OR Using only NAND Gates

An OR gate can be replaced by NAND gates as shown in the figure (The OR gate is replaced by a NAND gate with all its inputs complemented by NAND gate inverters).

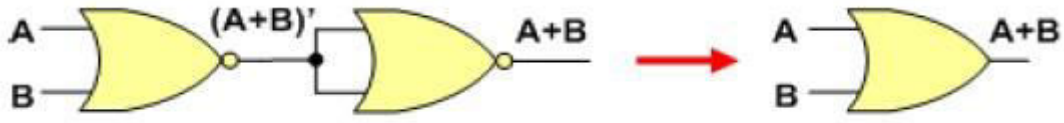


NOR Gate is a Universal Gate:

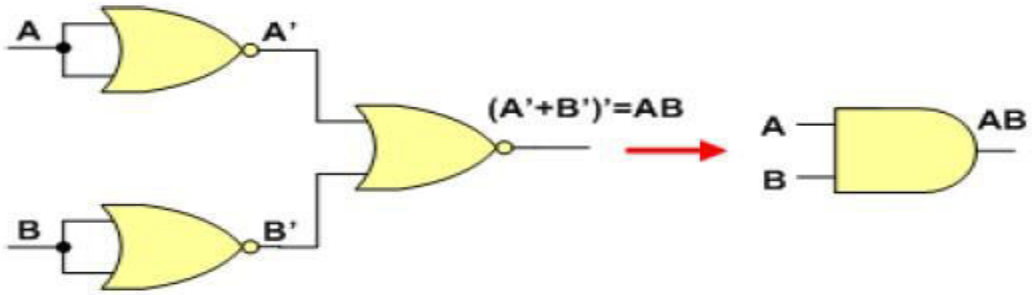
Implementing an Inverter Using only NOR Gate



Implementing OR Using only NOR Gates



Implementing AND Using only NOR Gates



Equivalent Gates: