ANNAMACHARYA UNIVERSITY

(ESTD UNDER AP PRIVATE UNIVERSITIES (ESTABLISHMENT AND REGULATION) ACT, 2016

New Boyanapalli, Rajampet,
Annamayya (Dist), Andhra Pradesh – 516 126

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

Course: DIGITAL CIRCUITS

Course Code**: 23A0252T**

Branch: EEE

Prepared by: Dr. J. Sreeranganayakulu

Designation: Assistant Professor

Department**: EEE**

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES RAJAMPET

(An Autonomous Institution)

Department of Electrical and Electronics Engineering

Title of the	DIGITAL CIRCUITS
Category:	PCC
Couse Code:	23A0252T
Year :	III
Semester	I
Branch/es:	EEE

Lecture Hours	Tutorial Hours	Practice Hours	Credits
3	-	-	3

Course Objectives:

1. To Learn Boolean algebra, logic simplification techniques, and combinational circuit design
2. To analyze combinational circuits like adders, subtractors, and code converters.
3. To explore combinational logic circuits and their applications in digital design.
4. To understand sequential logic circuits, including latches, flip-flops, counters, and shift registers.
5. To gain knowledge about programmable logic devices and digital IC's.

Course Outcomes:

At the end of the course, the student will be able to

1. Learn Boolean algebra, logic simplification techniques, and combinational circuit design. L1
2. Analyze combinational circuits like adders, subtractors, and code converters. L2
3. Explore combinational logic circuits and their applications in digital design. L3
4. Understand sequential logic circuits, including latches, flip-flops, counters, and shift registers. L1
5. Gain knowledge about programmable logic devices and digital IC's. L3

Unit 1 Logic Simplification and Combinational Logic Design: 9

Review of Boolean Algebra and De Morgan's Theorem, SOP & POS forms, Canonical forms, Introduction to Logic Gates, Ex-OR, Ex-NOR operations, Minimization of Switching Functions: Karnaugh map method, Logic function realization: AND-OR, ORAND and NAND/NOR realizations.

Unit 2 Introduction to Combinational Design 1: 9

Binary Adders, Subtractors and BCD adder, Code converters -Binary to Gray, Gray to Binary, BCD to excess3, BCD to Seven Segment display.

Unit 3 Combinational Logic Design 2: 9

Decoders, Encoders, Priority Encoder, Multiplexers, Demultiplexers, Comparators, Implementations of Logic Functions using Decoders and Multiplexers.

Unit 4 Sequential Logic Design: 9

Latches, Flip-flops, S-R, D, T, JK and Master-Slave JK FF, Edge triggered FF, set up and hold times, Ripple counters, Shift registers.

Unit 5 Programmable Logic Devices & Digital IC's: 9

Programmable Logic Devices: ROM, Programmable Logic Devices (PLA and PAL).

Digital IC's: Decoder (74x138), Priority Encoder (74x148), multiplexer (74x151) and de-multiplexer (74x155), comparator (74x85)

Prescribed Textbooks:

1. Digital Design, M.Morris Mano & Michel D. Ciletti, 5th Edition, Pearson Education, 1999
2. Switching theory and Finite Automata Theory, Zvi Kohavi and Nirah K.Jha, 2nd Edition, Tata
3. McGraw Hill, 2005.

Reference Books:

1. Fundamentals of Logic Design, Charles H Roth,Jr., 5th Edition, Brooks/cole Cengage Learning, 2004.

Online Learning Resources:

1. https://onlinecourses.nptel.ac.in/noc21_ee75/preview
2. https://www.nesoacademy.org/ee/05-digital-electronics

CO-PO Mapping:

Course Outcomes	Engineering Knowledge	Problem Analysis	Design/Development of solutions	Conduct investigations of	Modern tool usage	The engineer and society	Environment and sustainability	Ethics	Individual and team work	Communication	Project management and	Life-long learning	PSO1	PSO2
.1	**3**	**2**	**2**	**1**	**2**	**1**	**-**	**-**	**-**	**-**	**-**	**2**	**3**	**2**
.2	**3**	**3**	**2**	**2**	**2**	**1**	**-**	**-**	**-**	**-**	**-**	**2**	**3**	**2**
.3	**3**	**2**	**3**	**1**	**3**	**1**	**-**	**-**	**-**	**-**	**-**	**3**	**3**	**3**
.4	**3**	**2**	**2**	**2**	**2**	**1**	**-**	**-**	**-**	**-**	**-**	**2**	**3**	**2**
.5	**3**	**2**	**3**	**2**	**3**	**1**	**-**	**-**	**-**	**-**	**-**	**3**	**3**	**3**

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET
(AUTONOMOUS)

UNIT I

NUMBER SYSTEMS, CODES & BOOLEAN ALGEBRA

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

Topic

NUMBER SYSTEMS

OVERVIEW

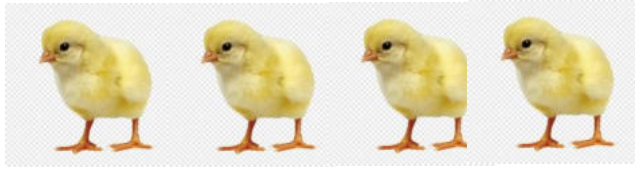
- Philosophy of number systems
- Applications
- Decimal Number System
- Binary Number System
- Octal Number System
- Hexadecimal Number System

Number System

= 1

= 3

= 4



= 4

= Chicks =3

Hen=1

- Number system is a basic for counting various items.
- The familiar decimal number system contains 10 unique symbols or values. i.e. (0 to 9).

Applications

1. Security

2. Computers

3. Military , Navy , Air force

Number System

As a computer programmer or an IT professional, you should understand the following number systems which are frequently used in computers.

- Number systems are divided into 4 types.

S.N.	Number Systems
1	Decimal Number System
2	Binary Number System
3	Octal Number System
4	Hexa Decimal Number System

Decimal Number System

The number system that we use in our day-to-day life is the decimal number system. Decimal number system has base 10 as it uses 10 digits from 0 to 9. In decimal number system, the successive positions to the left of the decimal point represents units, tens, hundreds, thousands and so on. 0 1 2 3 4 5 6 7 8 9

Each position represents a specific power of the base (10)

For example

The decimal number is **1234** → $1x10^3+2x10^2+3x10^1+4x10^0$

The digit **4** in the units position $1x1000+2x100+3x10+4x10$

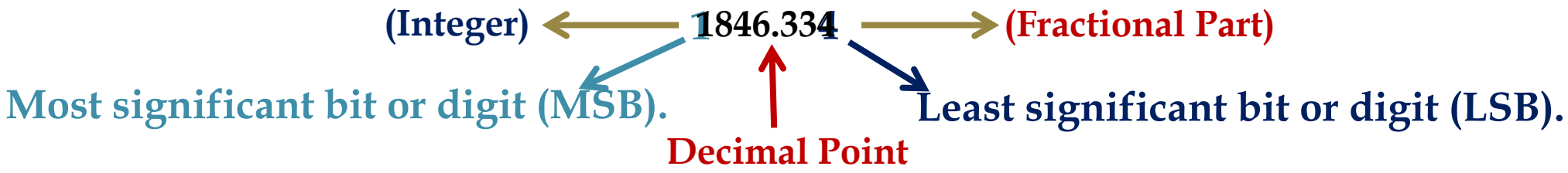
The digit **3** in the tens position $1000+200+30+40$

The digit **2** in the hundreds position

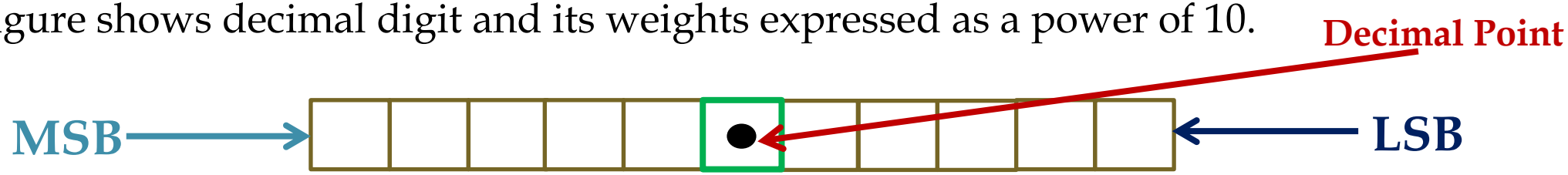
And **1** in the thousands position 1234

Decimal Number System

- The Decimal number system contains 10 unique symbols. Since counting in decimal denotes 10 symbols. We can say that its base or radix is 10.
- In Decimal number system we can express any decimal number in units, tens, hundreds….
- Each symbol in the number is called a digit.

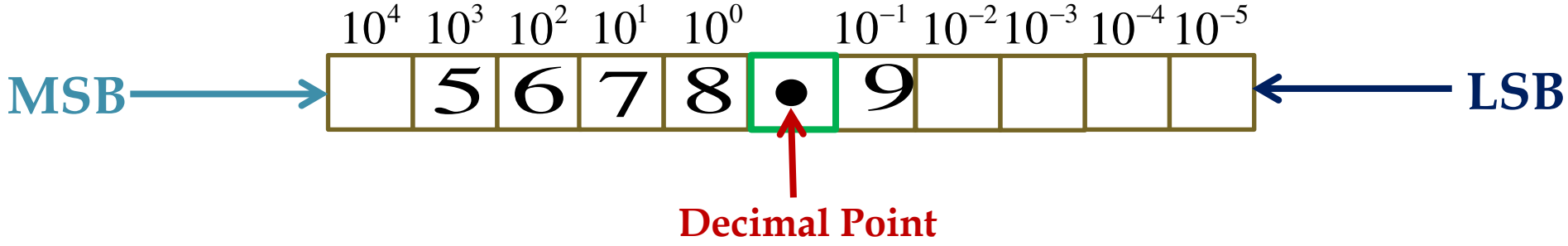


- The position of the digit with reference to decimal point determines its **value** or **weight**.
- The sum of all the digits multiplied by their weights gives the total number.
- The figure shows decimal digit and its weights expressed as a power of 10.



Decimal Number System

- In a Decimal number system weight is expressed as a power of 10 as shown in figure.



Example: $(5678.9)_{10}$

$\Rightarrow 5x10^3+6x10^2+7x10^1+8x10^0+9x10^{-1}$

$\Rightarrow 5x1000+6x100+7x10+8x1+9x\frac{1}{10}$

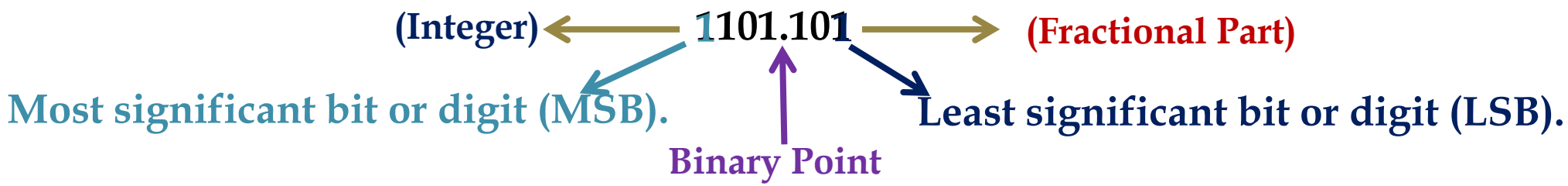
$\Rightarrow 5000+600+70+8+0.9$

$\Rightarrow (5678.9)_{10} = (5678.9)_{10}$

$\therefore (5678.9)_{10} = (5678.9)_{10}$

Binary Number System

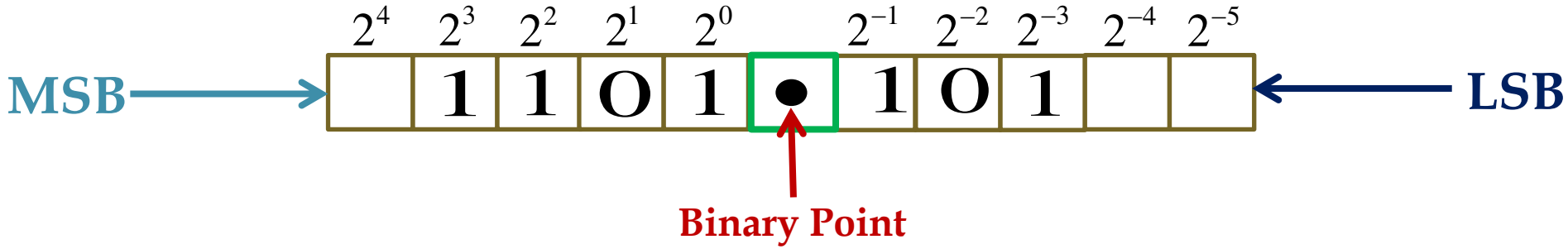
- It contains **2** unique symbols. The **2** binary digits(bits) are **0** and **1**. It is a base 2 system. (Base or radix =**2**)
- The digits computers understand only binary number system.



- The position of the bit with reference to binary point determines its **value** or **weight**.
- The sum of all the digits multiplied by their weights gives the total number.

Binary Number System

- In a binary number system weight is expressed as a power of 2 as shown in figure.



Example: $(1101.101)_2$

$1x2^3+1x2^2+0x2^1+1x2^0 + 1x2^{-1}+0x2^{-2}+1x2^{-3}$

$1x8 +1x4 +0x2 +1x1 + 1x\frac{1}{2}+0x\frac{1}{2^2}+1x\frac{1}{2^3}$

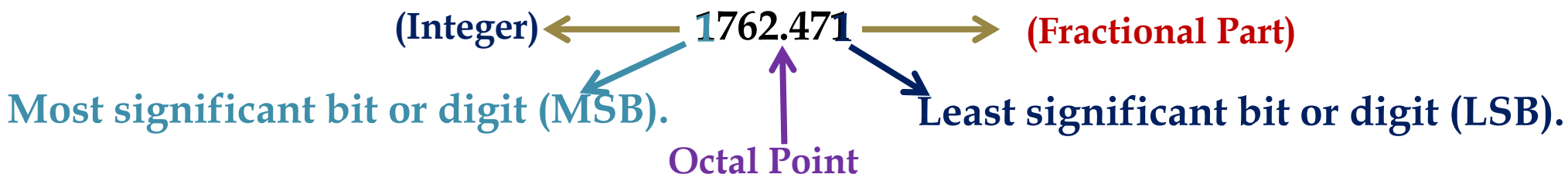
$8 + 4 + 0 + 1 + 0.5 + 0 + 0.125$

$(13.625)_{10} = (1101.101)_2$

$\therefore (1101.101)_2 = (13.625)_{10}$

Octal Number System

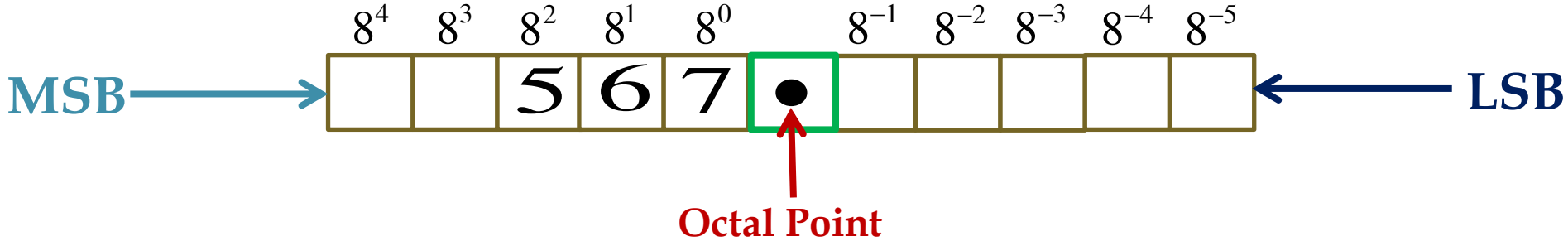
- It uses first 8 digits of decimal number system i.e. (0 to 7). 0,1,2,3,4,5,6,7
- It uses 8 digits so its base or radix is 8.



- The position of the bit with reference to Octal point determines its **value** or **weight**.
- The sum of all the digits multiplied by their weights gives the total number.

Octal Number System

- In a Octal number system weight is expressed as a power of 8 as shown in figure.



Example: $(567)_8$

$5x8^2+6x8^1+7x8^0$

$5x64 +6x8 +7x1$

$320 + 48 + 7$

$(375)_{10} = (567)_8$

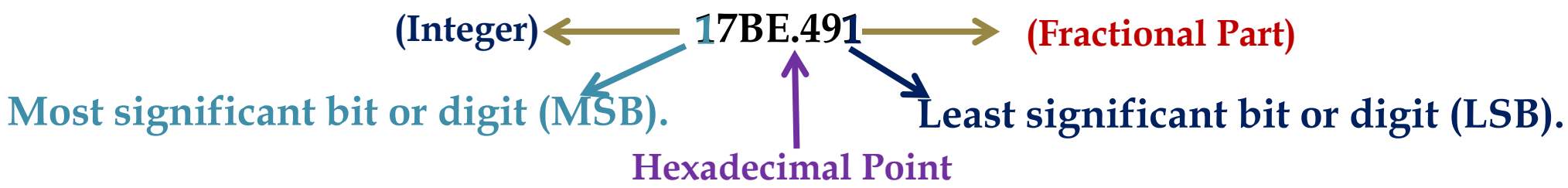
$\therefore (1101.101)_2 = (13.625)_{10}$

Hexadecimal Number System

- The hexadecimal number system has a base of 16 having 16 digits i.e. (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F).

A=10 B=11 C=12 D=13 E=14 F=15

- It uses 16 digits so its base or radix is 16.

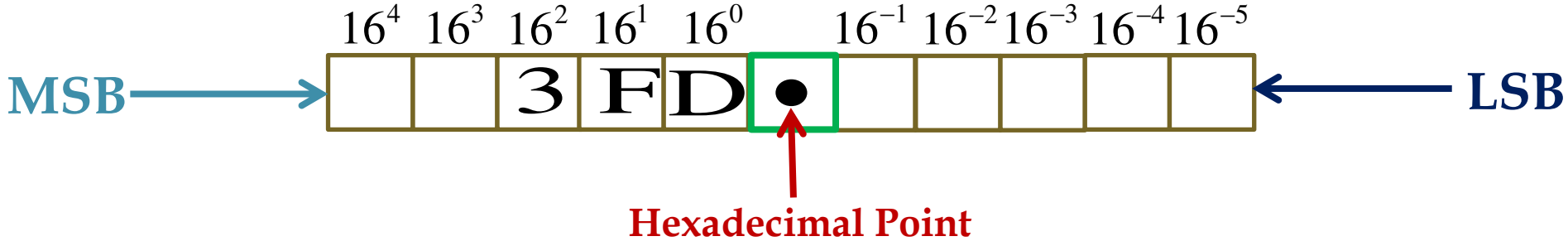


- The position of the bit with reference to hexadecimal point determines its **value** or **weight**.

- The sum of all the digits multiplied by their weights gives the total number.

Hexadecimal Number System

- In a Hexadecimal number system weight is expressed as a power of 16 as shown in figure.



Example: $(3FD)_{16}$

$3x16^2+Fx16^1+Dx16^0$

$3x256+15x16+13x1$

$768+ 240+ 13$

$(1021)_{10} = (3FD)_{16}$

$\therefore (3FD)_{16} = (1021)_{10}$

Number Systems

S.N.	Number Systems	Radix	Description
1	Decimal Number System	Base 10	Digits used: 0 to 9 (0, 1,2,3,4,5,6,7,8,9)
2	Binary Number System	Base 2	Digits used: 0, 1
3	Octal Number System	Base 8	Digits used: 0 to 7 (0,1,2,3,4,5,6,7)
4	Hexa Decimal Number System	Base 16	Digits used: 0 to 9, Letters used: A- F

0, 1, 2, 3, 4, 5, 6, 7, 8, 9	A ,B , C , D ,E ,F

A=10 , B=11 , C=12 , D=13 ,E=14 , F=15

OVERVIEW

- Format of Binary Numbers
- Counting in radix or base
- Decimal Number System to any base or radix conversion
- Binary Number System to any base or radix conversion
- Octal Number System to any base or radix conversion
- Hexadecimal Number System to any base or radix conversion

Format of Binary Numbers

Nibble	=	4 bits	=	2^4	Distinct values	=	16
Bytes	=	8 bits	=	2^8	Distinct values	=	256
Word	=	16 bits	=	2^{16}	Distinct values	=	65536
Double word	=	32 bits	=	2^{32}	Distinct values	=	4294967292

Counting in Radix or Base

S.N.	Number Systems	Base or Radix	Characters or Symbols
1	Decimal Number System	Base 10	Digits used: 0 to 9 (0, 1,2,3,4,5,6,7,8,9)
2	Binary Number System	Base 2	Digits used: 0, 1
3	Octal Number System	Base 8	Digits used: 0 to 7 (0,1,2,3,4,5,6,7)
4	Hexa Decimal Number System	Base 16	Digits used: 0 to 9, Letters used: A- F

0, 1, 2, 3, 4, 5, 6, 7, 8, 9	A ,B , C , D ,E ,F

A=10 , B=11 , C=12 , D=13 ,E=14 , F=15

Number Base Conversion

The digital circuits and systems work strictly in binary form. We are using octal and hexadecimal only as a convenience for the operation of the systems

Number Base Conversion

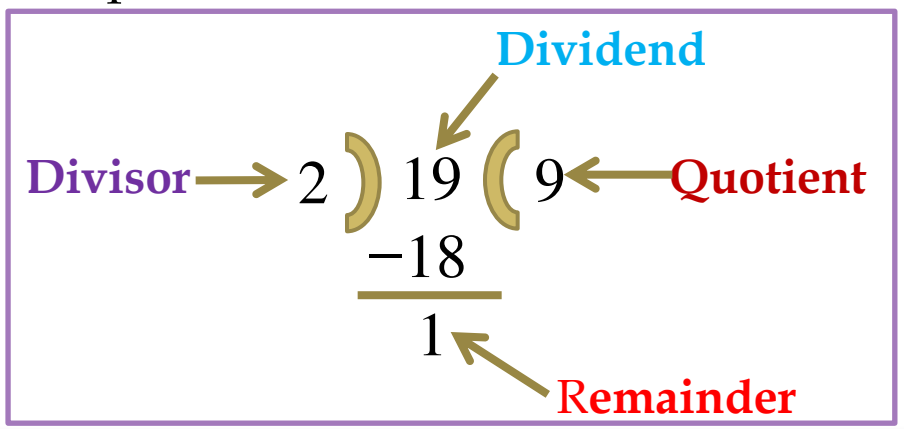
The digital circuits and systems work strictly in binary form. We are using octal and hexadecimal only as a convenience for the operation of the systems

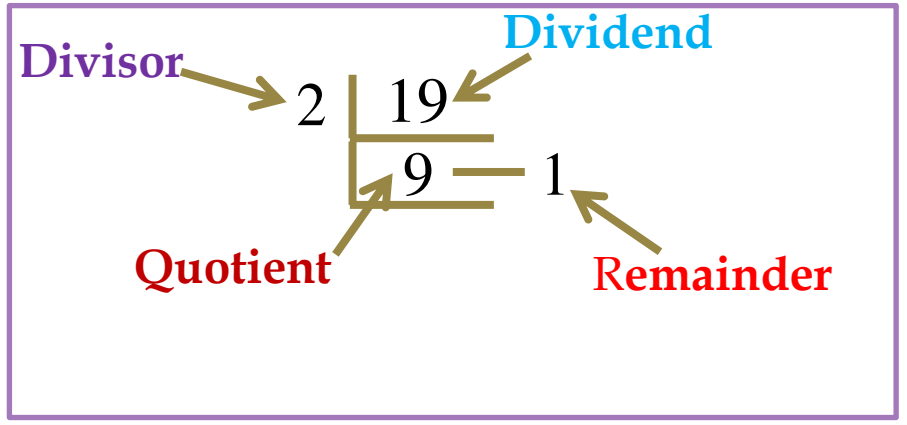
Conversion of Decimal numbers into any radix number

Steps in Successive Division Method:

1. Divide the **integral part of decimal number** by desired base number. (store quotient and remainder).
2. Consider the **quotient** as a new decimal number and **repeat step1** until **quotient** becomes zero.
3. List the **remainders** in reverse order. **(From bottom to Top)**

Example: $19 \div 2$





Conversion of Decimal numbers into any radix number

Steps in Successive Multiplication Method:

1. Multiply the fractional part of decimal number by desired base number.
2. Record the integer part of product as a carry and fractional part as new fractional.
3. Repeat step1 and step2 until fractional part of number becomes zero **(or)** until we have many digits as necessary for your application.
4. Read carry downwards to get desired base number.

Example: 1896.456

Integer = 1896

Fractional Part = .456

Conversion of Decimal numbers into Binary number

Example: Convert **decimal number** 37.95 into **binary equivalent**.

Solution: $(37.95)_{10} = (\ ?\ )_2$

Integer Part = 37

Fractional Part = 0.95

In Integer Part : By Successive division method

2	37	
2	18	1
2	9	0
2	4	1
2	2	0
	1	0

$(37)_{10} = (100101)_2$

In Fractional Part : By Successive Multiplication method

$0.95 \times 2 = 1.90 = 0.9 \rightarrow 1$

$0.9 \times 2 = 1.8 = 0.8 \rightarrow 1$

$0.8 \times 2 = 1.6 = 0.6 \rightarrow 1$

$0.6 \times 2 = 1.2 = 0.2 \rightarrow 1$

$0.2 \times 2 = 0.4 = 0.4 \rightarrow 0$

$0.4 \times 2 = 0.8 = 0.8 \rightarrow 0$

$0.8 \times 2 = 1.6 = 0.6 \rightarrow 1$

$(0.95)_{10} = (1111001)_2$

$\therefore (37.95)_{10} = (100101.1111001)_2$

Conversion of Decimal numbers into Binary number

Example: Convert **12.125** decimal number into binary equivalent.

Solution: $(12.125)_{10} = (\ ?\ )_2$

Integer Part = 12

Fractional Part = 0.125

In Integer Part : By Successive division method

2	12	
2	6	0
2	3	0
	1	1

$(12)_{10} = (1\ 1\ 0\ 0)_2$

In Fractional Part : By Successive Multiplication method

$0.125 \times 2 = 0.25 = 0.25 \rightarrow 0$

$0.25 \times 2 = 0.5 = 0.5 \rightarrow 0$

$0.5 \times 2 = 1.0 = 0.0 \rightarrow 1$

$0.0 \times 2 = 0.0 = 0.0 \rightarrow 0$

$(0.125)_{10} = (0\ 0\ 1\ 0)_2$

$\therefore (12.125)_{10} = (1100.0010)_2$

Conversion of Decimal numbers into Octal number

Example: Convert **decimal number** 658.825 into **Octal equivalent**.

Solution: $(658.825)_{10} = (\ ?\ )_8$

Integer Part = 658

Fractional Part = 0.825

In Integer Part : By Successive division method

8	658	
8	82	2
8	10	2
	1	2

$(658)_{10} = (1222)_8$

In Fractional Part : By Successive Multiplication method

$0.825 \times 8 = 6.6 = 0.6 \rightarrow 6$

$0.6 \times 8 = 4.8 = 0.8 \rightarrow 4$

$0.8 \times 8 = 6.4 = 0.6 \rightarrow 6$

$0.4 \times 8 = 3.2 = 0.2 \rightarrow 3$

$0.2 \times 8 = 1.6 = 0.6 \rightarrow 1$

$0.6 \times 8 = 4.8 = 0.8 \rightarrow 4$

$0.8 \times 8 = 6.4 = 0.6 \rightarrow 6$

$(0.825)_{10} = (6\,4\,6\,3\,1\,4)_8$

$\therefore (658.825)_{10} = (1222.646314)_8$

Conversion of Decimal numbers into Hexa decimal number

Example: Convert **5386.345 decimal number** into **Hexadecimal equivalent**.

Solution: $(5386.345)_{10} = (\ ?\ )_{16}$

Integer Part = 5386

Fractional Part = 0.345

In Integer Part : By Successive division method

16	5386	
16	236	10(A)
16	21	0
	1	5

$(5386)_{10} = (1\ 5\ 0\ A)_{16}$

In Fractional Part : By Successive Multiplication method

$0.345 \times 16 = 5.52 = 0.52 \rightarrow 5$

$0.52 \times 16 = 8.32 = 0.32 \rightarrow 8$

$0.32 \times 16 = 5.12 = 0.12 \rightarrow 5$

$0.12 \times 16 = 1.92 = 0.92 \rightarrow 1$

$(0.345)_{10} = (5851)_{16}$

$\therefore (5386.345)_{10} = (150A.5851)_{16}$

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET
(AUTONOMOUS)

Topic

NUMBER SYSTEM BASE CONVERSION

OVERVIEW

- Format of Binary Numbers
- Counting in radix or base
- Decimal Number System to any base or radix conversion
- Binary Number System to any base or radix conversion
- Octal Number System to any base or radix conversion
- Hexadecimal Number System to any base or radix conversion

Format of Binary Numbers

Nibble	=	4 bits	=	2^4	Distinct values	=	16
Bytes	=	8 bits	=	2^8	Distinct values	=	256
Word	=	16 bits	=	2^{16}	Distinct values	=	65536
Double word	=	32 bits	=	2^{32}	Distinct values	=	4294967292

Counting in Radix or Base

S.N.	Number Systems	Base or Radix	Characters or Symbols
1	Decimal Number System	Base 10	Digits used: 0 to 9 (0, 1,2,3,4,5,6,7,8,9)
2	Binary Number System	Base 2	Digits used: 0, 1
3	Octal Number System	Base 8	Digits used: 0 to 7 (0,1,2,3,4,5,6,7)
4	Hexa Decimal Number System	Base 16	Digits used: 0 to 9, Letters used: A- F

0, 1, 2, 3, 4, 5, 6, 7, 8, 9	A ,B , C , D ,E ,F

A=10 , B=11 , C=12 , D=13 ,E=14 , F=15

Number Base Conversion

The digital circuits and systems work strictly in binary form. We are using octal and hexadecimal only as a convenience for the operation of the systems

Number Base Conversion

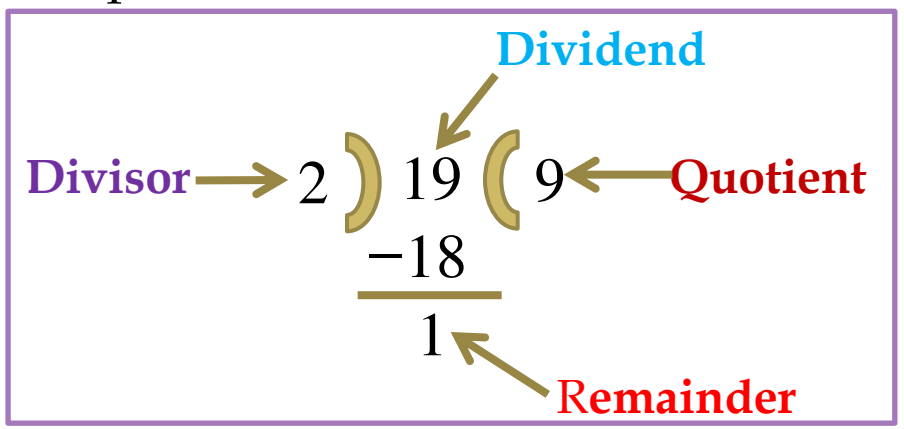
The digital circuits and systems work strictly in binary form. We are using octal and hexadecimal only as a convenience for the operation of the systems

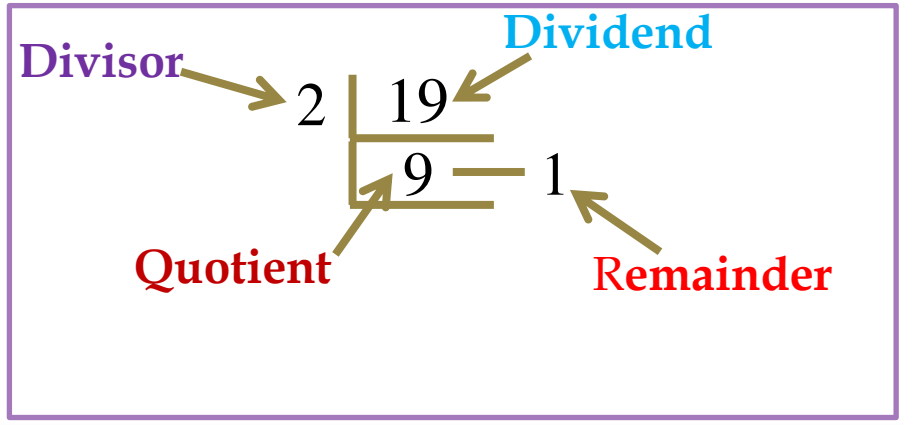
Conversion of Decimal numbers into any radix number

Steps in Successive Division Method:

1. Divide the **integral part of decimal number** by desired base number. (store quotient and remainder).
2. Consider the **quotient** as a new decimal number and **repeat step1** until **quotient** becomes zero.
3. List the **remainders** in reverse order. **(From bottom to Top)**

Example: $19 \div 2$





Conversion of Decimal numbers into any radix number

Steps in Successive Multiplication Method:

1. Multiply the fractional part of decimal number by desired base number.
2. Record the integer part of product as a carry and fractional part as new fractional.
3. Repeat step1 and step2 until fractional part of number becomes zero **(or)** until we have many digits as necessary for your application.
4. Read carry downwards to get desired base number.

Example: 1896.456

Integer = 1896

Fractional Part = .456

Conversion of Decimal numbers into Binary number

Example: Convert **decimal number** 37.95 into **binary equivalent**.

Solution: $(37.95)_{10} = (\ ?\ )_2$

Integer Part = 37

Fractional Part = 0.95

In Integer Part : By Successive division method

2	37	
2	18	1
2	9	0
2	4	1
2	2	0
	1	0

$(37)_{10} = (100101)_2$

In Fractional Part : By Successive Multiplication method

$0.95 \times 2 = 1.90 = 0.9 \rightarrow 1$

$0.9 \times 2 = 1.8 = 0.8 \rightarrow 1$

$0.8 \times 2 = 1.6 = 0.6 \rightarrow 1$

$0.6 \times 2 = 1.2 = 0.2 \rightarrow 1$

$0.2 \times 2 = 0.4 = 0.4 \rightarrow 0$

$0.4 \times 2 = 0.8 = 0.8 \rightarrow 0$

$0.8 \times 2 = 1.6 = 0.6 \rightarrow 1$

$(0.95)_{10} = (1111001)_2$

$\therefore (37.95)_{10} = (100101.1111001)_2$

Conversion of Decimal numbers into Binary number

Example: Convert **12.125** decimal number into binary equivalent.

Solution: $(12.125)_{10} = (\ ?\ )_2$

Integer Part = 12

Fractional Part = 0.125

In Integer Part : By Successive division method

2	12	
2	6	0
2	3	0
	1	1

$(12)_{10} = (1\ 1\ 0\ 0)_2$

In Fractional Part : By Successive Multiplication method

$0.125 \times 2 = 0.25 = 0.25 \rightarrow 0$

$0.25 \times 2 = 0.5 = 0.5 \rightarrow 0$

$0.5 \times 2 = 1.0 = 0.0 \rightarrow 1$

$0.0 \times 2 = 0.0 = 0.0 \rightarrow 0$

$(0.125)_{10} = (0\ 0\ 1\ 0)_2$

$\therefore (12.125)_{10} = (1100.0010)_2$

Conversion of Decimal numbers into Octal number

Example: Convert **decimal number** 658.825 into **Octal equivalent**.

Solution: $(658.825)_{10} = (\ ?\ )_8$

Integer Part = 658

Fractional Part = 0.825

In Integer Part : By Successive division method

8	658	
8	82	2
8	10	2
	1	2

$(658)_{10} = (1222)_8$

In Fractional Part : By Successive Multiplication method

$0.825 \times 8 = 6.6 = \square \rightarrow 6$

$0.6 \times 8 = 4.8 = 0.8 \rightarrow 4$

$0.8 \times 8 = 6.4 = 0.4 \rightarrow 6$

$0.4 \times 8 = 3.2 = 0.2 \rightarrow 3$

$0.2 \times 8 = 1.6 = 0.6 \rightarrow 1$

$0.6 \times 8 = 4.8 = 0.8 \rightarrow 4$

$0.8 \times 8 = 6.4 = 0.6 \rightarrow 6$

$(0.825)_{10} = (6\,4\,6\,3\,1\,4)_8$

$\therefore (658.825)_{10} = (1222.646314)_8$

Conversion of Decimal numbers into Hexa decimal number

Example: Convert **5386.345 decimal number** into **Hexadecimal equivalent**.

Solution: $(5386.345)_{10} = (\ ?\ )_{16}$

Integer Part = 5386

Fractional Part = 0.345

In Integer Part : By Successive division method

16	5386	
16	236	10(A)
16	21	0
	1	5

$(5386)_{10} = (1\ 5\ 0\ A)_{16}$

In Fractional Part : By Successive Multiplication method

$0.345 \times 16 = 5.52 = 0.52 \rightarrow 5$

$0.52 \times 16 = 8.32 = 0.32 \rightarrow 8$

$0.32 \times 16 = 5.12 = 0.12 \rightarrow 5$

$0.12 \times 16 = 1.92 = 0.92 \rightarrow 1$

$(0.345)_{10} = (5851)_{16}$

$\therefore (5386.345)_{10} = (150A.5851)_{16}$

Conversion of Decimal numbers into any radix number

Example: Convert **decimal number** into **Hexa decimal equivalent**.

Example: Convert **decimal number** into **Hexa decimal equivalent**.

Example: Convert **decimal number** into **Binary equivalent**.

Example: Convert **decimal number** into **Octal equivalent**.

Conversion of Binary numbers into any radix number

Example: Convert **100101.1111001** **Binary number** into **decimal equivalent**.

Solution: $(100101.1111001)_2 = (\ ?\ )_{10}$

$$2^5\,2^4\,2^3\,2^2\,2^1\,2^0 \quad 2^{-1}\,2^{-2}\,2^{-3}\,2^{-4}\,2^{-5}\,2^{-6}\,2^{-7}$$
$$(\ 1\ 0\ 0\ 1\ 0\ 1\ .\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ )_2$$

$\Rightarrow 1x2^5+0x2^4+0x2^3+1x2^2+0x2^1 + 1x2^{-1}+1x2^{-2}+1x2^{-3}+1x2^{-4}+0x2^{-5}+0x2^{-6}+1x2^{-7}$

$\Rightarrow 1x32+0x16+0x8+1x4+0x2+1x1 + 1x\frac{1}{2}+1x\frac{1}{2^2}+1x\frac{1}{2^3}+1x\frac{1}{2^4}+0x\frac{1}{2^5}+0x\frac{1}{2^6}+1x\frac{1}{2^7}$

$\Rightarrow 1x32+0x16+0x8+1x4+0x2+1x1 + 1x0.5+1x0.25+1x0.125+1x0.0625+0x0.03125$
$+0x0.015625+1x0.0078125$

$\Rightarrow 32+0+0+4+0+1 + 0.5+0.25+0.125+0.0625+0+0$
$+0.0078125$

$\Rightarrow (37.9453125)_{10} = (100101.1111001)_2$

Conversion of Binary numbers into Octal number

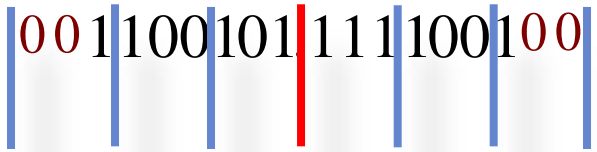
Procedure for Binary to Octal Conversion:

1. Base or Radix for binary number is 2 and octal is 8.
2. The base for octal number is the third power of the base for binary number.

   Base for octal number is $8 = 2^3$

3. Make a group of 3 bits starting from binary point towards MSB, add zero's at MSB side if required and starting from binary point towards LSB, add zero's at LSB side if required.

Example: $(1100101.1111001)_2$

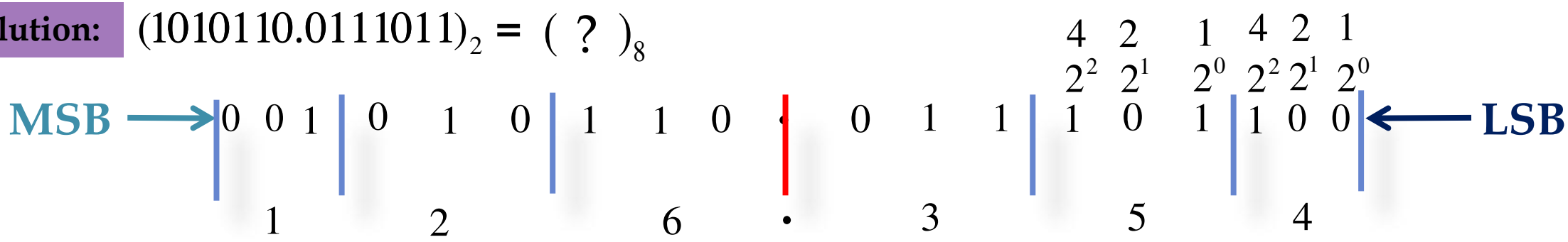


4. By grouping 3 digits of binary number and then convert each group bits to its octal equivalent.

Conversion of Binary numbers into Octal number

Example: Convert **1010110.0111011 Binary number** into **Octal equivalent**.

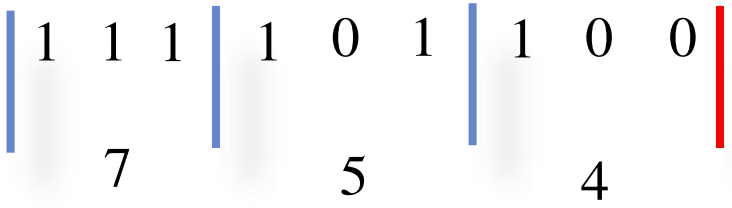
Solution: $(1010110.0111011)_2 = (\ ?\ )_8$



$\therefore (1010110.0111011)_2 = (126.354)_8$

Example: Convert **111101100 Binary number** into **Octal equivalent**.

Solution: $(111101100)_2 = (\ ?\ )_8$



$\therefore (111101100)_2 = (754)_8$

Conversion of Binary numbers into Hexadecimal number

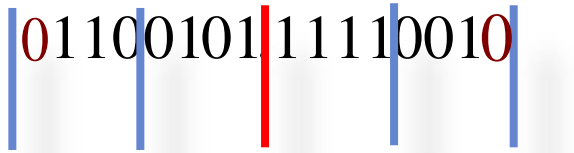
Procedure for Binary to Hexa Decimal Conversion:

1. Base or Radix for binary number is 2 and Hexa decimal is 16.
2. The base for Hexadecimal number is the fourth power of the base for binary number.

   Base for Hexadecimal number is $16 = 2^4$

3. Make a group of 4 bits starting from binary point towards MSB, add zero's at MSB side if required and starting from binary point towards LSB, add zero's at LSB side if required.

Example: $(1100101.1111001)_2$

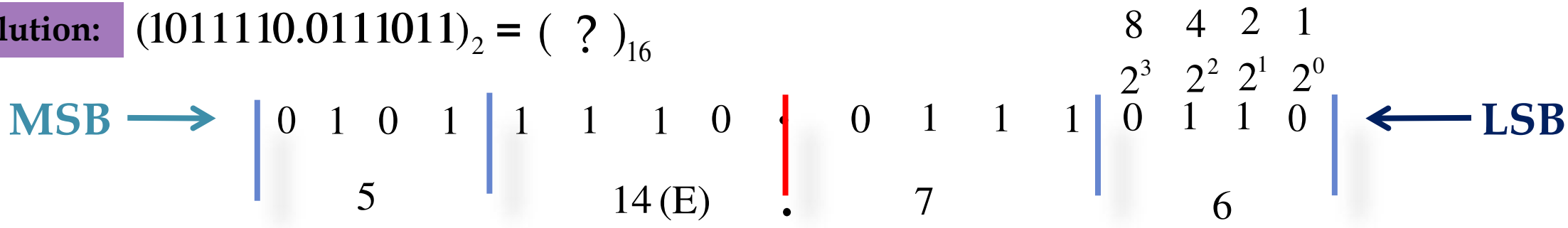


4. By grouping 4 digits of binary number and then convert each group bits to its hexadecimal equivalent.

Conversion of Binary numbers into Hexa Decimal number

Example: Convert **1011110.0111011 Binary number** into **Hexa decimal equivalent**.

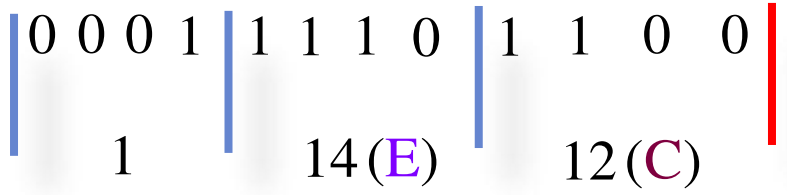
Solution: $(1011110.0111011)_2 = (\ ?\ )_{16}$



$\therefore (1011110.0111011)_2 = (5E.76)_{16}$

Example: Convert **111101100 Binary number** into **Hexa decimal equivalent**.

Solution: $(111101100)_2 = (\ ?\ )_{16}$



$\therefore (111101100)_2 = (1EC)_{16}$

Conversion of Binary numbers into any radix number

Example: Convert **1101100010011011 Binary number** into **Hexa decimal equivalent**.

Example: Convert **10010001011.00101110 Binary number** into **Hexa decimal equivalent**.

Example: Convert **1010110.0111011 Binary number** into **Decimal equivalent**.

Example: Convert **11110110.0111011011 Binary number** into **Octal equivalent**.

Conversion of Octal numbers into any radix number

Example: Convert **7456.732** Octal number into decimal equivalent.

Solution: $(7456.732)_8 = (\ ?\ )_{10}$

$$8^3\ 8^2\ 8^1\ 8^0 \quad 8^{-1}\ 8^{-2}\ 8^{-3}$$
$$(7\ 4\ 5\ 6\ .\ 7\ 3\ 2)_8$$

$$7x8^3 + 4x8^2 + 5x8^1 + 6x8^0 + 7x8^{-1} + 3x8^{-2} + 2x8^{-3}$$

$$7x512 + 4x64 + 5x8 + 6x1 + 7x\frac{1}{8} + 3x\frac{1}{8^2} + 2x\frac{1}{8^3}$$

$$3584 + 256 + 40 + 6 + 7x0.125 + 3x0.015625 + 2x0.001953$$

$$3584 + 256 + 40 + 6 + 0.875 + 0.046875 + 0.003906$$

$$(3886.92578)_{10} = (7456.732)_8$$

Conversion of Octal numbers into Binary number

Note: **Each digit** of octal number is **individually** converted into its **binary equivalent**.

Procedure for Octal to Binary Conversion:

Example: Convert octal number **634** into its **Binary equivalent.**

Solution: $(634)_8 = (\ ?\ )_2$

Step 1 : write equivalent 3 bits binary number for each octal digit.

6	3	4
↓	↓	↓
1 1 0	0 1 1	1 0 0

Step 2 : Remove any leading or tailing zeros.

$(110011100)_2 = (634)_8$

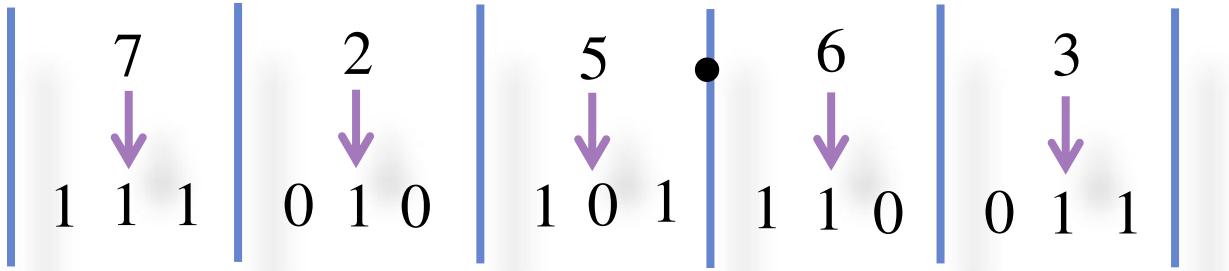
$\therefore (634)_8 = (110011100)_2$

Conversion of Octal numbers into Binary number

Example: Convert octal number **725.63** into its **Binary equivalent.**

Solution: $(725.63)_8 = (\ ?\ )_2$

Step 1 : write equivalent 3 bits binary number for each octal digit.



Step 2 : Remove any leading or tailing zeros.

$$(111010101.110011)_2 = (725.63)_8$$

$$\therefore (725.63)_8 = (111010101.110011)_2$$

Conversion of Octal numbers into Hexa Decimal number

Note: **Octal number:** → **Binary number:** → **Hexa Decimal number:**

Procedure for Octal to Hexa Decimal Conversion:

Example: Convert octal number **634.56** into its **Hexa decimal equivalent.**

Solution: $(634.56)_8 = (\ ?\ )_{16}$

Step 1 : Write equivalent 3 bits binary number for each octal digit.

6	3	4	5	6
1 1 0	0 1 1	1 0 0	1 0 1	1 1 0

Step 2 : Make a group of 4 bits starting from LSB for integer part and MSB for fractional part by adding zero's at ends if required.

Conversion of Octal numbers into Hexa Decimal number

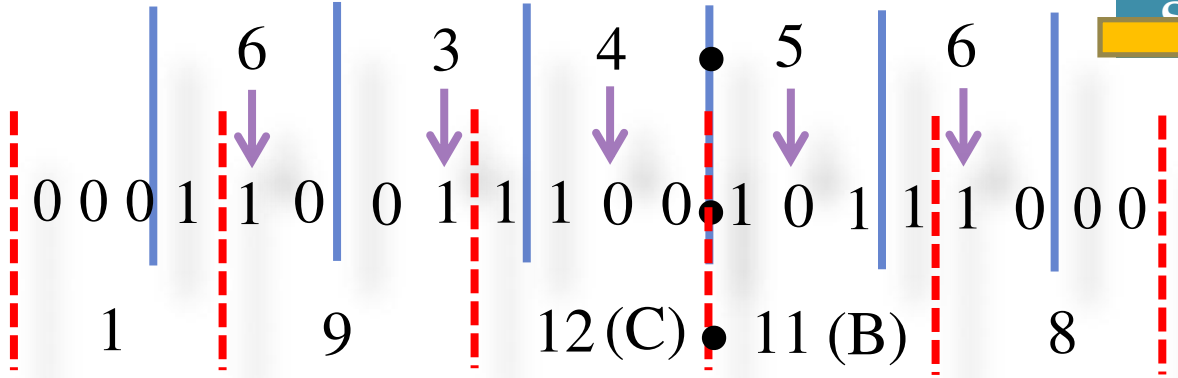
Note: **Octal number:** → **Binary number:** → **Hexa Decimal number:**

Procedure for Octal to Hexa Decimal Conversion:

Example: Convert octal number **634.56** into its **Hexa decimal equivalent.**

Solution: $(634.56)_8 = (\ ?\ )_{16}$

Step 1 : Write equivalent 3 bits binary number for each octal digit.



Step 2 : Make groups of 4 bits starting from LSB for integer part and MSB for fractional part by adding zero's at end.

Octal Number → **Binary Number** → **Hexadecimal Number**

$(19C.B8)_{16} = (634.56)_8$

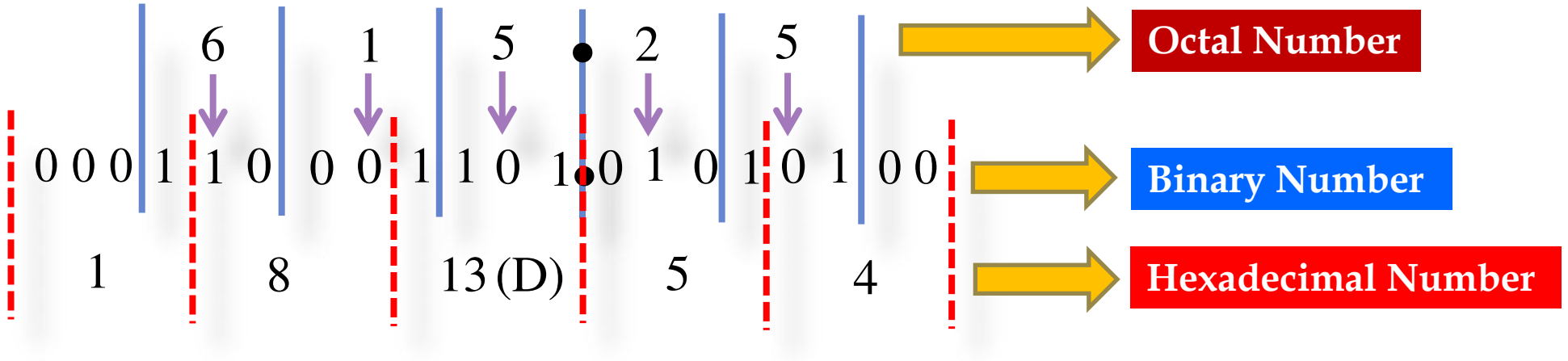
$\therefore (634.56)_8 = (19C.B8)_{16}$

Conversion of Octal numbers into Hexa Decimal number

Example: Convert octal number **615.25** into its **Hexa decimal equivalent.**

Solution: $(615.25)_8 = (\ ?\ )_{16}$

Step 1 : Write equivalent 3 bits binary number for each octal digit.



$(18D.54)_{16} = (615.25)_8$

$\therefore (615.25)_8 = (18D.54)_{16}$

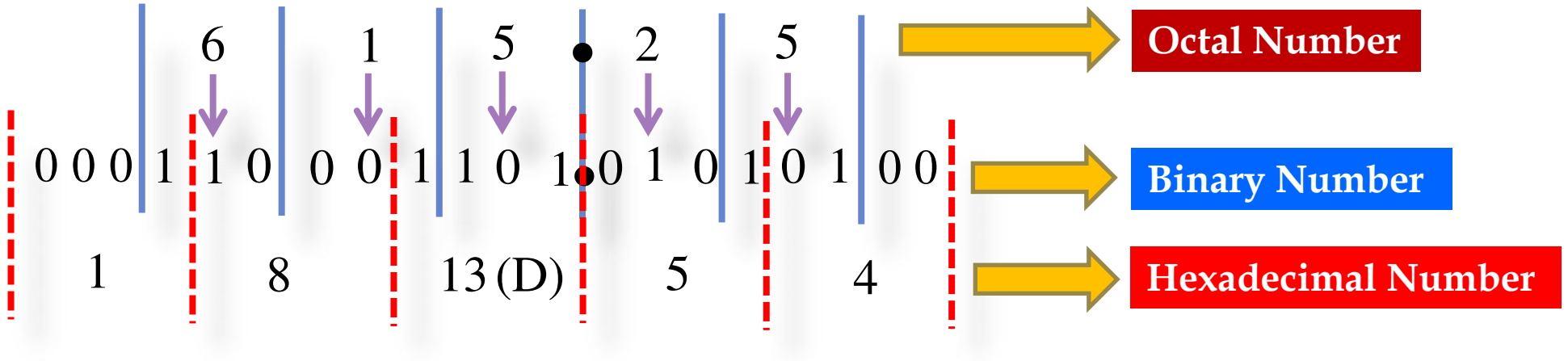
Step 2 : Make a group of 4 bits starting from LSB for integer part and MSB for fractional part by adding zero's at ends if required.

Conversion of Octal numbers into Hexa Decimal number

Example: Convert octal number **615.25** into its **Hexa decimal equivalent.**

Solution: $(615.25)_8 = (\ ?\ )_{16}$

Step 1 : Write equivalent 3 bits binary number for each octal digit.



$(18D.54)_{16} = (615.25)_8$

$\therefore (615.25)_8 = (18D.54)_{16}$

Step 2 : Make a group of 4 bits starting from LSB for integer part and MSB for fractional part by adding zero's at ends if required.

Conversion of Octal numbers into any radix number

Example:1 Convert **Octal numbers** into **Hexa decimal equivalent**.

Example:2 Convert **Octal numbers** into **Hexa decimal equivalent**.

Example:3 Convert **Octal numbers** into **Decimal equivalent**.

Example:4 Convert **Octal numbers** into **Octal equivalent**.

Conversion of Hexadecimal numbers into any radix number

Example: Convert **8A9.B4** **Hexadecimal number** into **decimal equivalent**.

Solution: $(8A9.B4)_{16} = (\ ?\ )_{10}$

$16^2 16^1 16^0 \ 16^{-1} 16^{-2}$

$(\ 8\ \ A\ \ 9\ .\ B\ \ 4\ )_{16}$

$8x16^2+Ax16^1+9x16^0 + Bx16^{-1}+4x16^{-2}$

$8x256+10x16+9x1 + 11x\frac{1}{16}+4x\frac{1}{16^2}$

$2048 + 160 + 9 + 11x0.0625+4x0.00390625$

$2048 + 160 + 9 + 0.6875+ 0.015625$

$(2217.70313)_{10} = (8A9.B4)_{16}$

Conversion of Hexadecimal number into Binary number

Example: Convert **Hexadecimal number 7CA.F3** into its **Binary equivalent.**

Solution: $(7CA.F3)_{16} = (\ ?\ )_2$

Note : Each digit of hexadecimal number is individually converted to its binary equivalent using 4 binary bits per digit.

Step 1 : write equivalent 4 bits binary number for each Hexadecimal digit.

7	C	A	F	3
0 1 1 1	1 1 0 0	1 0 1 0	1 1 1 1	0 0 1 1

Step 2 : Remove any leading or tailing zeros.

$(01111001010.11110011)_2 = (11111001010.11110011)_2 = (7CA.F3)_{16}$

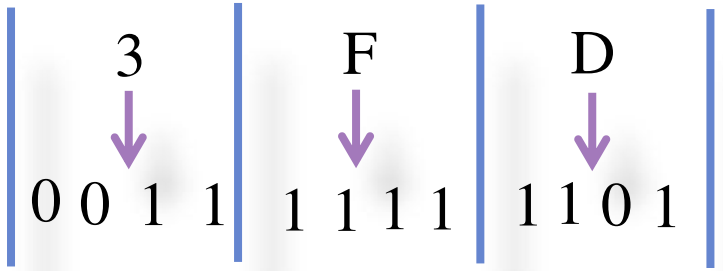
$\therefore (7CA.F3)_{16} = (11111001010.11110011)_2$

Conversion of Hexadecimal number into Binary number

Example: Convert **Hexadecimal number 3FD** into its **Binary equivalent.**

Solution: $(3FD)_{16} = (\ ?\ )_2$

Step 1 : write equivalent 4 bits binary number for each Hexadecimal digit.



Step 2 : Remove any leading or tailing zeros.

$$(00111111101)_2 = (111111101)_2 = (3FD)_{16}$$

$$\therefore (3FD)_{16} = (111111101)_2$$

Conversion of Hexadecimal number into Octal number

Procedure for Hexadecimal to Octal Conversion:

1. Convert **hexadecimal** to its **binary equivalent** and then convert binary to its **octal equivalent**
2. Write equivalent **4 bit** binary number of **each** hexadecimal digit.
3. Make a **group of 3 bits** starting from binary point towards MSB, add zero's at MSB side if required and starting from binary point towards LSB, add zero's at LSB side if required.

   Make a group of 3 bits starting from LSB for integer part and MSB for fractional part by adding zeros at ends if required.

Example: $(6D4.A1)_{16}$

6	D	4	A	1
0110	1101	0100	1010	0001

011	011	010	100	101	000	010
3	3	2	4	5	0	2

4. By grouping 3 digits of binary number and then convert each group bits to its octal equivalent.

Conversion of Hexadecimal number into Octal number

Note: Hexa Decimal number: → Binary number: → Octal number:

Procedure for Octal to Hexa Decimal Conversion:

Example: Convert **Hexa decimal number** **25B.DC** into its **octal equivalent.**

Solution: $(25B.DC)_{16} = (\ ?\ )_8$

Step 1 : Write equivalent 4 bits binary number for each Hexa digit.

2	5	B	D	C
↓	↓	↓	↓	↓
0010	0101	1011	1101	1100

Step 2 : Make a group of 4 bits starting from LSB for integer part and MSB for fractional part by adding zero's at ends if required.

Conversion of Hexadecimal number into Octal number

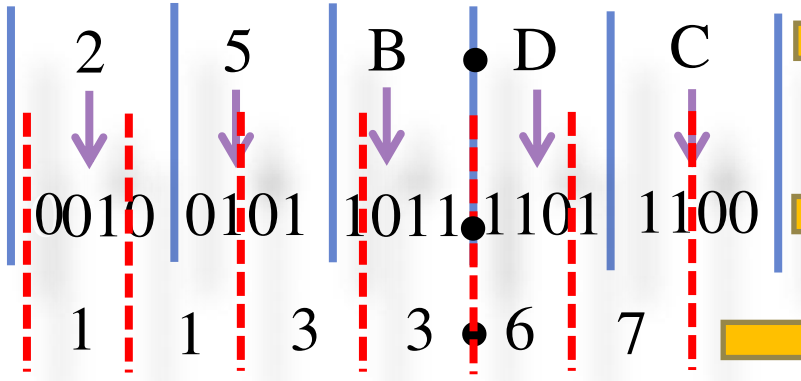
Note: Hexa Decimal number: → Binary number: → Octal number:

Procedure for Octal to Hexa Decimal Conversion:

Example: Convert **Hexa decimal number 25B.DC** into its **octal equivalent.**

Solution: $(25B.DC)_{16} = (\ ?\ )_8$

Step 1 : Write equivalent 4 bits binary number for each Hexa digit.



Hexadecimal Number

Binary Number

Octal Number

Step 2 : Make groups of 3 bits starting from LSB for integer part and MSB for fractional part by adding zero's at required.

$(1133.37)_8 = (25B.DC)_{16}$

$\therefore (25B.DC)_{16} = (2633.67)_8$

Conversion of Hexadecimal number into Octal number

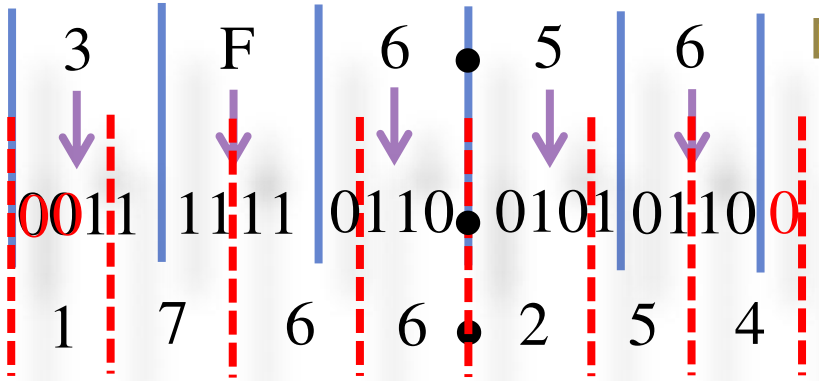
Note: Hexa Decimal number: → Binary number: → Octal number:

Procedure for Octal to Hexa Decimal Conversion:

Example: Convert **Hexa decimal number** **3F6.56** into its **octal equivalent.**

Solution: $(3F6.56)_{16} = (\ ?\ )_8$

Step 1 : Write equivalent 4 bits binary number for each hexadecimal digit.



Step 2 : Hexadecimal Number → ts starting from LSB for integer part and MSB for fractional part by adding zero's at ed.

Binary Number

Octal Number

$(1766.254)_8 = (3F6.56)_{16}$

$\therefore (3F6.56)_{16} = (1766.254)_8$

Conversion of Hexadecimal number into Octal number

Example:1 Convert **Hexa decimal number** into **Octal numbers equivalent**.

BC66.AF

Example:2 Convert **Hexa decimal number** into **Octal numbers equivalent**.

Example:3 Convert **Hexa decimal number** into **Decimal equivalent**.

Example:4 Convert **Hexa decimal number** into **Binary equivalent**.

Relation between Decimal ,Binary ,Octal and Hexadecimal number

S.No	Decimal number	Binary number	Octal number	Hexadecimal number
1	0	0 0 0 0	00	0
2	1	0 0 0 1	01	1
3	2	0 0 1 0	02	2
4	3	0 0 1 1	03	3
5	4	0 1 0 0	04	4
6	5	0 1 0 1	05	5
7	6	0 1 1 0	06	6
8	7	0 1 1 1	07	7
9	8	1 0 0 0	10	8
10	9	1 0 0 1	11	9
11	10	1 0 1 0	12	A
12	11	1 0 1 1	13	B
13	12	1 1 0 0	14	C
14	13	1 1 0 1	15	D
15	14	1 1 1 0	16	E
16	15	1 1 1 1	17	F

Conversion of any radix number into Decimal numbers

Where N=no.in decimal

A =digit ,

r =radix or base of a number

n = no.of digits in integer portion.

m = no.of digits in fractional portion.

$$N = A_{n-1}r^{n-1} + A_{n-2}r^{n-2} + A_{n-3}r^{n-3} \ldots\ldots\ldots + A_0r^0 + A_{-1}r^{-1} + A_{-2}r^{-2} + \ldots\ldots\ldots\ldots + A_{-m}r^{-m}$$

Conversion of any radix number into Decimal numbers

Example: Convert $(3102.12)_4$ into **decimal equivalent**.

Solution: $(3102.12)_4 = (\ ?\ )_{10}$

$$4^3\ 4^2\ 4^1\ 4^0 \quad 4^{-1}4^{-2}$$
$$(3\ \ 1\ \ 0\ \ 2\ .\ 1\ \ 2\ )_4$$

$\rightarrow$ $3x4^3$ [ ] $+0x4^1 +2x4^0 + 1x4^{-1}+2x4^{-2}$

$\rightarrow$ [ ] $+1x16 +0x4$ [ ] $+ 1x\frac{1}{4}+2x\frac{1}{4^2}$

$\rightarrow$ [ ] $+1x16+0x4 \quad +2x1 + 1x0.25+2x0.0625$

$\rightarrow$ $192 \quad + 16 \quad + 0 \quad + 2 \quad +$ [ ] $+0.125$

$(210.375)_{10} = (3102.12)_4$

Conversion of any radix number into Decimal numbers

Example: Convert $(11010010)_2$ into **base 4 number** .

Note: **Binary number:** → **Decimal number:** → **Any radix number:**

Solution: $(11010010)_2 = (\ ?\ )_{10}$

$$\begin{matrix} 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{matrix}$$

$$1x2^7 + 1x2^6 + 0x2^5 + 1x2^4 + 0x2^3 + 0x2^2 + 1x2^1 + 0x2^0$$

$$1x128 + 1x64 + 0x32 + 1x16 + 0x8 + 0x4 + 1x2 + 0x1$$

$$128 + 64 + 0 + 16 + 0 + 0 + 2 + 0$$

$$(\mathbf{210})_{10} = (11010010)_2$$

$$\therefore (11010010)_2 = (210)_{10} = (3102)_4$$

4	210	
4	52	2
4	13	0
	3	1

$$(3102)_4 = (210)_{10}$$

$$\therefore (210)_{10} = (3102)_4$$

Conversion of any radix number into Decimal numbers

Example: Given that $(16)_{10} = (100)_b$ find the base "**b**".

Solution: $(16)_{10} = (100)_b$

$$\begin{matrix} 10^1 10^0 & & b^2\, b^1\, b^0 \\ (\ 1\ \ 6\ )_{10} & = & (1\ \ 0\ \ 0)_b \end{matrix}$$

$$1x10^1+6x10^0 = 1xb^2+0xb^1+0xb^0$$

$$1x10 +6x1 = 1xb^2+0xb +0x1$$

$$10 + 6 = b^2 + 0 + 0$$

$$16 = b^2$$

$$b^2 = 16$$

$$b = \sqrt{16}$$

$$\therefore\ b = 4$$

Conversion of any radix number into Decimal numbers

Example: Determine the value of base x if $(193)_x = (623)_8$.

Solution: $(193)_x = (623)_8$

$x^2\ x^1\ x^0 \qquad 8^2\ 8^1\ 8^0$

$(\ 1\ \ 9\ \ 3\ )_x = (\ 6\ \ 2\ \ 3\ )_8$

$1x\,x^2 + 9x\,x^1 + 3x\,x^0 = 6x8^2 + 2x8^1 + 3x8^0$

$1x\,x^2 + 9x\,x + 3x\,1 = 6x64 + 2x8 + 3x1$

$x^2 + 9x + 3 = 6x64 + 2x8 + 3x1$

$x^2 + 9x + 3 = 384 + 16 + 3$

$x^2 + 9x + 3 = 403$

$x^2 + 9x = 403 - 3$

$x^2 + 9x = 400$

$x^2 + 9x - 400 = 0$

$x^2 + 25x - 16x - 400 = 0$

$x(x+25) - 16(x+25) = 0$

$(x-16)(x+25) = 0$

$(x-16) = 0\ ;\ (x+25) = 0$

$x = 16 \quad ; \quad x = -25$

$\therefore\ x = 16$

Conversion of any radix number into Decimal numbers

Example:1 Convert $(475.25)_8$ to its decimal number .

Example:2 Given that $(64)_{10} = (100)_b$ find the base "**b**".

Example:3 Determine the value of base 'a' for the following .

$(225)_a = (341)_8$

$(211)_a = (152)_8$

Example:4

Conversion of Hexadecimal number into Octal number

1. Convert the following to decimal and then binary

i) $(2311)_{16}$ ii) $(5446)_3$ iii) $(7444)_6$ iv) $(158)_5$ v) $(7667)_8$

2. Convert the following numbers

i) $(6753)_8 = (?)_{10}$ ii) $(95.75)_{10} = (?)_2$ iii) $(111101.0101)_2 = (?)_8 \,\&\, (?)_4$

3. Convert $(225.225)_{10}$ to binary , Octal and hexadecimal

4. Convert the following numbers

i) $(1776)_{10} = (?)_6$ ii) $(3.1415)_{10} = (?)_2$ iii) $(1023222)_4 = (?)_{16}$

iv) $(220012112021.102)_3 = (?)_9$

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET
(AUTONOMOUS)

Topic

BINARY ARITHMETIC

OVERVIEW

- Binary Arithmetic
- Complements
- Subtraction with (r-1) complement
- Subtraction with r complement

Binary addition

Binary addition:- Digital computers perform various arithmetic operations the most basic operation is addition of two binary bits.

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = 10$ with carry 1

Example: 1 Add binary number $(1101)_2$ and $(1101)_2$

$$\begin{array}{r} {\scriptstyle 1\quad\ 1} \\ 1\ 1\ 0\ 1 \\ 1\ 1\ 0\ 1 \\ \hline 1\ 1\ 0\ 1\ 0 \end{array}$$

$(1101.101)_2 + (111.011)_2$

$$\begin{array}{r} {\scriptstyle 1\ 1\ 1\ 1\ 1\ 1} \\ 1\ 1\ 0\ 1.1\ 0\ 1 \\ 1\ 1\ 1.0\ 1\ 1 \\ \hline 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0 \end{array}$$

Carry 1

Binary Subtraction

Binary Subtraction:- The following are the rules for binary subtractions.

$0 - 0 = 0$

$0 - 1 = 1$ with borrow of 1

$1 - 0 = 1$

$1 - 1 = 0$

Example: 1 subtract $(111.111)_2$ from $(1010.01)_2$

$$\begin{array}{r} 0\ 1\ {}^{1}0\ 1\ 1\ \cancel{0}\ {}^{1}0 \\ \cancel{1\ 0\ 1\ 0.0\ 1\ 0} \\ -\ 0\ 1\ 1\ 1.1\ 1\ 1 \\ \hline 0\ 0\ 1\ 0.0\ 1\ 1 \end{array}$$

$\therefore (1010.01)_2 - (111.111)_2 = (0010.011)_2$

Example: 2 $(1011.101)_2 - (101.01)_2$

$$\begin{array}{r} {}^{1}0\qquad {}^{1}0 \\ \cancel{1\ 0}\ 1\ 1.\cancel{1\ 0}\ 1 \\ 0\ 1\ 0\ 1.0\ 1\ 0 \\ \hline 0\ 1\ 1\ 0.0\ 1\ 1 \end{array}$$

$\therefore (1011.101)_2 - (101.01)_2 = (0110.011)_2$

Binary Multiplication

Binary Multiplication:- The rules for binary multiplication are

$0 \times 0 = 0$

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

Example: 1 Multiply $(1101)_2$ by $(110)_2$

$$\begin{array}{r} 1\ 1\ 0\ 1 \times 1\ 1\ 0 \\ \hline 0\ 0\ 0\ 0 \\ 1\ 1\ 0\ 1 \\ 1\ 1\ 0\ 1 \\ \hline 1\ 0\ 0\ 1\ 1\ 1\ 0 \end{array}$$

$\therefore (1101)_2 \times (110)_2 = (1001110)_2$

Example: 2 Multiply $(1011.101)_2$ by $(101.01)_2$

$$\begin{array}{r} 1\ 0\ 1\ 1.1\ 0\ 1 \times 1\ 0\ 1.0\ 1 \\ \hline 1 \\ 1\ 1\ 1\ 1\ 1 \\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1 \\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1 \\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\ 1\ 0\ 1\ 1\ 1\ 0\ 1 \\ \hline 1\ 1\ 1\ 1\ 0\ 1.0\ 0\ 0\ 0\ 1 \end{array}$$

$\therefore (1011.101)_2 \times (101.01)_2 = (111101.00001)_2$

Binary Division

Binary Division :

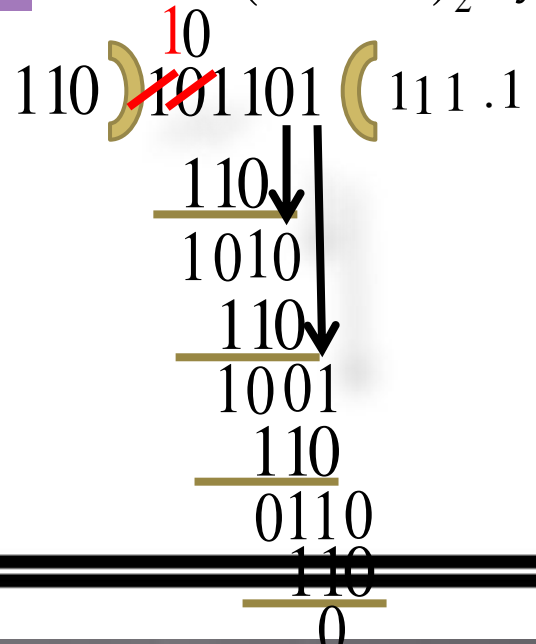
$0 \div 0 = 0$ Meaningless

$0 \div 1 = 0$

$1 \div 0 = 1$ Meaningless

$1 \div 1 = 1$

Example: 1 Divide $(101101)_2$ by $(110)_2$



Complements

Complements are used in digital computers for simplifying the subtraction operation and for logical manipulation.

They are 2 types of complement for each base r system

i	r's complement (or)Radix complement	ii	(r-1) complement (or)Diminished radix complement

Binary	= 2'S Complement	(2-1)=1'S Complement
Decimal	= 10'S Complement	(10-1)=9'S Complement
Octal	= 8'S Complement	(8-1)=7'S Complement
Hexa Decimal	=16'S Complement	(16-1)=15'S Complement

1's complement representation

i **1's complement representation:-**

The **1's complement** of a binary number is the number that results when we change **1's to 0's** and **0's to 1's.**

Example: Find the **1's complement** of 11010100.

Solution: $(11010100)_2$

$$(1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0)_2$$
$$0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1$$

1's complement of a binary number $(11010100)_2$ is $(001101011)_2$

Example: Find the **1's complement** of 110111101

Solution: $(110111101)_2$

$$(1\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1)_2$$
$$0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0$$

1's complement of a binary number $(110111101)_2$ is $(001000010)_2$

2's complement representation

ii **2's complement representation:**

The **2's complement** of the binary number is when we **add 1** to the **one's complement** of the binary number for **least significant bit**.

2's complement = **1's complement** + 1 (**at LSB**)

Example: Find the **2's complement** of $(11000100)_2$

Solution: **Given that** $(11000100)_2$

$$(1\ 1\ 0\ 0\ 0\ 1\ 0\ 0)_2$$

1's complement of $(11000100)_2$ is $(0\ 0\ 1\ 1\ 1\ 0\ 1\ 1)_2$

$$1\ 1$$

1's complement is $0\ 0\ 1\ 1\ 1\ 0\ 1\ 1$

$$+\qquad\qquad\qquad\qquad 1$$

2's complement is $0\ 0\ 1\ 1\ 1\ 1\ 0\ 0$

$\therefore$ **2's complement** of $(11000100)_2$ is $(111100)_2$

Note: **The 1's & 2's complement form is used to represent negative numbers.**

9's complement representation

iii **9's complement representation:-**

9's complement of a **decimal number** is obtained by **subtracting** each digit from **9.**

Example: Find the **9's complement** of decimal number 546700.

Solution: $(546700)_{10}$

$$\begin{array}{r} 9\ 9\ 9\ 9\ 9\ 9 \\ 5\ 4\ 6\ 7\ 0\ 0 \\ \hline 4\ 5\ 3\ 2\ 9\ 9 \end{array}$$

$\therefore$ **9's complement** of $(546700)_{10}$ is $(453299)_{10}$

Example: Find the **9's complement** of decimal number 25.639.

Solution: $(25.639)_{10}$

$$\begin{array}{r} 9\ 9\ 9\ 9\ 9 \\ 2\ 5\ .6\ 3\ 9 \\ \hline 7\ 4\ .3\ 6\ 0 \end{array}$$

$\therefore$ **9's complement** of $(25.639)_{10}$ is $(74.360)_{10}$

10's complement representation

iv **10's complement representation:**

The **10's complement** of the **decimal number** is when we **add 1** to the LSB of **9's complement** of the given decimal number.

10's complement = **9's complement** + 1 (**at LSB**)

Example: Find the **10's complement** of $(012398)_{10}$

Solution: **Given that** $(012398)_{10}$

$$\begin{array}{r} 9\ 9\ 9\ 9\ 9\ 9 \\ -\ 0\ 1\ 2\ 3\ 9\ 8 \\ \hline (9\ 8\ 7\ 6\ 0\ 1)_{10} \end{array}$$

9's complement of $(012398)_{10}$ is $(9\ 8\ 7\ 6\ 0\ 1)_{10}$

9's complement is $9\ 8\ 7\ 6\ 0\ 1$

$$\begin{array}{r} +\qquad\qquad 1 \\ \hline 9\ 8\ 7\ 6\ 0\ 2 \end{array}$$

10's complement is $9\ 8\ 7\ 6\ 0\ 2$

$\therefore$ **10s complement** of $(012398)_{10}$ is $(987602)_{10}$

Workout Problems

Example:1 Find 2's complement of $(01.0011)_2$

Example:2 Find 9's compliment of $(0.3267)_{10}$

Example:3 Find the 2's complement of $(0.0110)_2$

Example:4 Find 1's complement of $(0.0110)_2$

Subtraction with (r-1) complement

Procedure for Subtraction with (r-1) complement:

1. The subtraction of 2 positive numbers M and N is (M-N), both of base **r** system is as follows.

Minuend ⟶ M-N ⟵ **Subtrahend**

2. **Step1:** Add the minuend **M** to the **(r-1)'s** complement of the subtrahend **N**.

3. Inspect the result obtain in **step1** for an end carry

   a) If an **end carry** occurs **add 1 to LSB**. (**End around carry**).

   b) If an **end around carry** does not occurs take the **(r-1) complement** of the number obtain in **step1** and place a **negative sign** in front of it.

9's complement representation

Example: Using **9's complement** subtract 72532 - 03250

Solution: $M = (72532)_{10}$ $N = (03250)_{10}$

$$\begin{array}{rr} M = & 7\ 2\ 5\ 3\ 2 \\ N = & 0\ 3\ 2\ 5\ 0 \\ \hline (M\text{-}N) & 6\ 9\ 2\ 8\ 2 \end{array}$$

9's complement of **N** is

$$\begin{array}{r} 9\ 9\ 9\ 9\ 9 \\ -\ 0\ 3\ 2\ 5\ 0 \\ \hline 9\ 6\ 7\ 4\ 9 \end{array}$$

$\therefore$ **9's complement** of $(03250)_{10}$ is $(96749)_{10}$

Step 1: Add **M+ 9's** complement of **N**

$$\begin{array}{rr} & \overset{1}{7}\ \overset{1}{2}\ 5\ 3\ 2 \\ M = & \\ +\ N = & 1\ 9\ 6\ 7\ 4\ 9 \\ \hline \text{Carry} \rightarrow & (1)\ 6\ 9\ 2\ 8\ 1 \end{array}$$

Step 2: If **carry is generated** add carry to the LSB of result

End around Carry

$$\begin{array}{r} 6\ 9\ 2\ 8\ 1 \\ +\qquad (1) \\ \hline 6\ 9\ 2\ 8\ 2 \end{array}$$

$\therefore (72532)_{10} - (03250)_{10} = (69282)_{10}$

9's complement representation

Example: b) Using **9's complement** subtract 03250 - 72532

Solution: $M = (03250)_{10}$ $N = (72532)_{10}$

$$\begin{array}{rr} M = & 0\ 3\ 2\ 5\ 0 \\ N = & 7\ 2\ 5\ 3\ 2 \\ \hline (M-N) & 6\ 9\ 2\ 8\ 2 \end{array}$$

9's complement of **N** is

$$\begin{array}{r} 9\ 9\ 9\ 9\ 9 \\ -\ 7\ 2\ 5\ 3\ 2 \\ \hline 2\ 7\ 4\ 6\ 7 \end{array}$$

$\therefore$ **9's complement** of $(72532)_{10}$ is $(27467)_{10}$

Step 1: Add **M+ 9's** complement of **N**

$$\begin{array}{rr} & 1\ \ 1 \\ M = & 0\ 3\ 2\ 5\ 0 \\ +\ N = & 2\ 7\ 4\ 6\ 7 \\ \hline & 3\ 0\ 7\ 1\ 7 \end{array}$$

Step 2: If carry is not generated (r-1) = 10-1=9's complement of obtained result and place –ve sign to number .

$$\begin{array}{r} 9\ 9\ 9\ 9\ 9 \\ -3\ 0\ 7\ 1\ 7 \\ \hline -\ 6\ 9\ 2\ 8\ 2 \end{array}$$

$\therefore (72532)_{10} - (03250)_{10} = -(69282)_{10}$

1's complement representation

Example: a) Using **1's complement** subtract $(1010100)_2 - (1000100)_2$

Solution: $M = (1010100)_2$ $N = (1000100)_2$

$$\begin{array}{rr} M = & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ N = & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ \hline (M\text{-}N) & 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

$$1\ 0\ 0\ 0\ 1\ 0\ 0$$

1's complement of **N** is $0\ 1\ 1\ 1\ 0\ 1\ 1$

$\therefore$ **1's complement** of $(1000100)_2$ is $(0111011)_2$

Step 1: Add **M+ 1's** complement of **N**

$$\begin{array}{rr} & 1\ 1 \\ M = & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ +\ N = & 1\ 0\ 1\ 1\ 1\ 0\ 1\ 1 \\ \hline \text{Carry} \rightarrow & (1)\ 0\ 0\ 0\ 1\ 1\ 1\ 1 \end{array}$$

Step 2: If **carry is generated** add carry to the LSB of result

$$\begin{array}{rr} & 1\ 1\ 1\ 1 \\ & 0\ 0\ 0\ 1\ 1\ 1\ 1 \\ + & \qquad\qquad\ (1) \\ \hline & 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

End around Carry

$\therefore (1010100)_2 - (1000100)_2 = (0010000)_2$

1's complement representation

Example: b) Using **1's complement** subtract $(1000100)_2 - (1010100)_2$

Solution: $M = (1000100)_2$ $N = (1010100)_2$

$$\begin{array}{rr} M = & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ N = & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ \hline (M\text{-}N) & 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

$1\ 0\ 1\ 0\ 1\ 0\ 0$

1's complement of **N** is $0\ 1\ 0\ 1\ 0\ 1\ 1$

$\therefore$ **1's complement** of $(1010100)_2$ is $(0101011)_2$

Step 1: Add **M+ 1's** complement of **N**

$$\begin{array}{rr} M = & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ +\ N = & 0\ 1\ 0\ 1\ 0\ 1\ 1 \\ \hline & 1\ 1\ 0\ 1\ 1\ 1\ 1 \end{array}$$

Step 2: If carry is not generated (r-1) =2-1=1's complement of obtained result and place **–ve sign** to number .

result is $1\ 1\ 0\ 1\ 1\ 1\ 1$

1's complement result is $-\ 0\ 0\ 1\ 0\ 0\ 0\ 0$

$\therefore (1000100)_2 - (1010100)_2 = -(0010000)_2$

Workout Problems

Example:1 Perform $(28)_{10} - (15)_{10}$ using 6 bit 1's complement representation.

Example:2 Perform $(15)_{10} - (28)_{10}$ using 6 bit 1's complement representation.

Subtraction with r's complement

Procedure for Subtraction with r's complement:

1. The subtraction of 2 positive numbers M and N is (M-N), both of base **r** system is as follows.

2. **Step1:** Add the minuend **M** to the **r's** complement of the subtrahend **N**.

3. Inspect the result obtain in **step1** for an end carry

   a) If an **end carry** occurs **discard** it.

   b) If an **end around carry** does not occurs take the **r's complement** of the number obtain in **step1** and place a **negative sign** in front of it.

2's complement representation

Example: a) Using **2's complement** subtract $(1010100)_2 - (1000100)_2$

Solution: $M = (1010100)_2$ $N = (1000100)_2$

$$\begin{array}{rr} M = & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ N = & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ \hline (M\text{-}N) & 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

$$\begin{array}{rr} & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ & \quad\ \ 1\ 1 \\ \text{1's complement of N is} & 0\ 1\ 1\ 1\ 0\ 1\ 1 \\ + & \qquad\qquad 1 \\ \hline \text{2's complement of N is} & 0\ 1\ 1\ 1\ 1\ 0\ 0 \end{array}$$

$\therefore$ **2's complement** of $(1000100)_2$ is $(0111100)_2$

Step 1: Add **M+ 2's** complement of **N**

$$\begin{array}{rr} & \ \ 1\ 1\ 1\ 1 \\ M = & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ + N = & 1\ 0\ 1\ 1\ 1\ 1\ 0\ 0 \\ \hline \text{End Carry} \rightarrow & (1)\ 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

Step 2: If **carry is generated ,** then **discard** the carry.

Discard the Carry → (1) 0 0 1 0 0 0 0

$(0\ 0\ 1\ 0\ 0\ 0\ 0)_2$

$\therefore (1010100)_2 - (1000100)_2 = (0010000)_2$

2's complement representation

Example: b) Using **1's complement** subtract $(1000100)_2 - (1010100)_2$

Solution: $M = (1000100)_2$ $N = (1010100)_2$

$$\begin{array}{rr} M = & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ N = & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ \hline (M\text{-}N) & 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

$$\begin{array}{rr} & 1\ 0\ 1\ 0\ 1\ 0\ 0 \\ & \quad\quad\quad 1\ 1 \\ \text{1's complement of N is} & 0\ 1\ 0\ 1\ 0\ 1\ 1 \\ + & \quad\quad\quad\quad\quad 1 \\ \hline \text{2's complement of N is} & 0\ 1\ 0\ 1\ 1\ 0\ 0 \end{array}$$

$\therefore$ **2's complement** of $(1010100)_2$ is $(0101100)_2$

Step 1: Add **M+ 2's** complement of **N**

$$\begin{array}{rr} & \quad\quad 1\ 1 \\ M = & 1\ 0\ 0\ 0\ 1\ 0\ 0 \\ + N = & 0\ 1\ 0\ 1\ 1\ 0\ 0 \\ \hline & 1\ 1\ 1\ 0\ 0\ 0\ 0 \end{array}$$

Step 2: If carry is not generated , r's complement of obtained result and place **–ve sign** to number .

$$\begin{array}{rr} \text{result is} & 1\ 1\ 1\ 0\ 0\ 0\ 0 \\ & \quad\quad 1\ 1\ 1\ 1 \\ \text{1's complement result is} & 0\ 0\ 0\ 1\ 1\ 1\ 1 \\ + & \quad\quad\quad\quad 1 \\ \hline \text{2's complement result is} & -\ 0\ 0\ 1\ 0\ 0\ 0\ 0 \end{array}$$

$\therefore (1000100)_2 - (1010100)_2 = -(0010000)_2$

10's complement representation

Example: Using **10's complement** subtract 72532 - 03250

Solution: $M = (72532)_{10}$ $N = (03250)_{10}$

$$\begin{array}{rr} M = & 7\ 2\ 5\ 3\ 2 \\ N = & 0\ 3\ 2\ 5\ 0 \\ \hline (M\text{-}N) & 6\ 9\ 2\ 8\ 2 \end{array}$$

9's complement is $9\ 9\ 9\ 9\ 9$

$$-\ 0\ 3\ 2\ 5\ 0$$

$$\underline{\qquad\qquad 1}$$

9's complement of N is $9\ 6\ 7\ 4\ 9$

$$+\qquad\qquad 1$$

$$\underline{\qquad\qquad\qquad}$$

10's complement of N is $9\ 6\ 7\ 5\ 0$

$\therefore$ **10's complement** of $(03250)_{10}$ is $(96750)_{10}$

Step 1: Add **M+ 10's** complement of **N**

$$\begin{array}{rr} & 1 \\ M = & 7\ 2\ 5\ 3\ 2 \\ +\ N = & 9\ 6\ 7\ 5\ 0 \\ \hline \text{Carry} \rightarrow & 1\ 6\ 9\ 2\ 8\ 2 \end{array}$$

Step 2: If **carry is generated ,** then **discard** the carry

Discard the Carry → $1\ 6\ 9\ 2\ 8\ 2$

$(6\ 9\ 2\ 8\ 2)_{10}$

$\therefore (72532)_{10} - (03250)_{10} = (69282)_{10}$

10's complement representation

Example: Using **10's complement** subtract 03250 - 72532

Solution: $M = (03250)_{10}$ $N = (72532)_{10}$

$$\begin{array}{rr} M = & 0\ 3\ 2\ 5\ 0 \\ N = & 7\ 2\ 5\ 3\ 2 \\ \hline (M\text{-}N) & 6\ 9\ 2\ 8\ 2 \end{array}$$

9's complement is $9\ 9\ 9\ 9\ 9 - 7\ 2\ 5\ 3\ 2$

9's complement of N is $2\ 7\ 4\ 6\ 7 + 1$

10's complement of N is $2\ 7\ 4\ 6\ 8$

$\therefore$ **10's complement** of $(72532)_{10}$ is $(27468)_{10}$

Step 1: Add **M+ 10's** complement of **N**

$$\begin{array}{rr} & 1\ \ 1 \\ M = & 0\ 3\ 2\ 5\ 0 \\ +N = & 2\ 7\ 4\ 6\ 8 \\ \hline & 3\ 0\ 7\ 1\ 8 \end{array}$$

Step 2: If carry is not generated , r's complement of obtained result and place **–ve sign** to number

9's complement is $9\ 9\ 9\ 9\ 9 - 3\ 0\ 7\ 1\ 8$

9's complement of N is $6\ 9\ 2\ 8\ 1 + 1$

10's complement of N is $-6\ 9\ 2\ 8\ 2$

$\therefore (03250)_{10} - (72532)_{10} = -(69282)_{10}$

Workout Problems

Example:1 Perform subtraction operation using **9's** complement.

a) $(2928.54)_{10} - (416.73)_{10}$ b) $(416.73)_{10} - (2928.54)_{10}$

Example:2 Perform subtraction operation using **10's** complement.

a) $(2928.54)_{10} - (416.73)_{10}$ b) $(416.73)_{10} - (2928.54)_{10}$

Example:3 Perform subtraction operation using **1's & 2's** complement.

a) $(11010)_2 - (10000)_2$ b) $(10000)_2 - (11010)_2$

Example:4 Perform subtraction operation using **1's & 2's** complement.

a) $(11010)_2 - (1101)_2$ b) $(1101)_2 - (11010)_2$ c) $(1100)_2 - (1100)_2$

Comparison between 1's complement and 2's complement:

- The **1's complement** as the advantage of being easier to implement by digital components. since the only thing that **must be done** is to change **0's into 1's** and **1's into 0's**.
- The implementation of **2's complements** may obtained in **two ways**.
- By **adding 1** to the LSB of the **1's complement**.
- By leaving all **leading zeros** in the LSB position and the **first 1 unchanged** and only then change in all **1's to 0's** and **all 0's to 1's** .
- During **subtraction of two numbers** by **complements the 2's** complement is advantageous in that only **one arithmetic operation** is required. But the **1's complement** requires **two arithmetic additions** when an **end around carry occurs**.
- The **1's complement** has additional **disadvantages** of possessing **2 arithmetic zeros**. One with all 0's and one with all 1's.

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET
(AUTONOMOUS)

Topic

SIGNED BINARY NUMBERS

OVERVIEW

- Signed magnitude representation
- Signed binary numbers
- Octal Arithmetic

Signed magnitude representation

Signed binary numbers:

1. We can use **positive sign** to represent positive numbers and **negative sign** to represent negative numbers in **decimal system**.
2. However because hardware limitations in computer both **positive** and **negative** numbers are represented with only **binary digits**.
3. The **MSB (Left most bit)** of the number represents **sign** of the number.
4. The **sign bit** is **0** for **positive numbers** and **sign bit** is **1** for **negative numbers**.

The signed binary numbers are represented by using **three techniques**.

a) Signed magnitude representation.

b) 1's complement representation.

c) 2's complement representation.

Signed binary numbers

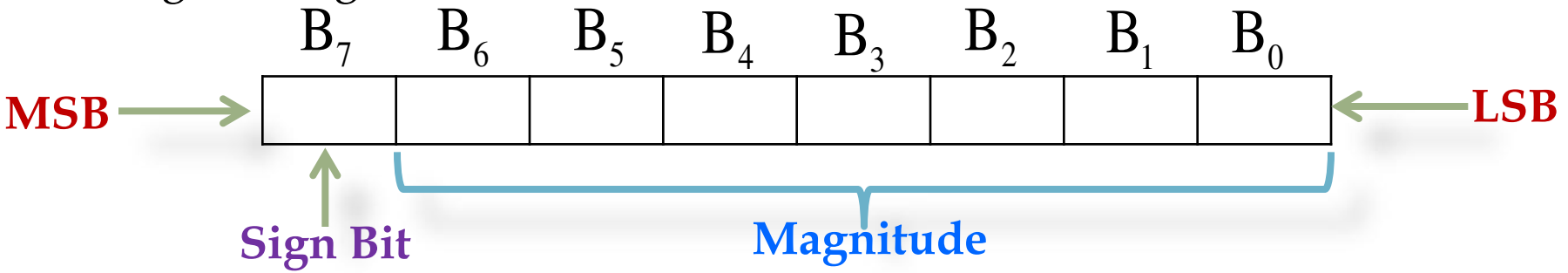
Signed magnitude representation:

1. The signed magnitude representation of binary numbers, the **MSB is reserved** for the **sign** of the numbers.

   Normally **MSB =0** **stands for positive numbers**

   MSB =1 **stands for negative numbers**

2. The disadvantage of the system is the size of the number that can be represented with a given length of register is reduced.



The above figure shows 8 **bit register**.

MSB=0 →+ve **MSB=1 → -ve**

Signed magnitude representation

Example: 1. +6 2. -14 3. +24 4. -64 represent in 8 bit sign magnitude form

Solution:

+6 = 0 0 0 0 0 1 1 0

–14 = 1 0 0 0 1 1 1 0

+24 = 0 0 0 1 1 0 0 0

–64 = 1 1 0 0 0 0 0 0

1's complement representation and 2's complement representation

Example: 1. **+9** 2. **-9** represent in **8** bit sign magnitude form

Solution:

+9 = 0 0 0 0 1 0 0 1

–9 = 1 0 0 0 1 0 0 1

Example: 1. **+9** 2. **-9** represent in **8** bit 1's complement form

Solution:

+9 = 0 0 0 0 1 0 0 1

–9 = 1 1 1 1 0 1 1 0

Example: 1. **+9** 2. **-9** represent in **8** bit 2's complement form

Solution:

+9 = 0 0 0 0 1 0 0 1

–9 = 1 1 1 1 0 1 1 1

1's complement representation and 2's complement representation

Note: In case of positive numbers signed **1's** and **2's** complement of the number is **same as it is** of **true form** of the number.

Note: The **maximum positive** and **minimum negative** numbers for a **n bit** binary numbers which can be represented in **sign magnitude form** $+ (2^{n-1}-1)$ to $- (2^{n-1}-1)$.

Example: if **n=3** the maximum and minimum number represented is $+ (2^{3-1}-1)$ to $- (2^{3-1}-1)$

$$+ (2^{3-1}-1) \text{ to } - (2^{3-1}-1)$$

$$+ (2^{2}-1) \text{ to } - (2^{2}-1)$$

$$+ (4-1) \text{ to } - (4-1)$$

$$+ 3 \text{ to } - 3$$

Subtraction of signed numbers:

Example: 2's complement of 7 bit register for following numbers.

Solution:

$+6 = 0\ 0\ 0\ 0\ 1\ 1\ 0$

$+9 = 0\ 0\ 0\ 1\ 0\ 0\ 1$

$+15 = 0\ 0\ 0\ 1\ 1\ 1\ 1$

Example: 2's complement of 7 bit register for following numbers.

Solution:

$-6 = 1\ 0\ 0\ 0\ 1\ 1\ 0$

$+9 = 0\ 0\ 0\ 1\ 0\ 0\ 1$

$\quad\quad 1\ 1\ 1$

$-6 = 1\ 1\ 1\ 1\ 0\ 1\ 0$ ← 2's complement

$+9 = 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1$

$+3 = 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1$

Discard the Carry

$-6 = 1\ 0\ 0\ 0\ 1\ 1\ 0$

$\quad\quad\quad\quad\quad\quad\quad\ 1$

1's complement $1\ 1\ 1\ 1\ 0\ 0\ 1$

$\quad\quad\quad\quad\quad\quad\quad\ +\ 1$

2's complement $1\ 1\ 1\ 1\ 0\ 1\ 0$

Comparison between Signed Magnitude , 1's and 2's Complement

To illustrate the effect of these 3 representations, we consider 4-bit binary representation and draw the below table. Carefully observe the differences in three methods.

Decimal	*Signed Magnitude*	*1's complement*	*2's complement*
+0	0 0 0 0	0 0 0 0	0 0 0 0
+1	0 0 0 1	0 0 0 1	0 0 0 1
+2	0 0 1 0	0 0 1 0	0 0 1 0
+3	0 0 1 1	0 0 1 1	0 0 1 1
+4	0 1 0 0	0 1 0 0	0 1 0 0
+5	0 1 0 1	0 1 0 1	0 1 0 1
+6	0 1 1 0	0 1 1 0	0 1 1 0
+7	0 1 1 1	0 1 1 1	0 1 1 1
–8	—	—	1 0 0 0
–7	1 1 1 1	1 0 0 0	1 0 0 1
–6	1 1 1 0	1 0 0 1	1 0 1 0
–5	1 1 0 1	1 0 1 0	1 0 1 1
–4	1 1 0 0	1 0 1 1	1 1 0 0
–3	1 0 1 1	1 1 0 0	1 1 0 1
–2	1 0 1 0	1 1 0 1	1 1 1 0
–1	1 0 0 1	1 1 1 0	1 1 1 1
–0	1 0 0 0	1 1 1 1	—

Different signed representation

Subtraction of signed numbers:

Example:1 Add **-17** to **+30** in **1's** and **2's** complement form using 8 bit.

Example:2 Add **-20** to **+26** using **5 bit** 1's and 2's complement representation.

Example:3 Subtract **14** from **46** using **8 bit** 2's complement Arithmetic

Example:4 Add **-75** to **+26** using **8 bit** 2's complement arithmetic.

Example:5 Add **-45.75** to **+87.5** using **12 bit** 2's complement arithmetic.

Representation of signed numbers using 2's and 1's complement method

Example:1 Add **-17** to **+30** in **1's** and **2's** complement form using 8 bit.

Octal arithmetic

Octal addition:-

Procedure:

1. The **sum** of **2 octal digits** is same as their **decimal sum** provided that the sum is **less than 8.**
2. If the **octal sum is 8** or **greater than 8** then **subtract 8** to obtained **octal digit**. A **carry** is **produced** when the **octal sum** is corrected in this way.

Example: Perform the octal addition for the following

Solution: $(4)_8 + (6)_8 = 10 - 8 = (2)_8$ **With carry 1**

$(2)_8 + (6)_8 = 8 - 8 = (0)_8$ **With carry 1**

Octal addition

Example: Perform the octal addition $(167)_8 + (325)_8$

Solution: $(167)_8 + (325)_8 = (\ ?\ )_8$

$$\begin{array}{cccc} & 1 & 1 & \\ & 1 & 6 & 7 \\ & 3 & 2 & 5 \\ \hline 5 & (9-8) & (12-8) & \\ \hline 5 & 1 & 4 & \\ \hline \end{array}$$

$\therefore (167)_8 + (325)_8 = (514)_8$

Note: If the octal sum of octal digit is **16** or **>16** then subtract **16** from **each digit** and set **carry=2**.

Octal Subtraction

Example: Find the 8's complement of $(346)_8$

Solution: $(346)_8$

8's complement = 7's complement + 1 (at LSB)

7's complement is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 3\ 4\ 6 \\ \hline \end{array}$$

7's complement is

$$\begin{array}{r} 4\ 3\ 1 \\ +\ \quad 1 \\ \hline \end{array}$$

8's complement is

$$\begin{array}{r} 4\ 3\ 2 \\ \hline \end{array}$$

$\therefore$ **8's complement** of $(346)_8$ is $(432)_8$

Octal Subtraction

Example: Subtract $(516)_8 - (413)_8$ using 8's complement method.

Solution: $(516)_8 - (413)_8 = (\ ?\ )_8$

$M = (516)_8 \quad N = (413)_8$

7's complement is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 4\ 1\ 3 \\ \hline 3\ 6\ 4 \end{array}$$

7's complement of N is $3\ 6\ 4$

$$\begin{array}{r} +\quad\ 1 \\ \hline 3\ 6\ 5 \end{array}$$

8's complement of N is $3\ 6\ 5$

$\therefore$ **8's complement** of $(413)_8$ is $(365)_8$

Step 1: Add **M+ 8's complement of N**

$$\begin{array}{rr} & 1\ \ 1 \\ M = & 5\ \ 1\ \ 6 \\ +\ N = & 1\ \ 3\ \ 6\ \ 5 \\ \hline \text{Carry} \rightarrow & 1\ \ (9-8)(8-8)(11-8) \\ & 1\ \ 1\ \ 0\ \ 3 \end{array}$$

Step 2: If **carry is generated** add carry to the LSB of result

Carry $\rightarrow$ $1\ 1\ 0\ 3$

$(103)_8$

$\therefore (516)_8 - (413)_8 = (103)_{10}$

Octal Subtraction

Example: Subtract $(413)_8 - (516)_8$ using 8's complement method.

Solution: $(413)_8 - (516)_8 = (\ ?\ )_8$

$M = (413)_8 \quad N = (516)_8$

7's complement is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 5\ 1\ 6 \\ \hline 2\ 6\ 1 \end{array}$$

7's complement of N is

$$\begin{array}{r} 2\ 6\ 1 \\ +\quad\ 1 \\ \hline 2\ 6\ 2 \end{array}$$

8's complement of N is

$\therefore$ **8's complement** of $(516)_8$ is $(262)_8$

Step 1: Add **M+ 8's** complement of **N**

$$\begin{array}{r} M = \quad 4\quad 1\quad 3 \\ +\ N = \quad 2\quad 6\quad 2 \\ \hline 6\quad 7\quad 5 \end{array}$$

Step 2: If carry is not generated, then take 8's complement of obtained result and place –ve sign to number .

Result is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 6\ 7\ 5 \\ \hline 1\ 0\ 2 \end{array}$$

7's complement of result is

$$\begin{array}{r} 1\ 0\ 2 \\ +\quad\ 1 \\ \hline -\ 1\ 0\ 3 \end{array}$$

8's complement of result is

$\therefore (413)_8 - (516)_8 = -(103)_8$

Workout Problems Octal arithmetic

Example:1 Perform the octal addition $(27.5)_8 + (74.4)_8$

Example:2 Subtract $(66)_8 - (45)_8$ using 8's complement method.

Example:3 Subtract $(73)_8 - (25)_8$ using 8's complement method.

Example:4 Subtract $(25)_8 - (73)_8$ using 8's complement method.

Hexadecimal arithmetic

Hexadecimal addition:-

Procedure:

1. The **sum** of **2 Hexadecimal digits** is same as their **decimal sum** provided that the sum is **less than 16**.
2. If the **Hexadecimal sum is 16** or **greater than 16** then **subtract 16** to obtained **Hexadecimal digit**. A **carry** is **produced** when the **Hexadecimal sum** is corrected in this way.

Example: Perform the Hexadecimal addition for the following

Solution: $(B)_{16} + (9)_{16} = 20 - 16 = (4)_{16}$ **With carry 1**

$(7)_{16} + (9)_{16} = 16 - 16 = (0)_{16}$ **With carry 1**

Hexadecimal addition

Example: Perform the octal addition $(6E)_{16} + (C5)_{16}$

Solution: $(6E)_{16} + (C5)_{16} = (\ ?\ )_{16}$

$$\begin{array}{cccc} & 1 & & \\ & 6 & & E \\ 1 & C & & 5 \\ \hline (19 & -16) & (19 & -16) \\ \hline 1 & 3 & & 3 \end{array}$$

$\therefore (6E)_{16} + (C5)_{16} = (133)_{16}$

Note: If the hexadecimal sum of hexadecimal digit is **32** or **>32** then subtract **32** from **each digit** and set **carry=2**.

Hexadecimal Subtraction

Example: Find the 16's complement of $(14A)_{16}$

Solution: $(14A)_{16}$

16's complement = **15's complement** + 1 (**at LSB**)

15's complement is

$$\begin{array}{r} 15\ 15\ 15 \\ -\ 1\ 4\ A \\ \hline E\ B\ 5 \end{array}$$

15's complement is

$$\begin{array}{r} E\ B\ 5 \\ +\ \ \ \ 1 \\ \hline E\ B\ 6 \end{array}$$

16's complement is

$\therefore$ **16's complement** of $(14A)_{16}$ is $(EB6)_{16}$

Hexadecimal Subtraction

Example: Subtract $(C4)_{16} - (7B)_{16}$ using 16's complement method.

Solution: $(C4)_{16} - (7B)_{16} = (\ ?\ )_{16}$

$M = (C4)_{16}$ $N = (7B)_{16}$

15's complement is

$$\begin{array}{r} 15\ \ 15 \\ -\ 7\ \ B \\ \hline 8\ \ 4 \end{array}$$

15's complement of N is

$$\begin{array}{r} 8\ \ 4 \\ +\ \ \ 1 \\ \hline 8\ \ 5 \end{array}$$

16's complement of N is

$\therefore$ **16's complement** of $(7B)_{16}$ is $(85)_{16}$

Step 1: Add **M+ 16's** complement of **N**

$$\begin{array}{r} M = \ \ C\ \ 4 \\ +\ N = 1\ \ 8\ \ 5 \\ \hline 1\ \ (20-16)\ \ 9 \\ 1\ \ 4\ \ 9 \end{array}$$

Carry → 1

Step 2: If **carry is generated,** add carry to the LSB of result

Discard the Carry → 1

$4\ \ 9$

$(4\ \ 9)_{16}$

$\therefore (C4)_{16} - (7B)_{16} = (49)_{16}$

Hexadecimal Subtraction

Example: Subtract $(7B)_{16} - (C4)_{16}$ using 16's complement method.

Solution: $(7B)_{16} - (C4)_{16} = (\ ?\ )_{16}$

$M = (7B)_{16} \quad N = (C4)_{16}$

15's complement is

$$\begin{array}{cc} 15 & 15 \\ -C & 4 \\ \hline 3 & B \end{array}$$

15's complement of N is

$$\begin{array}{cc} 3 & B \\ + & 1 \\ \hline 3 & C \end{array}$$

16's complement of N is

$\therefore$ **16's complement** of $(C4)_{16}$ is $(3C)_{16}$

Step 1: Add **M+ 16's** complement of **N**

$$\begin{array}{rcc} & 1 & \\ M = & 7 & B \\ +\,N = & 3 & C \\ \hline & B & (23-16) \\ & B & 7 \end{array}$$

Step 2: If carry is not generated, then take 16's complement of obtained result and place –ve sign to number .

Result is

$$\begin{array}{cc} 15 & 15 \\ -B & 7 \\ \hline 4 & 8 \end{array}$$

15's complement of result is

$$\begin{array}{cc} + & 1 \\ \hline -4 & 9 \end{array}$$

16's complement of result is

$\therefore (7B)_{16} - (C4)_{16} = -(49)_{16}$

Workout Problems Hexadecimal arithmetic

Example:1 Perform the Hexadecimal addition $(4A6)_{16} + (1B3)_{16}$

Example:2 Subtract $(3A)_{16} - (5D)_{16}$ using 16's complement method.

Example:3 Subtract $(5D)_{16} - (3A)_{16}$ using 16's complement method.

Example:4 Subtract $(4A6)_{16} - (1B3)_{16}$ using 16's complement method.

Example:4 Subtract $(1B3)_{16} - (4A6)_{16}$ using 16's complement method.

Representation of signed numbers using 2's and 1's complement method

- If the number is positive the magnitude is represented in its true binary form and a sign bit 0 is placed in front of MSB.
- If the number is negative the magnitude is represented in 2's or 1's complement form and sign bit 1 is placed in front of MSB.

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET
(AUTONOMOUS)

Topic

BINARY CODES

OVERVIEW

- Binary codes
- Classification of binary codes
- BCD code or (Natural BCD code)or (8421)

Binary Codes

1. The **digital data** is represented, **stored** and **transmitted** as a group of **binary bits**.

2. The **group of bits** known as **binary codes**. Represent both **numbers** and **letters** of the **alphabets** as well as many **special characters** and **control functions**.

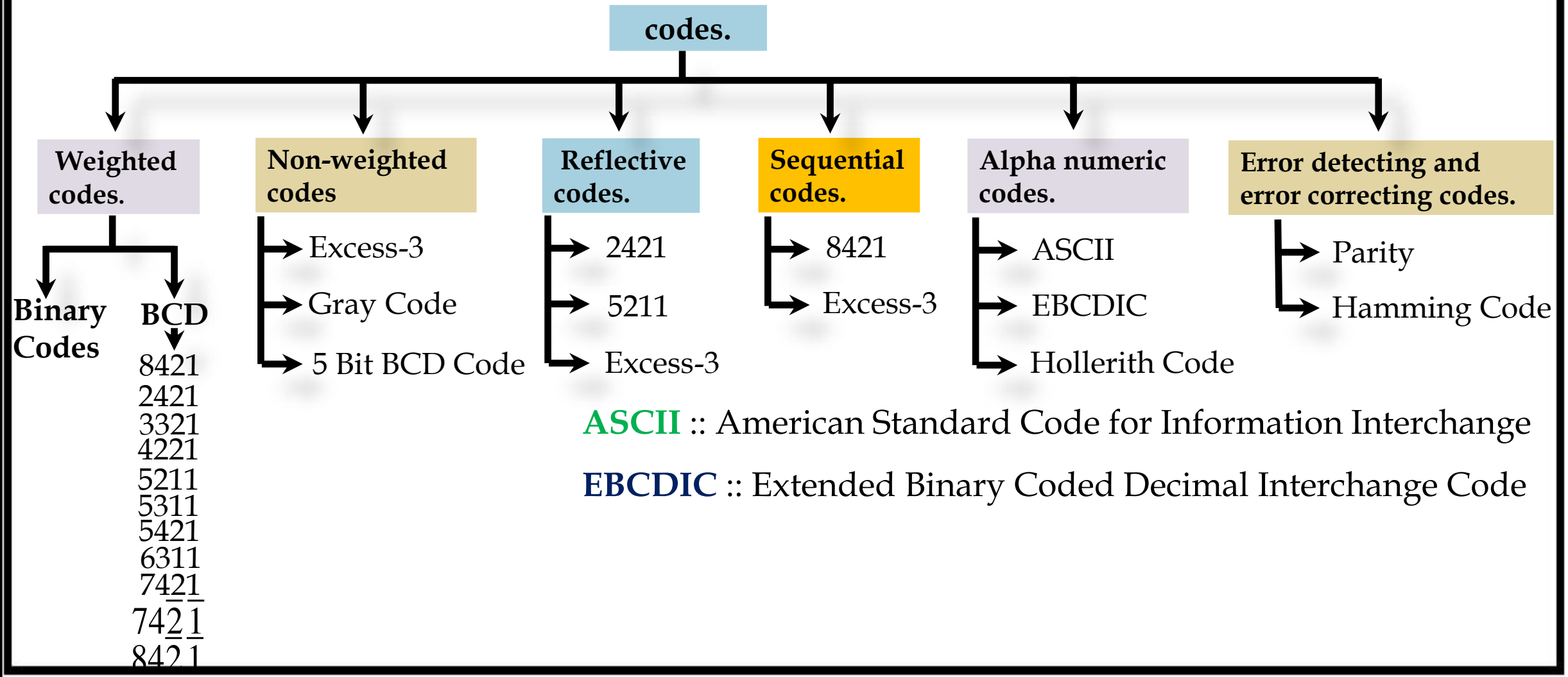
3. To represent a group of 2^n **distinct elements** in a binary code requires a minimum of **n bits**. i.e., it is possible to arrange **n bits** in 2^n **distinct ways**.

Classification of binary codes

Mainly binary codes are classified into 6 types

1. Weighted codes.
2. Non-weighted codes
3. Reflective codes.
4. Sequential codes.
5. Alpha numeric codes.
6. Error detecting and error correcting codes.

Classification of binary codes



ASCII :: American Standard Code for Information Interchange

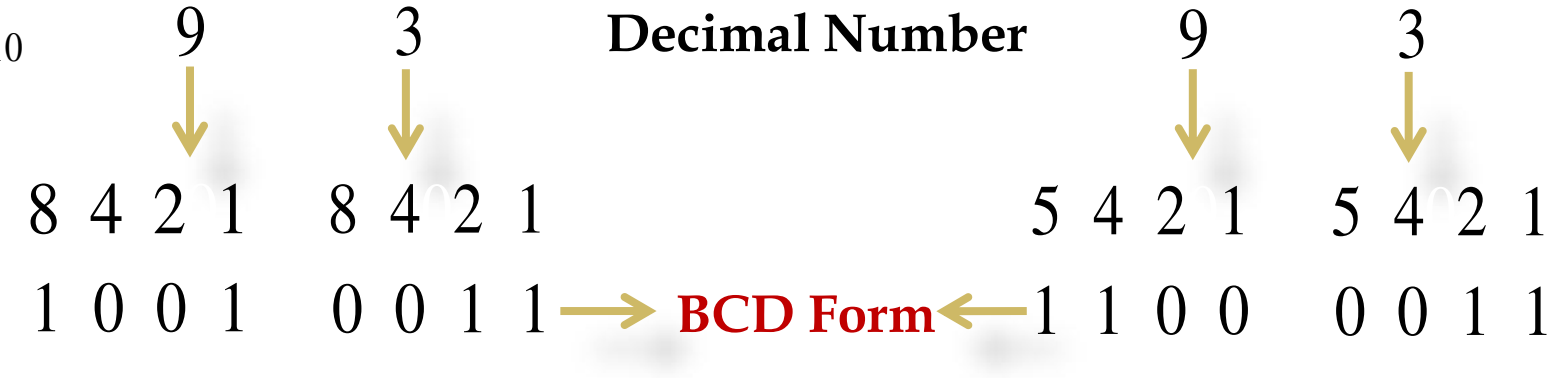
EBCDIC :: Extended Binary Coded Decimal Interchange Code

Classification of binary codes

1. Weighted codes:-

- In **weighted codes** each digit **position** of the number represents a **specific weight**.
- In **weighted code** each bit has a **weight 8, 4, 2 or 1** and **each decimal digit** is represented by a group **of 4 bits**.

Example: $(93)_{10}$



Classification of binary codes

2. Non weighted codes:-

- **Non weighted codes** are which are **not assigned** with **any weight** to each **digit** position i.e. **each digit** position within the number is **not assigned** a **fixed value**.
- **Excess-3**, **gray code** and **5-bit BCD** codes are **non weighted codes**.

3. Reflective codes:-

- A code is said to be **reflective** when the code for **9** is the **complement** for code **0**, **8** for **1**, **7** for **2**, …….
- **Excess-3**, **2421**, **5211** codes are **reflective codes**.

Example:

	5	2	1	1	
9 →	1	1	1	1	← Complement
0 →	0	0	0	0	←

	5	2	1	1	
8 →	1	1	1	0	← Complement
1 →	0	0	0	1	←

- Reflective is desirable in a code when the **1's** complement must be form.

Classification of binary codes

4. Sequential codes:-

- In this code **each succeeding** code is **1 binary** number **greater than** its **preceding code**.
- **8421** and **excess-3** are **sequential codes**.

5. Alpha numeric codes:-

- The codes which consist of both **numbers** and **alphabetic characters** are called **alpha numeric codes**.
- **ASCII**- American Standard code for Information Interchange.
- **EBCDIC**- Extended binary code decimal interchange code

Classification of binary codes

6. Error detecting and error correcting codes:-

- When the **digital** information is in **binary form**. It **transmitted** from **one system** to **another system** an **error may occur**. i.e. a **signal** corresponding to **0** may change to **1** or **vice versa**.
- Due to presence of **noise**.
- To maintain the **data integrity** between **transmitter** and **receiver**, **extra bit** or **more than 1 bit** are **added** in data.
- The data along the **extra bit/bits** forms the **code**.
- Codes which allow **only error detection** called **error detecting** codes.
- Codes which allow **error detection** and **error correction** called **error detecting and correcting codes.**

Eg:- **parity** **and** **hamming** **codes.**

BCD code or (Natural BCD code)or (8421)

BCD code or (Natural BCD code) or (8421)

- The most common **BCD code** is **8421**. BCD in which **each decimal digit** is represented by a **4 bit binary** number. This code is also called **natural binary code**.
- BCD is an abbreviation for **Binary Coded Decimal**.

Example: Represent 14 in BCD form

Solution:

$(14)_{10}$ = 1 1 1 0 (8 4 2 1)

1 4

8 4 2 1 8 4 2 1

0 0 0 1 0 1 0 0

14 in BCD form

$\therefore$ $(14)_{10}$ in BCD form is = 0001 0100

BCD code or (Natural BCD code)or (8421)

Decimal digit	8 4 2 1	2 4 2 1	8 4 -2 -1
0	0 0 0 0	0 0 0 0	0 0 0 0 (0)
1	0 0 0 1	0 0 0 1	0 1 1 1 (1)
2	0 0 1 0	0 0 1 0	0 1 1 0 (2)
3	0 0 1 1	0 0 1 1	0 1 0 1 (3)
4	0 1 0 0	0 1 0 0	0 1 0 0 (4)
5	0 1 0 1	0 1 0 1	1 0 1 1 (5)
6	0 1 1 0	0 1 1 0	1 0 1 0 (6)
7	0 1 1 1	0 1 1 1	1 0 0 1 (7)
8	1 0 0 0	1 1 1 0	1 0 0 0 (8)
9	1 0 0 1	1 1 1 1	1 1 1 1 (9)

This is valid for 0 to 9 and invalid for (10-15).

Binary addition :- (BCD)

Binary addition :- (BCD)

Procedure:

1. Add **2 BCD** numbers using ordinary **binary addition**.
2. If **4 bit sum** is **greater than equal to** or **less than 9** no correction is needed. The sum is in proper **BCD form**.
3. If the **4 bit sum** is **greater than 9** or if **a carry is generated** from the **4 bit sum** the sum is **invalid**.
4. **To correct** the **invalid sum** add (0110)2 to the **4 bit sum**.
5. If a **carry results** from this **addition** add it to the **next higher** order **BCD digit**.

Binary addition :- (BCD)

1. Sum=9 or less with carry 0

1. Sum=9 or less with carry 0

Example:1 Add +6 and +3

Solution:

```
          1 1
+6 =      0 1 1 0
+3 =      0 0 1 1
-------------------
+9 =      1 0 0 1   BCD for 9 "valid number"
-------------------
```

Carry 0

Binary addition :- (BCD)

2. Sum > 9 with carry 0

2. Sum>9 with carry 0

Example:1 Add +6 and +8

Solution:

+6 = 0 1 1 0 **BCD form for 6**

+8 = 1 0 0 0 **BCD form for 8**

+14 = 1 1 1 0 **BCD for 14 is "invalid number"**

Carry 0

Add +6 for the BCD sum because the BCD sum obtained is invalid.

```
      1 1
+14 = 1 1 1 0
+6 = 1 0 1 1 0
     ---------
     1 0 1 0 0
```

Carry → 1

0 0 0 1 | 0 1 0 0

1 | 4

14 in BCD form

$\therefore (6)_{10} + (8)_{10} = (14)_{10} = 0001\ 0100$

Binary addition :- (BCD)

3. Sum > 9 or less with carry 0

3. Sum equals 9 or less with carry 1

Example:1 Let us consider addition of +8 and +9 in BCD

Solution:

+8 = 1 0 0 0 **BCD form for 8**

+9 = 1 1 0 0 1 **BCD form for 9**

+17 = 1 0 0 0 1 **BCD for 17 is "invalid number"**

Carry

In this case , Result (**0001 0001**) valid BCD number. But it is incorrect. To get the correct BCD result . Correction factor 6 has to be added to the least significant digit as shown.

0 0 0 1 0 0 0 1 **Result**

0 1 1 0 6

0 0 0 1 0 1 1 1

1 7

17 in BCD form

$\therefore (8)_{10} + (9)_{10} = (17)_{10} =$ 0001 0111

Binary addition :- (BCD)

Example: $(24)_{10} + (18)_{10}$ Perform the addition in **8421 BCD**.

Solution:

$(24)_{10} =$ 0 0 1 0 0 1 0 0 **BCD form for 24**

$(18)_{10} =$ 0 0 0 1 1 0 0 0 **BCD form for 24**

$(42)_{10} =$ 0 0 1 1 1 1 0 0 **"invalid BCD number"**

3 12

Add 6 to 12

1 1 1 1
0 0 1 1 1 1 0 0
0 1 1 0
0 1 0 0 0 0 1 0

4 2

42 in BCD form

$\therefore (24)_{10} + (18)_{10} = (42)_{10} =$ 0100 0010

Workout Problems BCD Addition

Example:1 Perform the following addition in **8421 BCD**

a) $(48)_{10} + (58)_{10}$

b) $(175)_{10} + (326)_{10}$

c) $(589)_{10} + (199)_{10}$

d) $(679.6)_{10} + (536.8)_{10}$

BCD Subtraction

BCD subtraction

Procedure:

1. It is performed by **subtracting** the digits of **each 4 bit group** of the **subtrahend** from the corresponding **4 bit group** of **the minuend**.

2. If there is **no borrow** from the **next higher group** then **no correction** is required.

3. If there is **a borrow** from **next group** then **(0110)** is **subtracted** from the **difference term** of this **group**

BCD Subtraction

Example: $(38)_{10} - (15)_{10}$ Perform the BCD Subtraction in **8421 Code**.

Solution:

$$\begin{array}{rll} & 0\ 1\ 1\ 10 & \\ (38)_{10} = & 0\ 0\ 1\ 1\ \ 1\ 0\ 0\ 0 & \text{BCD form for 38} \\ -\ (15)_{10} = & 0\ 0\ 0\ 1\ \ 0\ 1\ 0\ 1 & \text{BCD form for 15} \\ \hline (23)_{10} = & 0\ 0\ 1\ 0\ \ 0\ 0\ 1\ 1 & \text{“valid BCD number”} \\ & \quad 2 \qquad\quad 3 & \end{array}$$

23 in BCD form

$\therefore (38)_{10} - (15)_{10} = (23)_{10} = 0010\ 0011$

BCD Subtraction

Example: $(206.7)_{10} - (147.8)_{10}$ Perform the BCD Subtraction in **8421 Code**.

Solution:

$(206.7)_{10} =$ 0 0 1 0 0 0 0 0 0 1 1 0 . 0 1 1 1 — **BCD form for 206.7**

$-(147.8)_{10} =$ 0 0 0 1 0 1 0 0 0 1 1 1 . 1 0 0 0 — **BCD form for 147.8**

$(058.9)_{10} =$ 0 0 0 0 1 0 1 1 1 1 1 0 . 1 1 1 1

0 0 0 0 1 0 1 1 1 1 1 0 . 1 1 1 1 — **"invalid BCD number"**

$-$ 0 1 1 0 0 1 1 0 0 1 1 0

0 0 0 0 0 1 0 1 1 0 0 0 . 1 0 0 1

0 5 8 9

$\therefore (206.7)_{10} - (147.8)_{10} = (058.9)_{10} =$ 0000 0101 1000 .1001

BCD subtraction using 9's and 10's complement method:

BCD subtraction 9's complement method

Procedure:

1. Find the **9's complement** of a **subtrahend**.
2. **Add 2 numbers** using **BCD addition** (**minuhend+9's complement of subtrahend**).
3. If **carry** is **generated**, **add it** to the **LSB of result**.
4. If **carry** is **not generated**, **takes 9's complement** of **result** and **result is negative** .

BCD subtraction using 9's and 10's complement method:

Example: Perform Subtract $(46)_{10} - (22)_{10}$ in BCD using 9's complement method.

Solution: $(46)_{10} - (22)_{10} = (\ ?\ )_{BCD}$

$M = (46)_{10}$ $N = (22)_{10}$

9's complement is $\begin{array}{r} 9\ 9 \\ -\ 2\ 2 \\ \hline 7\ 7 \end{array}$

9's complement of N is $7\ 7$

$\therefore$ **9's complement** of $(22)_{10}$ is $(77)_{10}$

Step 1: Add **M+ 9's** complement of **N**

$$\begin{array}{rl} & \overset{1}{0}\ 1\ 0\ 0\ \overset{1}{0}\ \overset{1}{1}\ 1\ 0 \\ M = 46 = & 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0 \quad \text{BCD form} \\ +\ N = 77 = & 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1 \quad \text{BCD form} \\ \hline & 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1 \quad \text{Invalid} \end{array}$$

Here the **result** is **Invalid**, for correction **add 6** for **1011** & **1101**

$$\begin{array}{rl} & 1\ 1\ 1\ 1\ 1 \\ \text{Invalid} & 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1 \\ \text{Adding 6} & 1\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0 \\ \hline \text{Carry} \rightarrow & (1)\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1 \end{array}$$

Step 2: If **carry is generated** add carry to the LSB of result

End around Carry

$$\begin{array}{rl} & 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1 \\ + & \qquad\qquad\qquad\ (1) \\ \hline & 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0 \end{array}$$

$\therefore (46)_{10} - (22)_{10} = (24)_{10} =$ 0010 0100

BCD subtraction using 9's and 10's complement method:

Example: Perform Subtract $(22)_{10} - (46)_{10}$ in BCD using 9's complement method.

Solution: $(22)_{10} - (46)_{10} = (\ ?\ )_{BCD}$

$M = (22)_{10}$ $N = (46)_{10}$

9's complement is $\begin{array}{r} 9\ 9 \\ -\ 4\ 6 \\ \hline 5\ 3 \end{array}$

9's complement of **N** is

$\therefore$ **10's complement** of $(46)_{10}$ is $(53)_{10}$

Step 1: Add **M+ 9's** complement of **N**

$$\begin{array}{rl} & \ 1 \\ M = 22 = & 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0 \quad \text{BCD form} \\ +\ N = 53 = & 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1 \quad \text{BCD form} \\ \hline & 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1 \quad \text{valid} \end{array}$$

Here the **result** is **valid**, then no correction **is required**

Valid BCD $0\ 1\ 1\ 1\ \ 0\ 1\ 0\ 1$

$7 \qquad 5$

Step 2: If **carry is not generated, then take 9's complement and place –Ve sign**

Result is $\begin{array}{r} 9\ 9 \\ -\ 7\ 5 \\ \hline -\ 2\ 4 \end{array}$

9's complement of **75** is

$\therefore (22)_{10} - (46)_{10} = -(24)_{10} = -0010\ 0100$

Workout Problems BCD Addition

Example:1 Perform decimal Subtract in BCD using 9's complement method.

a) $(305.5)_{10} - (168.8)_{10}$

b) $(679.6)_{10} - (885.9)_{10}$

c) $(46)_{10} - (58)_{10}$

d) $(86)_{10} - (58)_{10}$

BCD subtraction using 9's and 10's complement method:

BCD subtraction 10's complement method

Procedure:

1. Find the **10's complement** of a **subtrahend**.

2. **Add 2 numbers** using **BCD addition** (**minuhend+10's complement of subtrahend**).

3. If **carry** is **generated**, **discard the carry**.

4. If carry is **not generated** then the **result is negative** and takes **10's complement** of result and place **negative sign** in **front** of result.

BCD subtraction using 9's and 10's complement method:

Example: Perform Subtract $(46)_{10} - (22)_{10}$ in BCD using 10's complement method.

Solution: $M = (46)_{10}$ $N = (22)_{10}$

9's complement is

$$\begin{array}{r} 9\ 9 \\ -\ 2\ 2 \\ \hline 7\ 7 \end{array}$$

9's complement of N is

$$\begin{array}{r} 7\ 7 \\ +\ \ 1 \\ \hline 7\ 8 \end{array}$$

$\therefore$ **10's complement** of $(22)_{10}$ is $(78)_{10}$

Step 1: Add **M+ 10's** complement of **N**

$$\begin{array}{rl} & \ \ 1 \\ M = 46 = & 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0 \quad \text{BCD form} \\ +\ N = 78 = & 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0 \quad \text{BCD form} \\ \hline & 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0 \quad \text{Invalid} \end{array}$$

Here the **result** is **Invalid**, for correction **add 6** for **1011** & **1110**

$$\begin{array}{rl} & 1\ 1\ 1\ 1\ 1\ 1 \\ \text{Invalid} & 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0 \\ \text{Adding 6} & 1\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0 \\ \hline \text{Carry} \rightarrow & (1)\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0 \end{array}$$

Step 2: If **carry is generated, then discard the carry.**

$$\begin{array}{rl} \text{Discard Carry} \rightarrow & (1)\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0 \\ & 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0 \quad \text{Result} \end{array}$$

$\therefore (46)_{10} - (22)_{10} = (24)_{10} = 0010\ 0100$

BCD subtraction using 9's and 10's complement method:

Example: Perform Subtract $(22)_{10} - (46)_{10}$ in BCD using 10's complement method.

Solution: $M = (22)_{10}$ $N = (46)_{10}$

9's complement is

$$\begin{array}{r} 9\ 9 \\ -\ 4\ 6 \\ \hline 5\ 3 \end{array}$$

9's complement of N is

$$\begin{array}{r} 5\ 3 \\ +\ \ \ 1 \\ \hline 5\ 4 \end{array}$$

$\therefore$ 10's complement of $(46)_{10}$ is $(54)_{10}$

Step 1: Add **M+ 10's** complement of **N**

$$\begin{array}{lll} M = 22 = & 0\ 0\ 1\ 0\ \ 0\ 0\ 1\ 0 & \text{BCD form} \\ +N = 54 = & 0\ 1\ 0\ 1\ \ 0\ 1\ 0\ 0 & \text{BCD form} \\ \hline & 0\ 1\ 1\ 1\ \ 0\ 1\ 1\ 0 & \text{Valid} \end{array}$$

Here the **result** is **valid**, then no correction **is required**

Valid BCD $0\ 1\ 1\ 1\ \ 0\ 1\ 1\ 0$

$7 \qquad 6$

Step 2: If **carry is not generated, then take 10's complement and place –Ve sign**

Result is

$$\begin{array}{r} 9\ 9 \\ -\ 7\ 6 \\ \hline 2\ 3 \end{array}$$

9's complement of N is

$$\begin{array}{r} 2\ 3 \\ +\ \ \ 1 \\ \hline -\ 2\ 4 \end{array}$$

10's complement of N is

$\therefore (22)_{10} - (46)_{10} = -(24)_{10} = -0010\ 0100$

Workout Problems BCD Addition

Example:1 Perform $(24)_{10} - (56)_{10}$ Subtract in BCD using 10's complement method.

Example:2 Perform decimal Subtract in BCD using 10's complement method.

$a)\ (305.5)_{10} - (168.8)_{10}$

$b)\ (679.6)_{10} - (885.9)_{10}$

$c)\ (46)_{10} - (58)_{10}$

$d)\ (86)_{10} - (58)_{10}$

BCD subtraction using 9's and 10's complement method:

Example: Perform Subtract $(22)_{10} - (46)_{10}$ in BCD using 9's complement method.

Solution: $(22)_{10} - (46)_{10} = (\ ?\ )_{BCD}$

$M = (22)_{10}$ $N = (46)_{10}$

9's complement is

$$\begin{array}{r} 9\ 9 \\ -\ 4\ 6 \\ \hline 5\ 3 \end{array}$$

9's complement of N is

$\therefore$ **10's complement** of $(46)_{10}$ is $(53)_{10}$

Step 1: Add **M+ 9's** complement of **N**

$$\begin{array}{rl} & \ 1 \\ M = 22 = & 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0 \quad \text{BCD form} \\ +\ N = 53 = & 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1 \quad \text{BCD form} \\ \hline & 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1 \quad \text{valid} \end{array}$$

Here the **result** is **valid**, for no correction **is required**

Valid BCD $0\ 1\ 1\ 1\ \ 0\ 1\ 0\ 1$

$7 \qquad 5$

Step 2: If **carry is not generated, then take 9's complement and place –Ve sign**

$$\begin{array}{r} 9\ 9 \\ -\ 7\ 5 \\ \hline -\ 2\ 4 \end{array}$$

Result is

9's complement of **75** is

$\therefore (22)_{10} - (46)_{10} = -(24)_{10} = -0010\ 0100$

BCD Subtraction

Example: $(38)_{10} + (15)_{10}$ Perform the BCD Subtraction in **8421 Code**.

Solution:

$(38)_{10} =$ 0 0 1 0 0 1 0 0 **BCD form for 24**

$(15)_{10} =$ 0 0 0 1 1 0 0 0 **BCD form for 24**

$(42)_{10} =$ 0 0 1 1 1 1 0 0 **"invalid BCD number"**

3 12

Add 6 to 12

1 1 1 1
0 0 1 1 1 1 0 0
0 1 1 0
0 1 0 0 0 0 1 0

4 2

42 in BCD form

$\therefore (24)_{10} + (18)_{10} = (42)_{10} =$ 0100 0010

1's complement representation and 2's complement representation

Note: In case of positive numbers signed **1's** and **2's** complement of the number is **same as it is** of **true form** of the number.

Note: The **maximum positive** and **minimum negative** numbers for a **n bit** binary numbers which can be represented in **sign magnitude form** $+ (2^{n-1}-1)$ to $- (2^{n-1}-1)$.

Example: if **n=3** the maximum and minimum number represented is $+ (2^{3-1}-1)$ to $- (2^{3-1}-1)$

$$+ (2^{3-1}-1) \text{ to } - (2^{3-1}-1)$$

$$+ (2^{2}-1) \text{ to } - (2^{2}-1)$$

$$+ (4-1) \text{ to } - (4-1)$$

$$+ 3 \text{ to } - 3$$

Subtraction of signed numbers:

Example: 2's complement of 7 bit register for following numbers.

Solution:

$+6 = 0\ 0\ 0\ 0\ 1\ 1\ 0$

$+9 = 0\ 0\ 0\ 1\ 0\ 0\ 1$

$+15 = 0\ 0\ 0\ 1\ 1\ 1\ 1$

Example: 2's complement of 7 bit register for following numbers.

Solution:

$-6 = 1\ 0\ 0\ 0\ 1\ 1\ 0$

$+9 = 0\ 0\ 0\ 1\ 0\ 0\ 1$

$\quad\quad 1\ 1\ 1$

$-6 = 1\ 1\ 1\ 1\ 0\ 1\ 0$ ← 2's complement

$+9 = 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1$

$+3 = 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1$

Discard the Carry

$-6 = 1\ 0\ 0\ 0\ 1\ 1\ 0$

$\quad\quad\quad\quad\quad\quad\quad\ 1$

1's complement $1\ 1\ 1\ 1\ 0\ 0\ 1$

$\quad\quad\quad\quad\quad\quad\quad\ +\ 1$

2's complement $1\ 1\ 1\ 1\ 0\ 1\ 0$

Comparison between Signed Magnitude , 1's and 2's Complement

To illustrate the effect of these 3 representations, we consider 4-bit binary representation and draw the below table. Carefully observe the differences in three methods.

Decimal	*Signed Magnitude*	*1's complement*	*2's complement*
+0	0 0 0 0	0 0 0 0	0 0 0 0
+1	0 0 0 1	0 0 0 1	0 0 0 1
+2	0 0 1 0	0 0 1 0	0 0 1 0
+3	0 0 1 1	0 0 1 1	0 0 1 1
+4	0 1 0 0	0 1 0 0	0 1 0 0
+5	0 1 0 1	0 1 0 1	0 1 0 1
+6	0 1 1 0	0 1 1 0	0 1 1 0
+7	0 1 1 1	0 1 1 1	0 1 1 1
–8	—	—	1 0 0 0
–7	1 1 1 1	1 0 0 0	1 0 0 1
–6	1 1 1 0	1 0 0 1	1 0 1 0
–5	1 1 0 1	1 0 1 0	1 0 1 1
–4	1 1 0 0	1 0 1 1	1 1 0 0
–3	1 0 1 1	1 1 0 0	1 1 0 1
–2	1 0 1 0	1 1 0 1	1 1 1 0
–1	1 0 0 1	1 1 1 0	1 1 1 1
–0	1 0 0 0	1 1 1 1	—

Different signed representation

Subtraction of signed numbers:

Example:1 Add **-17** to **+30** in **1's** and **2's** complement form using 8 bit.

Example:2 Add **-20** to **+26** using **5 bit** 1's and 2's complement representation.

Example:3 Subtract **14** from **46** using **8 bit** 2's complement Arithmetic

Example:4 Add **-75** to **+26** using **8 bit** 2's complement arithmetic.

Example:5 Add **-45.75** to **+87.5** using **12 bit** 2's complement arithmetic.

Octal arithmetic

Octal addition:-

Procedure:

1. The **sum** of **2 octal digits** is same as their **decimal sum** provided that the sum is **less than 8.**
2. If the **octal sum is 8** or **greater than 8** then **subtract 8** to obtained **octal digit**. A **carry** is **produced** when the **octal sum** is corrected in this way.

Example: Perform the octal addition for the following

Solution: $(4)_8 + (6)_8 = 10 - 8 = (2)_8$ **With carry 1**

$(2)_8 + (6)_8 = 8 - 8 = (0)_8$ **With carry 1**

Octal addition

Example: Perform the octal addition $(167)_8 + (325)_8$

Solution: $(167)_8 + (325)_8 = (\ ?\ )_8$

$$\begin{array}{cccc} & 1 & 1 & \\ & 1 & 6 & 7 \\ & 3 & 2 & 5 \\ \hline 5 & (9-8) & (12-8) & \\ \hline 5 & 1 & 4 & \\ \hline \end{array}$$

$\therefore (167)_8 + (325)_8 = (514)_8$

Note: If the octal sum of octal digit is **16** or **>16** then subtract **16** from **each digit** and set **carry=2**.

Octal Subtraction

Example: Find the 8's complement of $(346)_8$

Solution: $(346)_8$

8's complement = 7's complement + 1 (at LSB)

7's complement is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 3\ 4\ 6 \\ \hline \end{array}$$

7's complement is

$$\begin{array}{r} 4\ 3\ 1 \\ +\ \quad 1 \\ \hline \end{array}$$

8's complement is

$$\begin{array}{r} 4\ 3\ 2 \\ \hline \end{array}$$

$\therefore$ **8's complement** of $(346)_8$ is $(432)_8$

Octal Subtraction

Example: Subtract $(516)_8 - (413)_8$ using 8's complement method.

Solution: $(516)_8 - (413)_8 = (\ ?\ )_8$

$M = (516)_8 \quad N = (413)_8$

7's complement is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 4\ 1\ 3 \\ \hline 3\ 6\ 4 \end{array}$$

7's complement of N is $3\ 6\ 4$

$$\begin{array}{r} +\quad\ 1 \\ \hline 3\ 6\ 5 \end{array}$$

8's complement of N is $3\ 6\ 5$

$\therefore$ **8's complement** of $(413)_8$ is $(365)_8$

Step 1: Add **M+ 8's complement of N**

$$\begin{array}{rr} & 1\ \ 1 \\ M = & 5\ \ 1\ \ 6 \\ +\ N = & 1\ \ 3\ \ 6\ \ 5 \\ \hline \text{Carry} \rightarrow & 1\ \ (9-8)(8-8)(11-8) \\ & 1\ \ 1\ \ 0\ \ 3 \end{array}$$

Step 2: If **carry is generated** add carry to the LSB of result

Carry $\rightarrow$ $1\ 1\ 0\ 3$

$(103)_8$

$\therefore (516)_8 - (413)_8 = (103)_{10}$

Octal Subtraction

Example: Subtract $(413)_8 - (516)_8$ using 8's complement method.

Solution: $(413)_8 - (516)_8 = (\ ?\ )_8$

$M = (413)_8 \quad N = (516)_8$

7's complement is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 5\ 1\ 6 \\ \hline 2\ 6\ 1 \end{array}$$

7's complement of N is

$$\begin{array}{r} 2\ 6\ 1 \\ +\quad\ 1 \\ \hline 2\ 6\ 2 \end{array}$$

8's complement of N is

$\therefore$ **8's complement** of $(516)_8$ is $(262)_8$

Step 1: Add **M+ 8's** complement of **N**

$$\begin{array}{r} M = \quad 4\quad 1\quad 3 \\ +\ N = \quad 2\quad 6\quad 2 \\ \hline 6\quad 7\quad 5 \end{array}$$

Step 2: If carry is not generated, then take 8's complement of obtained result and place –ve sign to number .

Result is

$$\begin{array}{r} 7\ 7\ 7 \\ -\ 6\ 7\ 5 \\ \hline 1\ 0\ 2 \end{array}$$

7's complement of result is

$$\begin{array}{r} 1\ 0\ 2 \\ +\quad\ 1 \\ \hline -\ 1\ 0\ 3 \end{array}$$

8's complement of result is

$\therefore (413)_8 - (516)_8 = -(103)_8$

Workout Problems Octal arithmetic

Example:1 Perform the octal addition $(27.5)_8 + (74.4)_8$

Example:2 Subtract $(66)_8 - (45)_8$ using 8's complement method.

Example:3 Subtract $(73)_8 - (25)_8$ using 8's complement method.

Example:4 Subtract $(25)_8 - (73)_8$ using 8's complement method.

Hexadecimal arithmetic

Hexadecimal addition:-

Procedure:

1. The **sum** of **2 Hexadecimal digits** is same as their **decimal sum** provided that the sum is **less than 16**.
2. If the **Hexadecimal sum is 16** or **greater than 16** then **subtract 16** to obtained **Hexadecimal digit**. A **carry** is **produced** when the **Hexadecimal sum** is corrected in this way.

Example: Perform the Hexadecimal addition for the following

Solution: $(B)_{16} + (9)_{16} = 20 - 16 = (4)_{16}$ **With carry 1**

$(7)_{16} + (9)_{16} = 16 - 16 = (0)_{16}$ **With carry 1**

Hexadecimal addition

Example: Perform the octal addition $(6E)_{16} + (C5)_{16}$

Solution: $(6E)_{16} + (C5)_{16} = (\ ?\ )_{16}$

$$
\begin{array}{cccc}
 & 1 & & \\
 & 6 & & E \\
1 & C & & 5 \\
\hline
(1 & 9 & -16) & 19 \\
\hline
1 & 3 & & 3 \\
\hline
\end{array}
$$

$\therefore (6E)_{16} + (C5)_{16} = (514)_{16}$

Note: If the octal sum of octal digit is **32** or **>32** then subtract **32** from **each digit** and set **carry=2**.

Hexadecimal Subtraction

Example: Find the 16's complement of $(14A)_{16}$

Solution: $(14A)_{16}$

16's complement = **15's complement** + 1 (**at LSB**)

15's complement is

$$\begin{array}{r} 15\ 15\ 15 \\ -\ 1\ 4\ A \\ \hline E\ B\ 5 \end{array}$$

15's complement is

$$\begin{array}{r} E\ B\ 5 \\ +\ \ \ \ 1 \\ \hline E\ B\ 6 \end{array}$$

16's complement is

$\therefore$ **16's complement** of $(14A)_{16}$ is $(EB6)_{16}$

Hexadecimal Subtraction

Example: Subtract $(C4)_{16} - (7B)_{16}$ using 16's complement method.

Solution: $(C4)_{16} - (7B)_{16} = (\ ?\ )_8$

$M = (C4)_{16}$ $N = (7B)_{16}$

15's complement is

$$\begin{array}{r} 15\ \ 15 \\ -\ 7\ \ B \\ \hline 8\ \ 4 \end{array}$$

15's complement of N is

$$\begin{array}{r} 8\ \ 4 \\ +\ \ \ 1 \\ \hline 8\ \ 5 \end{array}$$

16's complement of N is

$\therefore$ **16's complement** of $(7B)_{16}$ is $(85)_{16}$

Step 1: Add **M+ 16's** complement of **N**

$$\begin{array}{r} M = \ \ C\ \ 4 \\ +\ N = 1\ \ 8\ \ 5 \\ \hline 1\ \ (20-16)\ \ 9 \\ 1\ \ 4\ \ 9 \end{array}$$

Carry → 1

Step 2: If **carry is generated,** add carry to the LSB of result

Discard the Carry → 1

$4\ \ 9$

$(4\ \ 9)_{16}$

$\therefore (C4)_{16} - (7B)_{16} = (49)_{16}$

Hexadecimal Subtraction

Example: Subtract $(7B)_{16} - (C4)_{16}$ using 16's complement method.

Solution: $(7B)_{16} - (C4)_{16} = (\ ?\ )_8$

$M = (7B)_{16} \quad N = (C4)_{16}$

15's complement is

$$\begin{array}{cc} 15 & 15 \\ -C & 4 \\ \hline 3 & B \end{array}$$

15's complement of N is $3\ B$

$$\begin{array}{cc} + & 1 \\ \hline 3 & C \end{array}$$

16's complement of N is $3\ C$

$\therefore$ **16's complement** of $(C4)_{16}$ is $(3C)_{16}$

Step 1: Add **M+ 16's** complement of **N**

$$\begin{array}{rcc} & 1 & \\ M = & 7 & B \\ +\,N = & 3 & C \\ \hline & B & (23-16) \\ & B & 7 \end{array}$$

Step 2: If carry is not generated, then take 8's complement of obtained result and place –ve sign to number .

Result is

$$\begin{array}{cc} 15 & 15 \\ -B & 7 \\ \hline 4 & 8 \end{array}$$

15's complement of result is

$$\begin{array}{cc} + & 1 \\ \hline -4 & 9 \end{array}$$

16's complement of result is

$\therefore (C4)_{16} - (7B)_{16} = -(49)_{16}$

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET
(AUTONOMOUS)

Topic

EXCESS-3 CODE

OVERVIEW

- Excess-3 code
- Gray Code
- Error detecting and correcting codes

Excess-3 code

1. **Excess-3** code is a **modified form** of a **BCD number**.

2. The **excess-3 code** can be **derived from** the **natural BCD code** by **adding 3** to **each coded** number.

3. It is a **non weighted code**.

4. In **excess-3 code** we get **9's complement of a number** by just **complementing each bit** due to this excess-3 code is called **self complementing code**.

Excess-3 code

Decimal digit	BCD Form	Excess-3(Xs-3)
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 1

Valid numbers are **3 to 12** and **invalid numbers** are **0-2 and 13-15**

Excess-3 code

Example:1 Find **Excess-3 code** and its **9's complement** for the $(592)_{10}$ decimal numbers

Solution: $(592)_{10} = (\ ?\ )_{XS-3}$

5	9	2	Decimal Number
3	3	3	Adding 3
8	12	5	Excess-3
1000	1100	0101	

$\therefore (592)_{10} = (1000\ 1100\ 0101)_{XS-3}$

complement of XS-3

1's complement of $(1000\ 1100\ 0101)_{XS-3}$ is $(0111\ 0011\ 1010)_2$

9's complement of $(592)_{10}$

$$\begin{array}{r} 9\ 9\ 9 \\ -\ 5\ 9\ 2 \\ \hline 4\ 0\ 7 \end{array}$$

9's complement is 407

4	0	7	9's complement
3	3	3	Adding 3
7	3	10	Excess-3
0111	0011	1010	

9's complement of $(592)_{10} = (0111\ 0011\ 1010)_{XS-3}$

Note: 1's complement of Xs-3 code = 9's complement.

Excess-3 code

Example:1 Find **Excess-3 code** and its **9's complement** for the $(403)_{10}$ decimal numbers

Solution: $(403)_{10} = (\ ?\ )_{XS-3}$

4	0	3	Decimal Number
3	3	3	Adding 3
7	3	6	Excess-3

0111 0011 0110

$\therefore (403)_{10} = (0111\ 0011\ 0110)_{XS-3}$

complement of XS-3

1's complement of $(0111\ 0011\ 0110)_{XS-3}$ is $(1000\ 1100\ 1001)_2$

9's complement of $(403)_{10}$

$$\begin{array}{r} 9\ 9\ 9 \\ -\ 4\ 0\ 3 \\ \hline 5\ 9\ 6 \end{array}$$

9's complement is

5	9	6	9's complement
3	3	3	Adding 3
8	12	9	Excess-3

1000 1100 1001

9's complement of $(403)_{10} = (1000\ 1100\ 1001)_{XS-3}$

Note: 1's complement of Xs-3 code = 9's complement.

Excess-3 addition

Excsee-3 addition

Procedure:

1. Add **XS-3 numbers** (**binary form**)
2. If **carry** occurs **add 3** $(0011)_2$ (in binary form) to the **sum** of **2 numbers** (4 bit number).
3. If **no carry** occurs **subtract 3** $(0011)_2$ from the **sum** of **2 numbers** (4 bit number)

Excsee-3 addition

Example: Add **8** and **6** in **Xs-3 code**.

Solution: $(8)_{10} + (6)_{10} = (\ ?\ )_{XS\text{-}3}$

$8 \rightarrow 8 + 3 = 11$ **Excess-3 form**

$6 \rightarrow 6 + 3 = 9$ **Excess-3 form**

$$\begin{array}{rl} & \ 1\ 1 \\ (11)_{XS\text{-}3} = & 1\ 0\ 1\ 1 \\ +\ (9)_{XS\text{-}3} = & 1\ 1\ 0\ 0\ 1 \\ \hline (20)_{XS\text{-}3} = & 1\ 0\ 1\ 0\ 0 \end{array}$$

Carry

If **carry** occurs **add 3** $(0011)_2$ to the result of **each group**

$$\begin{array}{rl} & \ 1\ 1 \\ \text{Result is} & 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0 \\ \text{Adding 3} & 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1 \\ \hline & 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1 \end{array}$$

4 7

$\therefore (8)_{10} + (6)_{10} = (47)_{XS\text{-}3} = (0100\ 0111)_{XS\text{-}3}$

Excess-3 addition

Example: Add **1** and **2** in **Xs-3 code**.

Solution: $(1)_{10} + (2)_{10} = (\ ?\ )_{XS-3}$

$1 \rightarrow 1 + 3 = 4$ **Excess-3 form**

$2 \rightarrow 2 + 3 = 5$ **Excess-3 form**

$$\begin{array}{rl} & 1 \\ (4)_{XS-3} = & 0\ 1\ 0\ 0 \\ +\ (5)_{XS-3} = & 0\ 1\ 0\ 1 \\ \hline (9)_{XS-3} & 1\ 0\ 0\ 1 \end{array}$$

No Carry

If **no carry** occurs subtract **3** $(0011)_2$ to the result of **each group**

$$\begin{array}{rl} & 0\ 1\ 1\ 0 \\ \textbf{Result is} & \cancel{1}\ \cancel{0}\ \cancel{0}\ 1 \\ \textbf{Subtracting 3} & 0\ 0\ 1\ 1 \\ \hline & 0\ 1\ 1\ 0 \end{array}$$

6

$\therefore\ (1)_{10} + (2)_{10} = (6)_{XS-3} = (\ 0110\ )_{XS-3}$

Workout Problems Excess-3 Addition

Example:1 Perform the following addition in **in Xs-3**

$a)\quad (2)_{10} + (3)_{10}$

$b)\quad (37)_{10} + (28)_{10}$

$c)\quad (16)_{10} + (29)_{10}$

$d)\ (247.6)_{10} + (359.4)_{10}$

Excess-3 Subtraction

Excsee-3 addition

Procedure:

1. To **subtract** in **excess-3** by subtract **each 4 bit group** of **subtrahend** from the corresponding **4 bit group** of **minuend** starting **from LSB**.

2. If there is **no borrow** add $(0011)_2$ to the **difference term** of **such group**.

3. If there is **a borrow** subtract $(0011)_2$ from the **difference term**.

Excess-3 Subtraction

Example: $(267)_{10} - (175)_{10}$ Perform the **XS-3 Subtraction**.

Solution: $(267)_{10} - (175)_{10} = (\ ?\ )_{XS-3}$

2	6	7	Decimal Number
+ 3	3	3	Adding 3
5	9	10	Excess -3
0101	1001	1010	Excess -3 in binary

1	7	5	Decimal Number
+ 3	3	3	Adding 3
4	10	8	Excess -3
0100	1010	1000	Excess -3 in binary

$(267)_{10}$ = 0 1 0 1 1 0 0 1 1 0 1 0

$-(175)_{10}$ = 0 1 0 0 1 0 1 0 1 0 0 0

0 0 0 0 1 1 1 1 0 0 1 0

+0 0 1 1 - 0 0 1 1 +0 0 1 1

0 0 1 1 1 1 0 0 0 1 0 1

3 12 5

$\therefore (267)_{10} - (175)_{10} = (\ 3125\ )_{XS-3}$

Excess 3 Subtraction using 9's complement method

Excess 3 subtraction 9's complement method

Procedure:

1. Find the **9's complement** of a **subtrahend**.

2. **Add 2 numbers** using **Excess addition** (**minuhend+9's complement of subtrahend**).

3. If carry is **generated** from 4 bit sum, then add it to the **next bits** and consider this bits as **Invalid**. **Add 3** for **Invalid bits** and **subtract 3** for others

4. In final result, if **carry** is **occurred** then **add** it to the **LSB** remaining result.

5. If **carry** is **not generated**, **takes 9's complement** of **result** and **result is negative** .

Excess 3 Subtraction using 's complement method

Example: Perform Subtract $(983)_{10} - (187)_{10}$ in Excess -3 using 9's complement method.

Solution: $(983)_{10} - (187)_{10} = (\ ?\ )_{XS\text{-}3}$

$M = (983)_{10} \quad N = (187)_{10}$

9's complement is 9 9 9
− 1 8 7
9's complement of N is 8 1 2

Excess 3 code for $187 = 4\ \ 11\ \ 10 = 0\ 1\ 0\ 0\ \ 1\ 0\ 1\ 1\ \ 1\ 0\ 1\ 0$

9's complement of 187 in Xs-3 is $= 1\ 0\ 1\ 1\ \ 0\ 1\ 0\ 0\ \ 0\ 1\ 0\ 1$

Step 1: Add **M+ 9's** complement of **N**

Excess 3 code for $M = 983 = (12\ \ 11\ \ 6)_{XS\text{-}3} = 1\ 1\ 0\ 0\ \ 1\ 0\ 1\ 1\ \ 0\ 1\ 1\ 0$

Excess 3 code for $N = 812 = (11\ \ 4\ \ 5)_{XS\text{-}3} = 1\ 0\ 1\ 1\ \ 0\ 1\ 0\ 0\ \ 0\ 1\ 0\ 1$ +

End Carry

Addition : 1 0 1 1 1 1 1 1 1 1 0 1 1

If carry generated, add 3 otherwise subtract 3 : +0 0 1 1 − 0 0 1 1 − 0 0 1 1

Result : 1 1 0 1 0 1 1 0 0 1 0 0 0

End Carry

Excess 3 Subtraction using 9's complement method

Step 2: If **carry is generated** add carry to the LSB of result

Result : 1 0 1 0 1 1 0 0 1 0 0 0

End around Carry

1

1 0 1 0 1 1 0 0 1 0 0 1

Excess-3 value : 10 12 9

$\therefore (983)_{10} - (187)_{10} = (796)_{10} = (10129)_{XS-3}$

Excess 3 Subtraction using 10's complement method

Excess 3 subtraction 10's complement method

Procedure:

1. Find the **10's complement** of a **subtrahend**.

2. **Add 2 numbers** using **Excess addition** (**minuhend+10's complement of subtrahend**).

3. If carry is **generated** from 4 bit sum, then add it to the **next bits** and consider this bits as **Invalid**. **Add 3** for **Invalid bits** and **subtract 3** for others

4. In final result, if **carry** is **occured** then **discard** the carry.

5. If **carry** is **not generated**, **takes 10's complement** of **result** and **result is negative** .

Excess 3 Subtraction using 10's complement method

Example: Perform Subtract $(983)_{10} - (187)_{10}$ in Excess-3 using 10's complement method.

Solution: $(983)_{10} - (187)_{10} = (\ ?\ )_{XS\text{-}3}$

$M = (983)_{10} \quad N = (187)_{10}$

Excess 3 code for $187 = 4\ \ 11\ \ 10 = 0\ 1\ 0\ 0\ \ 1\ 0\ 1\ 1\ \ 1\ 0\ 1\ 0$

9's complement of 187 in Xs-3 is $= 1\ 0\ 1\ 1\ \ 0\ 1\ 0\ 0\ \ 0\ 1\ 0\ 1$

9's complement is $\begin{array}{r} 9\ 9\ 9 \\ -\ 1\ 8\ 7 \\ \hline 8\ 1\ 2 \end{array}$

9's complement of N is $\begin{array}{r} 8\ 1\ 2 \\ +\ \ \ \ 1 \\ \hline 8\ 1\ 3 \end{array}$

10's complement of N is

Step 1: Add **M+ 10's** complement of **N**

Excess 3 code for $M = 983 = (12\ \ 11\ \ 6)_{XS\text{-}3} = 1\ 1\ 0\ 0\ \ 1\ 0\ 1\ 1\ \ 0\ 1\ 1\ 0$

Excess 3 code for $N = 813 = (11\ \ 4\ \ 6)_{XS\text{-}3} = 1\ 0\ 1\ 1\ \ 0\ 1\ 0\ 0\ \ 0\ 1\ 1\ 0$

End Carry

Addition : $\begin{array}{r} 1\ 1 \\ 1\ 0\ 1\ 1\ \ 0\ 1\ 0\ 0\ \ 0\ 1\ 1\ 0 \\ 1\ 1\ 1 \\ \hline 1\ 0\ 1\ 1\ \ 1\ 1\ 1\ 1\ \ 1\ 1\ 0\ 0 \end{array}$

If carry generated, add 3 otherwise subtract 3 : $+0\ 0\ 1\ 1\ \ -0\ 0\ 1\ 1\ \ -0\ 0\ 1\ 1$

Result : $1\ 1\ 0\ 1\ 0\ \ 1\ 1\ 0\ 0\ \ 1\ 0\ 0\ 1$

End Carry

Excess 3 Subtraction using 10's complement method

Step 2: If **carry is generated,** then discard the carry.

Result : (1) 1 0 1 0 1 1 0 0 1 0 0 1

Discard the Carry

1 0 1 0 1 1 0 0 1 0 0 1

Excess-3 value : 10 12 9

$\therefore (983)_{10} - (187)_{10} = (796)_{10} = (10129)_{XS\text{-}3}$

Gray code

Gray code

1. Gray code is a **non weighted code**.

2 Gray code is a special case of **unit distance code** i.e. Bit pattern for **2 conjugate numbers** differ only **1 bit** position.

3. These codes are also called **cyclic code** and **reflective codes**.

Binary to Gray conversion

Binary to Gray conversion:-

Procedure:

Step1: Record the **MSB** of the **binary** as the **MSB** of the **gray code**.

Step2: **Add MSB** of the **binary** to the **next bit** in binary **record the sum** and **ignore the carry**. The **sum** is the **next bit** of **Gray code**.

Step3: **Add second** bit of **binary** number to the **3rd bit** of binary. **3rd** bit to **4th** bit and so on……

Step4: **Record** the **successive sums** as the successive bits of **gray codes**. Until **all the bits** of binary number are **completed**.

Binary to Gray conversion

Example: 1 **1 0 0 1** convert to **gray code**

Solution: $(1001)_2 = (\ ?\ )_{gray}$

Binary number 1 ⊕ 0 ⊕ 0 ⊕ 1

Gray code 1 1 0 1

$(1001)_2 = (1101)_{gray}$

Note : If carry is generated discord it.

Example: 2 **1 1 0 1 0 1** convert to **gray code**

Solution: $(110101)_2 = (\ ?\ )_{gray}$

Binary number 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1

Gray code 1 0 1 1 1 1

$(110101)_2 = (101111)_{gray}$

Gray to Binary conversion

Gray to Binary conversion:-

Procedure:

Step1: The **MSB** of the **binary number** is same as the **MSB** of **gray code** number.

Step2: **Add MSB** of **binary number** to the **next significant** bit of **gray code**. Record the **sum** and **ignore** the carry.

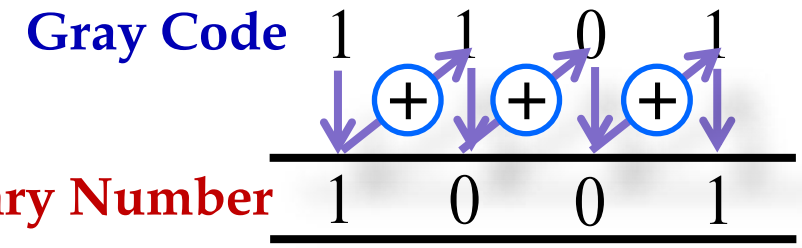
Step3: Add **2nd bit** of **binary** to the **3rd bit** of the **gray code**, **3rd bit** of **binary** to the **4th bit** of **gray code**.

Step4: **Continue this** till **Gray bits** are **exhausted**. Then the **sequence of bits** written is **binary** which are equivalent of **Gray code**.

Gray Code

Example: 1 **1 1 0 1** connect gray **to binary**

Solution: $(1101)_{gray} = (\ ?\ )_2$

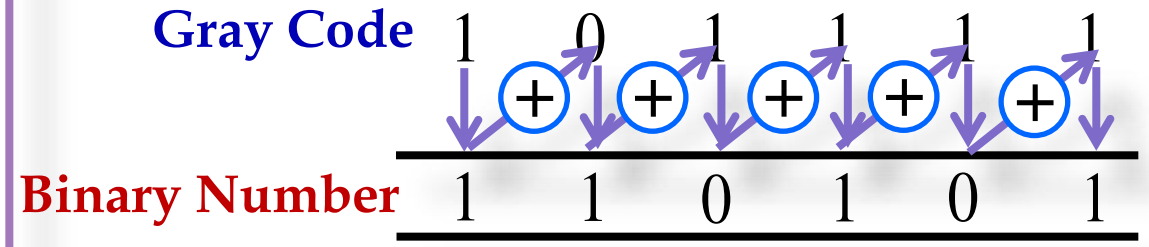


$(1101)_{Gray} = (1001)_2$

Note : If carry is generated discord it.

Example: 1 **1 0 1 1 1 1** connect **gray code** to binary

Solution: $(101111)_{gray} = (\ ?\ )_2$



$(101111)_{Gray} = (110101)_2$

Gray code

Write down the **Gray codes** for **(0-9) decimal** numbers.

Decimal digit	BCD Form	Gray Code
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1

Gray code

Write down the **Gray codes** for **Excess Code**.

Decimal digit	8421 Form	Excess-3(Xs-3)	Gray code
0	0 0 0 0	0 0 1 1	0 0 1 0
1	0 0 0 1	0 1 0 0	0 1 1 0
2	0 0 1 0	0 1 0 1	0 1 1 1
3	0 0 1 1	0 1 1 0	0 1 0 1
4	0 1 0 0	0 1 1 1	0 1 0 0
5	0 1 0 1	1 0 0 0	1 1 0 0
6	0 1 1 0	1 0 0 1	1 1 0 1
7	0 1 1 1	1 0 1 0	1 1 1 1
8	1 0 0 0	1 0 1 1	1 1 1 0
9	1 0 0 1	1 1 0 0	1 0 1 0

Workout Problems Gray Code

Example:1 Convert the 101011 Gray code into Binary.

Example:2 Convert the 10110010 Gray code into Hexadecimal, octal and Decimal.

Example:3 Convert the following into Gray code.

$a)\ (3A7)_{16}$ $b)\ (527)_{8}$ $c)\ (652)_{10}$

Example:4 Convert the 10111010 Gray code into .

$a)$ Hexadecimal $b)$ Octal $c)$ Decimal

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

Topic

ERROR DETECTING AND CORRECTING CODES

N.Sreeramula Reddy,
Assistant Professor,
Department of EEE,
AITS-Rajampet.

Error detecting and correcting codes

Error detecting and error correcting codes:-

- When the **digital** information is in **binary form**. It **transmitted** from **one system** to **another system** an **error may occur**. i.e. a **signal** corresponding to **0** may change to **1** or **vice versa**.
- Due to presence of **noise**.
- To maintain the **data integrity** between **transmitter** and **receiver**, **extra bit** or **more than 1 bit** are **added** in data.
- The data along the **extra bit/bits** forms the **code**.
- Codes which allow **only error detection** called **error detecting** codes.
- Codes which allow **error detection** and **error correction** called **error detecting and correcting codes.**

Eg:- **parity** **and** **hamming** **codes.**

Error detecting and correcting codes

Parity bit

1. A **parity bit** is used for the **purpose** of **detecting error** during the **transmission** of **binary information**.
2. A **parity bit** is an **extra bit** included with a **binary message** to make the number of **1's** either **even** or **odd**.
3. The **circuit** that **generated** parity bit in the **transmitter** is called **parity generator**.
4. The **circuit** that **checks** the parity in the **receiver** is called **parity checker**.
5. In **even parity** the **added parity** bit will move the **total number** of **1's** an **even**.
6. In **odd parity** the **added parity** bit will move the **total number** of **1's** an **odd**.
7. It detects only bits **error present**.

Error detecting and correcting codes

Example:-odd parity

3-bit message				Message with odd parity	
	A	B	C	Message	Odd parity
0	0	0	0	000	1
1	0	0	1	001	0
2	0	1	0	010	0
3	0	1	1	011	1
4	1	0	0	100	0
5	1	0	1	101	1
6	1	1	0	110	1
7	1	1	1	111	0

Error detecting and correcting codes

Example:-Even parity

3-bit message				Message with Even parity	
	A	B	C	Message	Even parity
0	0	0	0	000	0
1	0	0	1	001	1
2	0	1	0	010	1
3	0	1	1	011	0
4	1	0	0	100	1
5	1	0	1	101	0
6	1	1	0	110	0
7	1	1	1	111	1

Error detecting and correcting codes

Example: In **even parity** scheme which of the following words contain error.

Solution:

$a)\ (10101010)_2 \longrightarrow (1\ 0\ 1\ 0\ 1\ 0\ 1\ 0)_2$

1's→ 4 **Even Parity**

No error

$b)\ (11110110)_2 \longrightarrow (1\ 1\ 1\ 1\ 0\ 1\ 1\ 0)_2$

1's→ 6 **Even Parity**

No error

$c)\ (1011101)_2 \longrightarrow (1\ 0\ 1\ 1\ 1\ 0\ 1)_2$

1's→ 5 **Odd Parity**

It has an error to correct it 1 is added

$(1\ 0\ 1\ 1\ 1\ 0\ 1\ 1)_2$

Error detecting and correcting codes

Example: In **Odd parity** scheme which of the following words contain error.

Solution:

$a)\ (11101010)_2 \longrightarrow (1\ 1\ 1\ 0\ 1\ 0\ 1\ 0)_2$

1's→ 5 Odd Parity

No error

$b)\ (10110111)_2 \longrightarrow (1\ 0\ 1\ 1\ 0\ 1\ 1\ 1)_2$

1's→ 6 Even Parity

It has an error to correct it 1 is added $(1\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 1)_2$

$c)\ (10011010)_2 \longrightarrow (1\ 0\ 0\ 1\ 1\ 0\ 1\ 0)_2$

1's→ 4 Even Parity

It has an error to correct it 1 is added

$(1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1)_2$

Hamming code

Hamming code

1. **Hamming code** not only provide **detection** of a **error**, but also **identifies** which **bit** or **digit** is in **error**. So that it can **correct** the error.

2. Thus the **hamming code** is called **error detecting** and **correcting code**.

3. The **code** uses a **number of parity bits** (depends upon the **no.of information bits**) located at **certain positions** in the **code group**.

Hamming code

Number of parity bits:-

$P =$ no.of parity bits

$X =$ information

$$2^P \geq P + X + 1$$

If the **no.of information bits** is **X** then the **no.of parity bits** P can be **determined** by the following relationship.

$$2^P \geq P + X + 1$$

Hamming code

Example: For example **X=4** information bits then **P** is found by **trial and error** by using equation

Solution:

Let $P = 0 \Rightarrow 2^P \geq P + X + 1$

$2^0 \geq 0 + 4 + 1$

$1 \not\geq 5$

Let $P = 1 \Rightarrow 2^P \geq P + X + 1$

$2^1 \geq 1 + 4 + 1$

$2 \not\geq 6$

Let $P = 2 \Rightarrow 2^P \geq P + X + 1$

$2^2 \geq 2 + 4 + 1$

$4 \not\geq 7$

Let $P = 3 \Rightarrow 2^P \geq P + X + 1$

$2^3 \geq 8 + 4 + 1$

$8 \geq 8$ (satisfied)

$\therefore$ **3 Parity bit are sufficient**

- **3 parity bits** are required to provide **single error** correction for **4 information bits**.
- Hence the **total number of bits** in **transmission code** is **X+P** .

$$X + P = 4 + 3 = 7$$

Hamming code

Location of parity bits in the code

1. To each group of **M** information bits **(D_1, D_2, D_3,……D_m)**, **N** parity bits **(P_1, P_2, P_3,……P_n)** are added to form **(M+N)** information code.

2. The **parity bits** are **located** in the **position** that are **numbered** corresponding to **ascending** powers of **2**. **(1,2,4,8,16,……)**.

3. In above example for **7 bit codes** location for **parity bits** and **information bits** are shown in below.

   Information bits = X+P=4+3=7

7	6	5	4	3	2	1
D_7	D_6	D_5	P_4	D_3	P_2	P_1

Here P_1,P_2,P_4 are parity bits D_3,D_6,D_5,D_7 are information bits.

P_n represents parity bits, D_m represents information bits. And **n** represents location number.

Hamming code

Assigning values to parity bits

In hamming code each parity bit provides a check on certain other bits in the total code.

Bit designation	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Information bits(D_m)							
Parity bits(D_n)							

$P_1 \longrightarrow 1, 3, 5, 7$

$P_2 \longrightarrow 2, 3, 6, 7$

$P_4 \longrightarrow 4, 5, 6, 7$

Hamming code

$P_1 \rightarrow$ checks **bit location** 1,3,5,7 and assigns P_1 according of parity bit P_1 has a **1** for its **right most bit**. This **parity bit checks** all the **bit location** including **itself** that have **1** in the **same location** in the **binary location** number. For **even** or **odd** parity

$P_2 \rightarrow$ checks **bit location** 2,3,6,7 and assign P_2 according to **even** or **odd** parity.

$P_4 \rightarrow$ checks **bit location** 4,5,6,7 and assign P_4 according to **even** or **odd** parity.

Hamming code

Example: Encode the binary word **1011** into **7** bit even parity hamming code.

Solution:

$X = 4$ information

$P = ?$ no.of parity bits

$$2^P \geq P + X + 1$$

$$2^3 \geq 3 + 4 + 1$$

$$8 \geq 8$$

$\therefore$ No.of parity bits = **3** i.e, **P_1, P_2, P_4**

Total no.of information bits = **Information bits +Parity bits**

= X + P

= 4 + 3

= 7

Hamming code

Cont.....

Bit designation	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Information bits(D_m)	1	0	1		1		
Parity bits(D_n)						0	1

For $P_1 \rightarrow$ checks bit location 1,3,5,7 and assign P_1 according to even parity

1	3	5	7	
P_1	1	1	1	$\longrightarrow$ $P_1 = 1$ For Even Parity

For $P_2 \rightarrow$ checks bit location 2,3,6,7 and assign P_2 according to even parity

2	3	6	7	
P_2	1	0	1	$\longrightarrow$ $P_2 = 0$ For Even Parity

Hamming code

Cont.....

Bit designation	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Information bits(D_m)	1	0	1		1		
Parity bits(D_n)				0		0	1
Hamming Code	1	0	1	0	1	0	1

For P_4 → checks bit location 4,5,6,7 and assign P_4 according to even or odd parity

4	5	6	7			
P_4	1	0	1	→	$P_4 = 0$	For Even Parity

∴ The resultant hamming code is **(1010101)**

Hamming code

Example: Determine the single error correcting code for the information code **10111** for **odd parity**.

Solution:

$X = 5$ information

$P = ?$ no.of parity bits

$$2^P \geq P + X + 1$$

$$2^4 \geq 4 + 5 + 1$$

$$16 \geq 10$$

$\therefore$ No.of parity bits = **4** **i.e,** P_1, P_2, P_4, P_8

Total no.of information bits = **Information bits +Parity bits**

= X + P

= 5 + 4

= 9

Hamming code

Cont.....

Bit designation	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	9	8	7	6	5	4	3	2	1
Bit location number	1001	1000	0111	0110	0101	0100	0011	0010	0001
Information bits(D_m)	1		0	1	1		1		
Parity bits(D_n)								1	0

For $P_1 \rightarrow$ checks bit location 1,3,5,7,9 and assign P_1 according to Odd parity

1	3	5	7	9	
P_1	1	1	0	1	$\rightarrow$ $P_1 = 0$ For Odd Parity

For $P_2 \rightarrow$ checks bit location 2,3,6,7 and assign P_2 according to Odd parity

2	3	6	7	
P_2	1	1	0	$\rightarrow$ $P_2 = 1$ For Odd Parity

Hamming code

Cont.....

Bit designation	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	9	8	7	6	5	4	3	2	1
Bit location number	1001	1000	0111	0110	0101	0100	0011	0010	0001
Information bits(D_m)	1		0	1	1		1		
Parity bits(D_n)		0				1		1	0
Hamming Code	1	0	0	1	1	1	1	1	0

For $P_4 \rightarrow$ checks bit location 4,5,6,7 and assign P_4 according to odd parity

4	5	6	7	
P_4	1	1	0	$\longrightarrow$ $P_4 = 1$ For Odd Parity

For $P_8 \rightarrow$ checks bit location 8,9 and assign P_8 according to odd parity

8	9	
P_8	1	$\longrightarrow$ $P_8 = 0$ For Odd Parity

$\therefore$ The resultant hamming code is **(100111110)**

Hamming code

Detecting and Correcting an error

1. For **correct** of individual **parity check** is marked by **0** ,whereas **wrong result** is marked by **1.**

2. After all **parity bit checks**, Binary **word is formed** taking resulting bit for P_1 as **LSB**.

3. This what gives **bit location** where error has **occurred**.

Hamming code

Example: 1 Assume that the **even parity** hamming code is **transmitted** and that is **0100011 received**. The receiver does not know what was transmitted. Determine bit locations where has error occurred **using** received code.

Solution:

The given hamming code is **0100011**

The given hamming code has **7 bits**. So it contains three parity bits.

No.of parity bits = **3** i.e, P_1, P_2, P_4

Hamming code

Cont.....

Bit designation	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Hamming Code	0	1	0	0	0	1	1

Check for error bit location

For $P_1 \rightarrow$ 1 3 5 7 → Odd Parity → $P_1 = 1$

1 0 0 0 ✗

For $P_2 \rightarrow$ 2 3 6 7 → Even Parity → $P_2 = 0$

1 0 1 0 ✓

For $P_4 \rightarrow$ 4 5 6 7 → Odd Parity → $P_4 = 1$

0 0 1 0 ✗

Parity bits are: P_4 P_2 P_1 = 1 0 1 (5)

The resultant is $P_4,P_2,P_1 = 101$. It gives bit location 5.

In location 5 there is a bit error. It is 0 , then it should be 1.

∴ The corrected code is (0110011)

Hamming code

Example: 2 The hamming code **101101101** is **received**. **Correct it** , if any **error** occurs. There are **4 parity bits** and **odd parity** is used.

Solution: The given hamming code is **101101101**

The given hamming code has **9 bits**. So it contains **4 parity** bits.

No.of parity bits = **4** i.e, P_1, P_2, P_4, P_8

Bit designation	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	9	8	7	6	5	4	3	2	1
Bit location number	1001	1000	0111	0110	0101	0100	0011	0010	0001
Hamming Code	1	0	1	1	0	1	1	0	1

Check for error bit location

For $P_1 \rightarrow$ 1 3 5 7 9 → Even Parity → **$P_1 = 1$**

1 1 0 1 1 ✗

Hamming code

Cont.....

Bit designation	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	9	8	7	6	5	4	3	2	1
Bit location number	1001	1000	0111	0110	0101	0100	0011	0010	0001
Hamming Code	1	0	1	1	0	1	1	0	1

Check for error bit location

For $P_2 \rightarrow$ 2 3 6 7 → Odd Parity → $P_2 = 0$

0 1 1 1 ✓

For $P_4 \rightarrow$ 4 5 6 7 → Odd Parity → $P_4 = 0$

1 0 1 1 ✓

For $P_8 \rightarrow$ 8 9 → Odd Parity → $P_8 = 0$

0 1 ✓

Parity bits are:

P_8	P_4	P_2	P_1
0	0	0	1

1

The resultant is $P_8,P_4,P_2,P_1 = 0001$. It gives bit location 1.

In location 1 there is a bit error. It is 1 , then it should be 0.

∴ The corrected code is (10110110**0**)

Hamming code

Example: 1 The message below code in the **7 bit hamming code** is transmitted through a noisy channel. Decode the message assuming that at most a single error occurred in the each word in even parity.

a) 1001001

b) 0111001

c) 1110110

d) 0011011

THANK YOU

Classification of binary codes

Classification of binary codes

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

Topic

BOOLEAN ALGEBRA

OVERVIEW

- Boolean Algebra
- Fundamental Postulates/Law of Boolean Algebra
- Error detecting and correcting codes

Boolean Algebra

* Boolean Algebra is a system of Mathematical logic.
* It differs from both ordinary algebra and Binary number system.
* It is a unique system.

Variable :-

* The symbol which represent an arbitary element of an Boolean algebra. It can have (receive) either 0 or 1.

Constant :-

$Y = A + 1$.

In the above expression A = variable, 1 = constant.

Complement :-

Complement is represented by Bar ($\bar{}$) over the letter.

Ex :- complement of variable A is $\bar{A}$.

If $A = 0$ complement of $A \Rightarrow \bar{A} = \phi$

if $A = 1 \Rightarrow \bar{A} = 0$.

Sometimes complement is represented by prime symbol (') is used to denote complement.

* logic "AND" operator of two variables is represented either by writing a (·) b/n 2 variables, such as A·B or by simply writing 2 variables such as AB.

* Logical "OR" operator of 2 variables is represented by writing '+' sign b/n variables such as A+B.

Logic OR operation:-

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = 1$

Logic "AND" operation:-

$0 \cdot 0 = 0$

$0 \cdot 1 = 0$

$1 \cdot 0 = 0$

$1 \cdot 1 = 1$

Fundamental Postulates/Law of Boolean Algebra:-

Fundamental postulates/law of Boolean Algebra:-

* Boolean Algebra is formulated by a defined set of elements, together with 2 binary operators.

laws of Boolean Algebra:-

1st law : law of Intersection

law 1 : $A \cdot 1 = A$

A	1	A·1
0	1	0
1	1	1

law 2 : $A \cdot 0 = 0$

A	0	A·0
0	0	0
1	0	0

2nd law: law of Identity (union)

law 1 : $A + 1 = 1$

A	1	A+1
0	1	1
1	1	1

law 2 : $A + 0 = A$

A	0	A+0
1	0	1
0	0	0

3rd law: law of Idempotency

law 1 : $A + A = A$

A	A	A+A
0	0	0
1	1	1

law 2 : $A \cdot A = A$

A	A	A·A
0	0	0
1	1	1

4th law: law of complements

law 1: $A+\bar{A}=1$

A	$\bar{A}$	$A+\bar{A}$
1	0	1
0	1	1

law 2: $A \cdot \bar{A} = 0$

A	$\bar{A}$	$A \cdot \bar{A}$
1	0	0
0	1	0

$$A+\bar{A}=1$$

$$A \cdot \bar{A}=0$$

5th law :- Law of Commutative

law 1 : $A \cdot B = B \cdot A$

A	B	A·B	B·A
0	1	0	0
1	1	1	1
0	0	0	0
1	0	0	0

$\therefore$ $\boxed{A \cdot B = B \cdot A}$

law 2 : $A + B = B + A$

A	B	A+B	B+A
0	1	1	1
1	1	1	1
0	0	0	0
1	0	1	1

$\therefore$ $\boxed{A + B = B + A}$

6th law :- law of Associative

law 1 :- $(A \cdot B)C = A \cdot (B \cdot C)$

A	B	C	A·B	(A·B)·C	B·C	A·(B·C)
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

$$(A \cdot B) \cdot C = A \cdot (B \cdot C)$$

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

6th law :- law of Associative

Law 2 : $(A+B)+C = A+(B+C)$

A	B	C	A+B	(A+B)+C	B+C	A+(B+C)
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1
1	0	0	1	1	0	1
1	1	0	1	1	1	1
0	1	1	1	1	1	1
1	1	1	1	1	1	1
1	0	1	1	1	1	1

7th law :- law of Distribution

law 1 :- $A \cdot (B+C) = (A \cdot B) + (A \cdot C)$

A	B	C	B+C	A·(B+C)	A·B	A·C	(A·B)+(A·C)
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

$\therefore A \cdot (B+C) = (A \cdot B) + (A \cdot C)$

7th law :- law of Distribution

law 2 :- $(A+B)\cdot(A+C) = (A+B)C$

A	B	C	A+B	A+C	(A+B)(A+C)	B·C	A+(B·C)
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

$\therefore (A+B)(A+C) = A+B\cdot C$

8th law :- law of Absorption

Law 1 :- $A \cdot (A+B) = A$

A	B	A+B	A·(A+B)	A
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

$\therefore$ $\boxed{A \cdot (A+B) = A}$

Law 2 :- $A + AB = A$

A	B	AB	A+AB	A
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	1

$\therefore$ $\boxed{A + AB = A}$

8th law :- law of Absorption

law 3 :- $A(\bar{A}+AB) = AB$

A	B	$\bar{A}$	AB	$\bar{A}+AB$	$A(\bar{A}+AB)$
0	0	1	0	1	0
1	0	0	0	0	0
0	1	1	0	1	0
1	1	0	1	1	1

$\therefore$ $A(\bar{A}+AB) = AB$

8th law :- law of Absorption

law 4 :- $AB + \bar{B} = A + \bar{B}$

A	B	AB	$\bar{B}$	$AB + \bar{B}$	$A + \bar{B}$
0	0	0	Ø 1	Ø 1	Ø 1
1	0	0	1	1	1
0	1	0	0	0	0
1	1	1	0	1	1

$\therefore$ $\boxed{AB + \bar{B} = A + \bar{B}}$

8th law :- law of Absorption

law 5 :- $(A + \bar{A}B) = A + B$

A	B	$\bar{A}$	$\bar{A}B$	$A + \bar{A}B$	$A + B$
0	0	1	0	0	0
1	0	0	0	1	1
0	1	1	1	1	1
1	1	0	0	1	1

$\therefore$ $\boxed{A + \bar{A}B = A + B}$

9th law :- law of double Negation

law 1 :- $\overline{\overline{A}} = A$

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

$\therefore$ $\boxed{\overline{\overline{A}} = A}$

Duality theorem :-

* change each OR sign to AND sign.
* change each AND sign to OR sign.
* complement any zero's or one's appearing in the expression.

Ex:- Duality of $A+\bar{A}=1$ is $A\cdot\bar{A}=0$

Duality of $A\cdot 0=0$ is $A+1=1$

$A+0=A \Rightarrow A\cdot 1=A$

Demorgan's Theorem :-

In Demorgan's theorem $\overline{AB} = \overline{A} + \overline{B}$

A	B	$\overline{A}$	$\overline{B}$	$\overline{AB}$	$\overline{A}\overline{B}$	$\overline{A} + \overline{B}$
0	0	1	1	0	01	1
1	0	0	1	0	1	1
0	1	1	0	0	1	1
1	1	0	0	01	0	0

Demorgan's Theorem :-

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

A	B	$\overline{A}$	$\overline{B}$	A+B	$\overline{A+B}$	$\overline{A} \cdot \overline{B}$
0	0	1	1	0	1	1
1	0	0	1	1	0	0
0	1	1	0	1	0	0
1	1	0	0	1	0	0

Example: Simplify the Boolean expression to a minimum number of literals. $xyz' + x'yz + xyz + x'y'z'$

Solution:

$\Rightarrow xyz' + x'yz + xyz + x'y'z'$

$\Rightarrow xy(z'+z) + x'y(z+z')$

$\Rightarrow (z+z')(xy + x'y)$

According to complement law $z+z'=1$

$\Rightarrow xy + x'y$

$\Rightarrow y(x+x')$ According to complement law

$\Rightarrow \boxed{y}$

Example: Simplify the Boolean expression to a minimum number of literals. $(BC' + A'D)(AB' + CD')$

Solution:

$\Rightarrow (BC' + A'D)(AB' + CD')$

$\Rightarrow BC'(AB' + CD') + A'D(AB' + CD')$

$\Rightarrow BC'AB' + BC'CD' + A'DAB' + A'DCD'$

$\Rightarrow AC'BB' + BD'CC' + DB'AA' + DD'AC$

According to complements law

$A \cdot A' = 0 \qquad BB' = CC' = DD' = 0$

$\Rightarrow AC'(0) + BD'(0) + DB'(0) + AC(0)$

$\Rightarrow \boxed{0}$

Digital Logic Gates

Digital Logic Gates

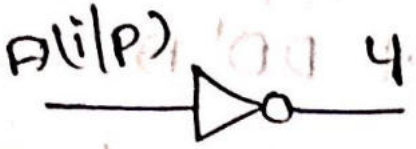
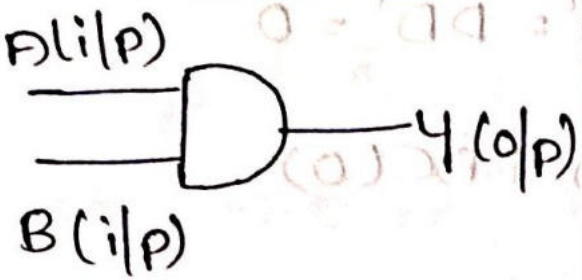
* Digital Logic gates are the Basic elements that makes up a digital system.

* The gate is a digital circuit with one or more input but only one output and is a able to operate on a number of Binary inputs. In order to perform a particular logical function.

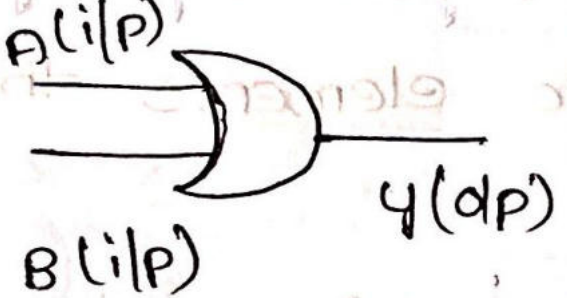
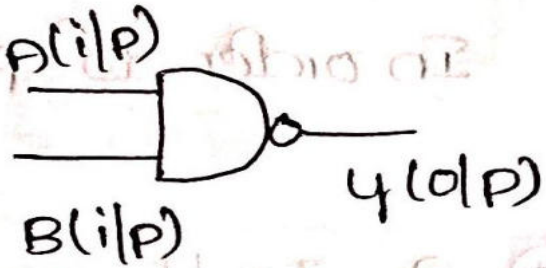
* The types of gates available are NOT, AND, OR, NAND, NOR, XOR, (exclusive OR), XNOR (exclusive NOR)

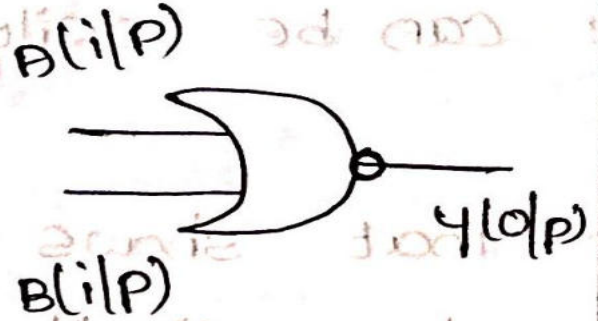
* The operation of a logic gate can be easily understood with the help of truth table.

* A truth table is a table that shows all the inputs and output possibilities of a logic circuit. i.e, the outputs for different possibilities of i/p.

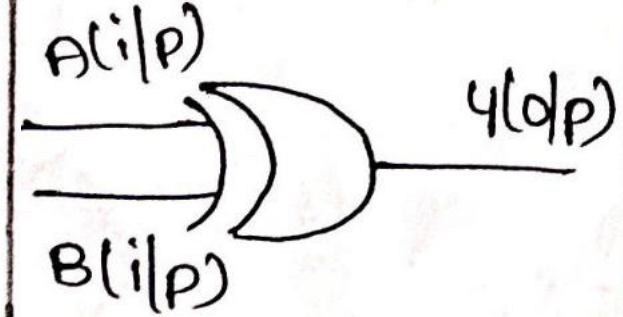
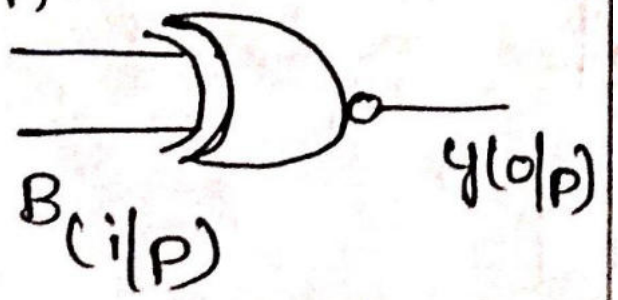
Name	Symbol	Algebric function	Truth table
NOT		$Y = \bar{A}$	i/p | o/p A | Y=A 0 | 1 1 | 0
AND		$Y = AB$	A | B | Y=AB 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1

output is high when all the inputs are high.

Name	Symbol	Algebric function	Truth table
OR		$y = A + B$	A B A+B 0 0 0 0 1 1 1 0 1 1 1 1
NAND		$y = \overline{AB}$	A B $y = \overline{AB}$ 0 0 0 1 0 1 0 1 1 0 0 1 1 1 1 0

Name	Symbol	Algebric function	Truth table
NOR		$y = \overline{A+B}$	

A	B	$y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Name	Symbol	Algebric function	Truth table
XOR (Exclusive OR)		$y = A \oplus B$ $= A\bar{B} + \bar{A}B$	
XNOR (Exclosive NOR)		$y = A \odot B$ $= AB + \bar{A}\bar{B}$	

A	B	$\bar{A}$	$\bar{B}$	$A\bar{B}+\bar{A}B$
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

A	B	$\bar{A}$	$\bar{B}$	$AB+\bar{A}\bar{B}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1

Properties of XOR Gate

Properties of XOR Gate

i) $A \oplus A = 0$

Proof :→ $A \oplus A = A \cdot \bar{A} + \bar{A} \cdot A$

$= 0 + 0$

$= 0$

$\therefore \boxed{A \oplus A = 0}$

ii) $A \oplus \bar{A} = 1$

Proof :→

$A \oplus \bar{A} = A \cdot \bar{\bar{A}} + \bar{A} \cdot \bar{A}$

$= A \cdot A + \bar{A} \cdot \bar{A}$

$= A + \bar{A}$

$= 1$

$\therefore \boxed{A \oplus \bar{A} = 1}$

iii) $A \oplus 1 = \bar{A}$ XOR as a inverter

Proof :- $A \oplus 1 = A \cdot \bar{1} + \bar{A} \cdot 1$

$= A \cdot 0 + \bar{A}$

$= 0 + \bar{A}$

$= \bar{A}$

A	1	$\bar{A}$
0	1	1
1	1	0

$\therefore$ $\boxed{A \oplus 1 = \bar{A}}$

iv) $A \oplus 0 = A$ XOR as non Inverter.

Proof :-

$A \oplus 0 = A \cdot \bar{0} + \bar{A} \cdot 0$

$= A \cdot 1 + 0$

$= A + 0$

$= A$

v) XOR as a Modulo 2 adder.

The **exclusive-OR gate** can be used as a **modulo 2 adder** because its truth table is **same as** the truth table of **modulo 2 adder.**

Modulo 2 adder		**Exclusive-OR gate**
$0 + 0 = 0$	$\equiv$	$0 \oplus 0 = 0$
$0 + 1 = 1$		$0 \oplus 1 = 1$
$1 + 0 = 1$		$1 \oplus 0 = 1$
$1 + 1 = 0$		$1 \oplus 1 = 0$

(vi) $(AB) \oplus (AC) = A(B \oplus C)$

A	B	C	$AB \oplus AC$	$A (B \oplus C)$
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	1	1
1	1	0	1	1
1	1	1	0	0

vii) $A \oplus B = C$ then $B \oplus C = A$, $A \oplus C = B$

$A \oplus B \oplus C = 0$.

If $A \oplus B = C$, then

$A \oplus C = B$

$B \oplus C = A$ and

$A \oplus B \oplus C = 0$

A	B	$A \oplus B = C$	$A \oplus C = B$	$B \oplus C = A$	$A \oplus B \oplus C = 0$
0	0	0	0	0	0
0	1	1	1	0	0
1	0	1	0	1	0
1	1	0	1	1	0

Note: The three input EX-OR, output is logic 1 only for odd number of logic 1 inputs.

Note: No similar terms for EX-NOR gate.

$$A \odot B = C \quad \text{then}$$

$$A \odot C = B$$

$$B \odot C = A \quad \text{and}$$

$$A \odot B \odot C = 1$$

A⊙B=C; then A⊙C=B, B⊙C=A and A⊙B⊙C=1

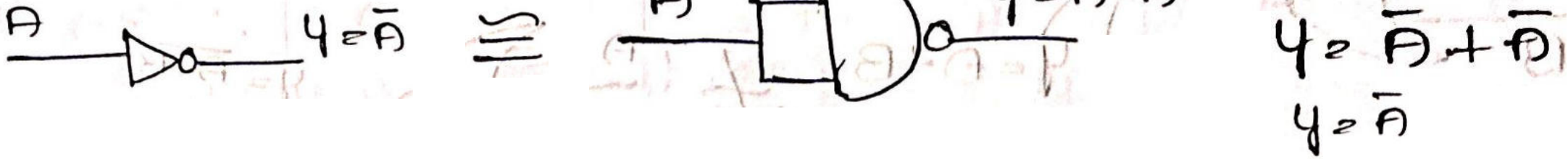
Universal Gates

Universal Gates

* The NAND and NOR gates are known as universal gates. since any logic function can be implemented by using NAND and NOR gates.

Implementation of Logic Gates Using Only NAND Gates

i) NOT gate

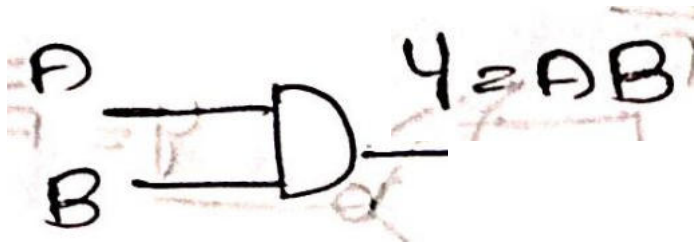


$Y = \overline{A \cdot A}$

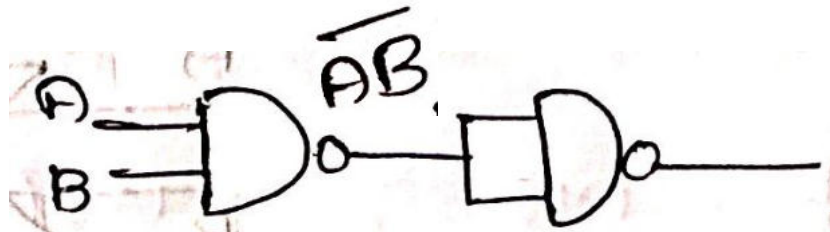
$Y = \bar{A} + \bar{A}$

$Y = \bar{A}$

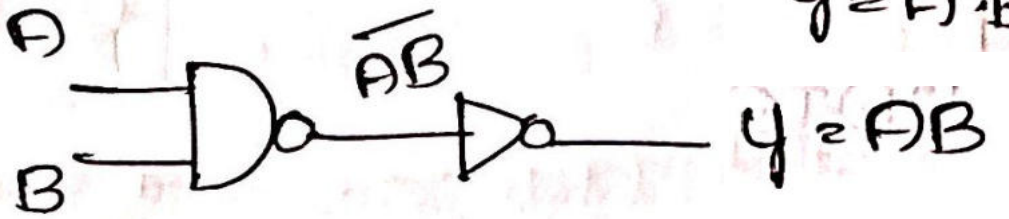
ii) AND gate



$\equiv$



$\not\equiv$



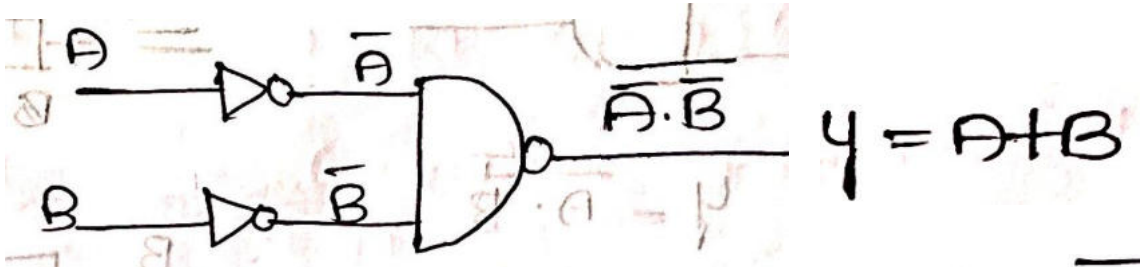
iii) OR gate

According to Demorgons law

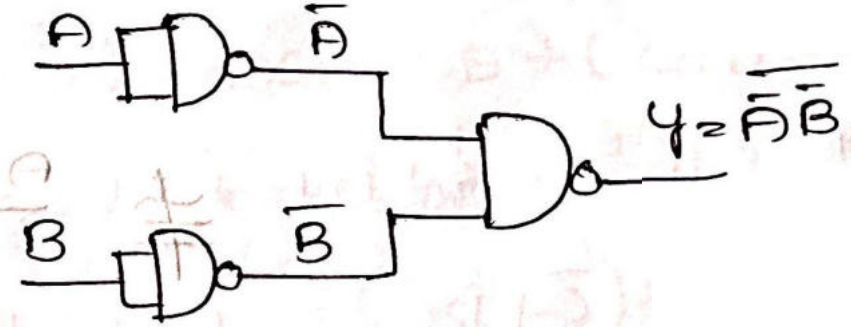
$$Y = \overline{\overline{A} + \overline{B}}$$

$$Y = \overline{\overline{A} \cdot \overline{B}}$$

$\not\equiv$



$\equiv$

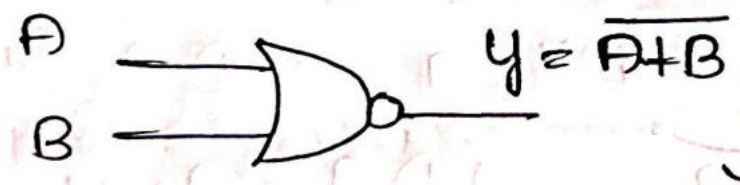


$$Y = \overline{\overline{A} \cdot \overline{B}}$$

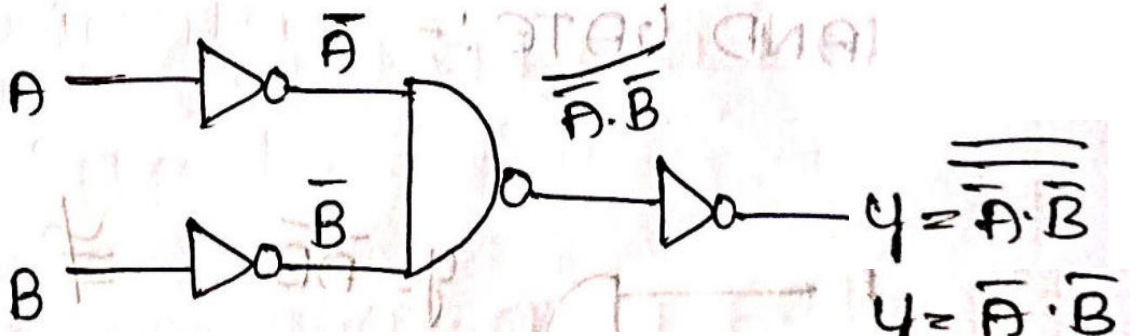
$$Y = \overline{\overline{A}} + \overline{\overline{B}}$$

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

iv) NOR gate :-



$\neq$



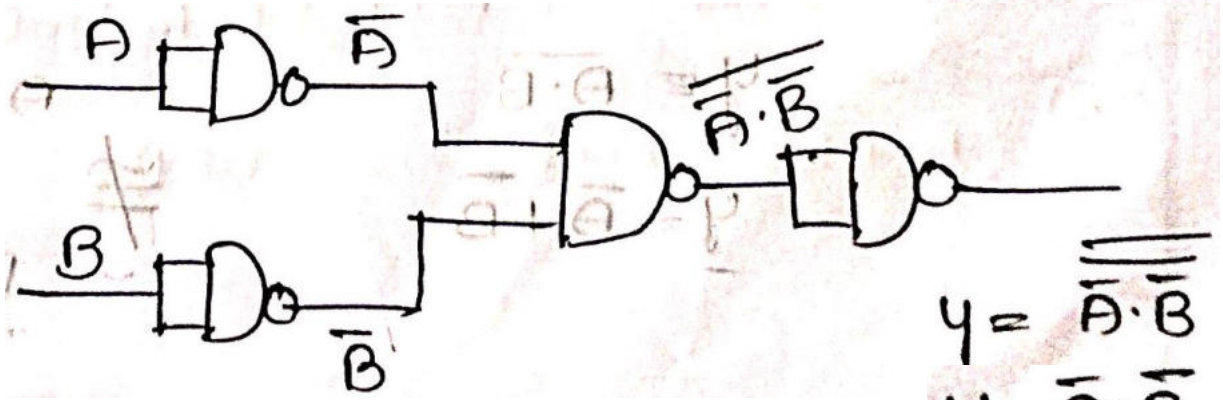
$Y = \overline{A+B}$

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

$Y = \overline{\bar{A}\cdot\bar{B}}$

$Y = A+B$

$=$

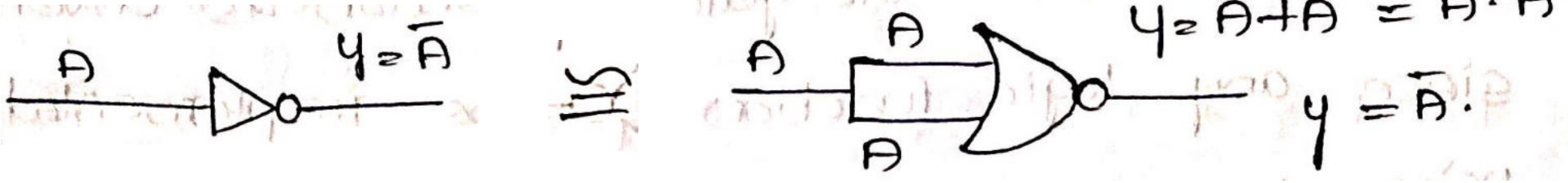


$Y = \bar{A}\cdot\bar{B}$

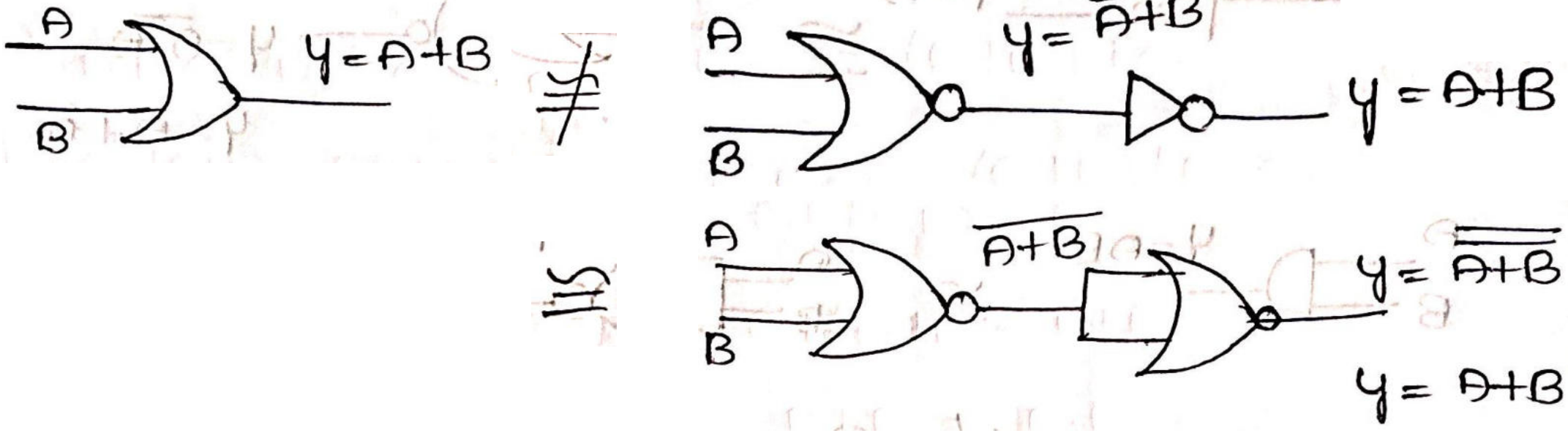
$\boxed{Y = \overline{A+B}}$

Implementation of Logic Gates Using Only NOR Gates

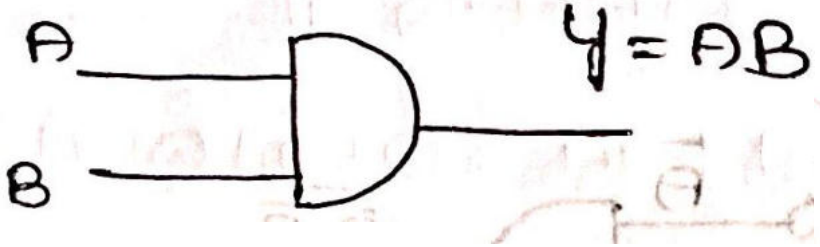
1) NOT GATE



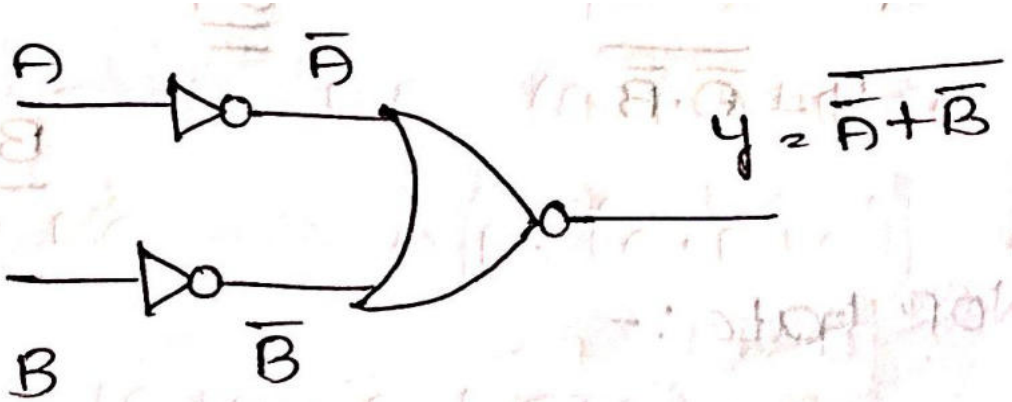
2) OR GATE



3) AND GATE



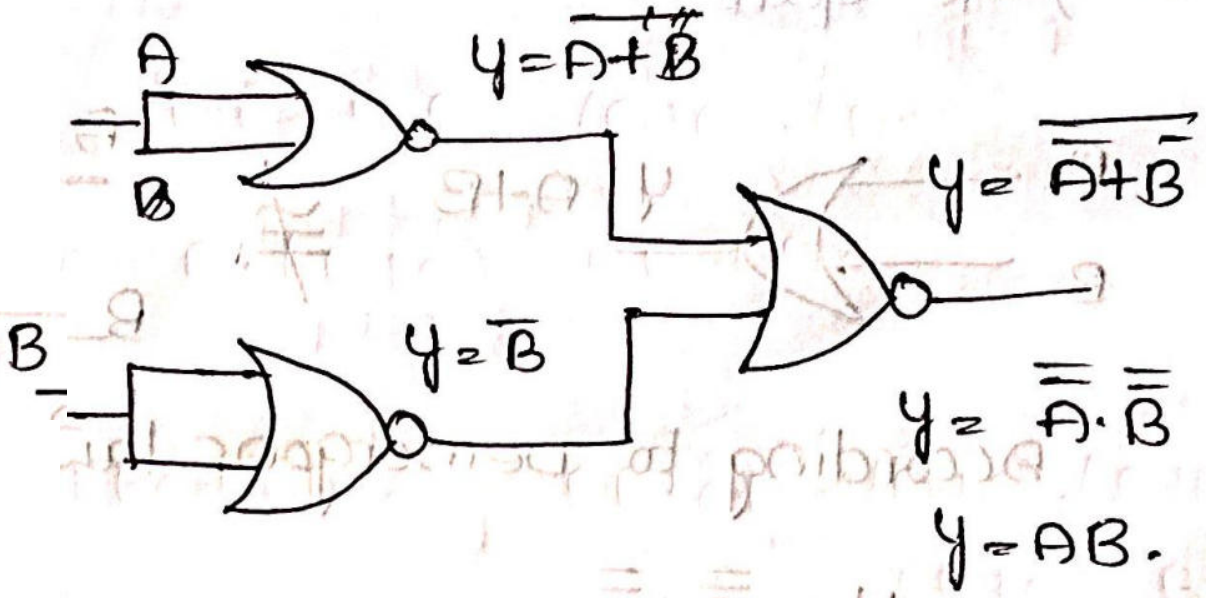
$\neq$



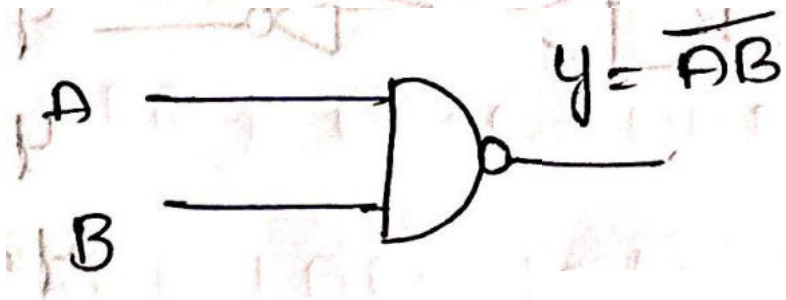
$$Y = \bar{\bar{A}} \cdot \bar{\bar{B}}$$

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

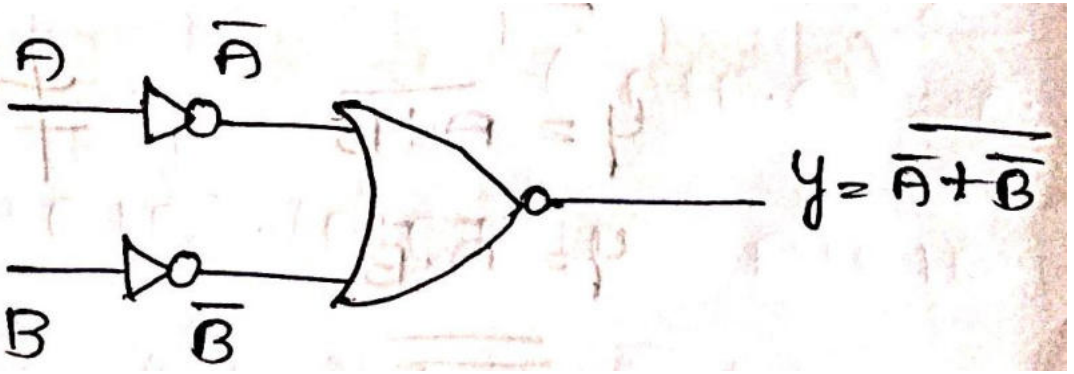
$=$



4) NAND GATE :-



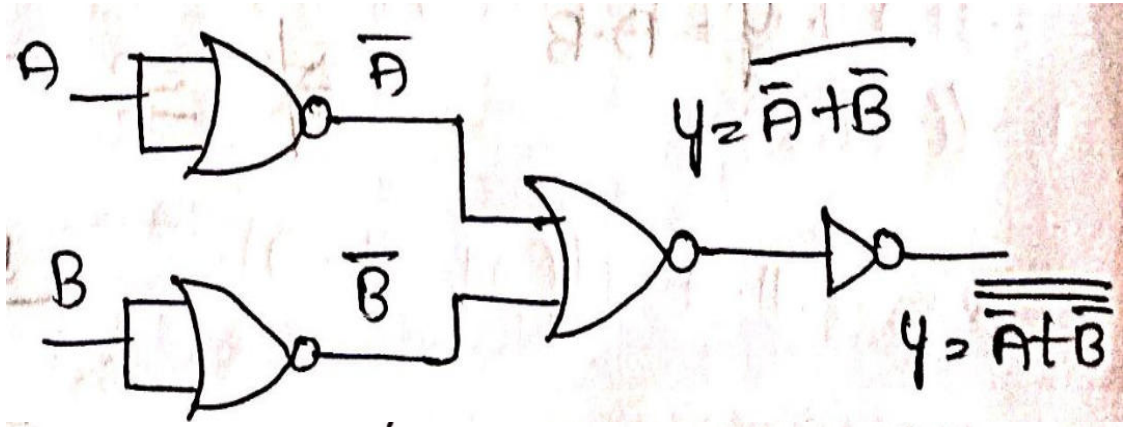
$\neq$



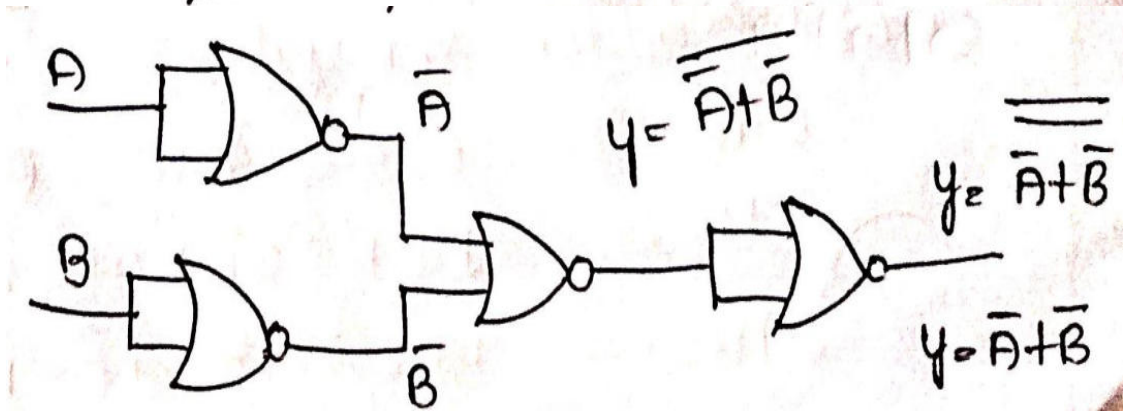
$Y = \overline{A \cdot B}$

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

$\neq$



$\neq$



$\equiv$

Workout Problems

Example:1 Reduce the following function into minimum number of literals. $(A'+C')(A'+C)(A+B+C+D)$

Example:2 Simplify following Boolean functions into minimum no. of literals.

1) $F = ABC + ABC' + A'B$. 2) $F = (A+B)'(A'+B')$.

Example:3 prove the Boolean function identity.

$$x + yz = (x+y)(x+z)$$

Workout Problems

Example:4 find the complement of following functions

1) $F_1 = \bar{x}y\bar{z} + \bar{x}\bar{y}z$ (Apply Demorgan's law).

2) $F_2 = x(\bar{y}\bar{z} + yz)$

Example:5 Demorganise the following functions

1) $F = \overline{(A+\bar{B})(C+\bar{D})}$ Ans:- $A\cdot B + \bar{C}\cdot B$

2) $F = \overline{\overline{AB}(CD+\bar{E}F)(\overline{AB}+\overline{CD})}$ Ans:- $AB + (\bar{C}+\bar{D})(\bar{E}+\bar{F}) + ABCD$

Workout Problems

Example:6

Reduce the following Boolean expressions.

1) $F = A\left[B + \bar{C}\left(\overline{AB + A\bar{C}}\right)\right]$

2) $F = A + B\left[AC + (B + \bar{C})D\right]$

3) $F = \left(\overline{A + \bar{B}C}\right)\left(A\bar{B} + ABC\right)$

4) $F = (B + BC)(B + \bar{B}C)(B + D)$

Answers:-

1) AB 2) $A + BD$ 3) 0 4) B

Example:7

Show that following fonctions

i) $AB + A\bar{B}C + B\bar{C} = AC + B\bar{C}$

ii) $A\bar{B}C + B + B\bar{D} + AB\bar{D} + A\bar{C} = B + C$

Duality Theorem

Expression	Dual
i) $\bar{0} = 1$	$\bar{1} = 0$
ii) $0 \cdot 1 = 0$	$1 + 0 = 1$
iii) $0 \cdot 0 = 0$	$1 + 1 = 1$
iv) $A \cdot 0 = 0$	$A + 1 = 1$
5) $1 \cdot 1 = 1$	5) $0 + 0 = 0$
6) $A \cdot 1 = A$	$A + 0 = A$
7) $A \cdot A = A$	$A + A = A$

Expression	Dual
8) $A \cdot \bar{A} = 0$	$A + \bar{A} = 1$
9) $A \cdot B = B \cdot A$	$A + B = B + A$
10) $A(B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$
11) $A \cdot (B + C) = AB + AC$	$A + (B \cdot C) = (A + B)(A + C)$
12) $A \cdot (A + B) = A$	$A + (A \cdot B) = A$
13) $A \cdot (A \cdot B) = AB$	$A + (A + B) = A + B$
14) $\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A} \cdot \bar{B}$
15) $(A+B)(\bar{A}+C)(B+C) =$ $(A+B)(\bar{A}+C)$	$A \cdot B + \bar{A} \cdot C + BC = A \cdot B + \bar{A} \cdot C$

Expression	Dual
16) $(A+C)(\bar{A}+B) = AB+\bar{A}C$	$A\cdot C+\bar{A}\cdot B = (A+B)\cdot(\bar{A}+C)$
17) $A+\bar{B}C = (A+\bar{B})(A+C)$	$A\cdot(\bar{B}+C) = A\bar{B}+AC$
18) $(A+B)(C+D) = AC+AD+BC+BD$	$AB+C\cdot D = (A+C)\cdot(A+D)\cdot(B+C)\cdot(B+D)$
19) $A+B = AB+\bar{A}B+A\bar{B}$	$A\cdot B = (A+B)\cdot(\bar{A}+B)\cdot(A+\bar{B})$
20) $A+B(\overline{C+\bar{D}E}) = A+B\bar{C}DE$	$A\left[B+\left(\overline{C\cdot\bar{D}+E}\right)\right] = A(B+\bar{C}+D+E)$
21) $\overline{\bar{A}B+\bar{A}+AB} = 0$	$\overline{(\bar{A}+B)\bar{A}\cdot(A+B)} = 1$
22) $AB+\bar{A}C+A\bar{B}C(AB+C) = 1$	$(A+B)\cdot(\bar{A}+C)\cdot(A+\bar{B}+C)+(A+B\cdot C) = 0$
23) $ABD+ABCD = ABD$	$(A+B+D)\cdot(A+B+C+D) = A+B+D$

Expression	Dual

24) $\overline{A\bar{B} + ABC} + A(B + A\bar{B}) = 0 \longrightarrow \overline{(A+\bar{B})\cdot(A+B+C)\cdot((A+B)(A+\bar{B}))} = 1$

25) $A + \bar{B}C(A+\bar{B}C) = A + \bar{B}C \longrightarrow (A\cdot(\bar{B}+C)) + A\cdot(\bar{B}+C) = A\cdot(\bar{B}+C)$

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

UNIT II

SWITCHING FUNCTIONS AND THEIR MINIMIZATION

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

Topic

CANONICAL AND STANDARD FORMS

OVERVIEW

- Canonical and Standard forms
- Converting expression to Standard SOP and POS forms
- Minterms and MaXterms Notation (M Notations)

Canonical and Standard forms

1. **Boolean functions** are constructed by connecting in the **Boolean constants** and **variables** with the **Boolean operations**. (**AND** & **OR**)
2. We use Boolean expressions to describe Boolean functions.
3. In Boolean functions the **literals (variables)** are appeared either in a **Complemented** or **uncomplemented** form .
4. The variables and terms (sum terms or product terms) are arranged in one of the two forms.

    1. Sum of products form (SOP)

    2. Products of sum form (POS)

Sum of products form (SOP):- Normal form

The sum of products is a group of 'AND' product terms 'OR'ed together

Ex:- Function A, B, C $F(A,B,C) = ABC + A\bar{B}C$

$F(A,B,C) = \bar{A}BC + AC\bar{B} + ACB.$

$F(P,Q,R,S) = \bar{P}Q + QR + RS.$

Products of Sum form (POS):- Normal form

A product of sum is any groups of 'OR' sum terms 'AND'ed together.

Ex:- $F(A,B,C) = (A+B)(\bar{A}+C)$

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

$F(P,Q,R,S) = (P+Q)(Q+R)(R+S)$

Canonical form (standard SOP and POS form) :-

canonical forms are special cases of sop and pos forms.

Standard SOP form (or) Min term canonical form :-

* In sop form all the indivudual terms do not involve all literals (or) variables.
* In each term of sop form contains all the literals then sop form is called as standard or canonical sop form.
* Each individual term in standard sop form is called min term.

Ex :- $F(A,B,C) = AB + A\bar{B}C + AC$ (Normal SOP form)

$F(A,B,C) = \underbrace{A\bar{B}C}_{\text{minterm.}} + AB\bar{C} + \bar{A}BC$ (standard form)

Standard POS form (or) Max term canonical form :-

* In each term in pos form contains all the literals then pos form is called standard (or) canonical pos form.
* Each individual term in the standard pos form is called max term.

Ex :- $F(A,B,C) = (A+B)(A+\bar{B}+C)$ (Normal pos form)

$F(A,B,C) = \underbrace{(A+\bar{B}+C)}_{\text{Max term}}(A+B+C)$ (standard pos form)

Converting expression to Standard SOP and POS forms:-

Standard SOP forms:-

Step 1:- find the missing literals or variables in each product term if in.

Step 2 :- AND each product term having literals with terms form by OR ing the literals and its complement.

Step 3 :- Expand the terms by applying distributive law. and reorder the literals or variables in product terms.

step 4:- Reduce the expression by omitting repeated product terms if any. ($A+A+A=A$)

Example: Convert the expression in standard sop form:-

1) $F(A,B,C) = AC+AB+CB$

Solution: Given function is $F(A,B,C) = AC+AB+BC$.

Step: 1 Find missing literals or variables in each product term.

$$F(A,B,C) = AC + AB + BC$$

$$\downarrow \quad \downarrow \quad \downarrow$$

$$B \quad C \quad A$$

Step: 2 $F(A,B,C) = AC \cdot 1 + AB \cdot 1 + BC \cdot 1$

By complements law $A + \bar{A} = 1$

$F(A,B,C) = AC(B+\bar{B}) + AB(C+\bar{C}) + BC(A+\bar{A})$

Step: 3 By distributive law $A(B+C) = AB + AC$.

$F(A,B,C) = ACB + A\bar{B}C + ABC + AB\bar{C} + ABC + \bar{A}BC$.

Step: 4 Reduce the expressions by omitting repeated product terms.

$F(A,B,C) = ACB + A\bar{B}C + AB\bar{C} + \bar{A}BC$

Example: Convert the expression in standard sop form :-

$$F(A,B,C) = A + BCA$$

Solution:

$$F(A,B,C) = A + ABC$$

$\downarrow$ B, C

$$F(A,B,C) = A \cdot 1 + ABC$$

$B + \bar{B} = 1, \quad C + \bar{C} = 1$

$$= A(B+\bar{B})(C+\bar{C}) + ABC$$

$$= (AB + A\bar{B})(C+\bar{C}) + ABC$$

$$= ABC + AB\bar{C} + A\bar{B}C + ABC + A\bar{B}\bar{C}$$

$$F(A,B,C) = ABC + AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C}$$

Standard POS forms:-

step1 :- find the missing literals in each sum term if any.

step2 :- 'OR' each sum term having missing literals with the terms form by 'AND' ing the literals and its complement.

step3 :- Expand the term by applying distributive law and re order the literals in sum terms.

Step 4:- Reduce the expression by omitting repeated sum terms if any. $(A \cdot A = A)$

Example: Convert the given expression in standard pos form :-

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

Solution: Given function is $F(A,B,C) = (A+B)(B+C)(A+C)$

Step: 1 find the missing literals in the function.

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

$\downarrow C \quad \downarrow A \quad \downarrow B$

Missing terms.

Step: 2 $F(A, B, C) = (A+B+0)\ (B+C+0)\ (A+C+0)$

According to complements law $C \cdot \bar{C} = 0$

$= (A+B+C \cdot \bar{C})\ (B+C+A \cdot \bar{A})\ (A+C+B \cdot \bar{B})$

Step: 3 Distributive law $A+B \cdot C = (A+B)(A+C)$

$F(A,B,C) = (A+B+C)(A+B+\bar{C})\ (B+C+A)(B+C+\bar{A})(A+C+B)\ (A+C+\bar{B})$

Step: 4 Reduce the expression by omitting repeated sum terms

$F(A,B,C) = (A+B+C)\ (A+B+\bar{C})\ (B+C+\bar{A})\ (A+C+\bar{B})$

Example: Convert the given expression in standard pos form :-

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

Solution: $F(A,B,C) = A(A+B+C)$

↓

B, C missing

$$F(A,B,C) = (A+0+0)(A+B+C)$$

$$= (A+B\cdot\bar{B}+C\cdot\bar{C})(A+B+C)$$

$$= (A+B\cdot\bar{B}+C)(A+B\cdot\bar{B}+\bar{C})(A+B+C)$$

$$= (A+C+B\cdot\bar{B})(A+\bar{C}+B\cdot\bar{B})(A+B+C)$$

$$= (A+C+B)(A+C+\bar{B})(A+\bar{C}+B)(A+\bar{C}+\bar{B})(A+B+C)$$

$$F(A,B,C) = (A+B+C)(A+C+\bar{B})(A+\bar{C}+B)(A+\bar{B}+\bar{C})$$

Minterms and MaXterms Notation (M Notations)

* Each individual term in standard sop form is called Min term and in standard pos form is Max term.
* This concept allow us to introduce a convinient short hand notation to express logical functions.
* In general for an n variable logical function there are 2^n minterms and an equal no. of max terms.
* Minterms are represented in 'm' and max terms with 'M'

Variables				Minterm	Maxterm
				m_i	M_i
	A	B	C		
0	0	0	0	$\bar{A}\,\bar{B}\,\bar{C} = m_0$	$A+B+C = M_0$
1	0	0	1	$\bar{A}\,\bar{B}\,C = m_1$	$A+B+\bar{C} = M_1$
2	0	1	0	$\bar{A}\,B\,\bar{C} = m_2$	$A+\bar{B}+C = M_2$
3	0	1	1	$\bar{A}\,B\,C = m_3$	$A+\bar{B}+\bar{C} = M_3$
4	1	0	0	$A\,\bar{B}\,\bar{C} = m_4$	$\bar{A}+B+C = M_4$
5	1	0	1	$A\,\bar{B}\,C = m_5$	$\bar{A}+B+\bar{C} = M_5$
6	1	1	0	$A\,B\,\bar{C} = m_6$	$\bar{A}+\bar{B}+C = M_6$
7	1	1	1	$A\,B\,C = m_7$	$\bar{A}+\bar{B}+\bar{C} = M_7$

Example: $F(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}BC + AB\bar{C}$ convert into min term notation.

Solution:

$$F(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}BC + AB\bar{C}$$

$$= 000 \quad 001 \quad 011 \quad 110$$

$$F(A,B,C) = m_0 + m_1 + m_3 + m_6$$

$$F(A,B,C) = \sum (m_0, m_1, m_3, m_6)$$

$$= \sum m(0,1,3,6)$$

Example: Convert into Maxterm notation

$$F(A,B,C) = (A+B+\bar{C})\ (A+\bar{B}+\bar{C})\ (\bar{A}+\bar{B}+C)$$

Solution:

$$F(A,B,C) = (A+B+\bar{C})\ (A+\bar{B}+\bar{C})\ (\bar{A}+\bar{B}+C)$$

$$= (0+0+1)\ (0+1+1)\ (1+1+0)$$

$$= M_1\ M_3\ M_6$$

$$= \Pi(M_1, M_3, M_6)$$

$$F(A,B,C) = \Pi M(1,3,6)$$

Note :- Σ denotes sum of products, Π denotes products of sums.

Example:

Truth Table

A	B	C	y	
0	0	0	0	} max
0	0	1	0	
0	1	0	1	} min
0	1	1	1	
1	0	0	0	} max
1	0	1	0	
1	1	0	1	} min
1	1	1	1	

* The logical expression corresponding to a given truth table can be written in a standard sop from by writing one product term for each input combination that produces an output of 1.

* The logical expression corresponding to a given truth table can be written in standard pos form by writing one Sum term for each output of 0.

* The logical expression corresponding to a given truth table can be written in a standard SOP from by writing one product term for each input combination that produces an output of 1.

* The logical expression corresponding to a given truth table can be written in standard POS form by writing one Sum term for each output of 0.

Truth Table

A	B	C	y	
0	0	0	0	} max
0	0	1	0	
0	1	0	1	} min
0	1	1	1	
1	0	0	0	} max
1	0	1	0	
1	1	0	1	} min
1	1	1	1	

Minterms :-

$F(A,B,C) = 010 + 011 + 110 + 111$

$= \bar{A}B\bar{C} + \bar{A}BC + AB\bar{C} + ABC$

$= m_2 + m_3 + m_6 + m_7$

$F(A,B,C) = \sum m(2,3,6,7)$

Max terms :-

$F(A,B,C) = (0+0+0)(0+0+1)(1+0+0)(1+0+1)$

$= (\bar{A}+B+C)(A+B+\bar{C})(\bar{A}+B+C)(\bar{A}+B+\bar{C})$

$= M_0 \cdot M_1 \cdot M_4 \cdot M_5$

$F(A,B,C) = \pi M(0,1,4,5)$

Example:

Truth Table

A	B	C	y	
0	0	0	1	min
0	0	1	1	min
0	1	0	0	max
0	1	1	0	max
1	0	0	1	min
1	0	1	1	min
1	1	0	0	max
1	1	1	0	max

Min terms :-

$F(A,B,C) = 000 + 001 + 010 + 101$

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

$= m_0 + m_1 + m_4 + m_5$

$= \Sigma m(0,1,4,5)$

$F(A,B,C) = \Sigma m(0,1,4,5).$

Max terms :-

$F(A,B,C) = (0+1+0)(0+1+1)(1+1+0)(1+1+1)$

$= (A+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})$

$= M_5 \cdot M_3 \cdot M_6 \cdot M_7$

$F(A,B,C) = \Pi M(2,3,6,7)$

Conversion between Canonical forms

Conversion b/n canonical forms :-

* To convert from one to another canonical form interchanging the symbols Σ to Π and those numbers are from the original form. (Missing terms).

∴ Total no. of min terms or max terms $= 2^n$

Where n is the no. of variables.

Example: If $F(x,y,z) = \Pi M(0,1,3,5) 7)$

$F(x,y,z) = \Sigma m(2,4,6) \longrightarrow$ min term

Example: Simplify the 3 variable expression using Boolean algebra and convert into canonical form. $y = \Sigma m(1,3,5,7)$

Solution: Given function is $y = \Sigma m(1,3,5,7)$

$y = \Sigma m(1,3,5,7)$

$= m_1 + m_3 + m_5 + m_7$

$= 001 + 011 + 101 + 111$

$= \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC$

$= BC(A + \bar{A}) + \bar{A}\bar{B}C + A\bar{B}C$

$= BC + \bar{B}C(A + \bar{A})$

$= BC + \bar{B}C = C(B + \bar{B}) = C$

$y = C \Rightarrow \boxed{y(A,B,C) = C}$

$Y = \Sigma m(1,3,5,7)$

$Y = \Pi M(0,2,4,6) \Rightarrow$ max terms

$= M_0 \cdot M_2 \cdot M_4 \cdot M_6.$

$Y = (A+B+C)(A+\bar{B}+C)(\bar{A}+B+C)(\bar{A}+\bar{B}+C)$

In POS form

$Y(A,B,C) = C$

$\downarrow$

A, B missing

$Y(A,B,C) = C+0+0$

$= (C + A \cdot \bar{A}) + B \cdot \bar{B}$

$= (C+A \cdot \bar{A} + B)(C+A \cdot \bar{A} + \bar{B})$

$$= ((C+B)+A\cdot\bar{B})\,((C+\bar{B})+A\cdot\bar{A})$$

$$= (C+B+A)(C+B+\bar{B})(C+\bar{B}+A)(C+\bar{B}+\bar{A})$$

$$Y(A,B,C) = (A+B+C)(\bar{A}+B+C)(A+\bar{B}+C)(\bar{A}+\bar{B}+C)$$

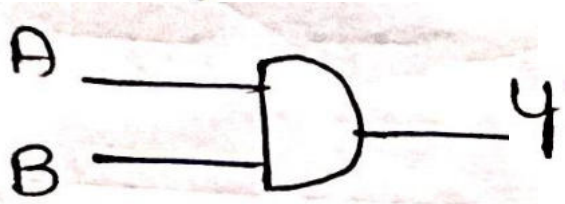
Example: Simplify the 3 variable expression using Boolean algebra and convert into canonical form.

$$F(x,y,z) = \pi M(0,3,6,7)$$

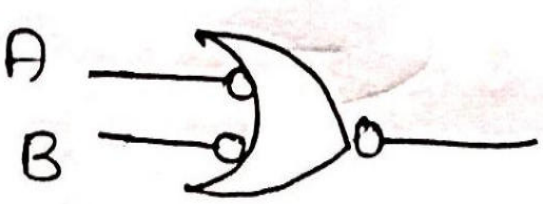
Alternate Gate Representation

The five basic logic gates NAND, NOR, NOT, OR, AND and their standard symbols are used to represent them on logic network diagram. In some networks an alternative set of symbols is used in addition to the standard logic symbols. These alternative symbols are equivalent to the standard symbols and their equivalents can be proved by using Demorgans theorem.

AND Gate:-



$Y = AB \equiv$

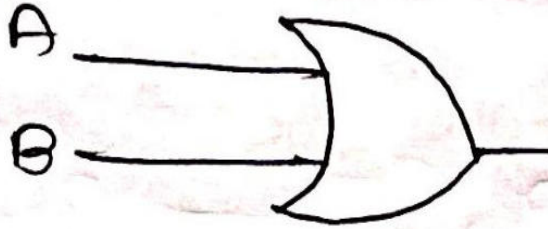


$$Y = \overline{\bar{A} + \bar{B}}$$

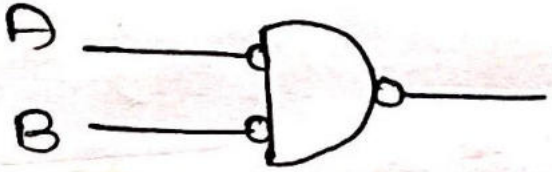
$$= \overline{\bar{A} + \bar{B}} = \bar{\bar{A}} \cdot \bar{\bar{B}}$$

$$\boxed{Y = AB}$$

OR Gate:-



$Y = A + B \equiv$

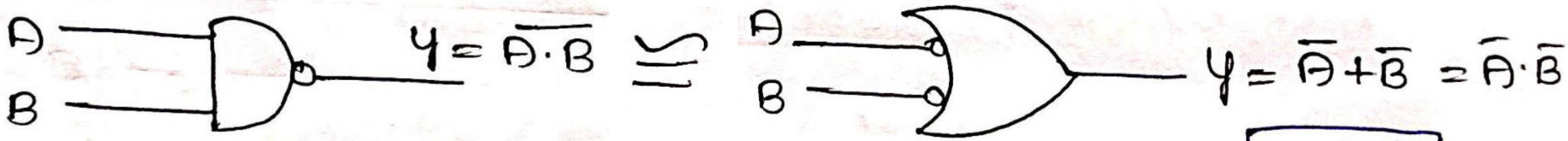


$$Y = \overline{\bar{A} \cdot \bar{B}}$$

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

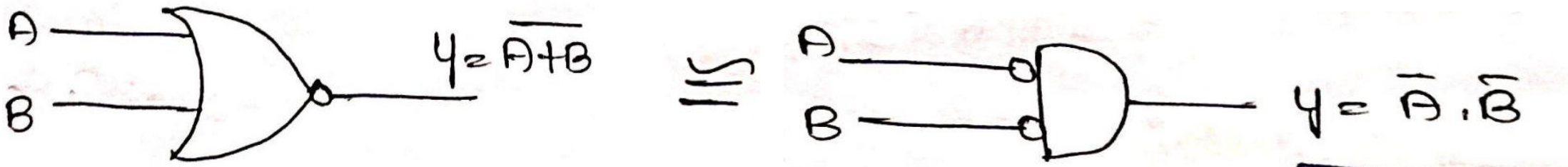
$$\boxed{Y = A + B}$$

NAND Gate :-



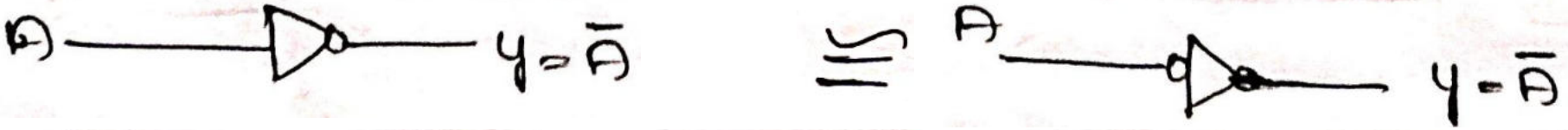
$$Y = \overline{AB}$$

NOR Gate :-



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

NOT Gate :-



NOTE :- The equivalents of standard and alternate symbols are valid for gates with any no. of inputs.

Conversion of AND/ OR/ NOT logic to NAND/NOR logic:-

Steps for Converting to NAND or NOR logic:-

Step : 1 Draw AND | OR | NOT logic.

Step : 2 If NAND hardware has been choosen Add bubbles on the output of each AND gate and bubbles on input side of OR gates.

Step : 3 If NOR hardware has been choosed add bubbles on output side of OR gate and input side of each AND gate.

Step : 4 Add or substract an inverter on each line that receive a bubble in step ② or step ③.

Step : 5 Replace bubbled OR. by NAND and bubbled AND by NOR logic.

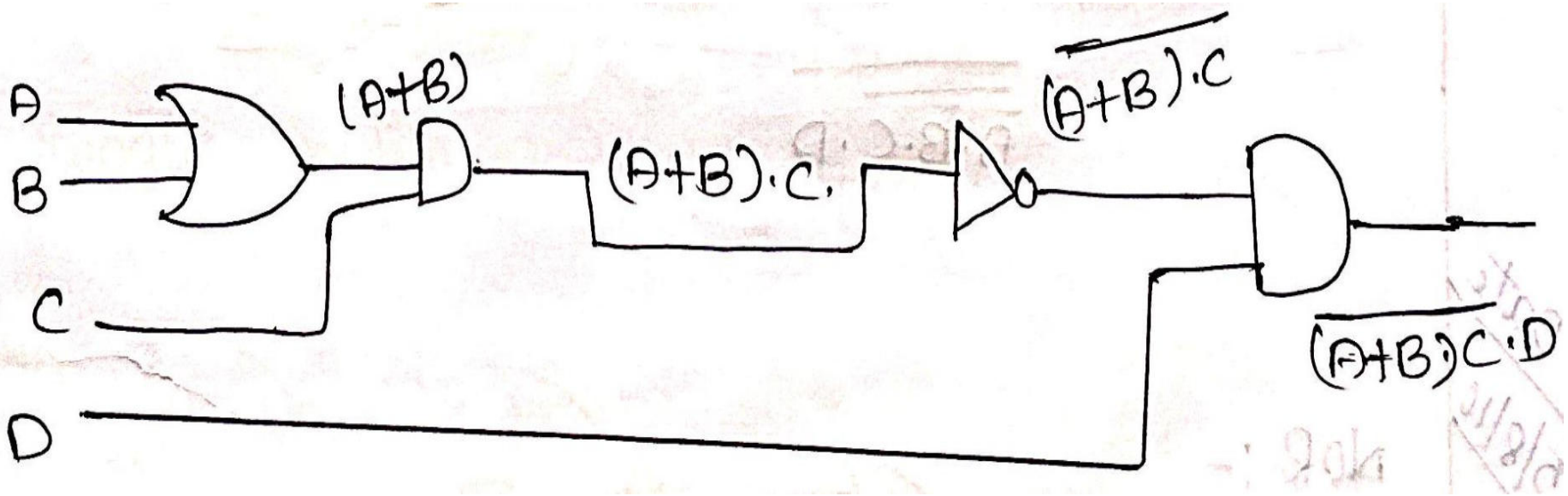
Step : 6 Eliminate double inversions and replace not gates by NAND / NOR logic.

Example: Implement boolean expression $\overline{(A+B)\cdot C}\cdot D$ only NAND and NOR gates.

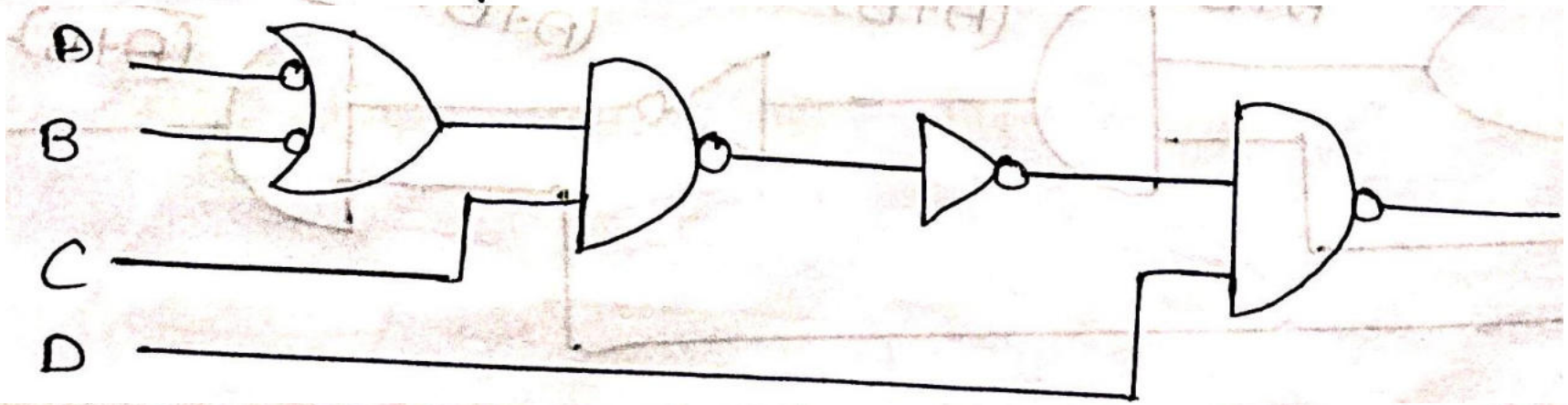
Solution: Given function is $\overline{(A+B)\cdot C}\cdot D$

Implementation using NAND

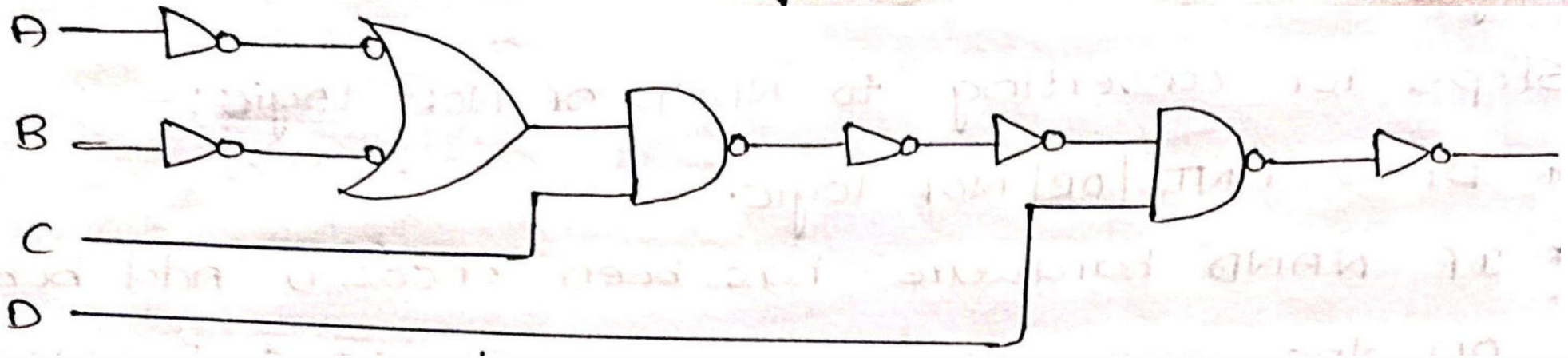
Step : 1 Draw AND or OR gates



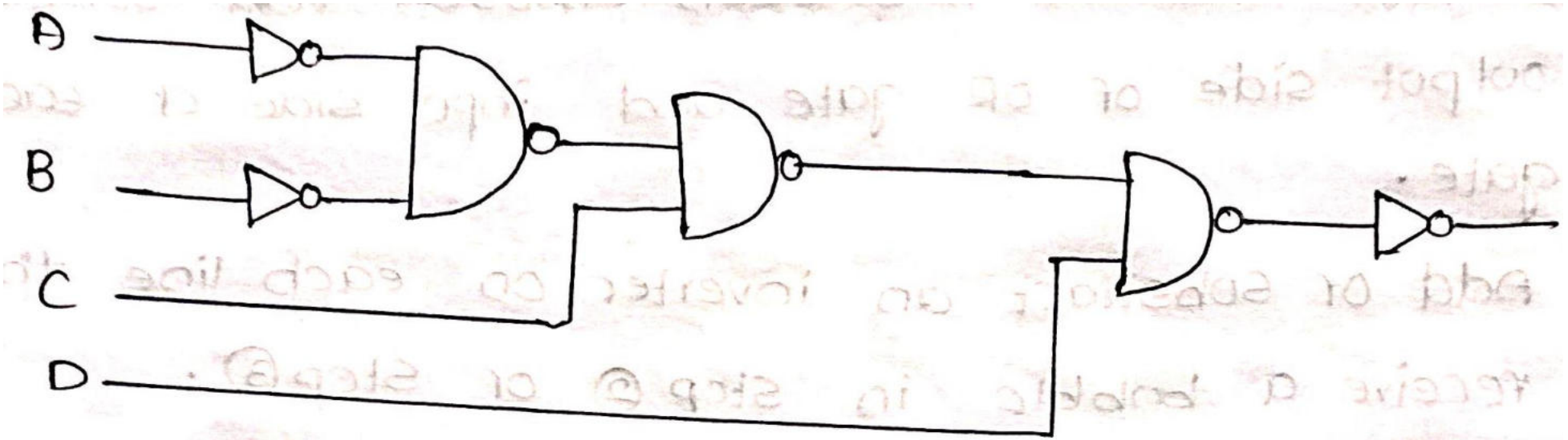
Step : 2 Add bubbles to o/p to each AND and i/p of each OR gates.



Step : 3 Add or Substract Not gate for each bubble line.

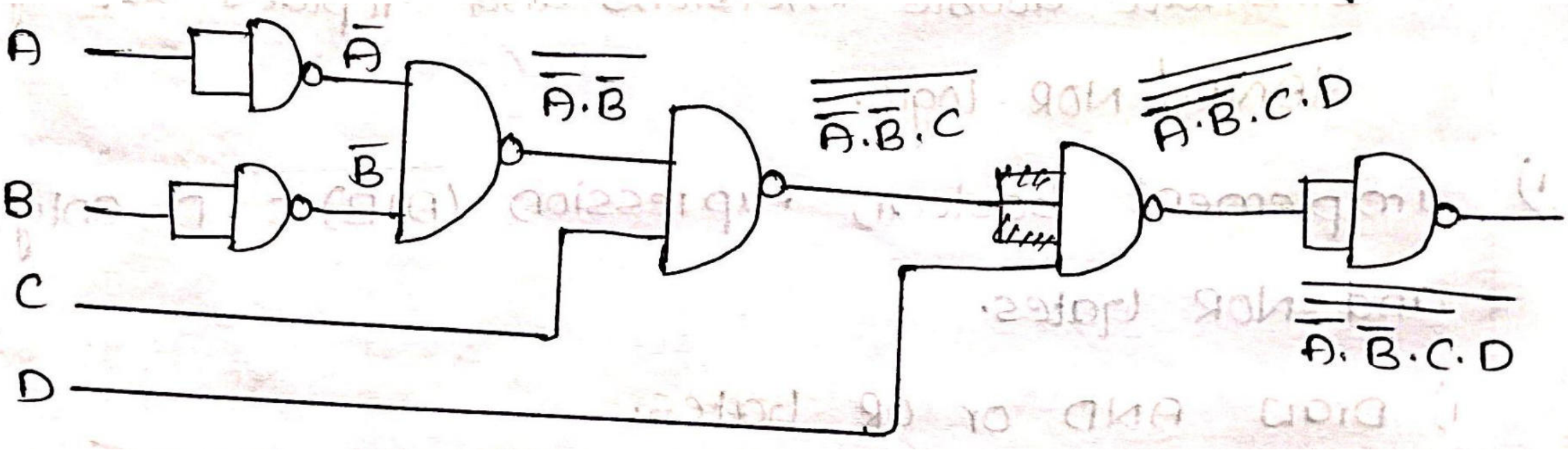


Step : 4 Replace bobbled gate with NAND gate



Step : 5 eliminate doubled inversion gates

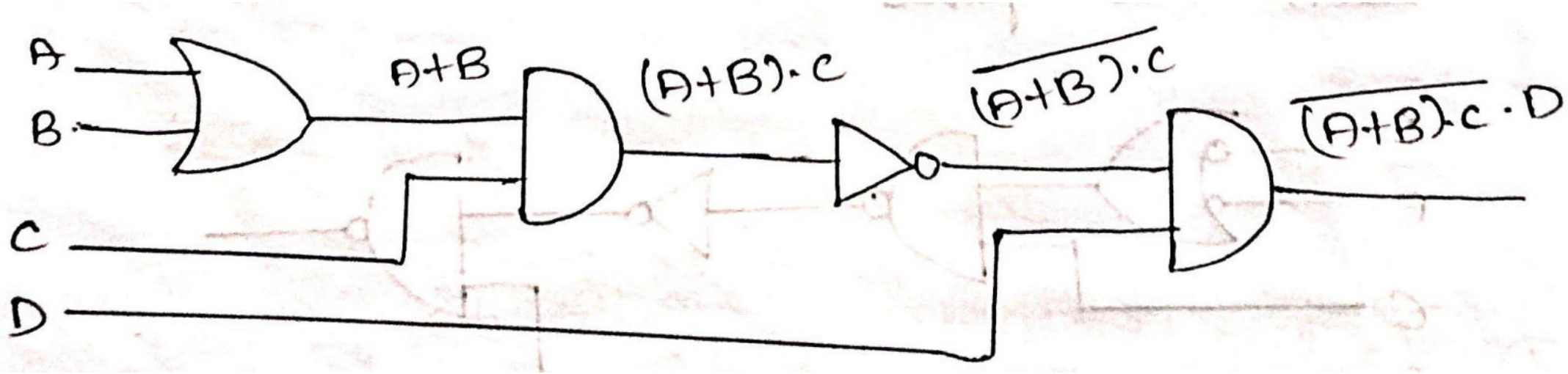
Step : 6 Each NOT gate represent in the form of NAND gate



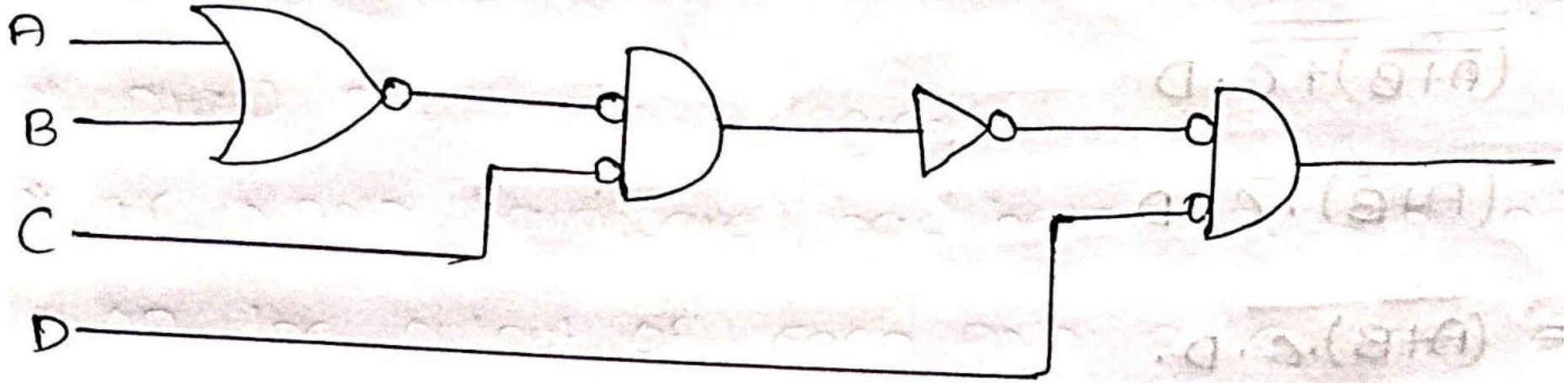
Implementation using NOR

Solution: Given function is $\overline{(A+B)\cdot C}\cdot D$

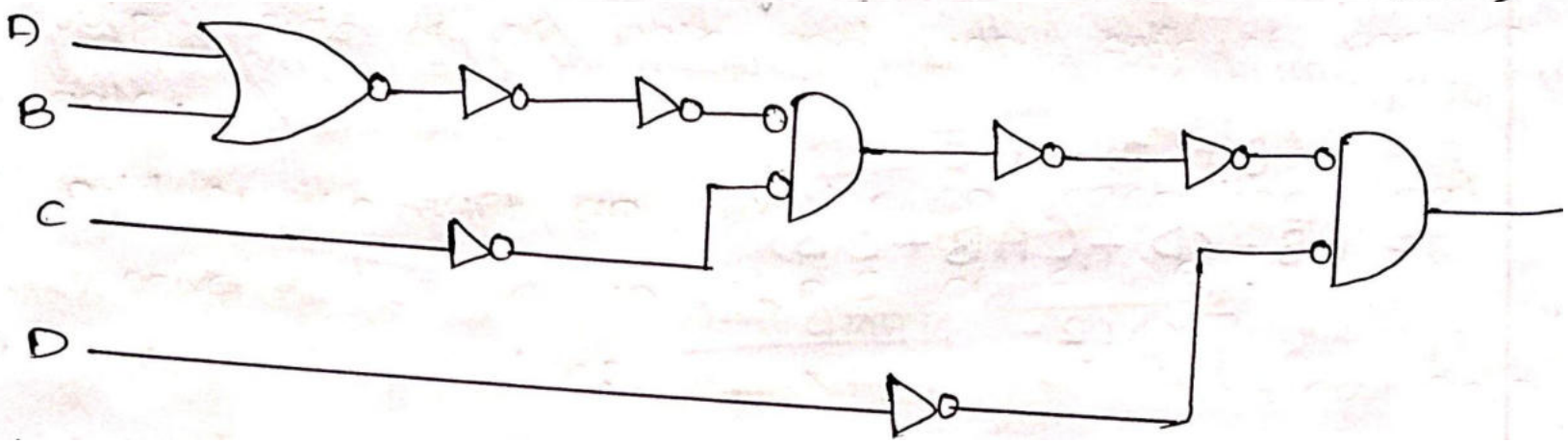
Step : 1 Draw AND or OR gates



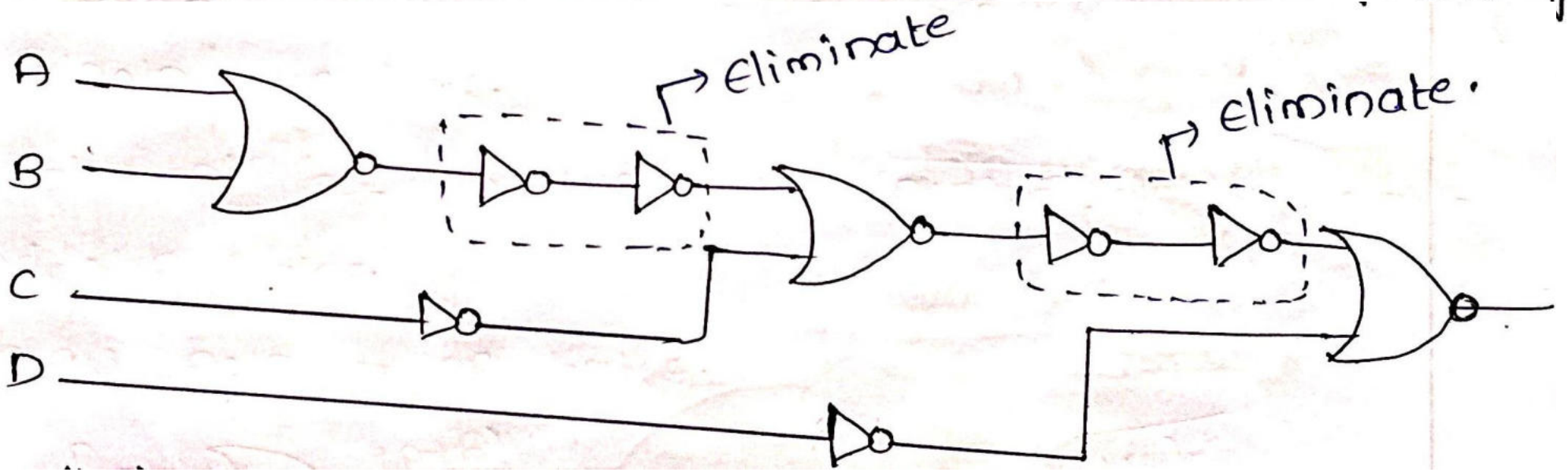
Step : 2 Add bubbles to i/p side of AND and o/p side of OR gate.



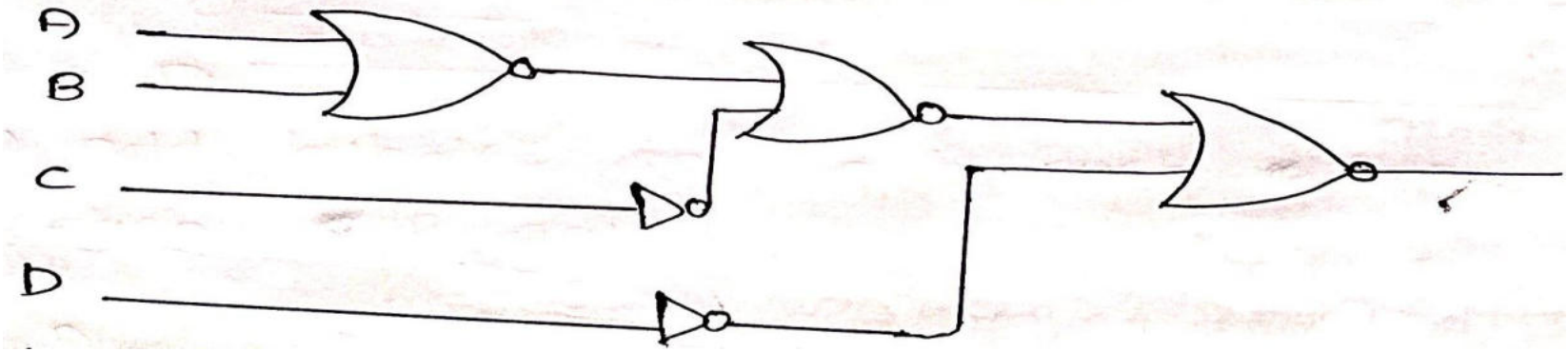
Step : 3 Add or subtract an inverter for each bubble line.



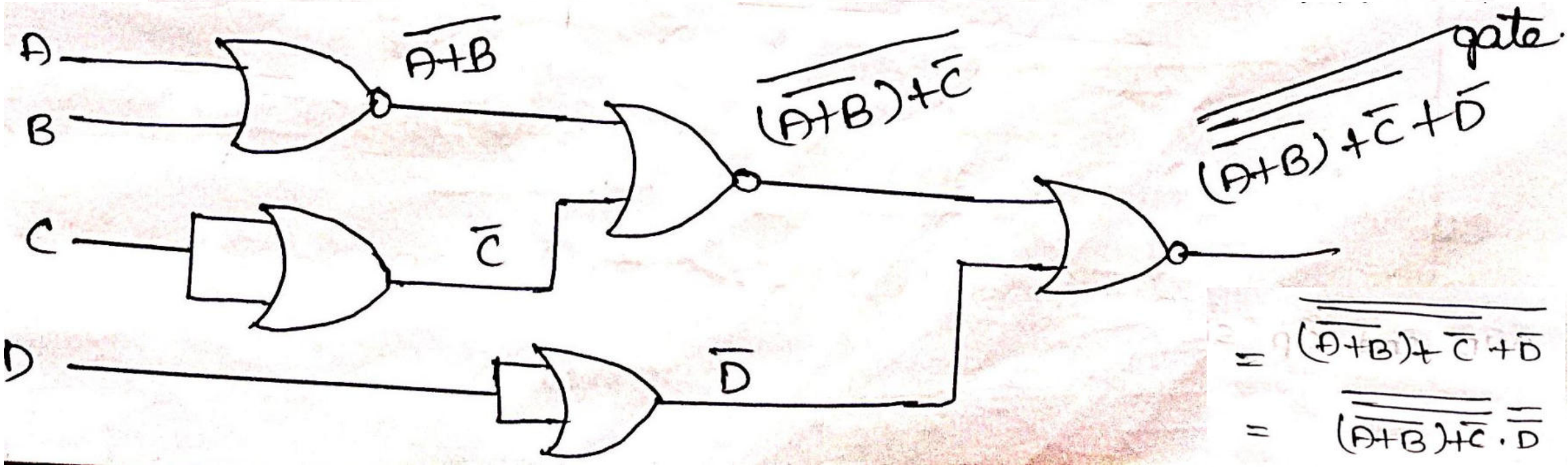
Step : 4 Replace bubbled AND gate interms of NOR gate



Step : 5 eliminate doubled inversion gates



Step : 6 Each NOT gate is represent in the form of NOR gate.



$$= \overline{\overline{(A+B)+\overline{C}}+\overline{D}}$$

$$= \overline{\overline{(A+B)+\overline{C}}}\cdot\overline{D}$$

$$= \overline{\overline{(A+B)+\overline{C}}}\cdot D$$

$$= \overline{(A+B)\cdot C}\cdot D$$

$$= \overline{(A+B)\cdot C\cdot D}.$$

Workout Problems

Example:1 Implement Boolean expression for exclusive OR gate (XOR gate) using NAND and NOR gates.

Example:2 Implement the following Boolean function using only **NAND** gates

$$F = A(B + CD) + B\bar{C}$$

$$Y = A\bar{B} + CD + C(\bar{A}B + \bar{C}\bar{D})$$

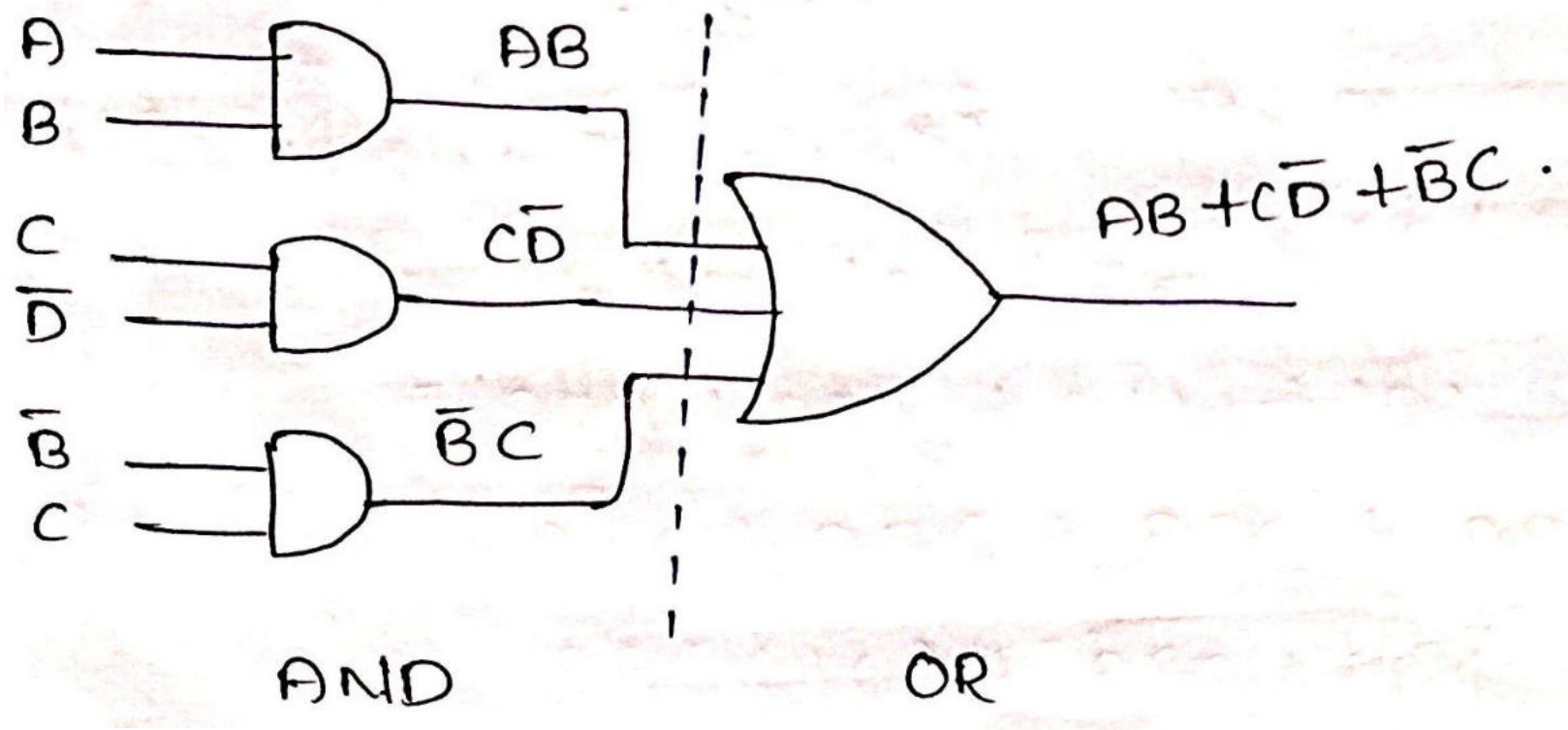
Example:3 Implement Boolean expression for **XNOR** gate using only **NAND & NOR** gates

Implementation of logic functions using gates

SOP form (AND-OR)

Example: $F = AB + C\bar{D} + \bar{B}C$

Solution:

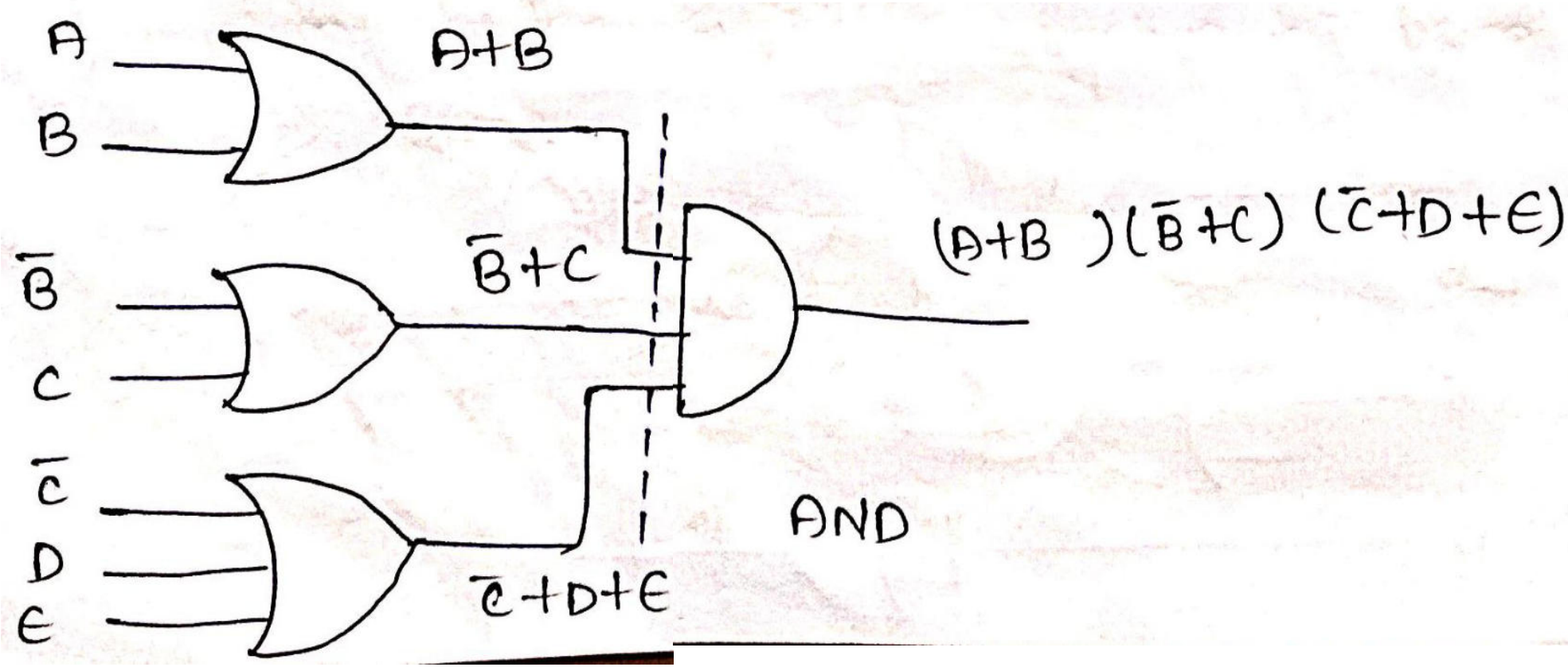


POS form (OR-AND)

Example

$F = (A+B)(\bar{B}+C)(\bar{C}+D+E)$

Solution:



SOP form (NAND - NAND):

NAND-NAND Implementation:-

Step : 1 Simplify the given Boolean function and express it in terms of sum of products (SOP) and implement using NAND-OR logic.

Step : 2 Draw a NAND gate for each product term of the function that has 2 or more literals. The inputs to each NAND gate are the literals of the term.

Step : 3 The Boolean function includes any single literal or literals draw NAND gate for each single literal and connect corresponding literals as a input to NAND gate

Step : 4 Draw a single NAND gate in the second level with inputs coming from outputs of first level gates.

Example: Implement the following Boolean functions with NAND-NAND logic. $F = AC + ABC + \bar{A}BC + AB + D$

Solution: Given function is $F = AC + ABC + \bar{A}BC + AB + D$

Step : 1 simplify the given function

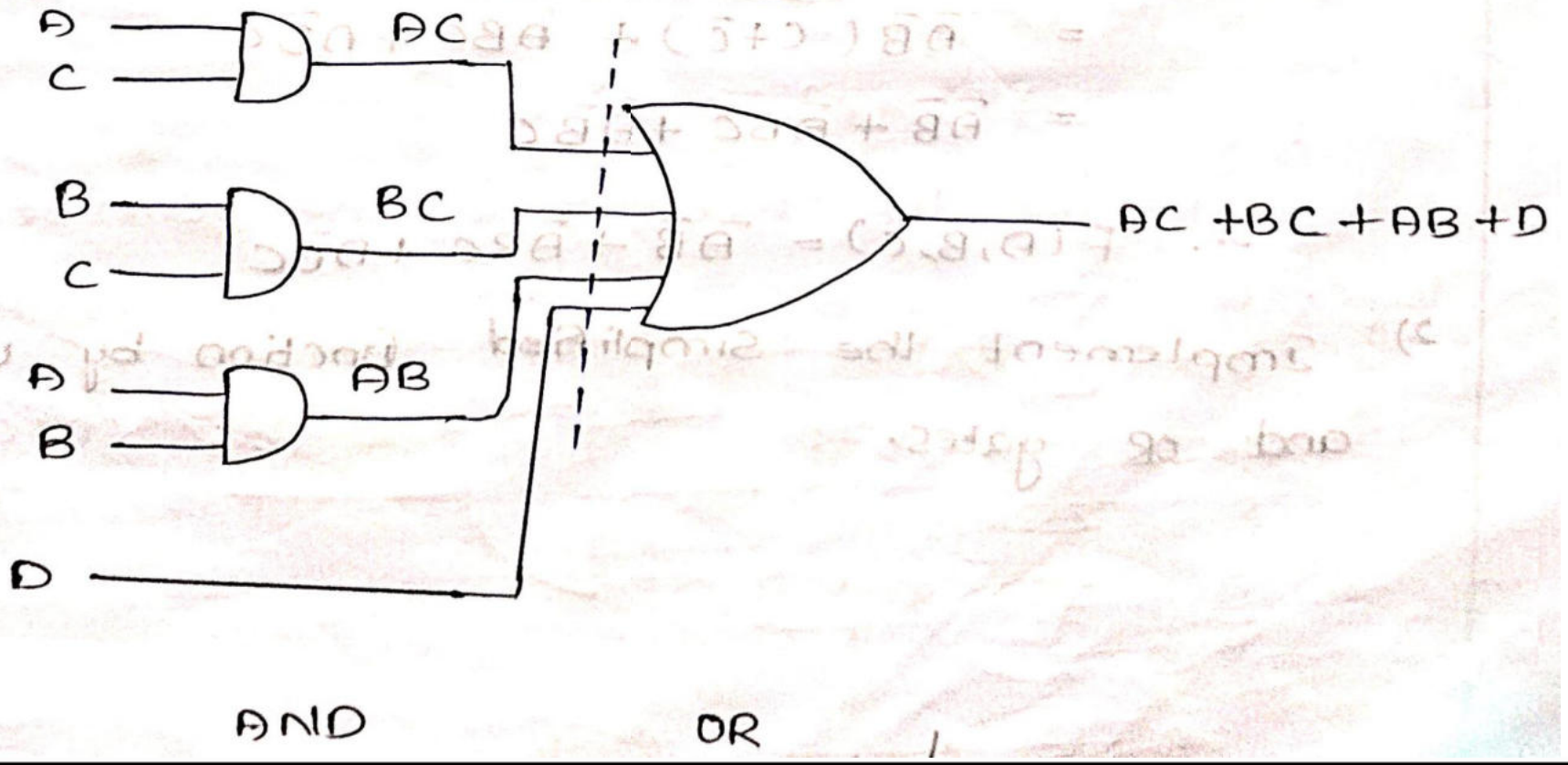
$$F = AC + ABC + \bar{A}BC + AB + D$$

$AC + A\bar{A}BC + \bar{A}BC + AB + D$

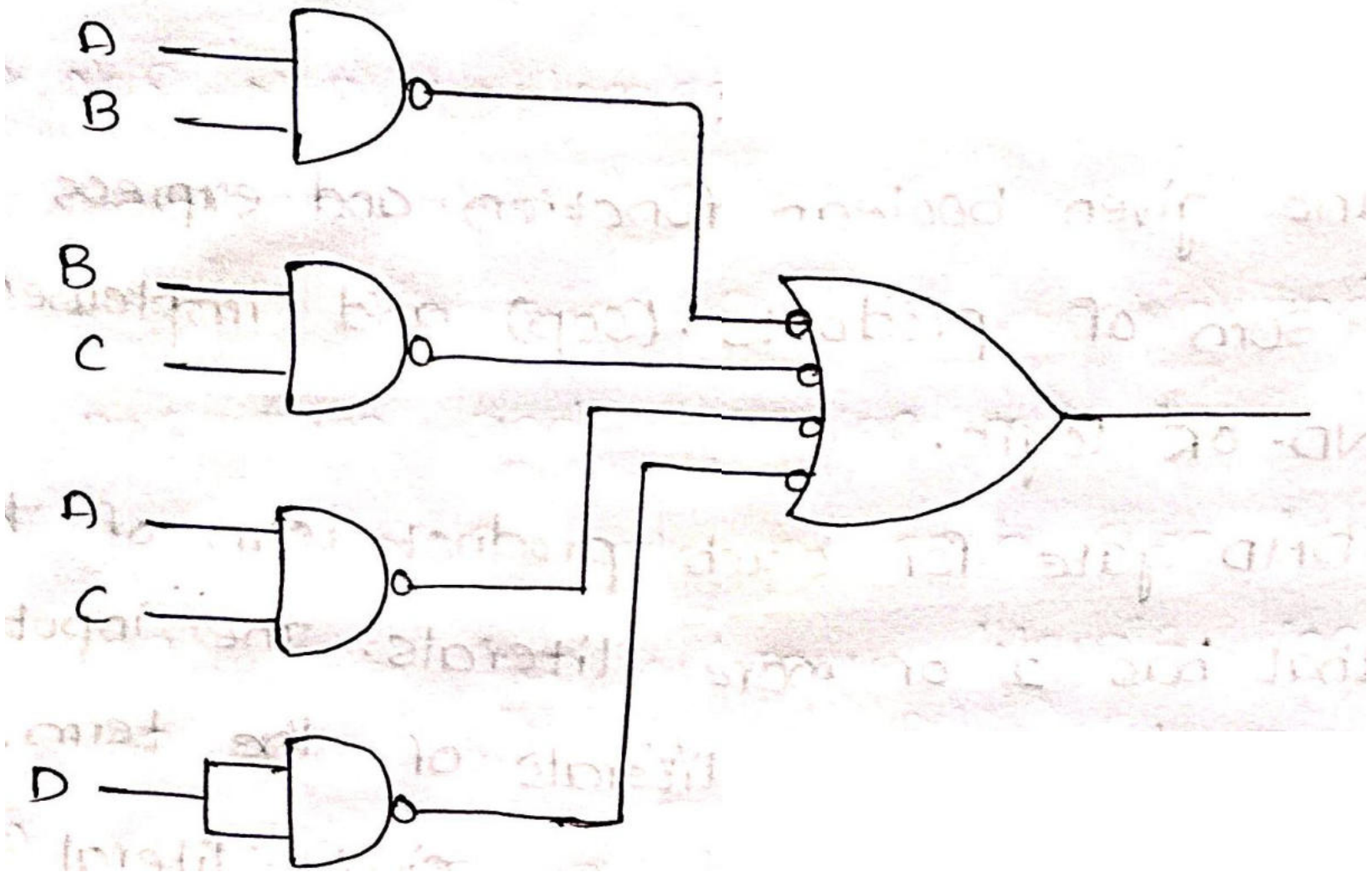
$AC + (A + \bar{A})BC + AB + D$

$AC + BC + AB + D$.

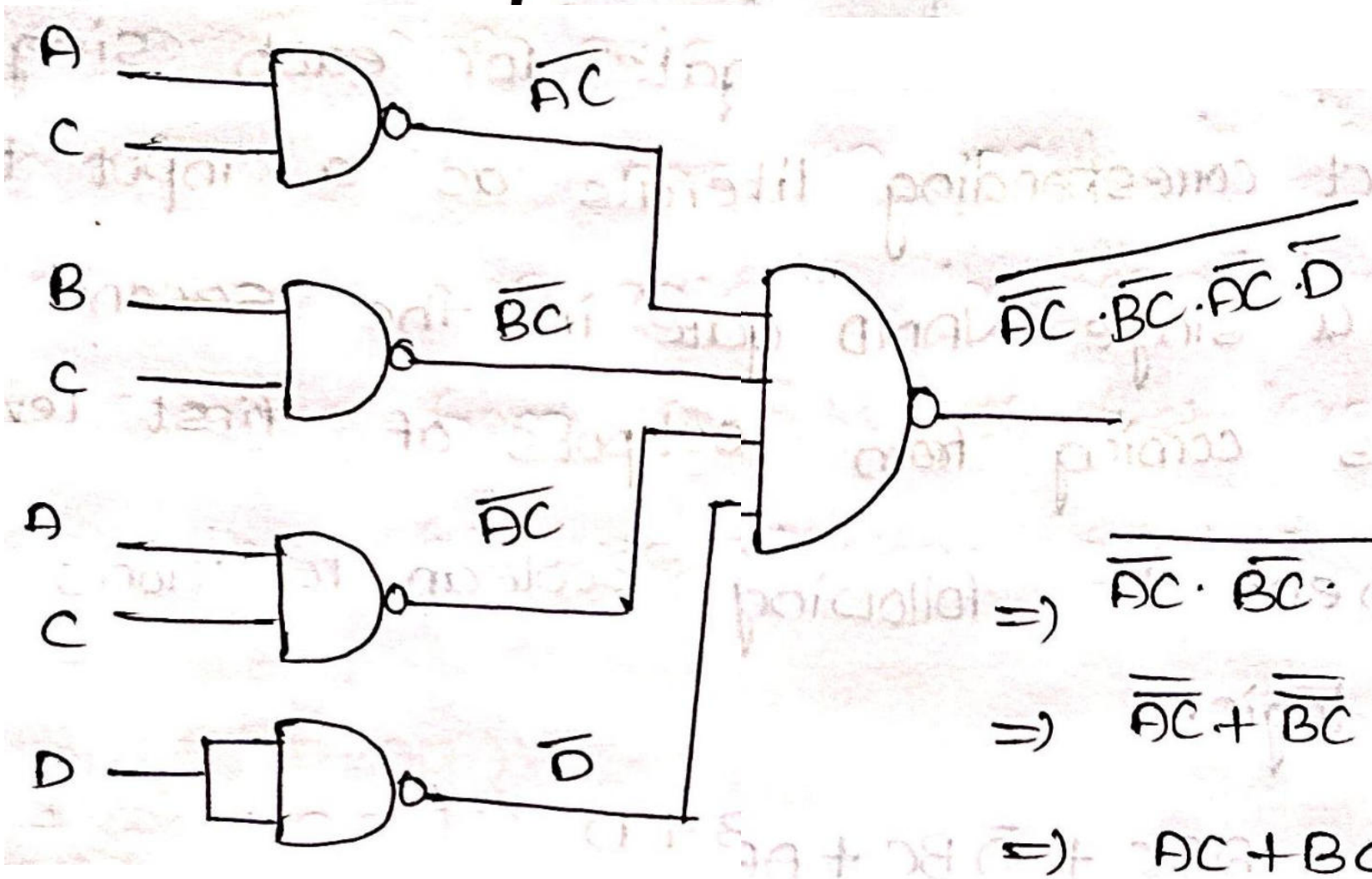
Step : 2 Implement by using AND-OR logic



Step : 3 Implement AND-OR gate in NAND - NAND gate.



Step : 4 Replace OR gate with NAND gate.



$$\Rightarrow \overline{\overline{AC} \cdot \overline{BC} \cdot \overline{AC} \cdot \overline{D}}$$

$$\Rightarrow \overline{\overline{AC}} + \overline{\overline{BC}} + \overline{\overline{AC}} + \overline{\overline{D}}$$

$$\Rightarrow AC + BC + AC + D.$$

POS form (NOR - NOR):

NOR-NOR Implementation:-

Note: NOR - NOR Implementation is the duals of NAND NAND Implementation.

Step : 1 Simplify the given Boolean function and express it in pos form and draw OR - AND logic.

Step : 2 Draw a NOR gate for each sum term of the function that has two or more literals. The input to each NOR gate are literals of the term.

Step : 3 if Boolean function includes any single literal or literals. Draw NOR gate for each single literal and corresponding literal has an input to the NOR gate.

Step : 4 Draw a single NOR gate in the second level with the inputs coming from outputs of first level gates.

Example: Implement the following Boolean function with NOR NOR logic. $Y = AC + BC + \bar{A}B + D$

Solution: Given function is $Y = AC + BC + \bar{A}B + D$

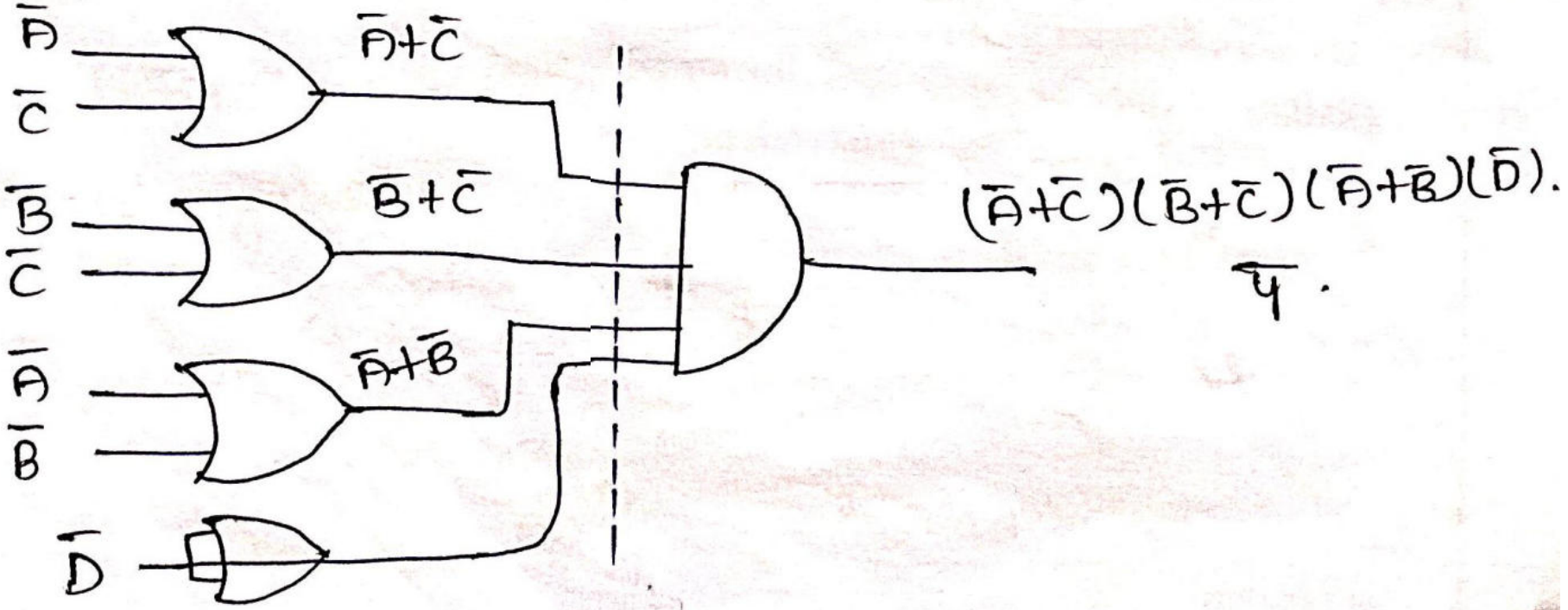
Given Boolean expression is in the form of SOP form convert SOP form into POS form i.e (By taking complement).

$$\bar{Y} = \overline{AC + BC + AB + D}$$

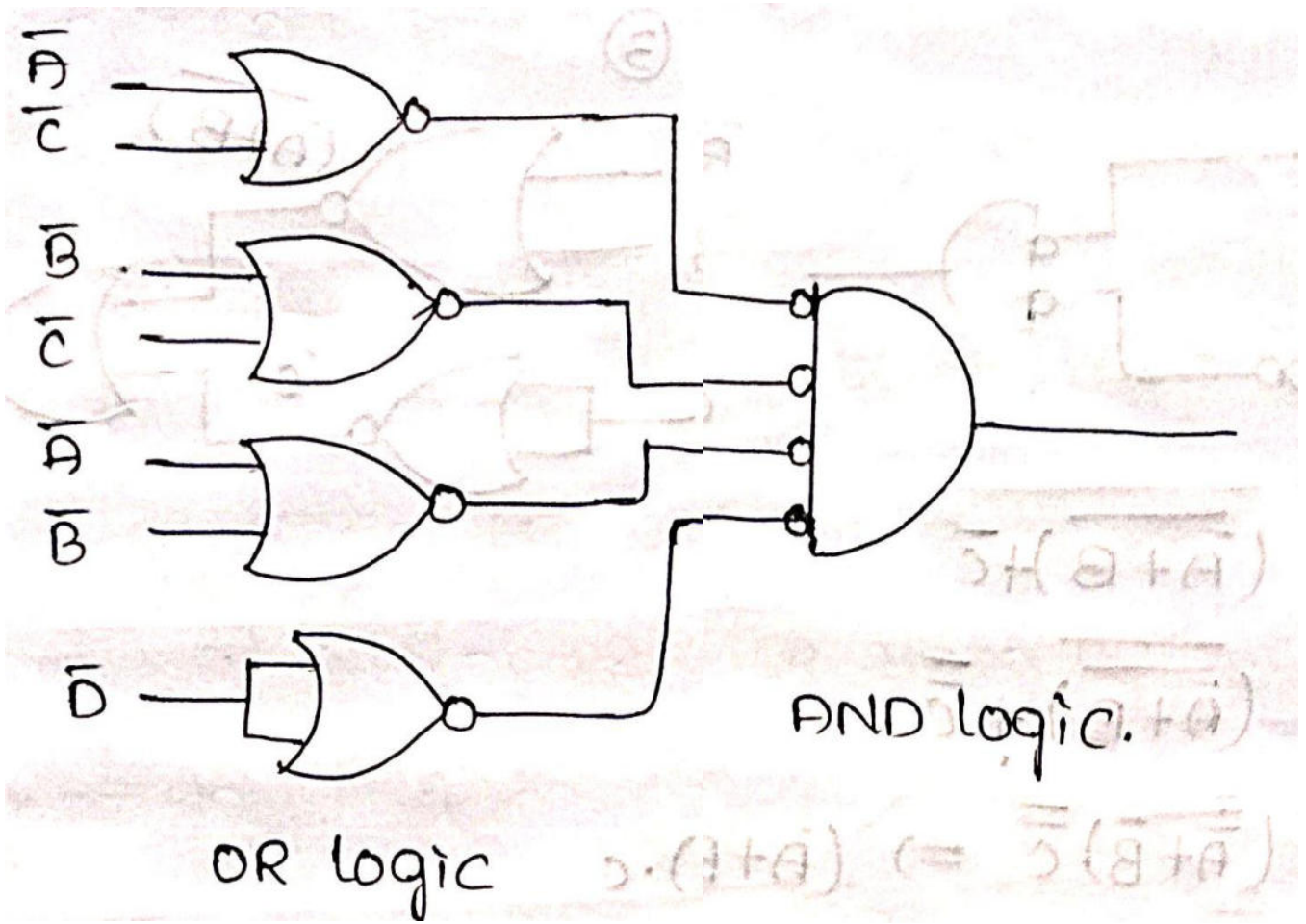
$$= \bar{A}C \cdot \bar{B}C \cdot \bar{A}B \cdot \bar{D}$$

$$\bar{Y} = (\bar{A}+\bar{C})(\bar{B}+\bar{C})(\bar{A}+\bar{B}) \cdot \bar{D}$$

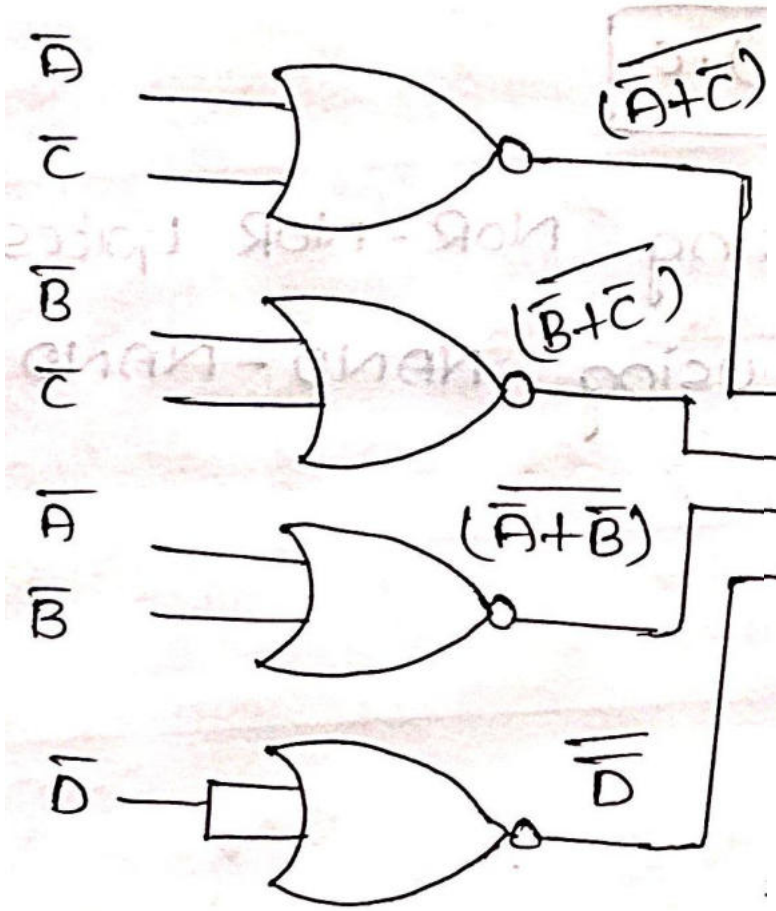
Step : 1 Implement the function in the form of OR-AND logic.



Step : 2



Step : 3



$(\overline{\overline{A}+\overline{C}})(\overline{\overline{B}+C})+(\overline{\overline{A}+\overline{B}})D$

$\overline{Y}$ → Y

$$\Rightarrow \overline{(\overline{\overline{A}+\overline{C}})+(\overline{\overline{B}+\overline{C}})+(\overline{\overline{A}+\overline{B}})+D}$$

$$\Rightarrow (\overline{\overline{\overline{A}+\overline{C}}})(\overline{\overline{\overline{B}+\overline{C}}})(\overline{\overline{\overline{A}+\overline{B}}})+\overline{D}$$

$$\overline{Y} \Rightarrow (\overline{A}+\overline{C})(\overline{B}+\overline{C})(\overline{A}+\overline{B})\overline{D}$$

$$\overline{\overline{Y}} \Rightarrow \overline{(\overline{A}+\overline{C})(\overline{B}+\overline{C})(\overline{A}+\overline{B})\overline{D}}$$

$$\overline{\overline{Y}} = \overline{(\overline{A}+\overline{C})} + \overline{(\overline{B}+\overline{C})} + \overline{(\overline{A}+\overline{B})} + \overline{\overline{D}}$$

$$Y = \overline{\overline{A}}\cdot\overline{\overline{C}} + \overline{\overline{B}}\cdot\overline{\overline{C}} + \overline{\overline{A}}\cdot\overline{\overline{B}} + \overline{\overline{D}}$$

$$\therefore Y = AC + BC + AB + D.$$

Workout Problems

Example:1 Implement the following Boolean functions with NAND-NAND logic. $F(A,B,C) = \Sigma m(0,1,3,5)$

Example:2 Implement the following Boolean function with NOR NOR logic. $Y = (\bar{A}+B)\cdot C$

Example:3 Implement XOR gate using NOR-NOR Gates

Example:4 Implement XNOR gate using NAND-NAND gates

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

Topic

MINIMIZATION: K-MAP METHODS

OVERVIEW

- Minimization: K-Map methods
- Minimal SOP and POS forms
- Prime implicants, don’t care combinations
- Tabular Method, Prime-Implicant chart, simplification rules.

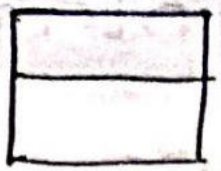
Map Method (K-Map)

1. Simplification of **Boolean function** is very important as it saves the **hardware require** and the **cost for** design of **Specific Boolean function**.
2. Map method gives us a **systematic approach** for simplifying **Boolean expression**.
3. It was first proposed by **veitch** and modified by **Karnaugh**. Hence it is known as **veitch diagram** or **karn diagram** or **map**.
4. The basics of this method is **a graphical chart** known as **K-map**. It contains boxes called **Cells**.
5. Each Cell can represents one of the 2^n possible products that can be formed from **n variables**.

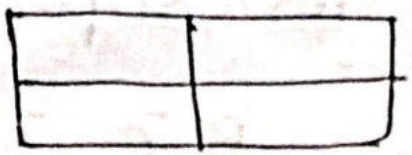
A one variable map Contains 2^1 = 2 cells

A two variable map contains 2^2 = 4 Cells.

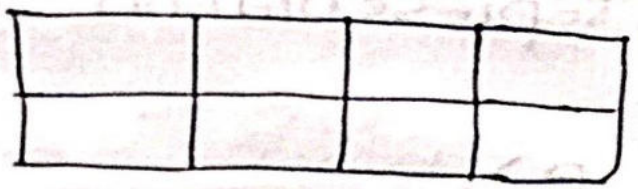
2^3 = 8 cells (**3 variable**) 2^4 = 16 cells (**4 variable**)



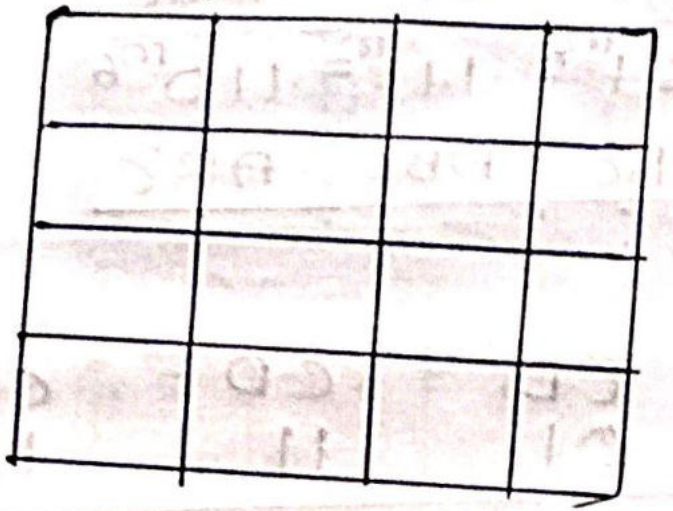
1 variable map
(2 cells)



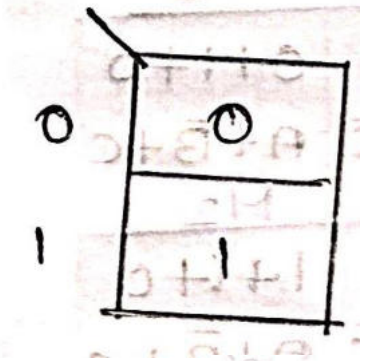
2 variable map
(4 cells)

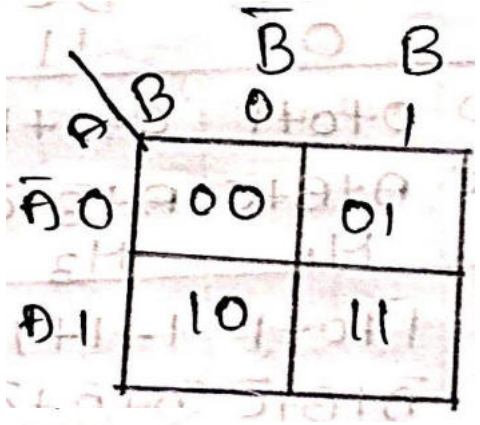


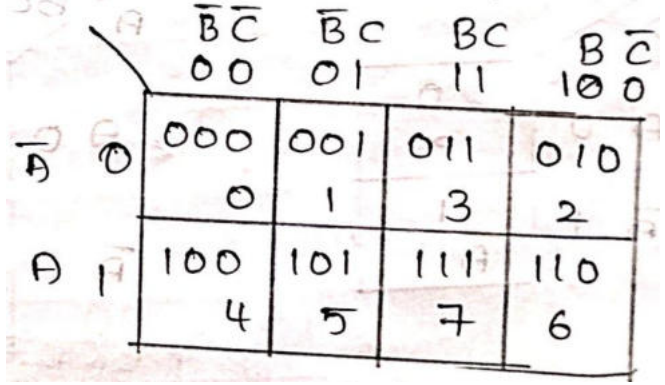
3 variable map
(8 cells).

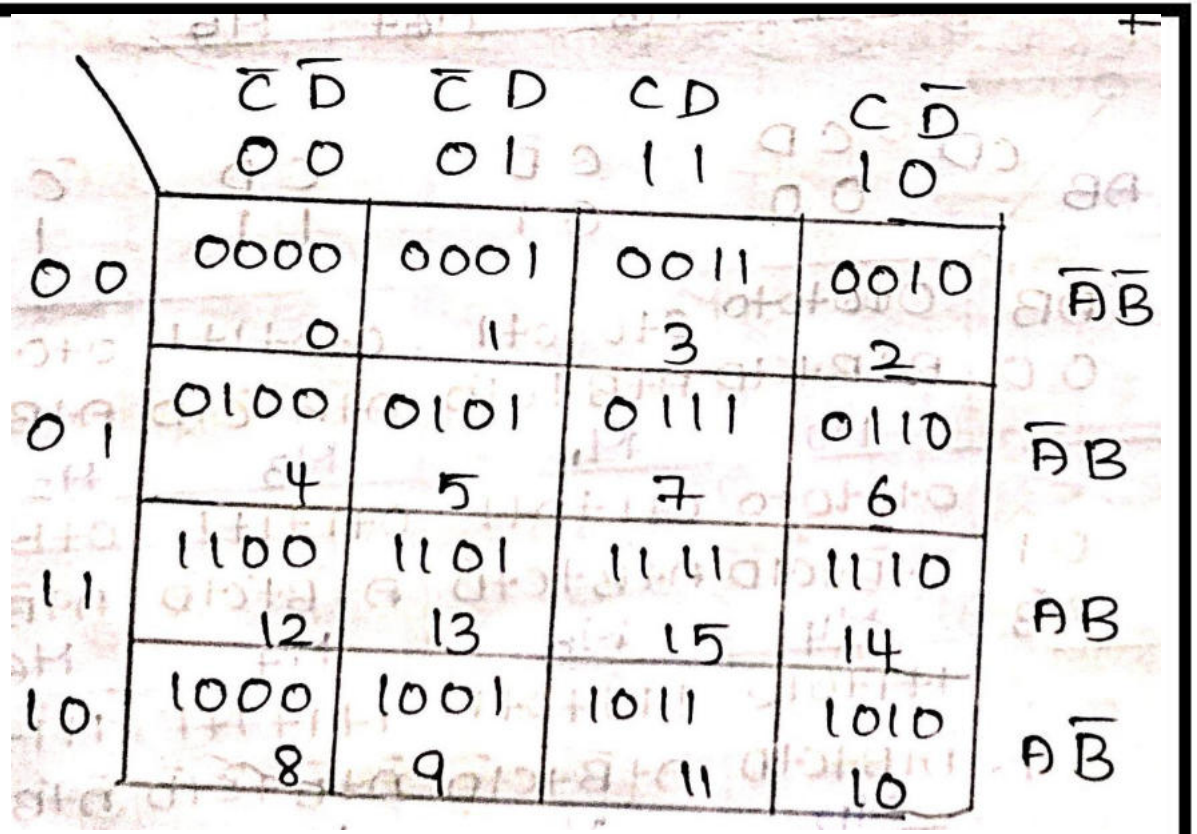


4 variable map
(16 cells)





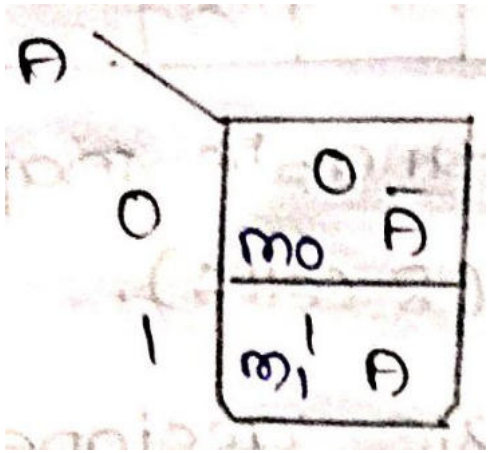


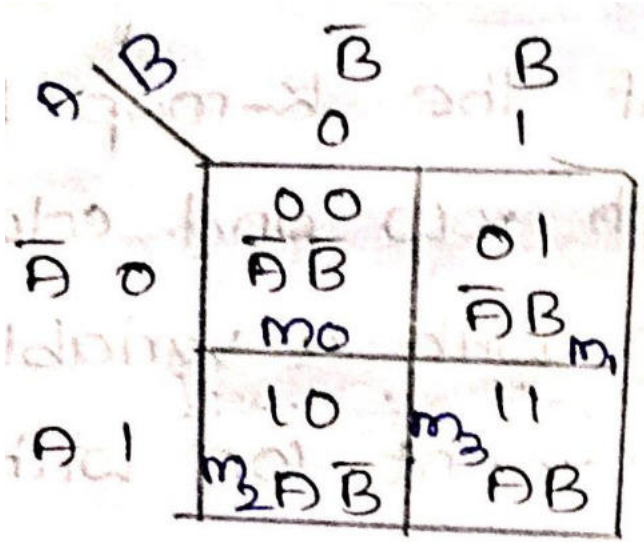


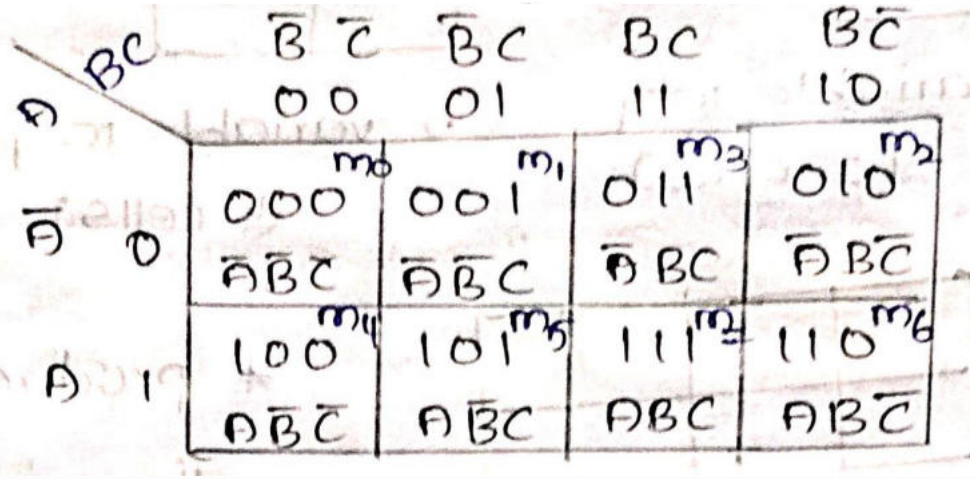
- **Product teams** are assigned to the **cells** of the **K-map** by labelling each row and column of the map with a **variable**, with its **Complement** (or) with a combination of **variables** and **complements.**
- It is **important** to note that when we move from **one cell** to the **next cell** along **any row** or **column**, one and only **one variable** changes in the **product term**.
- Irrespective of **no. of variables** the labels along any **row** or **column** must confirm to the **single change rule**.

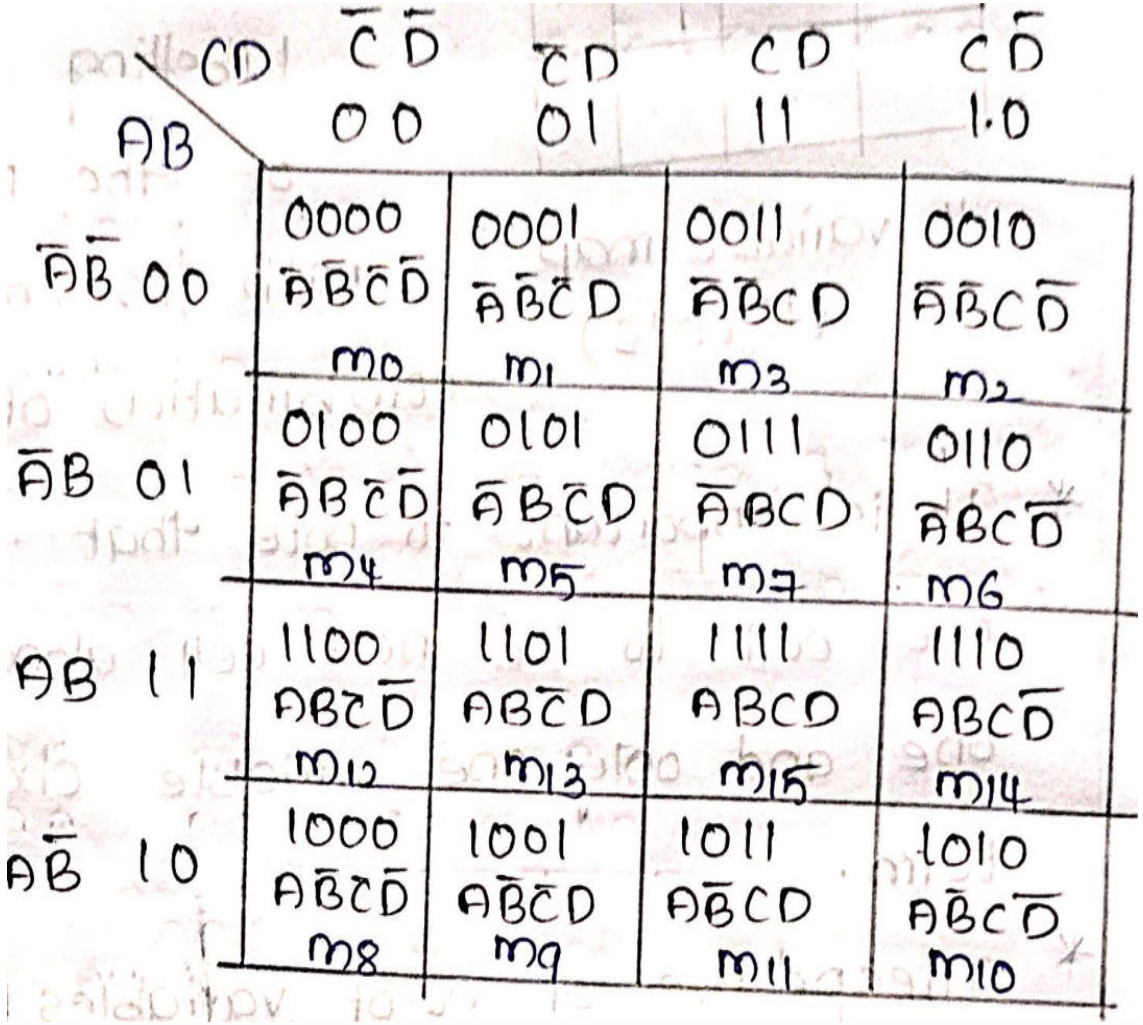
❖ The **Gray code** is used **to label** the **rows** and **column** of K-map

Representation of K – Map in SOP form:- (Minterms)









Representation of K–Map in POS form:- (Maxterms)

A \	0	1
A	0	
$\bar{A}$		1

A \ B	B = 0	$\bar{B}$ = 1
A	0	
	A+B	A+$\bar{B}$
	M_0	M_1
$\bar{A}$	1	
	$\bar{A}$+B	$\bar{A}$+$\bar{B}$
	M_2	M_3

A \ BC	BC = 00	B$\bar{C}$ = 01	$\bar{B}\bar{C}$ = 11	$\bar{B}$C = 10
A	0+0+0	0+0+1	0+1+1	0+1+0
	A+B+C	A+B+$\bar{C}$	A+$\bar{B}$+$\bar{C}$	A+$\bar{B}$+C
	M_0	M_1	M_3	M_2
$\bar{A}$	1+0+0	1+0+1	1+1+1	1+1+0
	$\bar{A}$+B+C	$\bar{A}$+B+$\bar{C}$	$\bar{A}$+$\bar{B}$+$\bar{C}$	$\bar{A}$+$\bar{B}$+C
	M_4	M_5	M_{67}	M_6

AB \ CD	CD = 00	C$\bar{D}$ = 01	$\bar{C}\bar{D}$ = 11	$\bar{C}$D = 10
AB	0+0+0+0	0+0+0+1	0+0+1+1	0+0+1+0
	A+B+C+D	A+B+C+$\bar{D}$	A+B+$\bar{C}$+$\bar{D}$	A+B+$\bar{C}$+D
	M_0	M_1	M_3	M_2
A$\bar{B}$	0+1+0+0	0+1+0+1	0+1+1+1	0+1+1+0
	A+$\bar{B}$+C+D	A+$\bar{B}$+C+$\bar{D}$	A+$\bar{B}$+$\bar{C}$+$\bar{D}$	A+$\bar{B}$+$\bar{C}$+D
	M_4	M_5	M_7	M_6
$\bar{A}\bar{B}$	1+1+0+0	1+1+0+1	1+1+1+1	1+1+1+0
	$\bar{A}$+$\bar{B}$+C+D	$\bar{A}$+$\bar{B}$+C+$\bar{D}$	$\bar{A}$+$\bar{B}$+$\bar{C}$+$\bar{D}$	$\bar{A}$+$\bar{B}$+$\bar{C}$+D
	M_{12}	M_{13}	M_{15}	M_{14}
$\bar{A}$B	1+0+0+0	1+0+0+1	1+0+1+1	1+0+1+0
	$\bar{A}$+B+C+D	$\bar{A}$+B+C+$\bar{D}$	$\bar{A}$+B+$\bar{C}$+$\bar{D}$	$\bar{A}$+B+$\bar{C}$+D
	M_8	M_9	M_{11}	M_{10}

Plotting a K Map:-

Plotting a K Map:-

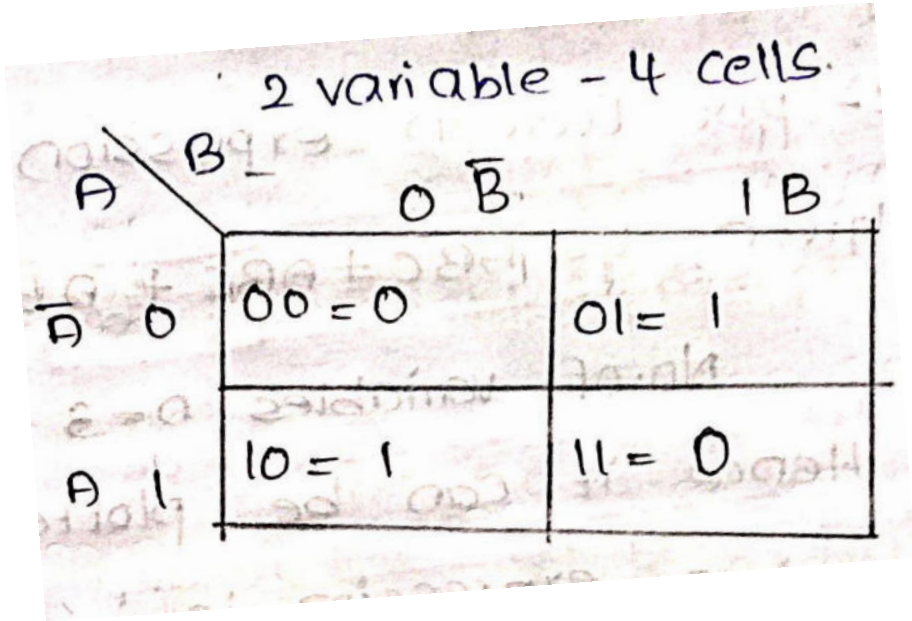
The logical functions can be represented in various forms such as

1. Truth Table
2. SOP Boolean expression
3. POS Boolean expression

1. Truth Table

Ex:-

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



Ex:-

A	B	C	y
0	0	0	0
0	0	1	0
0	01	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

3-variable k-map

A \ BC	00	101	11	010
0	0 0	0101=0 1	011=0 3	0010=0 2 1
1	100=1 4	110=0 101=1 5	111=1 7	110=0 6

Ex:-

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

4 variable k-map

AB \ CD	00	01	11	10
00	0000 1 0	0001 0 1	0011 1 3	0010 1 2
01	0100 1 4	0101 0 5	0111 1 7	0110 1 6
11	1100 0 12	1101 1 13	1111 1 15	1110 1 14
10	1000 0 8	1001 1 9	1011 1 11	1010 1 10

2. Standard SOP Boolean expression

Standard SOP in. K - map:-

Boolean expression in **SOP form** can be plotted on the **K map** by a placing in each cell corresponding to a **minterm** in SOP expression. Remaining cells are filled with **zeros**.

Example: plot Boolean expression $y = AB\bar{C} + ABC + \bar{A}\bar{B}C$

Solution: Given $y = AB\bar{C} + ABC + \bar{A}\bar{B}C$

No. of variables $n = 3$

Hence it can be plotted by using 3 variable Kmap.

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0 (0)	$\bar{A}\bar{B}C$ 1 (1)	0 (3)	0 (2)
A 1	0 (4)	0 (5)	ABC 1 (7)	$AB\bar{C}$ 1 (6)

The given expression is in standard sop form.

$$y = \Sigma m(1,7,6)$$

Example: plot Boolean expression

$$Y = \bar{A}B\bar{C}\bar{D} + A\bar{B}C\bar{D} + \bar{A}BCD + A\bar{B}CD + AB\bar{C}D$$

Solution: Given $Y = \bar{A}B\bar{C}\bar{D} + A\bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}CD + AB\bar{C}D$

$n = 4$

Hence it can be plotted using 4 variable K map.

The given Boolean expression is in standard Sop form.

$$Y = \Sigma m(4, 6, 10, 11, 13)$$

AB \ CD	$\bar{C}\bar{D}$ (00)	$\bar{C}D$ (01)	$C\bar{D}$ (11)	CD (10)
$\bar{A}\bar{B}$ (00)	0	0	0	0
	0	1	3	2
$\bar{A}B$ (01)	$\bar{A}B\bar{C}\bar{D}$	0	0	$\bar{A}B\bar{C}D$
	1 4	5	7	1 6
AB (11)	0	$AB\bar{C}D$	0	0
	12	1 13	15	14
$A\bar{B}$ (10)	0	0	$A\bar{B}CD$	$A\bar{B}C\bar{D}$
	8	9	1 11	1 10

3. Standard POS on K-map:

A Boolean expression in **Standard POS** form can be plotted on the k map by placing a **0** in each cell Corresponding to a **maxterm** in the expression. Remaining cells are filled with **one's**.

Example: plot Boolean expression $y = (A+\bar{B}+C)(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(A+B+\bar{C})$

Solution: Given $y = (A+\bar{B}+C)(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(A+B+\bar{C})$

No: of variables = 3

Hence it can be plotted by using 3 variable K map.

A \ BC	00	01	11	10
	$B+C$	$B+\bar{C}$	$\bar{B}+\bar{C}$	$\bar{B}+C$
A 0	1 0	$A+B+\bar{C}$ 0 1	$\bar{A}+\bar{B}+\bar{C}$ 0 3	$A+\bar{B}+C$ 0 2
$\bar{A}$ 1	1 4	1 5	1 7	$\bar{A}+\bar{B}+C$ 0 6

$y = \Pi M(1,2,3,6)$

$y = \Sigma m(0,4,5,7)$ → SOP form.

Grouping cells for simplification:-

Grouping cells for simplification:-

- Once the **Boolean function** is plotted on the **K map**. We have to use **grouping techniques** to simplify the **Boolean function**.
- The **grouping** is nothing but **Combining** terms in **adjacent cells**.
- The **simplification** is achieved by grouping adjacent **one's** or **zeros** in a group of 2^i where i = 1,2,3,4,...,n where is the **no of variables**

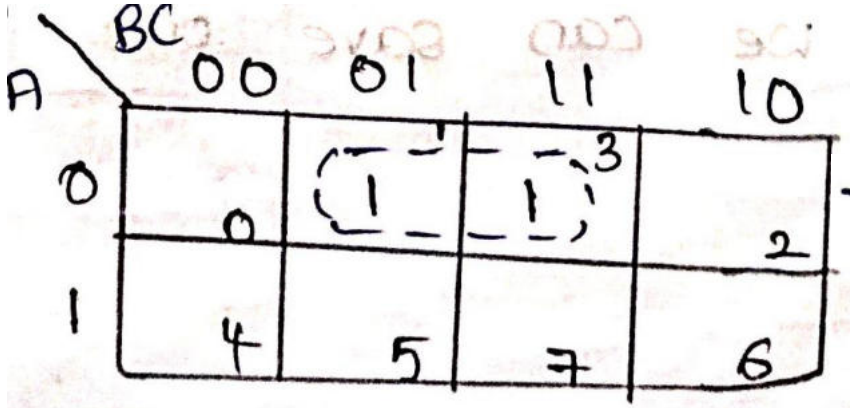
Pair

Two adjacent cells are grouped together is called pair.

* (in a row or column) in a k map.
* It cancels one variable in a k-map simplification.

Example:

a) Horizontal Grouping:-



→ horizontal grouping

$y = \bar{A}\bar{B}C + \bar{A}BC$

$= \bar{A}C(B+\bar{B})$

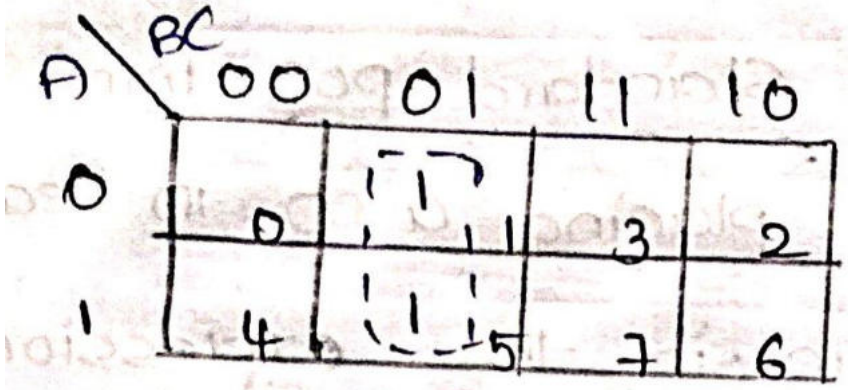
$\boxed{y = \bar{A}C.}$

Group 1,3

001 = $\bar{A}\bar{B}C$

011 = $\bar{A}BC$

b) Vertical Grouping:-



Group 1,5

$\bar{A}\bar{B}C$ — 1

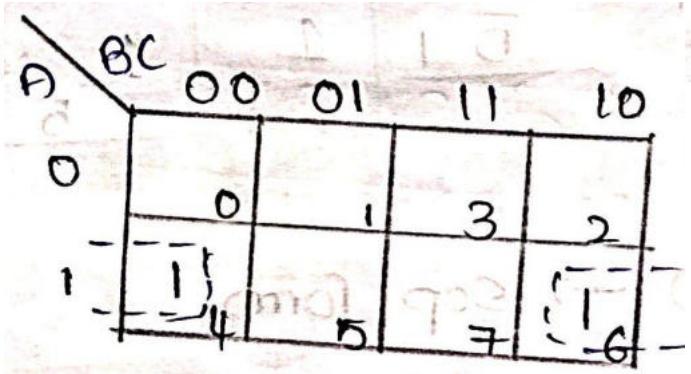
$A\bar{B}C$ — 5

$y = \bar{A}\bar{B}C + A\bar{B}C$

$= \bar{B}C(\bar{A}+A)$

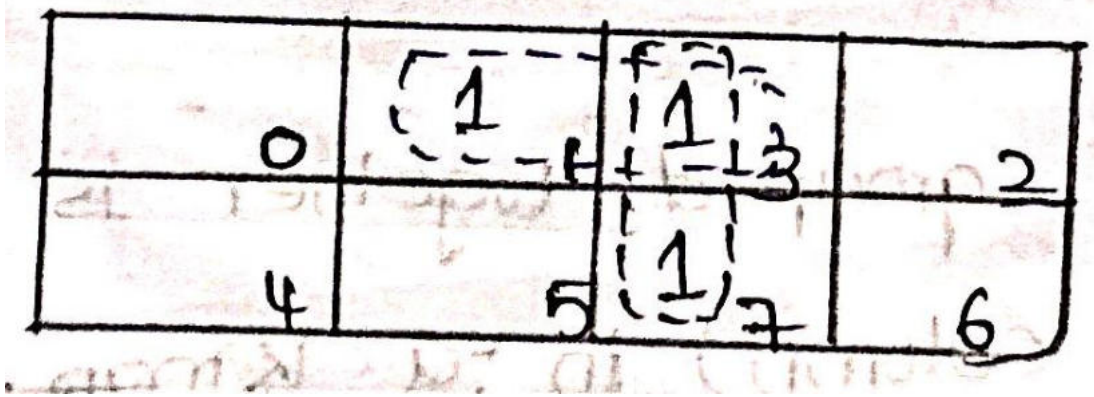
$\Rightarrow \boxed{y = \bar{B}C}$

c) In a Kmap the corresponding cells in the left most column and right most colomn are considered to be adjacent.

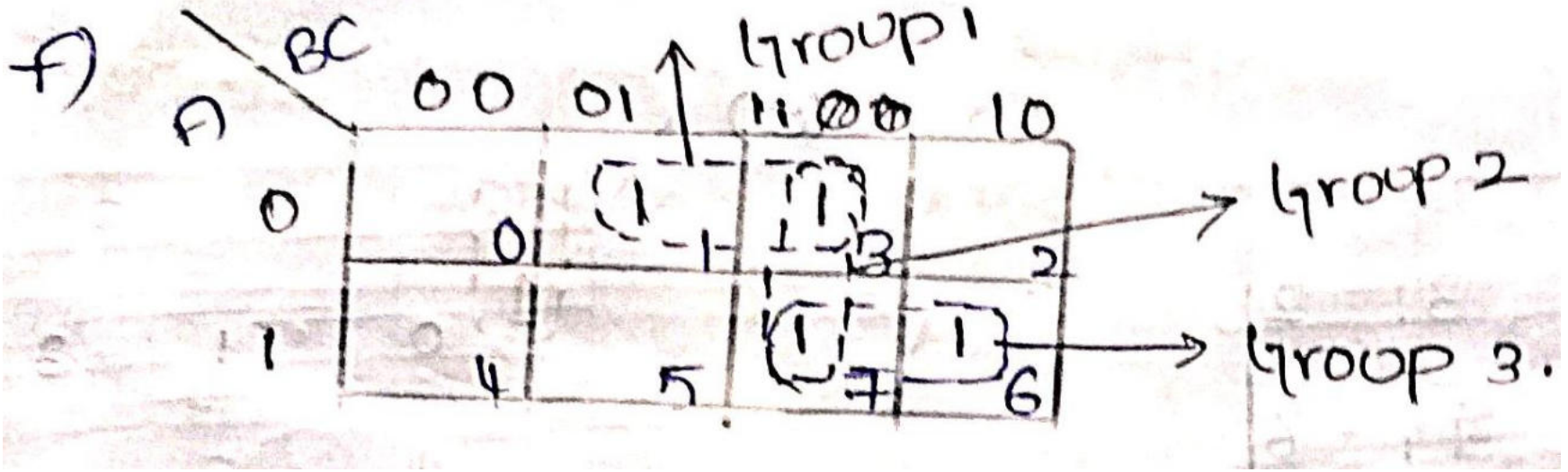


d) In a K-map the top rows and bottom rows are considered to be

e) In a k map that has two overlapping pairs of ones. this shows that we can save one term by the two pairs.



f)



group 1 :- $\bar{A}\bar{B}C-1$ $A\bar{B}C-3$

$B\bar{C}$

group 2 :- $\bar{A}BC-3$ $ABC-7$

BC

group 3 :- $AB\bar{C}-7$ $AB\bar{C}-6$.

AB

$$Y = \bar{A}C + BC + AB$$

$$= \bar{A}C(B+\bar{B}) + BC + AB$$

$$= \bar{A}BC + \bar{A}\bar{B}C + BC + AB$$ } → wrong

$$= \bar{A}C + BC(A+\bar{A}) + AB$$

$$= \bar{A}C + ABC + \bar{A}BC + AB = \bar{A}C + AB(C+1) + \bar{A}BC$$

$$= \bar{A}C + AB + \bar{A}BC = \bar{A}C(B+1) + AB$$

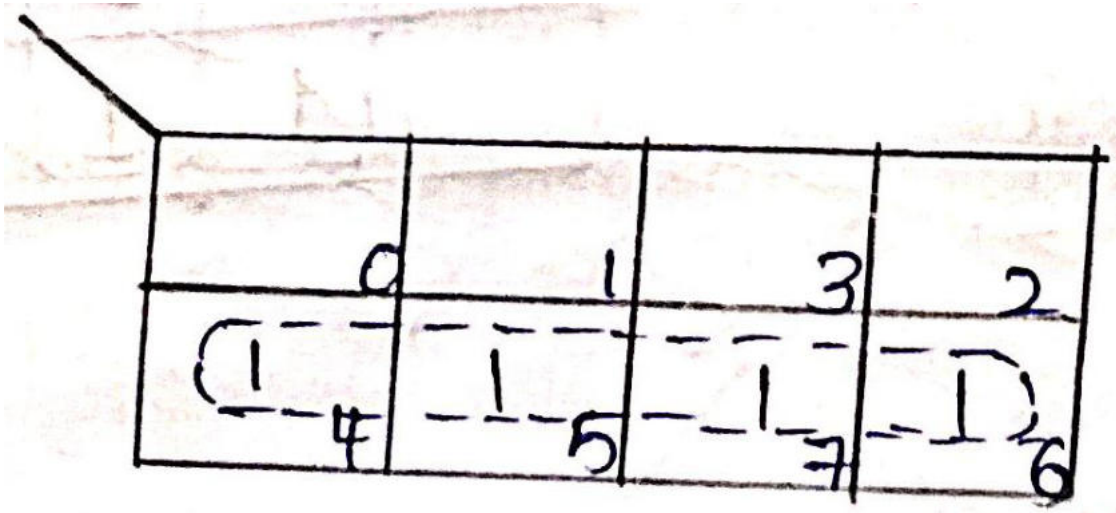
$$\boxed{Y = \bar{A}C + AB}$$

In a K-map where 3 groups of pair can be formed. But only two pairs are enough to include all ones in a kmap. In such cases third pair is not required. (2nd group).

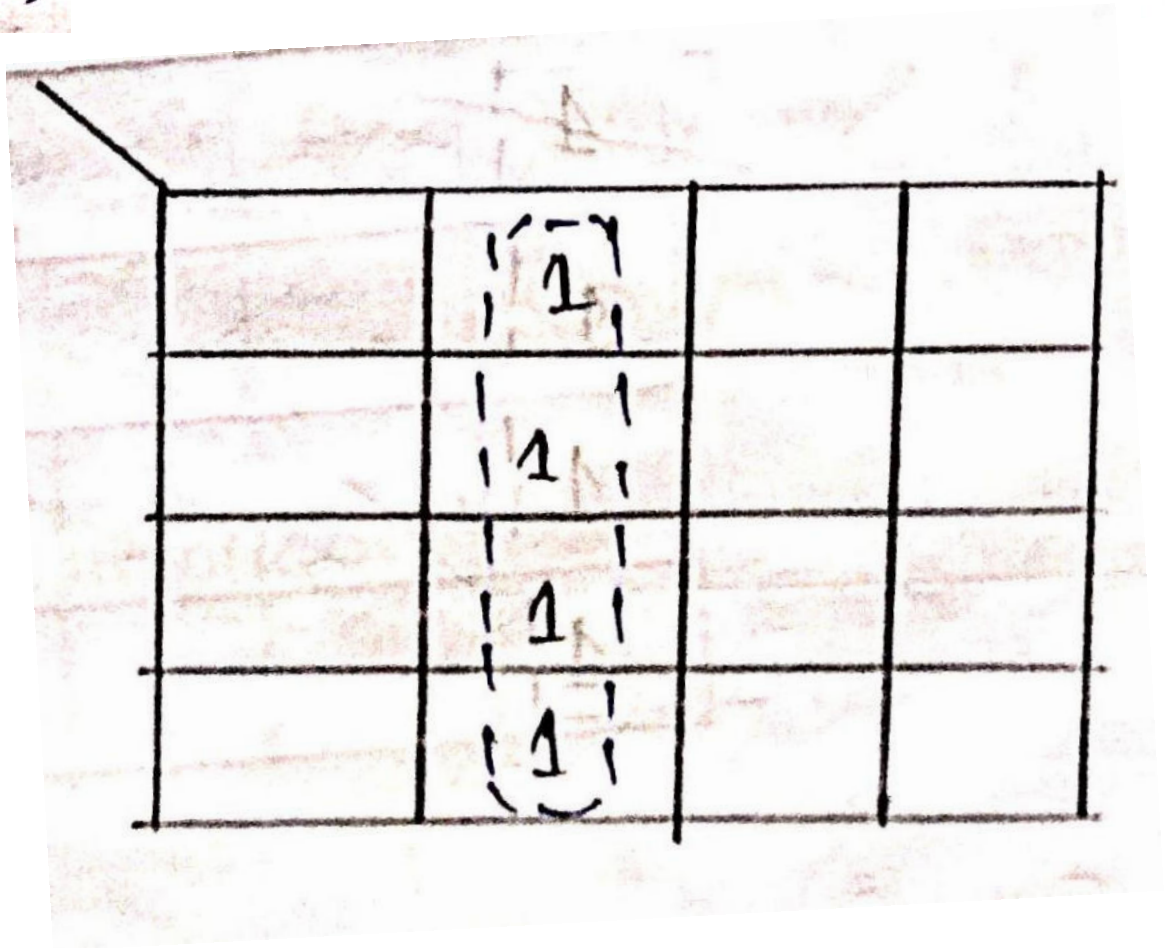
Quad (4 adjacent ones):

In a kmap we can prove 4 adjacent ones the resultant group is called Quad.

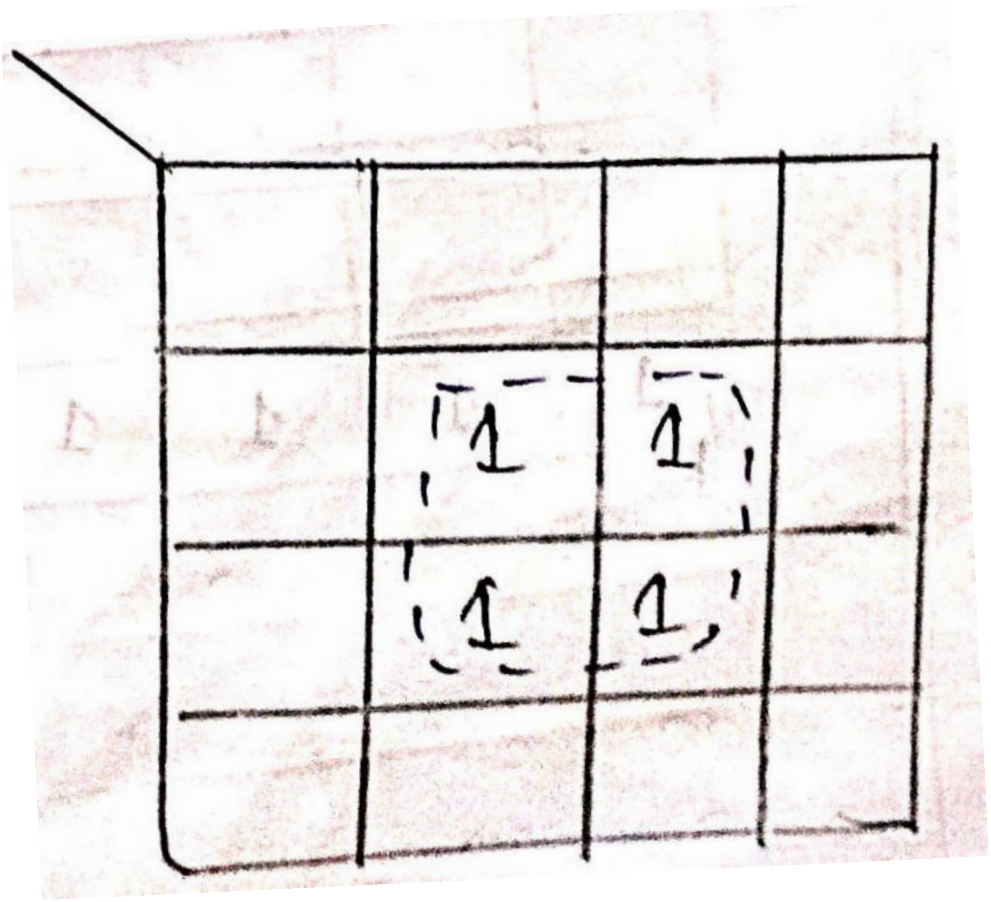
a) Quad is possible in 3 variable and 4 variable kmap.



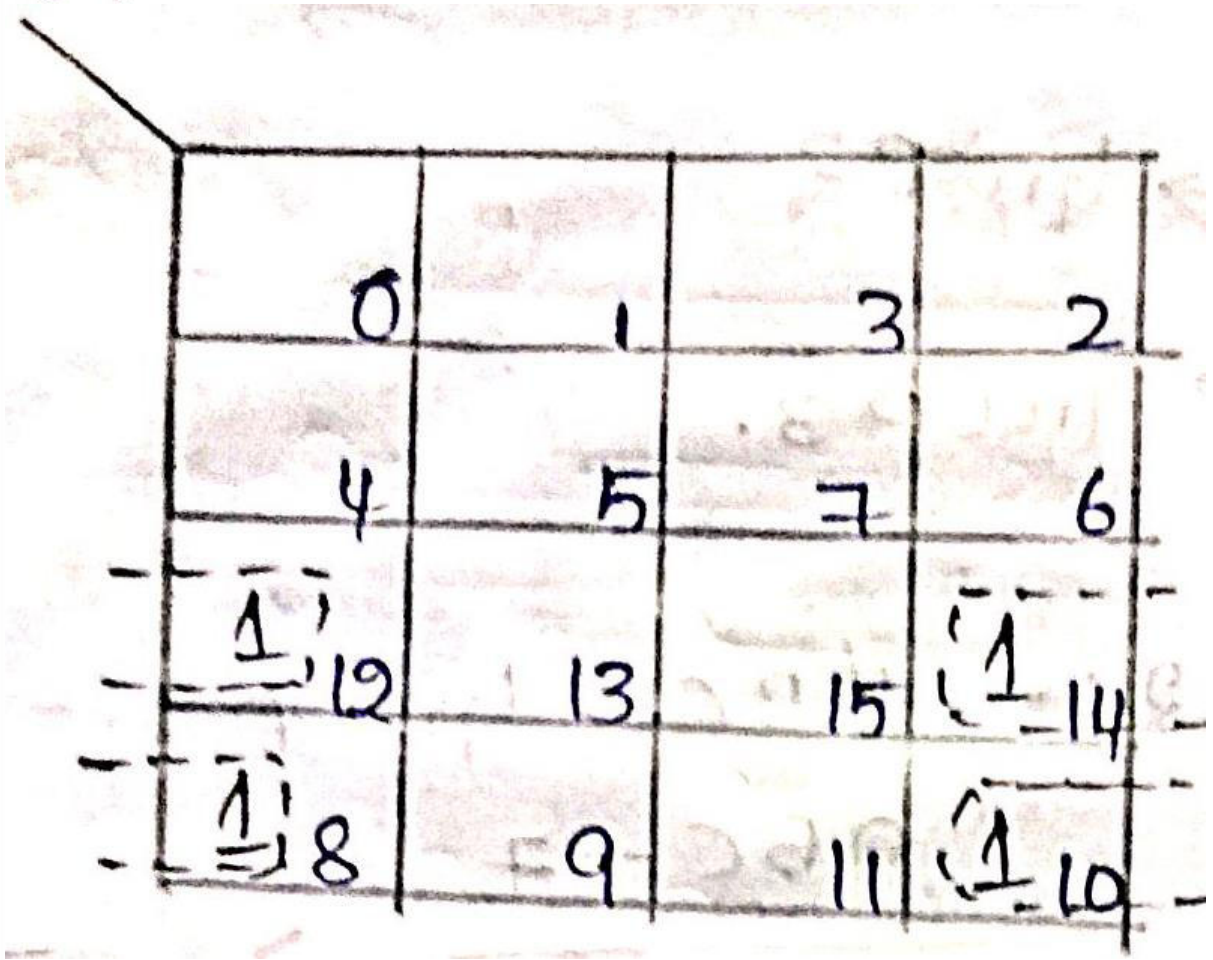
b)



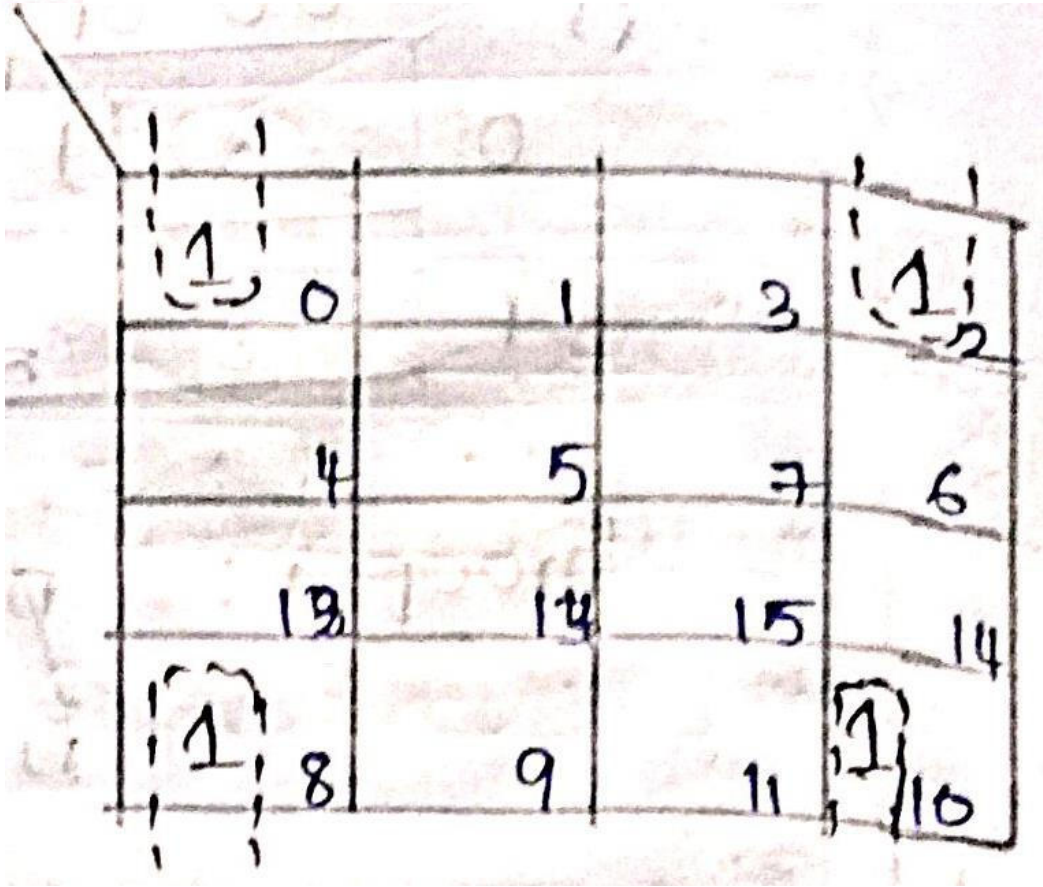
(c)



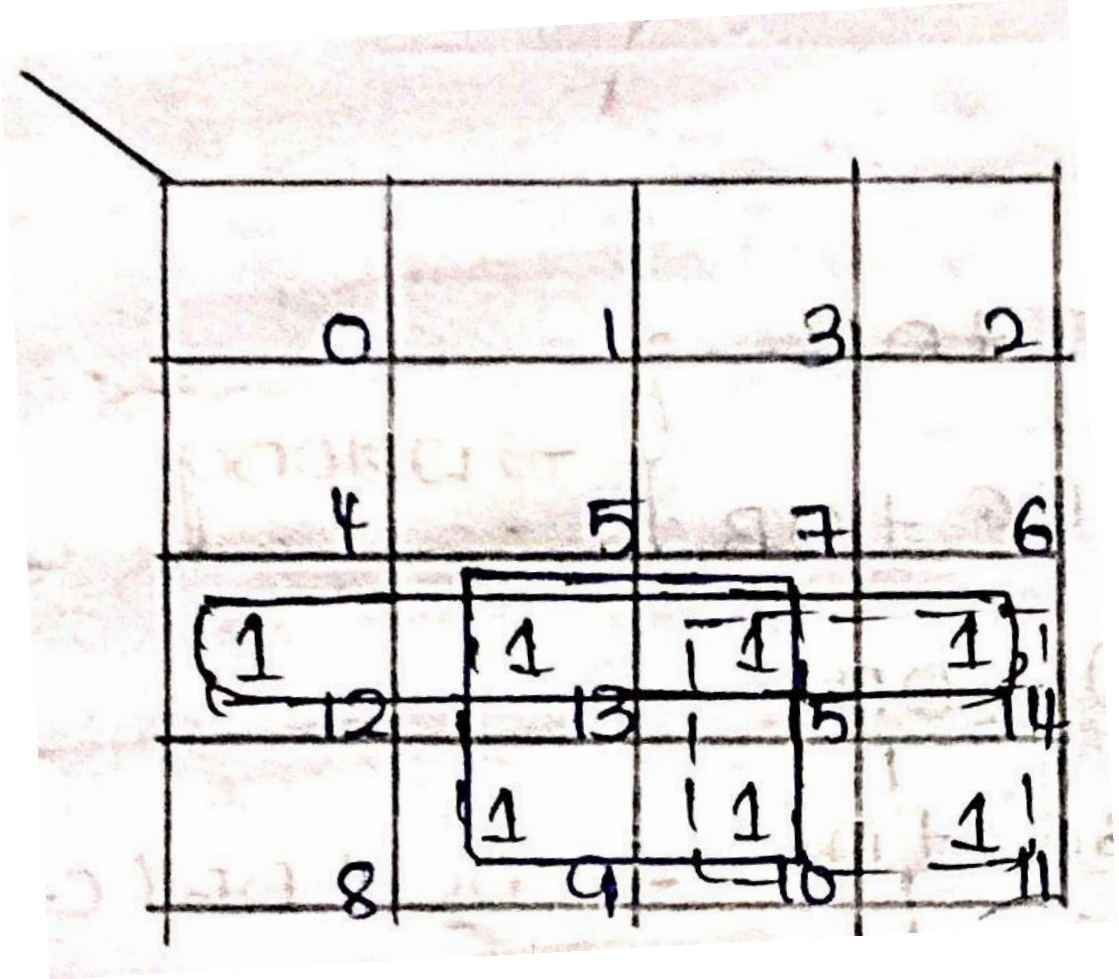
(d)


0
1
3
2
4
5
7
6
1
12
13
15
1
14
1
8
9
11
1
10

(e)


1
0
1
3
1
2
4
5
7
6
12
14
15
14
1
8
9
11
1
10

(F)

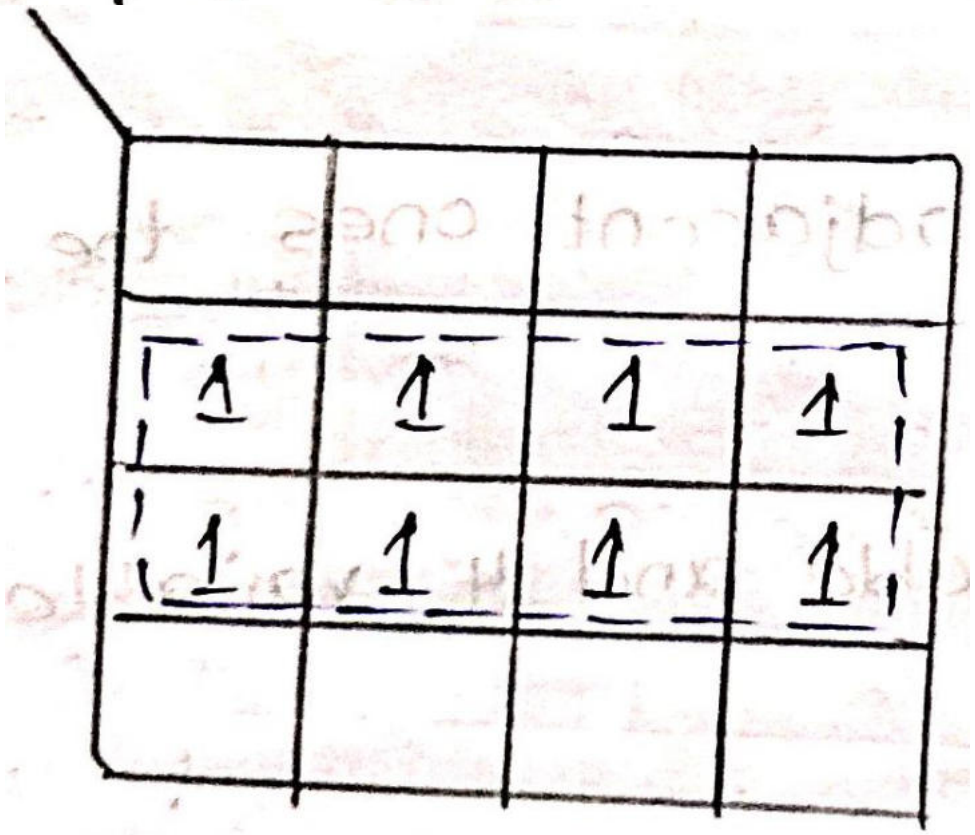


NOTE: Quad is a group of 4 adjacent cells in a K-map it cancels to variables in a K-map simplification.

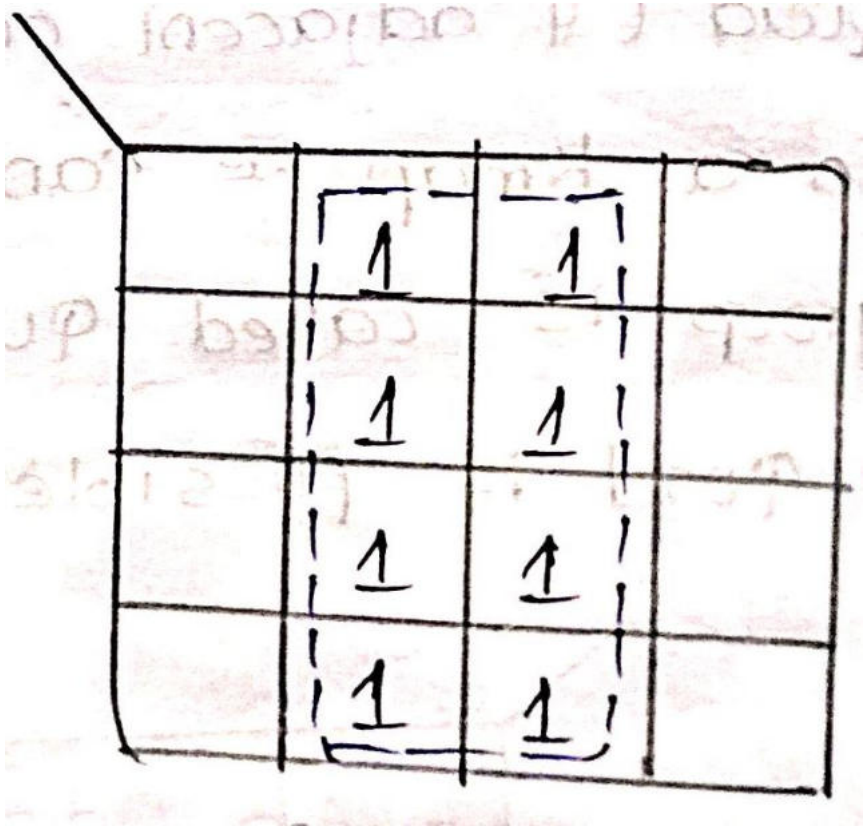
Octat (8 adjacent ones):

In a Kmap we can group 8 adjacent ones the resultant group is called as octet.

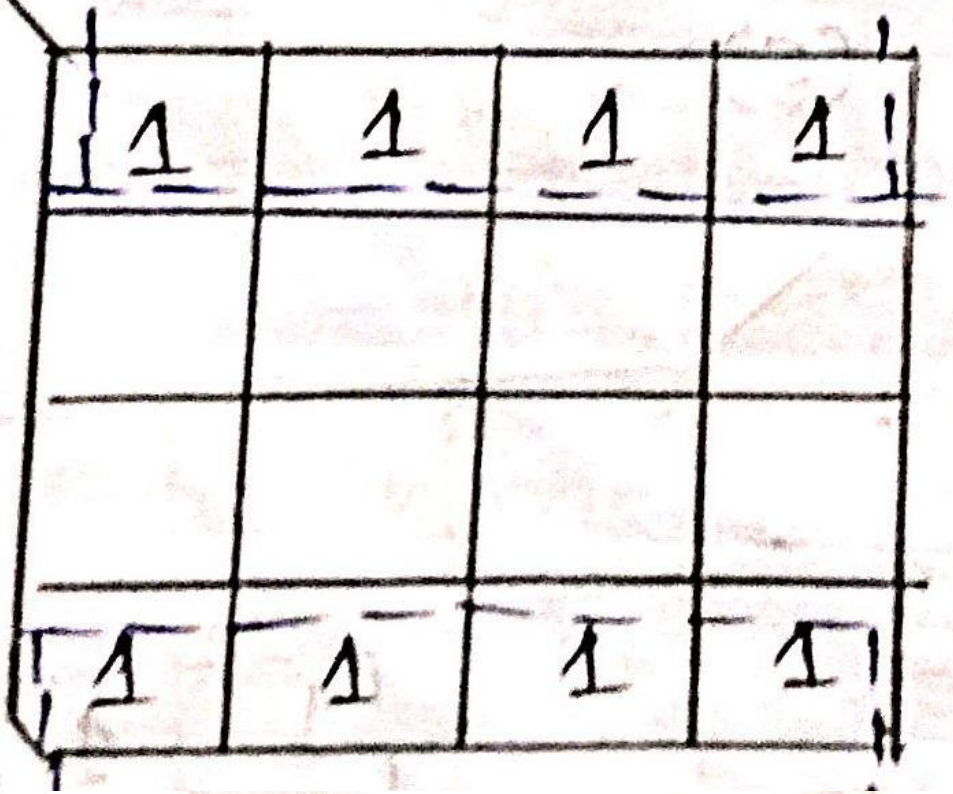
a)



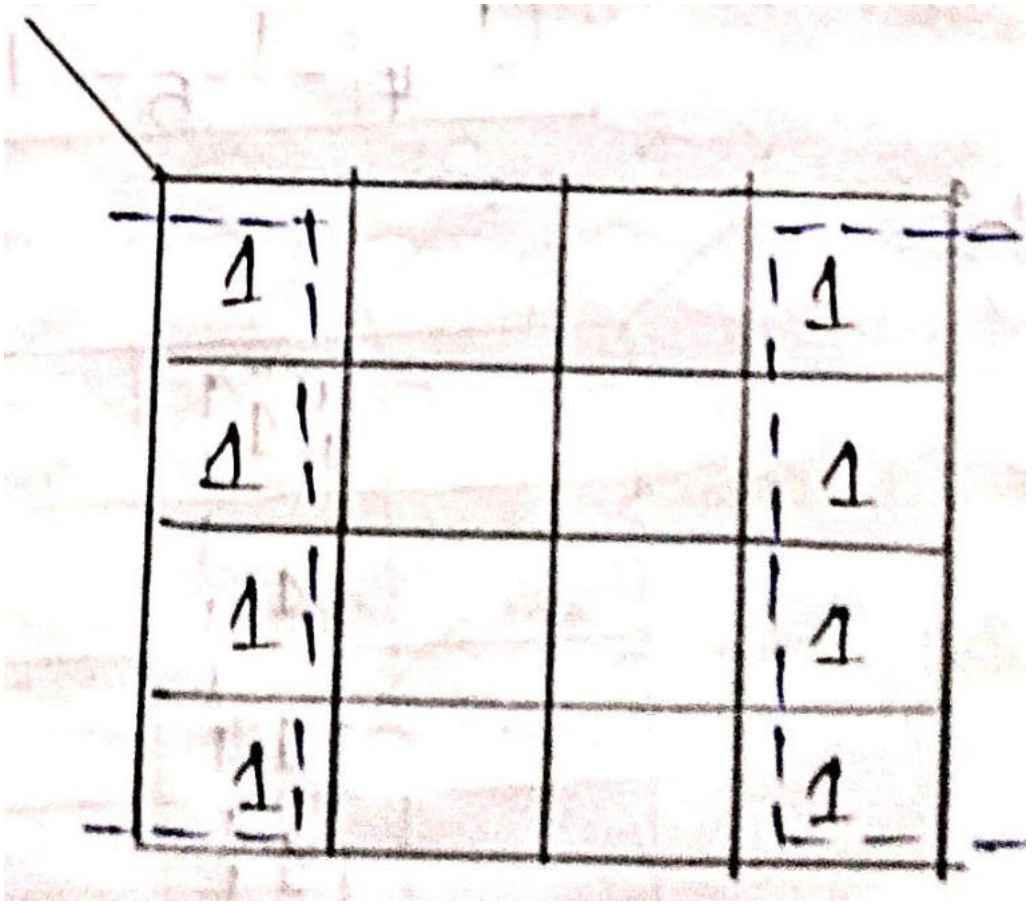
b)



(c)



(d)



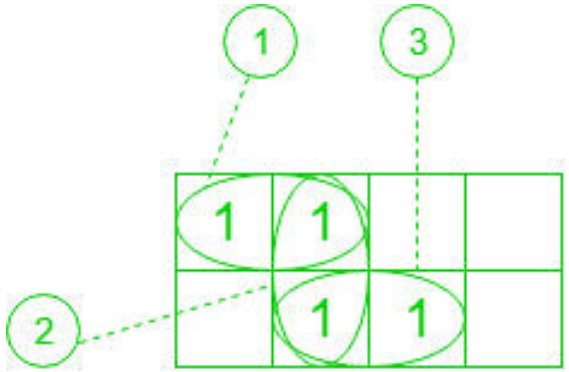
NOTE: Octet is a group of 8 adjacent cells in a K-map.

It can cancel 3 variables in K-map simplification.

1. Prime Implicants

* Each group on the K-map represents 1 prime implicant

Example: 1

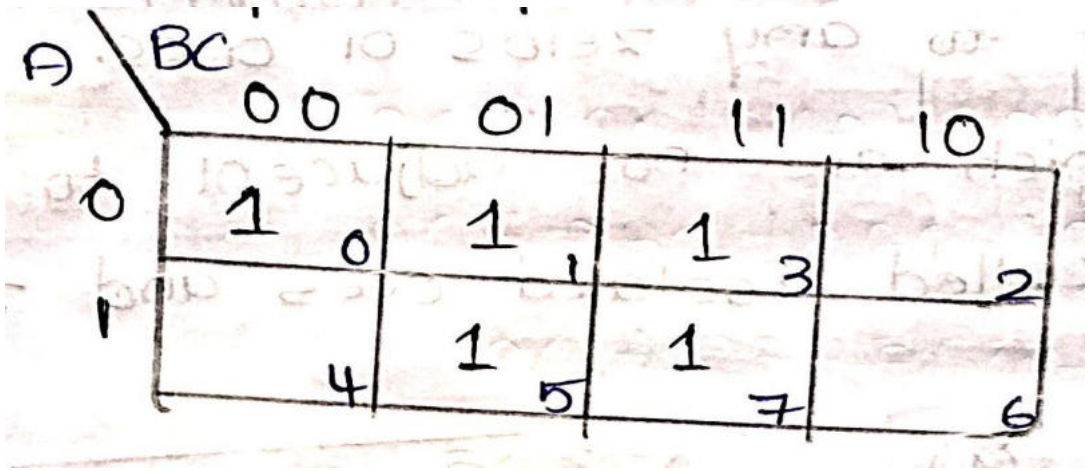


No. of Prime Implicants = 3

Example: 2 Ex:- Obtain the prime Implicants for a given boolean expression using K maps. $F(A,B,C) = (0,1,3,5,7)$

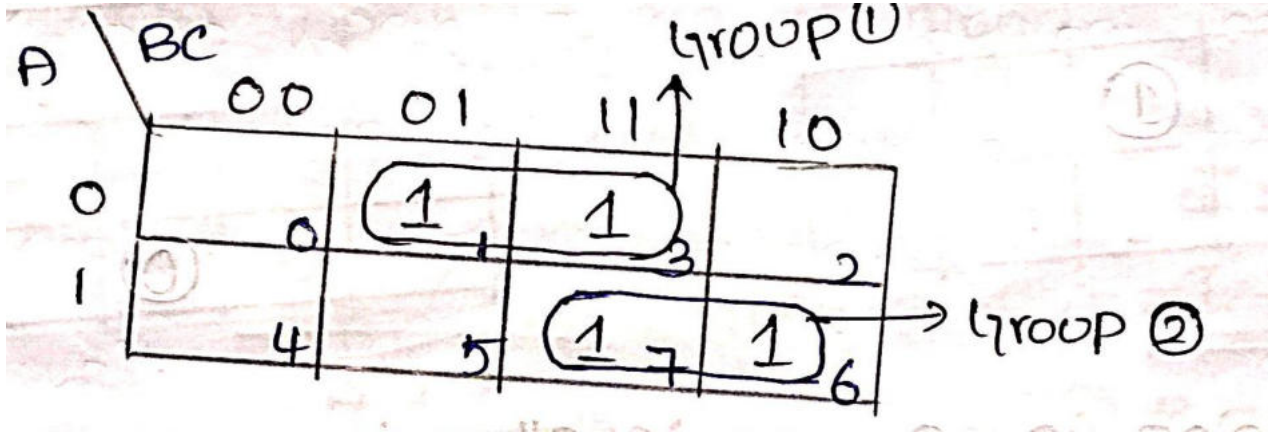
Solution: Given expression is a 3 variable. Hence it can be represent by using 3 variable K-map.

$F(A,B,C) = (0,1,3,5,7)$



Example: 3 Ex:- Obtain the prime Implicants for a given boolean expression using K maps. $F(A, B, C) = (1, 3, 7, 6)$

Solution: Given expression is a 3 variable. Hence it can be represent by using 3 variable K-map.

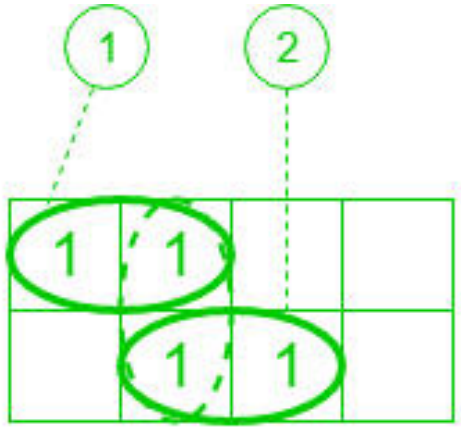


The prime implicants for given boolean expression are

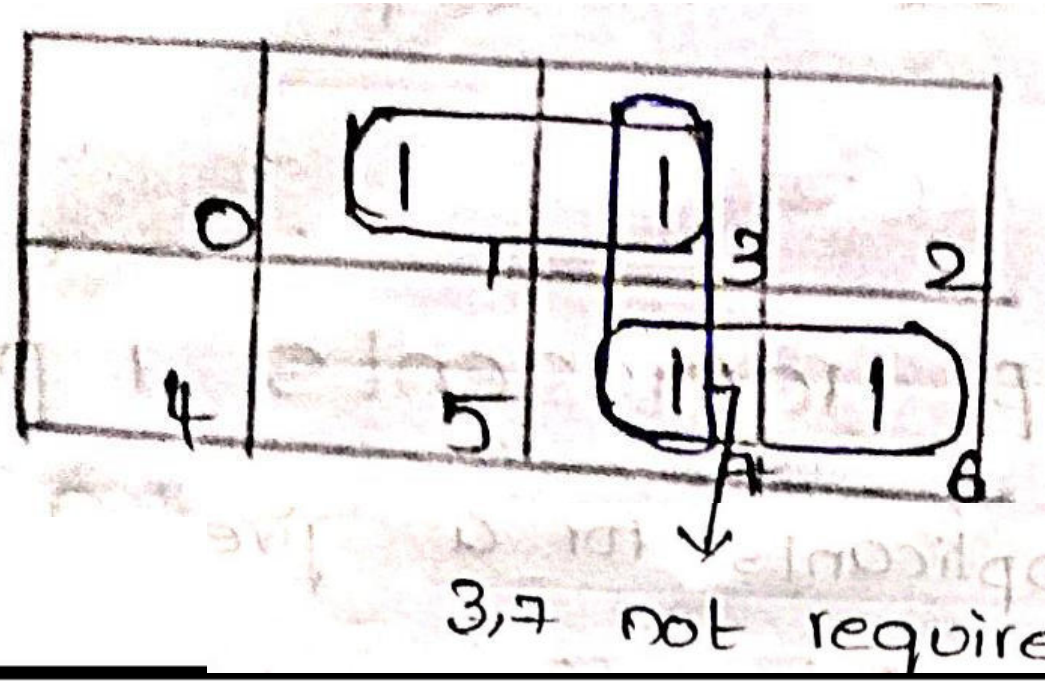
2. Essential Prime Implicants

The minimum no. of prime Implicants requires to include all ones present in the kmap are called essential prime implicants.

NOTE: Essential prime implicants(EPI) are those prime implicants which always appear in final solution.

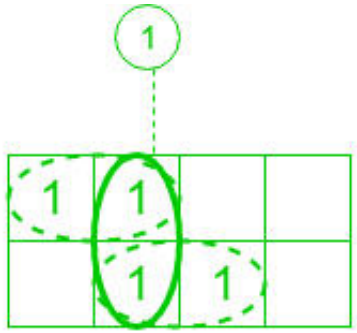


No. of Essential Prime Implicants = 2



3. Redundant Prime Implicants

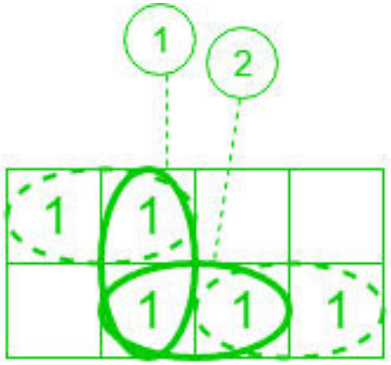
The prime implicants for which are not essential prime implicant are **redundant prime implicants (RPI)**. This prime implicant never appears in final solution.



No. of Redundant Prime Implicants = 1

4. Selective Prime Implicants

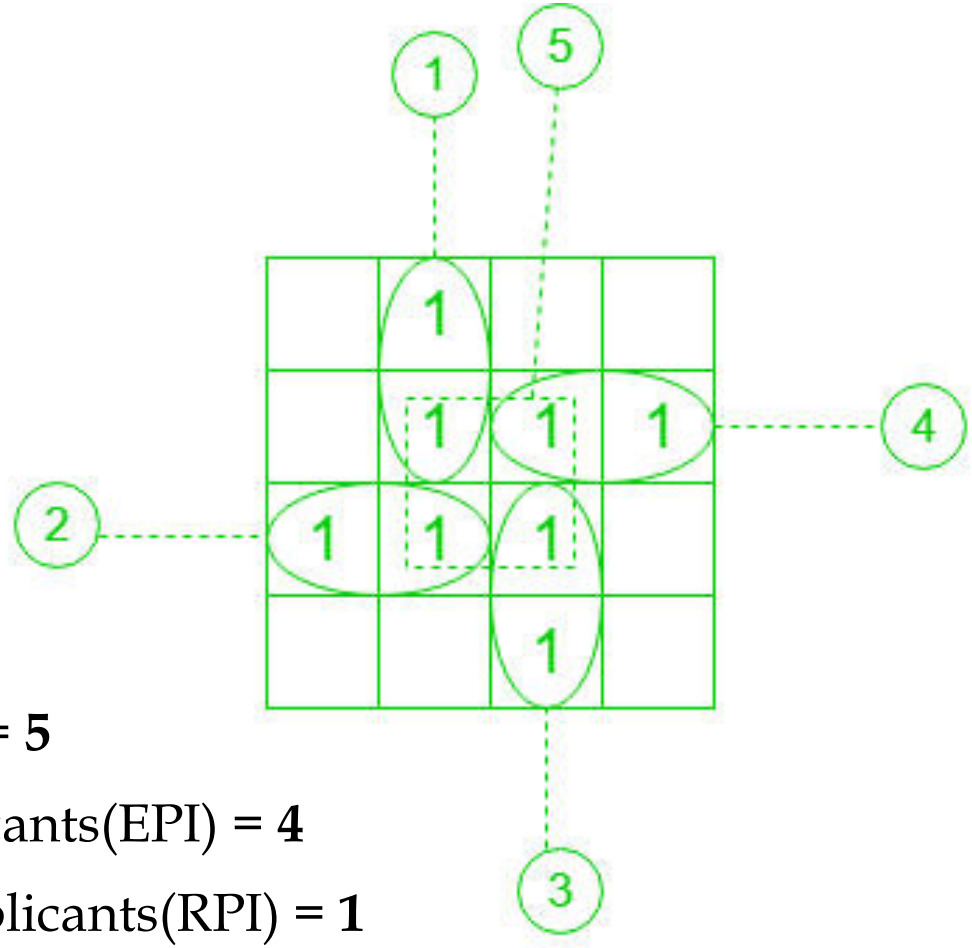
The prime implicants for which are neither essential nor redundant prime implicants are called selective prime implicants(SPI). These are also known as non-essential prime implicants. They may appear in some solution or may not appear in some solution.



No. of Selective Prime Implicants = 2

Example: 1 Given F = ∑m(1, 5, 6, 7, 11, 12, 13, 15), find number of implicant, PI, EPI, RPI and SPI.

Solution:



No. of Implicants = 8

PI = (1,2,3,4,5)

EPI = (1,2,3,4)

RPI = (5)

No. of Implicants = **8**

No. of Prime Implicants(PI) = **5**

No. of Essential Prime Implicants(EPI) = **4**

No. of Redundant Prime Implicants(RPI) = **1**

No. of Selective Prime Implicants(SPI) = **0**

F = 1 + 2 + 3 + 4

Example: 2 Given F = ∑m(0, 1, 5, 8, 12, 13), find number of implicant, PI, EPI, RPI and SPI.

Solution:

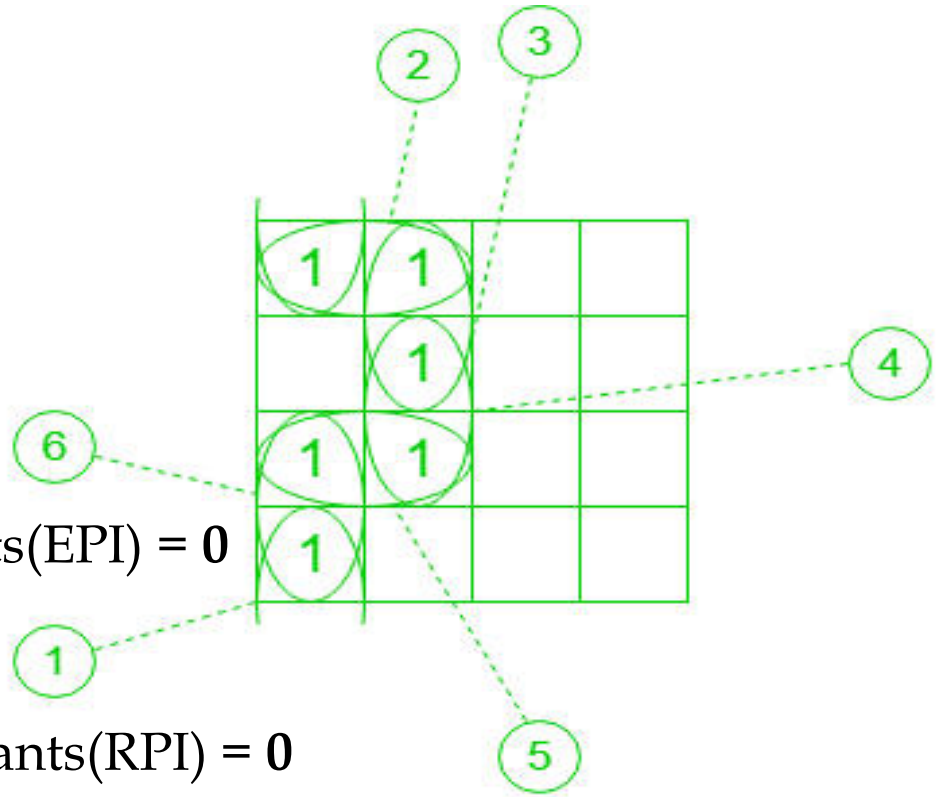
No. of Implicants = **6**

No. of Prime Implicants(PI) = **6**

No. of Essential Prime Implicants(EPI) = **0**

No. of Redundant Prime Implicants(RPI) = **0**

No. of Selective Prime Implicants(SPI) = **6**



No. of Implicants = 6

PI = (1,2,3,4,5,6)

SPI = (1,2,3,4,5,6)

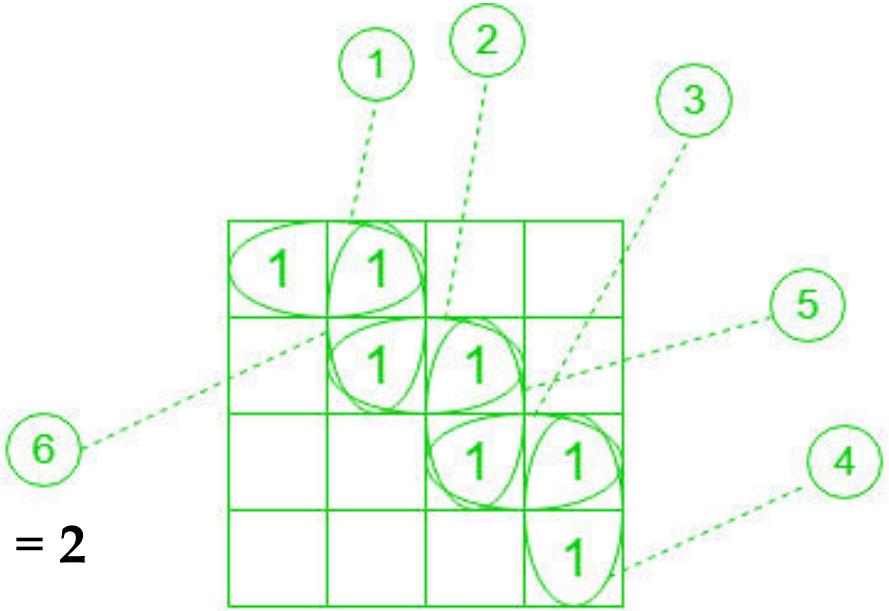
F = 1 + 3 + 5

OR

F = 2 + 4 + 6

Example: 3 Given F = ∑(0, 1, 5, 7, 15, 14, 10), find number of implicant, PI, EPI, RPI and SPI.

Solution:



No. of Implicants = 7

PI = (1,2,3,4,5,6)

EPI = (1,4)

SPI = (2,3,5,6)

No. of Implicants = **7**

No. of Prime Implicants(PI) = **6**

No. of Essential Prime Implicants(EPI) = **2**

No. of Redundant Prime Implicants(RPI) = **2**

No. of Selective Prime Implicants(SPI) = **4**

F = (1) + (2) + (3) + (4)

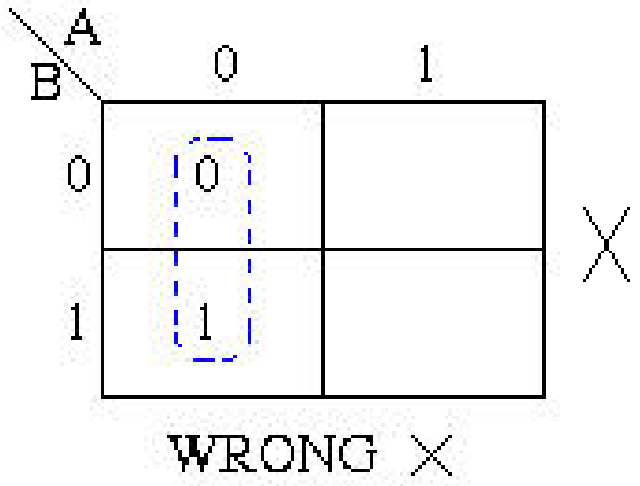
OR

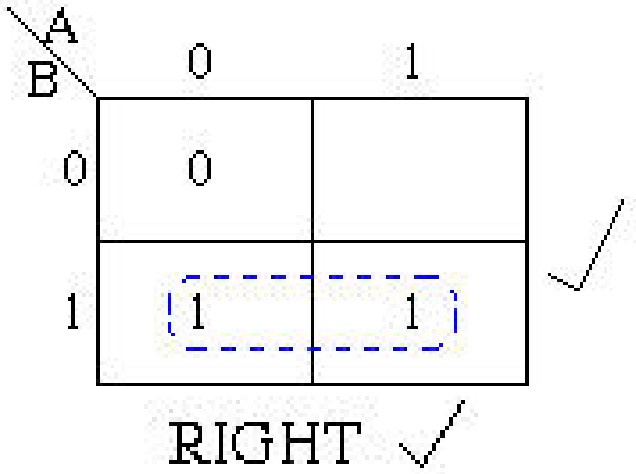
F = (1) + (5) + (6) + (4)

Karnaugh Maps - Rules of Simplification

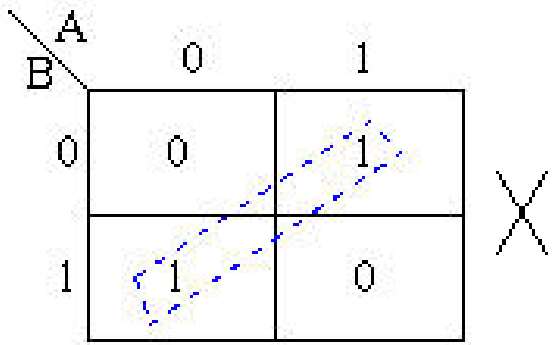
The Karnaugh map uses the following rules for the simplification of expressions by grouping together adjacent cells containing ones

1. Groups may not include any cell containing a zero

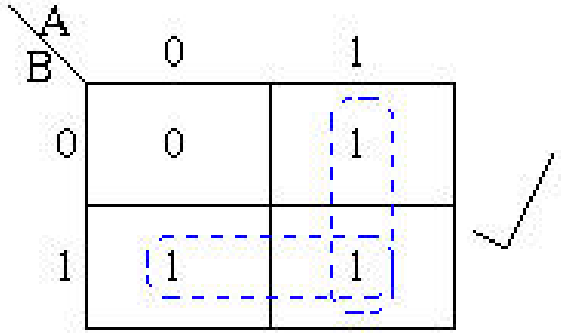




2. Groups may be horizontal or vertical, but not diagonal.

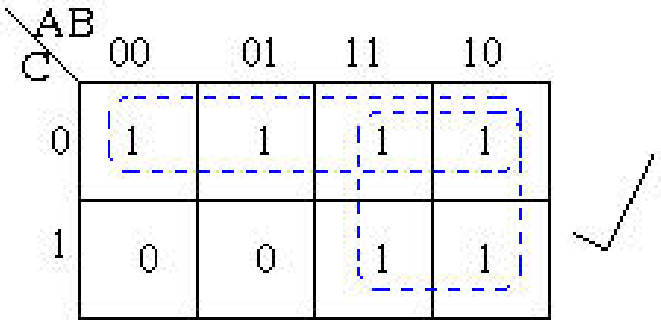


WRONG ✗

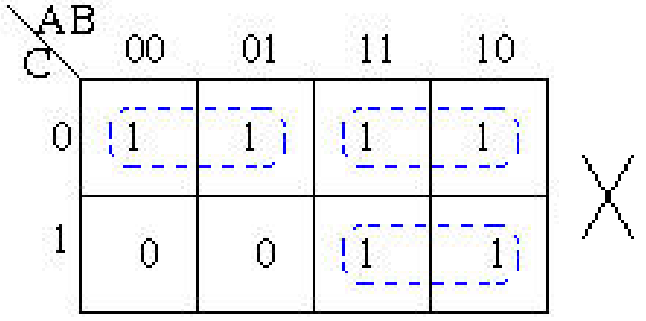


RIGHT ✓

3. Groups may be horizontal or vertical, but not diagonal.



RIGHT ✓

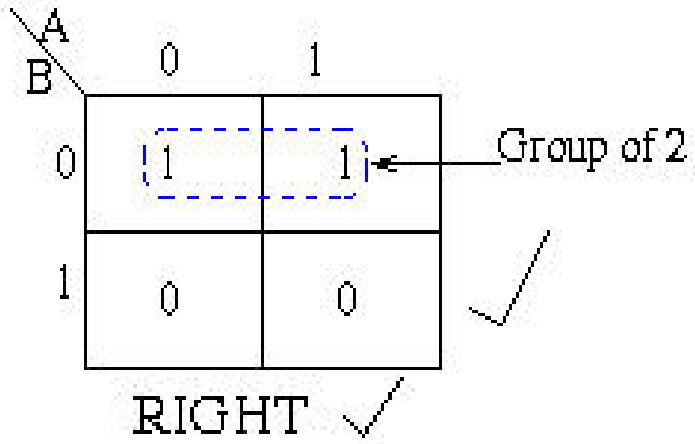


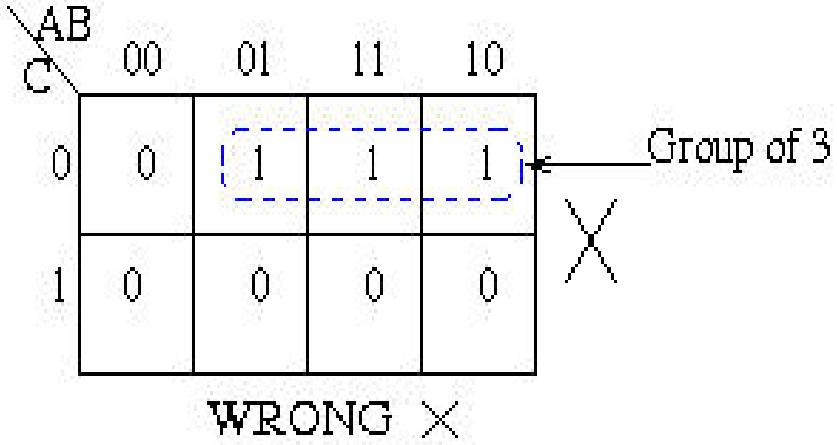
WRONG ✗

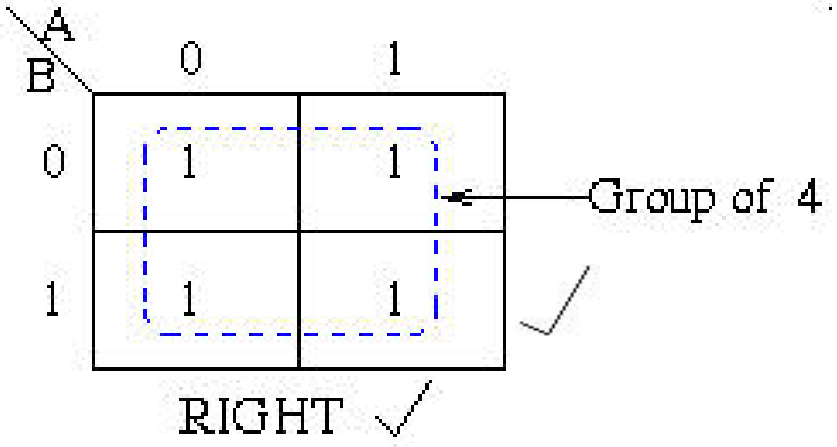
4. Groups must contain 1, 2, 4, 8, or in general 2n cells.

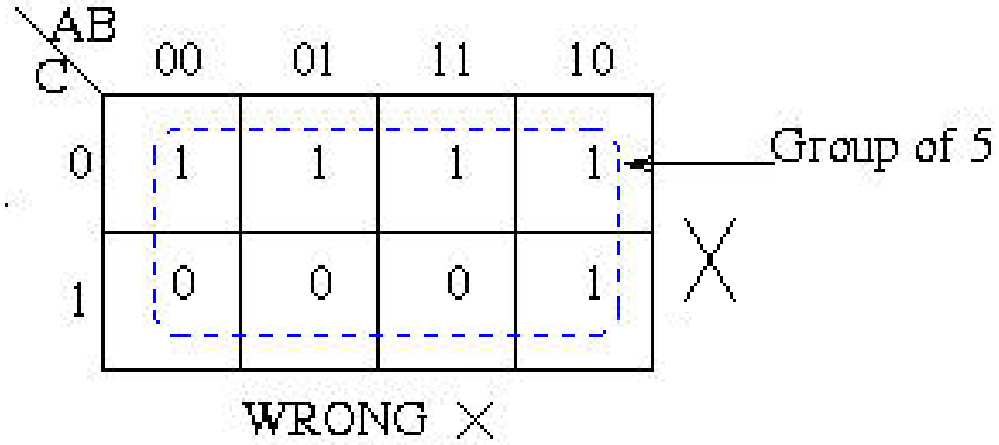
That is if n = 1, a group will contain two 1's since $2^1 = 2$.

If n = 2, a group will contain four 1's since $2^2 = 4$.

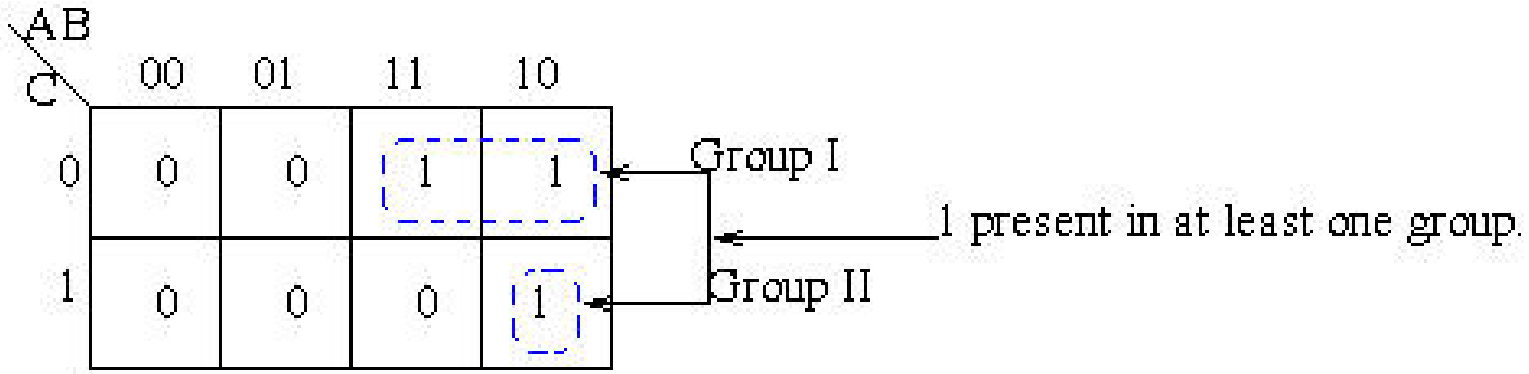




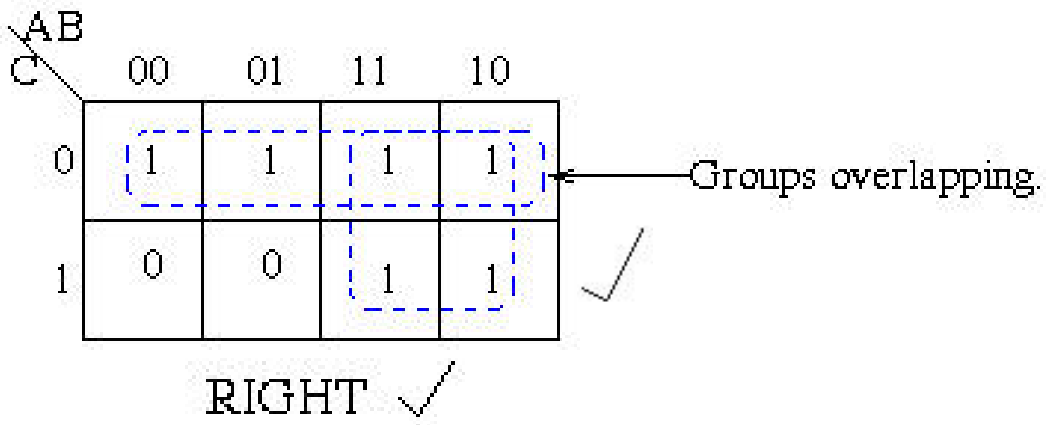


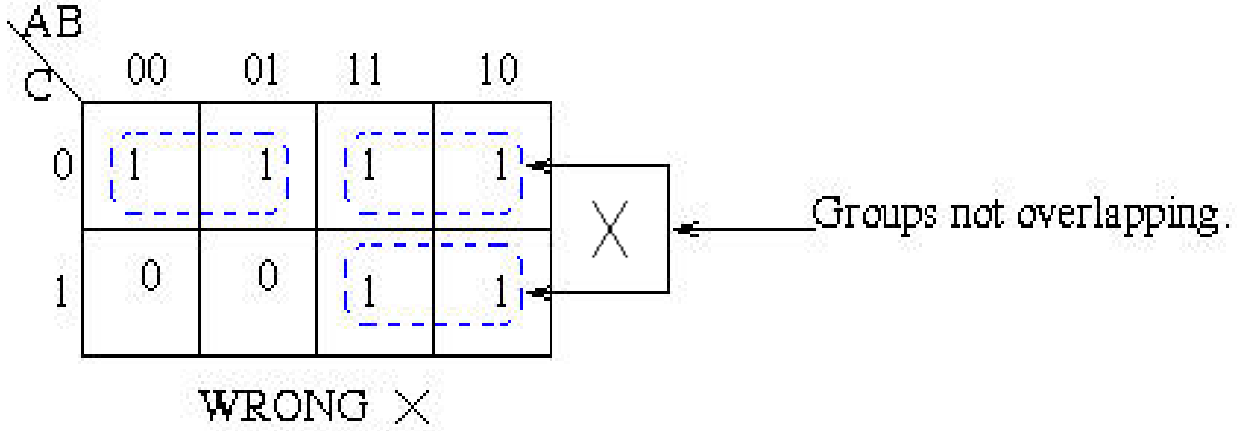


5. Each cell containing a one must be in at least one group.

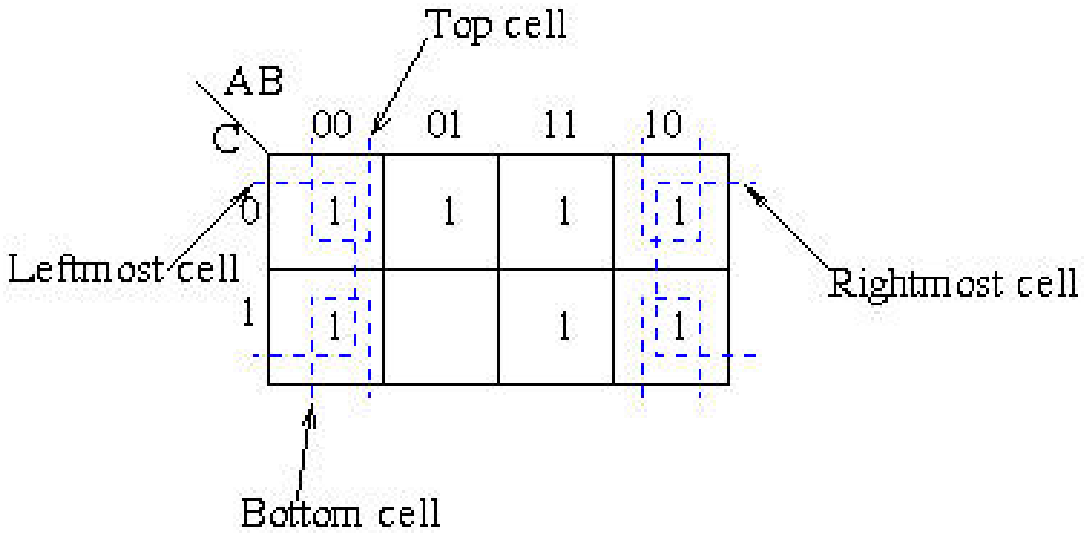


6. Groups may overlap.

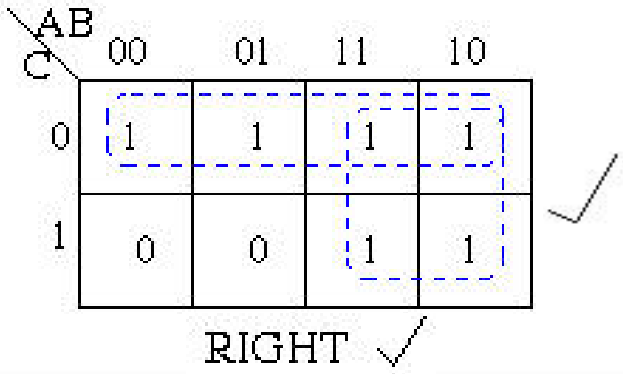


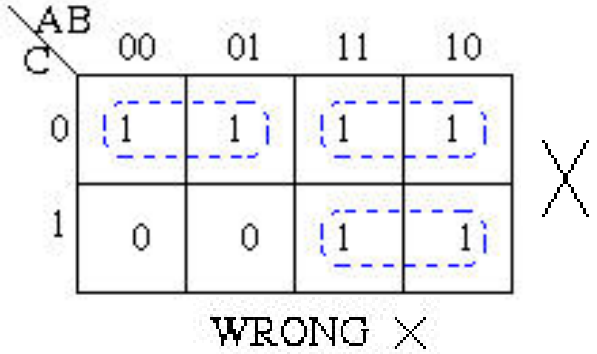


7. Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.



8. There should be as few groups as possible, as long as this does not contradict any of the previous rules.



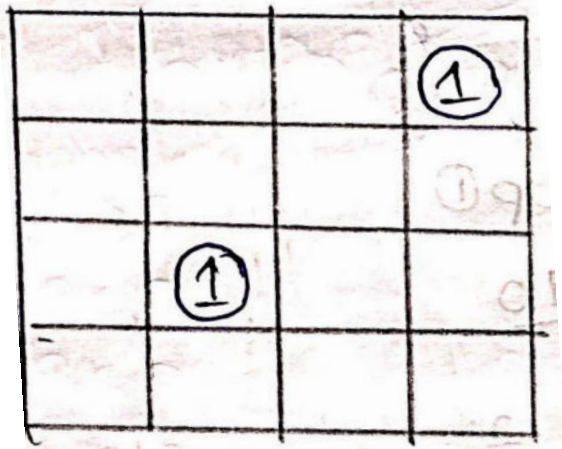


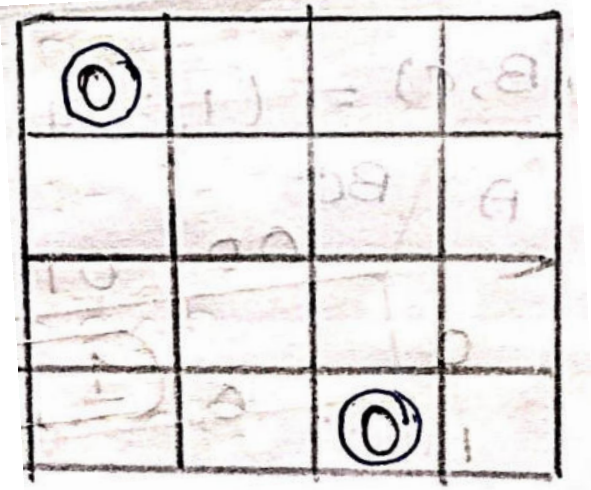
Note:1

Isolated zeros and ones :-

Isolated zeros or ones are the ones and zeros which are not adjacent to any zeros or ones.

In a k map which are not adjacent to any one's or zero's is called Isolated ones and zeros.



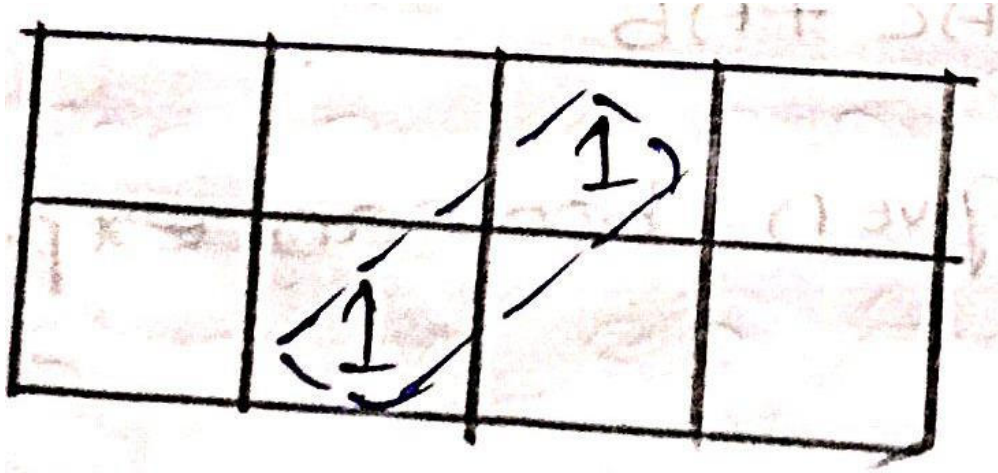


Note: 2

* overlapping in groups is allowed.
* Minimum no. of groups are required.

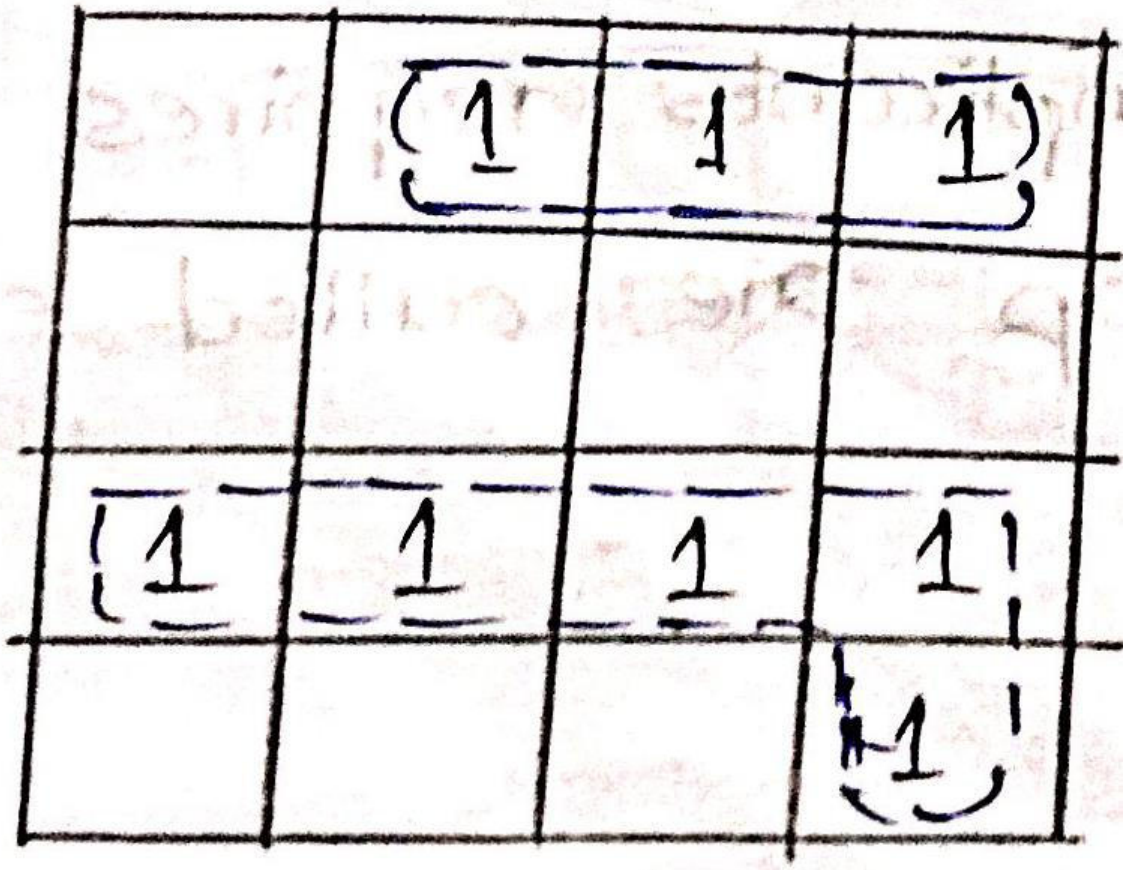
* Illegal grouping:-

① Illegal grouping is diagnol grouping



→ Diagnol grouping.

② Grouping odd no. of one's is also Illegal grouping.



→ odd grouping.

Summmary:

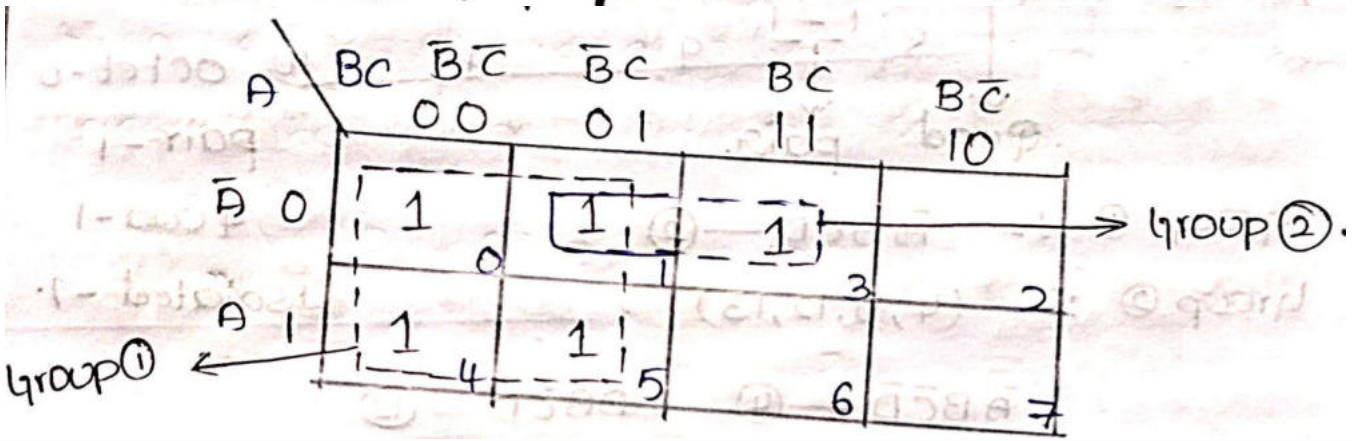
1. No zeros allowed.
2. No diagonals.
3. Only power of 2 number of cells in each group.
4. Groups should be as large as possible.
5. Every one must be in at least one group.
6. Overlapping allowed.
7. Wrap around allowed.
8. Fewest number of groups possible

Example:1 Minimise the expression using K-map

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

Solution: Identify the given expression whether it is in standard SOP form or not.

The given expression is in standard SOP form and it contains 3 variables. Hence it requires 3 variable K-map to simplify the expression.



octet = 0

Quad = 1

pair = 1.

Grouping of octet is not possible

Cell no's 0, 1, 4, 5 form a Quad.

Cell no's 1, 3 form a pair.

Group ① :- $\bar{A}\bar{B}\bar{C}$ — ①

$\bar{A}\bar{B}C$ — ②

$A\bar{B}\bar{C}$ — ④

$A\bar{B}C$ — ⑤

$\bar{B}$

Group ② :- $\bar{A}\bar{B}C$ — ①

$\bar{A}BC$ — ③

$\bar{A}C$

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

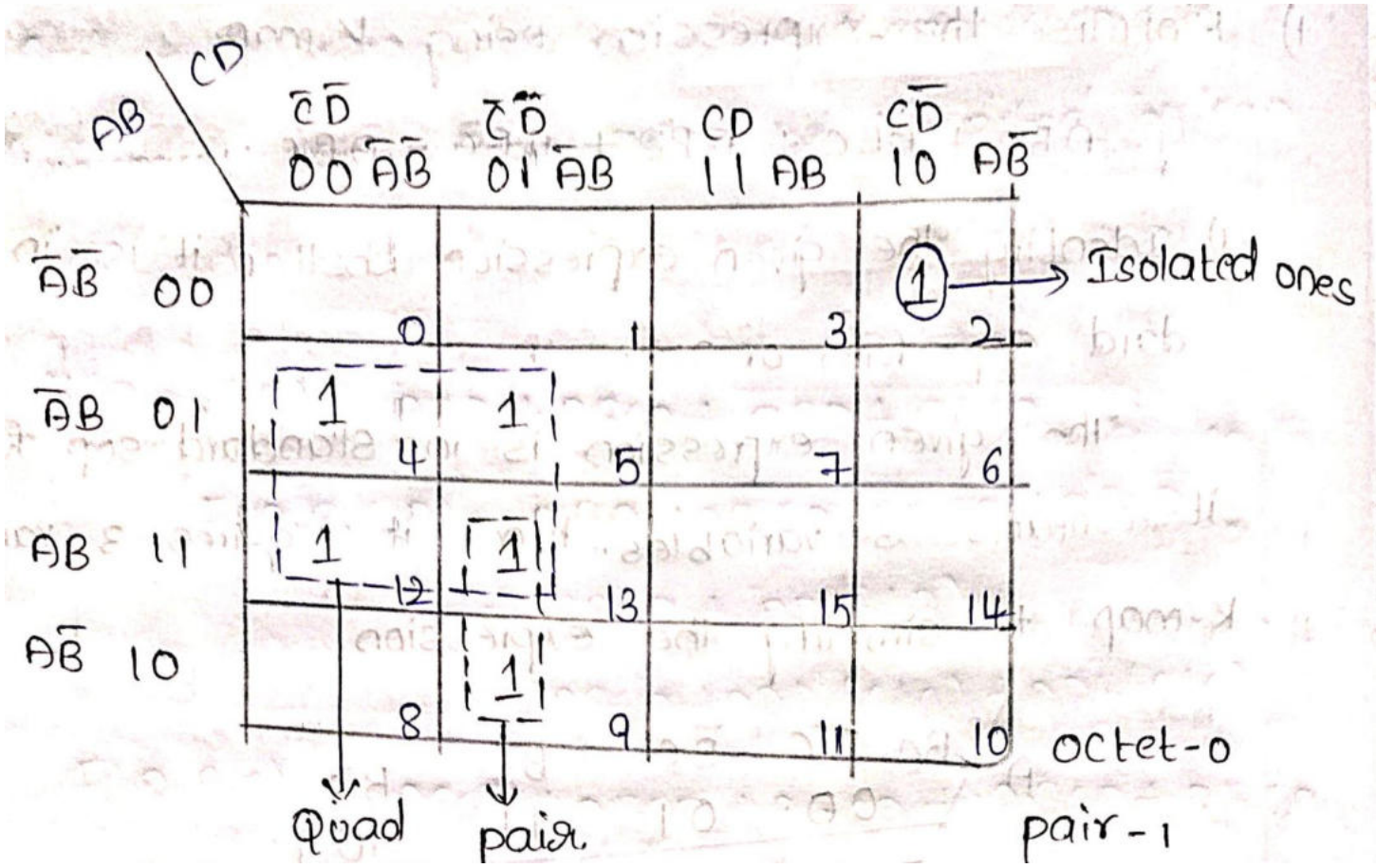
Example: 2 minimize Boolean expression using K-map

$$Y = \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + AB\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + AB\bar{C}D + A\bar{B}\bar{C}D$$

Solution: Given function is

$$Y = \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + AB\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + AB\bar{C}D + A\bar{B}\bar{C}D$$

The given expression is 4 variable and it is in standard SOP form. Hence it can be solve by using 4 variable K-map.



Group ① :- (2)

$\bar{A}\bar{B}C\bar{D}$

Group ② :- (4, 5, 12, 13)

$\cancel{A}B\bar{C}\cancel{D}$

$\cancel{A}B\bar{C}D$

$\cancel{A}B\bar{C}\cancel{D}$

$\cancel{A}B\bar{C}D$

$B\bar{C}$

Group ③ :- 9, 13

$A\cancel{B}\bar{C}D$

$A\cancel{B}\bar{C}D$

$A\bar{C}D$

$$Y = \bar{A}\bar{B}C\bar{D} + B\bar{C} + A\bar{C}D.$$

Example:3 minimize Boolean expression using K-map

$Y = \bar{A}\bar{B}C\bar{D} + AB\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}CD + A\bar{B}\bar{C}D + AB\bar{C}\bar{D} + \bar{A}BCD + \bar{A}\bar{B}\bar{C}\bar{D}$

Solution: Given function is

$Y = \bar{A}\bar{B}C\bar{D} + AB\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}CD + A\bar{B}\bar{C}D + AB\bar{C}\bar{D} + \bar{A}BCD + \bar{A}\bar{B}\bar{C}\bar{D}$

The given expression is 4 variable and it is in standard SOP form. Hence it requires 4 variable K map to simplify given boolean expression.

$$Y = \bar{A}\bar{B}C\bar{D} + AB\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}\bar{D} + A\bar{B}CD + A\bar{B}\bar{C}D + AB\bar{C}\bar{D} + \bar{A}BCD + \bar{A}\bar{B}\bar{C}\bar{D}$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	1	3	2
$\bar{A}B$	4	5	7	6
AB	12	13	15	14
$A\bar{B}$	8	9	11	10

Workout Problems

Example:1 Simplify the Boolean expression using K-map.

$Y = \bar{A}\bar{B}C\bar{D} + AB\bar{C}\bar{D} + A\bar{B}C\bar{D} + A\bar{B}CD + A\bar{B}\bar{C}\bar{D} + AB\bar{C}D + \bar{A}\bar{B}CD + \bar{A}\bar{B}\bar{C}\bar{D}$

Example:2 Minimise using K-map

$Y = F(A,B,C,D) = \Sigma(0,1,4,8,9,10)$

Ans:- $F(A,B,C,D) = \bar{A}\bar{C}\bar{D} + A\bar{B}\bar{D} + \bar{B}\bar{C}$

Example:3 Simplify the Boolean expression using K-map.

$F(A,B,C,D) = \Pi M(0,2,3,8,9,12,13,15)$

Ans:- $F(A,B,C,D) = (\bar{A}+C)(\bar{A}+\bar{B}+\bar{D})(A+B+\bar{C})(A+B+D).$

Workout Problems

Example:4 Use a Karnaugh map to simplify each of the following logic expressions as much as possible.

(*a*) $F = A'B'CD + AB'C'D' + A'B'CD' + ABC'D + ABCD$

Solution. $F = ABD + A'B'D + B'C'$

(*b*) $G = A'C + B'C + AB'C' + A'B$

Solution. $G = AB' + A'B + A'C$ or $AB' + A'B + B'C$

(*c*) $H = B'CD + CD' + A'B'C'D' + A'B'C$

Solution. $H = B'CD + CD' + A'B'C'D' + A'B'C$

(*d*) $F = (A' + B + C')(A + B + C)(A + B + C')$

Solution. $F = B + AC'$

Example:5 Simplify the following logic expressions using K-maps

(*a*) $F(A, B, C) = A'C + B'C + AB'C'$

(*b*) $G(A, B, C, D) = B'CD + CD' + A'B'C'D + A'B'C$

Workout Problems

Example:6

A	B	C	F_1
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

A	B	C	F_2
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Transfer the input-output specifications for F_1 and F_2 given above to 3 variable Karnaugh maps.

Example:7

Simplify the following using Boolean Algebra

(a) $z = w.x + w.\overline{x}.y$

(b) $z = \overline{(x + y) \,.\, (\overline{x} + \overline{y})}$

(c) $z = x.y + w.\overline{y} + w.x + x.y.v$

(d) $z = (x + y).(x + \overline{w}).[y.(x + \overline{w}) + \overline{y}]$

Workout Problems

Example:8 Obtain the simplified expression in s-of-p for the following Boolean functions:

(a) $xy + x'y'z' + x'yz'$

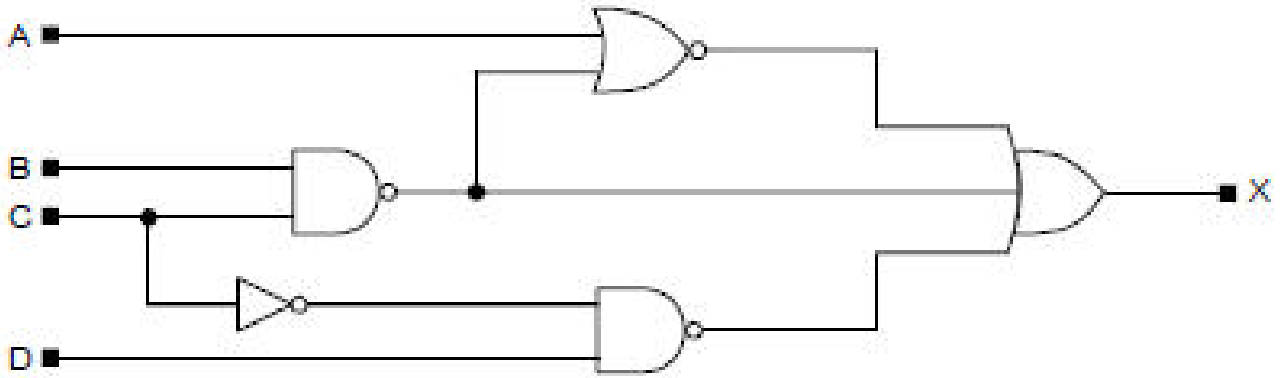
(b) $ABD + A'C'D' + A'B + ACD + AB'D'$

(c) $x'z + w'xy' + w(x'y + xy')$

(d) $F(x, y, z) = \Sigma(2, 3, 6, 7)$

(e) $F(A, B, C, D) = \Sigma(7, 13, 14, 15)$

Example:9 Write the logic expression and simplify it as much as possible and draw a logic diagram that implements the simplified expression.



Example: Reduce following function using K-map technique. Implement using Basic gates, the simplified boolean expression.

$$F(A,B,C,D) = \bar{A}\bar{B}D + AB\bar{C}\bar{D} + \bar{A}BD + AB\bar{C}D$$

Solution: Given function is

$$F(A,B,C,D) = \bar{A}\bar{B}D + AB\bar{C}\bar{D} + \bar{A}BD + AB\bar{C}D$$

The given Boolean function is in **normal SOP** from, it is **necessary** to convert **standard canonical SOP form** for simplifying using K-Map.

$$F(A,B,C,D) = \bar{A}\bar{B}D + AB\bar{C}\bar{D} + \bar{A}BD + AB\bar{C}D$$

↓ C — **Missing Literals** — ↓ C

$$F(A,B,C,D) = \bar{A}\bar{B}D + ABC\bar{D} + \bar{A}BD + ABC\bar{D}$$

$$= \bar{A}D\bar{B}(C+\bar{C}) + ABC\bar{D} + \bar{A}DB(C+\bar{C})$$

$$= ABC\bar{D} + \bar{A}BCD + \bar{A}B\bar{C}D + \bar{A}\bar{B}CD + \bar{A}\bar{B}\bar{C}D$$

$$F(A,B,C) = ABC\bar{D} + \bar{A}BCD + \bar{A}B\bar{C}D + \bar{A}\bar{B}CD + \bar{A}\bar{B}\bar{C}D$$

$\bar{A}\bar{B}$	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
	0	1	3	2
$\bar{A}B$	4	5	7	6
AB	12	13	15	14
$A\bar{B}$	8	9	11	10

Group 2

Group 1

Group ① :- 14

$ABC\bar{D}$

Group ② :- 1,3,5,7

$\bar{A}\bar{B}\bar{C}D$

$\bar{A}\bar{B}CD$

$\bar{A}B\bar{C}D$

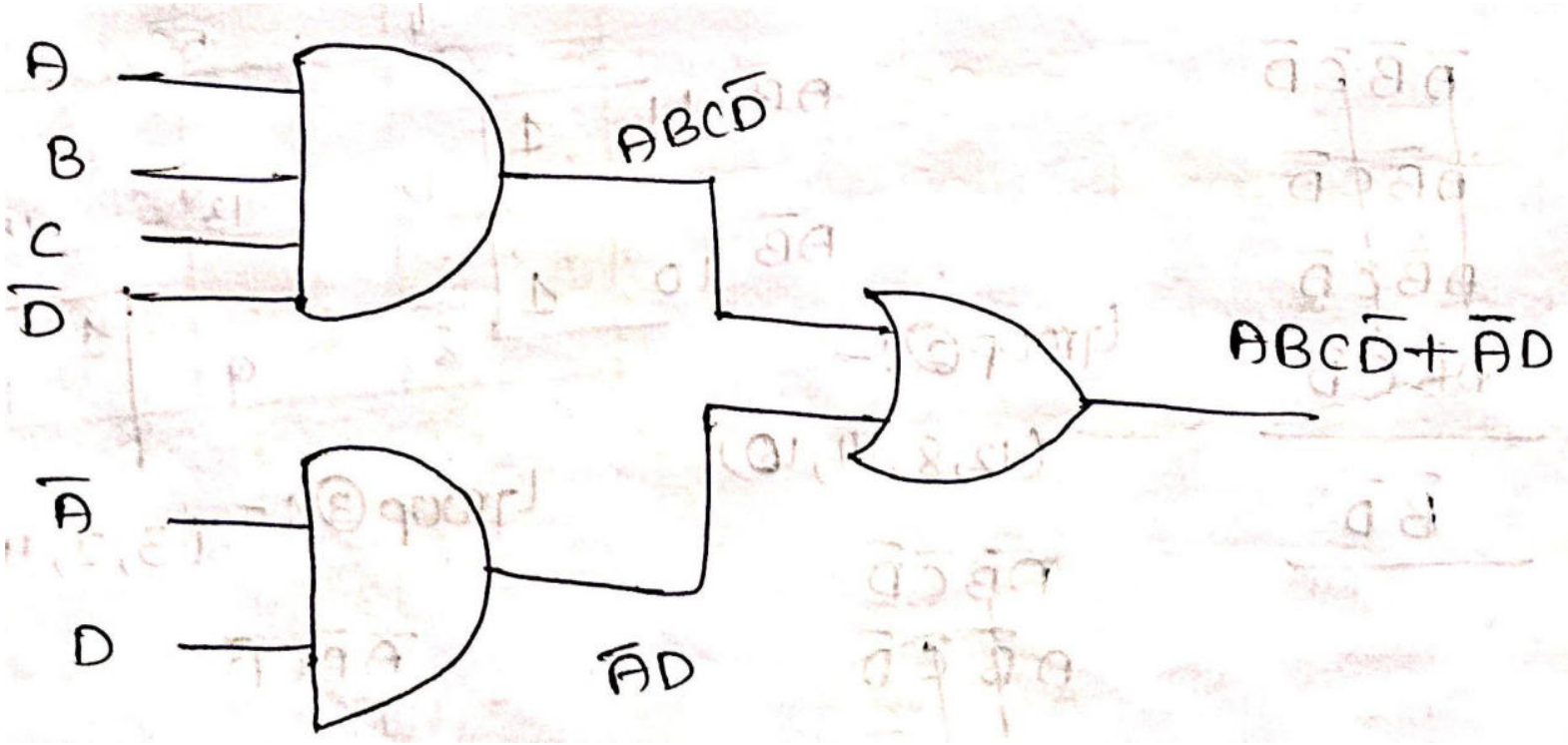
$\bar{A}\bar{B}CD$

$\bar{A}D$

$$Y = ABC\bar{D} + \bar{A}D.$$

Implementation the simplified Boolean function Using basic gates

$$Y = A\,B\,C\,\bar{D} + \bar{A}\,D$$



Example: Reduce following function using K-map technique. Implement using Basic gates, the simplified boolean expression.

$$Y = (A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+B+C)(A+B+C)$$

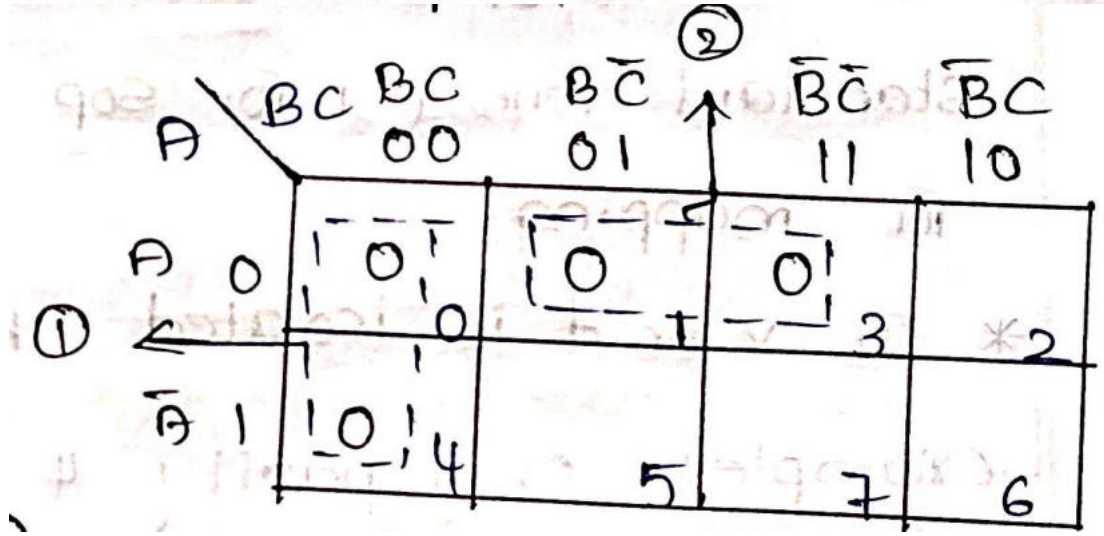
Solution: Given function is

$$Y = (A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+B+C)(A+B+C)$$

The given Boolean function is in **standard canonical POS form**.

The given expression is 3 variable. Hence we can solve it by using 3 variable K-map.

$Y = (A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+B+C)(A+B+C)$



Group ① :-

$A+B+C$

$\bar{A}+B+C$

$B+C$

Group ② :-

$A+B+\bar{C}$

$A+\bar{B}+\bar{C}$

$A+\bar{B}+\bar{C}$

Don't care conditions

Don't care conditions :-

* In some logic circuits certain input conditions never occur. Therefore the corresponding outputs never appear In such cases it can be either high or low.

* These output levels are indicated by "x" or "d". In the truth tables are called don't care outputs or Don't care conditions or incompletley specified functions.

* A circuit designer is free to make the o/p for any Don't care condition either a '0' or a '1'. In order to produce the simpliest output expression.

Note :- * X or d is considered as 1 or 0. As per the standard form (1 for SOP form and 0 for POS form) for mapping.

* If X or d is isolated then it can be neglect.

Example:

Even parity 4 bit BCD.

S.No	A B C D	even parity
0	0 0 0 0	0
1	0 0 0 1	1
2	0 0 1 0	1
3	0 0 1 1	0
4	0 1 0 0	1
5	0 1 0 1	0
6	0 1 1 0	0
7	0 1 1 1	1
8	1 0 0 0	1
9	1 0 0 1	0
10	1 0 1 0	x
11	1 0 1 1	x
12	1 1 0 0	x
13	1 1 0 1	x
14	1 1 1 0	x
15	1 1 1 1	x

→ Don't care conditions.

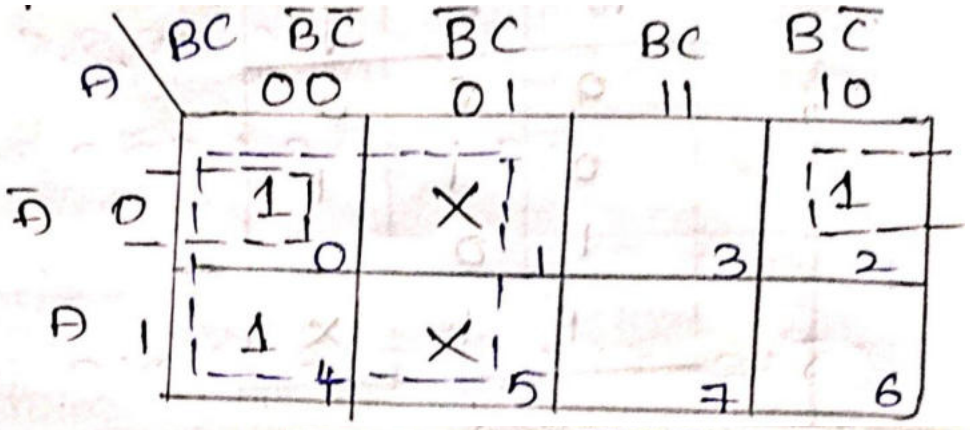
Example: Simplify the given function using K-map.

$$F(A,B,C,D) = m\Sigma(0,2,4) + d(1,5)$$

Solution: Given function is

$$F(A,B,C,D) = m\Sigma(0,2,4) + d(1,5)$$

The given boolean is having 3 variable. Hence we require 3 variable K-map.



$$F(A,B,C) = (\bar{B} + \bar{A}\bar{C})$$

Group ① :- (0, 1, 4, 5)

$\bar{A}\bar{B}\bar{C}$
$\bar{A}\bar{B}C$
$A\bar{B}\bar{C}$
$A\bar{B}C$

$\bar{B}$

Group ② :- (0, 2)

$\bar{A}\bar{B}\bar{C}$
$\bar{A}B\bar{C}$

$\bar{A}\bar{C}$

Example: Simplify the given function using K-map.

$$F(A,B,C) = m\Sigma(1,3,5) + d(6,7)$$

Solution: Given function is

$$F(A,B,C) = m\Sigma(1,3,5) + d(6,7)$$

The given boolean expression is having 3 variables.

Hence we require 3 variable K map.

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	1 (1)	1 (3)	2
A 1	4	1 (5)	X (7)	⊗ 6

$Y = C$

Group ①:- (1, 3, 5, 7)

$\bar{A}\bar{B}C$

$\bar{A}BC$

$A\bar{B}C$

ABC

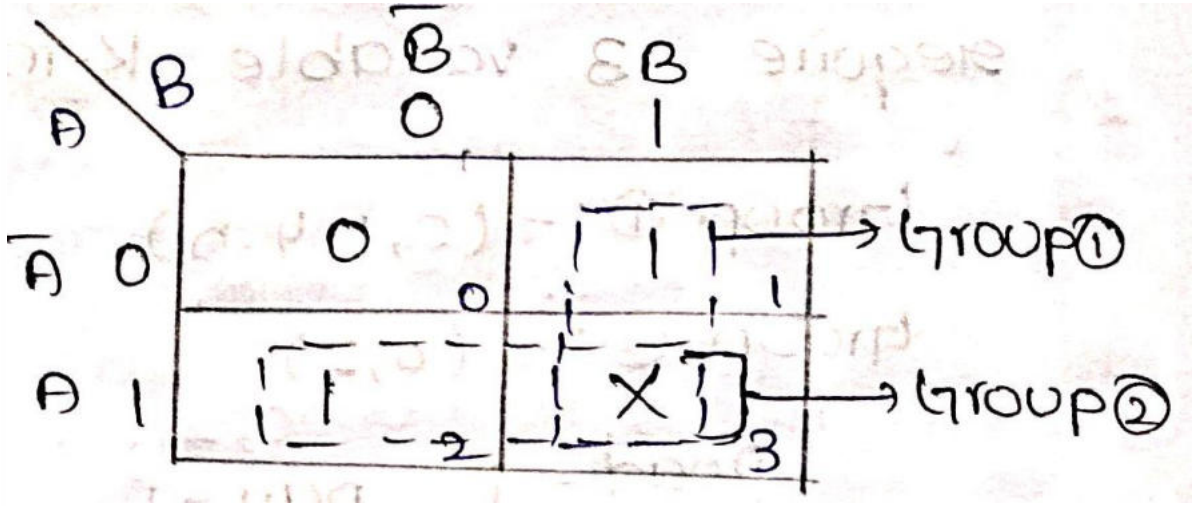
C

Example: Simplify the given function using K-map.

A	B	y
0	0	0
0	1	1
1	0	1
1	1	x

Solution: Given function is

$$F(A,B) = \Sigma m(1,2) + d(3).$$



Group ① :- 1,3

$\bar{A}B$

$A\bar{B}$

AB

B

Group ② :-

$A\bar{B}$

AB

A

$F(A,B) = A+B$

Limitations of K Map

Limitations of K map:-

* K map is continous as long as the no. of variables does not exceed 5.
* As no. of variable increases it is difficult to make which combinations to form the minimum expression in K-map's.
* K map is manual technique and simplification process depend upon the human abilities.

Workout Problems

Example:1

find the reduce sop form for the following functions.

i) $F(A,B,C,D) = \Sigma m(1,3,7,11,15) + \Sigma d(0,2,4)$

ii) $F(A,B,C,D) = \Sigma m(5,6,7,12,13) + \Sigma d(4,9,14,15)$

iii) $F(A,B,C) = \Sigma m(0,1,3,7) + \Sigma d(2,5)$

iv) $F(W,X,Y,Z) = \Sigma m(0,7,9,8,10,12) + \Sigma d(2,5,13)$

v) $F(A,B,C) = \Pi M(2,5,7) + \Pi d(1,3)$

THANK YOU

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES :: RAJAMPET (AUTONOMOUS)

Topic

MINIMIZATION USING QUINE-MCCLUSKEY (TABULAR) METHOD

OVERVIEW

- Minimization using Quine-Mccluskey (Tabular) Method)
- Prime-Implicant chart

Minimization using Quine-Mccluskey (Tabular) Method)

Quine - Mc cluskey method or Tabular method :-

* W. B. Quine and E.j Mc cluskey developed an exact tabular method to simplify boolean expression.
* The fundamental principle of the Q.M. method is that the minterms whose binary equivalent differs only in one place can be combine to reduce the minterms.

* The simplified product terms or sum terms are called an Implicant.

* the remaining terms that did not match during simplification are called prime Implicant.

* Summing one or more prime Implicant gives the simplified boolean expression.

Minimization using Quine-Mccluskey (Tabular) Method)

Procedure

Step: 1 Arrange all minterms in groups, such that all terms in the same group have same number of 1's in their binary representation. Start with the least number of 1's and continue with grouping of increasing number of 1's, the number of 1's in each term is called the index of that term i.e., all the minterms of same index are placed in a same group. The lowest value of index is zero. Separate each group by a thick line. This constitutes the I stage

Step: 2 Compare every term of the lowest index (say i) group with each term in the successive group of index (say, i + 1). If two minterms differ in only one variable, that variable should be removed and a dash (–) is placed at the position, thus a new term with one less literal is formed. If such a situation occurs, a check mark (✔) is placed next to both minterms. After all pairs of terms with indices i and (i + 1) have been considered, a thick line is drawn under the last terms.

When the above process has been repeated for all the groups of I stage, one stage of elimination have been completed. This constitutes the II stage.

Step: 3 The III stage of elimination should be repeated of the newly formed groups of second stage. In this stage, two terms can be compared only when they have dashes in same positions.

The process continues to next higher stages until no further comparisons are possible. (i.e., no further elimination of literals).

Step: 4 All terms which remain unchecked (No ✔ sign) during the process are considered to be prime implicants (PIs). Thus, a set of all PIs of the function is obtained.

Step: 5 From the set of all prime implicates, a set of essential prime implicants (EPIs) must be determined by preparing prime implicant chart as follow.

(a) The PIs should be represented in rows and each minterm of the function in a column.

(b) Crosses should be placed in each row corresponding to minterms that makes the PIs.

(c) A complete PIs chart should be inspected for columns containing only a single cross.

PIs that cover minterms with a single cross in their column are called EPIs

Step: 6 The minterms which are not covered by the EPIs are taken into consideration and a minimum cover is obtained form the remaining PIs.

Example: Simplify the given function using tabular method.
$F(A, B, C, D) = \sum m(0, 2, 3, 6, 7, 8, 10, 12, 13)$

Solution: The given function is $F(A, B, C, D) = \sum m(0, 2, 3, 6, 7, 8, 10, 12, 13)$

Step: 1 The minterms of the function are represened in binary form. The binary represented are grouped into a number of sections interms of the number of 1's index as shown in Table

Minterms	*Binary ABCD*	*No. of 1's*	*Minterms Group*	*Index*	*Binary ABCD*
m_0	0000	0	m_0	0	0000
m_2	0010	1	m_2	1	0010
m_3	0011	2	m_8		1000
m_6	0110	2	m_3	2	0011
m_7	0111	3	m_6		0110
m_8	1000	1	m_{10}		1010
m_{10}	1010	2	m_{12}		1100
m_{12}	1100	2	m_7	3	0111
m_{13}	1101	3	m_{13}		1101

Step: 2 Compare each binary term with every term in the adjacent next higher category. If they differ only by one position put a check mark and copy the term into the next column with (–) in the place where the variable is unmatched, which is shown in next Table

Minterms	*Binary ABCD*	*No. of 1's*	*Minterms Group*	*Index*	*Binary ABCD*
m_0	0000	0	m_0	0	0000✔
m_2	0010	1	m_2	1	0010✔
m_3	0011	2	m_8		1000✔
m_6	0110	2	m_3	2	0011✔
m_7	0111	3	m_6		0110✔
m_8	1000	1	m_{10}		1010✔
m_{10}	1010	2	m_{12}		1100✔
m_{12}	1100	2	m_7	3	0111✔
m_{13}	1101	3	m_{13}		1101✔

Minterm Group	*Binary* A	B	C	D
0, 2	0	0	–	0
0, 8	–	0	0	0
2, 3	0	0	1	–
2, 6	0	–	1	0
2, 10	–	0	1	0
8, 10	1	0	–	0
8, 12	1	–	0	0
3, 7	0	–	1	1
6, 7	0	1	1	–
12, 13	1	1	0	–

Step: 3 Apply same process to the resultant column of above Table and continue until no further elimination of literals. This is shown in below Table.

Minterm Group	*Binary* A	B	C	D	
0, 2	0	0	–	0	✔
0, 8	–	0	0	0	✔
2, 3	0	0	1	–	✔
2, 6	0	–	1	0	✔
2, 10	–	0	1	0	✔
8, 10	1	0	–	0	✔
8, 12	1	–	0	0	PI
3, 7	0	–	1	1	✔
6, 7	0	1	1	–	✔
12, 13	1	1	0	–	PI

Minterm Group	*Binary* A	B	C	D	
0, 2, 8, 10	–	0	–	0	PI
0, 8, 2, 10	–	0	–	0	PI eliminated
2, 3, 6, 7	0	–	1	–	PI
2, 6, 3, 7	0	–	1	–	PI eliminated.

Step: 4 All terms which remain unchecked are the PIs. However note that the minterms combination (0, 2) and (8, 10) form the same combination (0, 2, 8, 10) as the combination (0, 8) and (2. 10). The order in which these combinations are placed does not prove any effect. Moreover, as we know that x + x = x, thus, we can eliminate one of these combinations.

The same occur with combination (2, 3) and (6, 7).

Step: 5 Now we prepare a PI chart to determine EPIs as follows shown in below Table

Prime Implicants	*Minterms*								
	0	*2*	*3*	*6*	*7*	*8*	*10*	*12*	*13*
(8, 12)						×		×	
(12, 13) *								×	×
(0, 2, 8, 10) *	×	×				×	×		
(2, 3, 6, 7) *		×	×	×	×				
	✔		✔	✔	✔		✔		✔

(a) All the PIs are represented in rows and each minterm of the function in a column.

(b) Crosses are placed in each row to show the composition of minterms that make PIs.

(c) The column that contains just a single cross, the PI corresponding to the row in which the cross appear is essential. Prime implicant. A tick mark is part against each column which has only one cross mark. A star (*) mark is placed against each. EPI.

Step: 6 All the minterms have been covered by EPIs.
Finally, the sum of all the EPIs gives the function in its minimal SOP form

EPIs.	*Binary representation*				*Variable Representation*
	A	*B*	*C*	*D*	
12, 13	1	1	0	–	ABC′
0, 2, 8, 10	–	0	–	0	B′D′
2, 3, 6, 7	0	–	1	–	A′C

Therefore, F = ABC' + B'D' + A'C.

Note: If don't care conditions are given, they are also used to find the prime implicating, but it is not compulsory to include them in the final simplified expression.

Minterms	Binary ABCD	No. of 1's
m_0	0000	0
m_2	0010	1
m_3	0011	2
m_6	0110	2
m_7	0111	3
m_8	1000	1
m_{10}	1010	2
m_{12}	1100	2
m_{13}	1101	3

Minterms Group	Index	Binary ABCD
m_0	0	0000✔
m_2	1	0010✔
m_8		1000✔
m_3	2	0011✔
m_6		0110✔
m_{10}		1010✔
m_{12}		1100✔
m_7	3	0111✔
m_{13}		1101✔

Minterm Group	Binary A	B	C	D	
0, 2	0	0	–	0	✔
0, 8	–	0	0	0	✔
2, 3	0	0	1	–	✔
2, 6	0	–	1	0	✔
2, 10	–	0	1	0	✔
8, 10	1	0	–	0	✔
8, 12	1	–	0	0	PI
3, 7	0	–	1	1	✔
6, 7	0	1	1	–	✔
12, 13	1	1	0	–	PI

Minterm Group	Binary A	B	C	D
0, 2, 8, 10	–	0	–	0
0, 8, 2, 10	–	0	–	0
2, 3, 6, 7	0	–	1	–
2, 6, 3, 7	0	–	1	–

PI chart to determine EPIs

Prime Implicants	Minterms 0	2	3	6	7	8	10	12	13
(8, 12)						×		×	
(12, 13) *								×	×
(0, 2, 8, 10) *	×	×				×	×		
(2, 3, 6, 7) *		×	×	×	×				
	✔		✔	✔	✔		✔		✔

EPIs.	Binary representation A	B	C	D	Variable Representation
12, 13	1	1	0	–	ABC'
0, 2, 8, 10	–	0	–	0	B'D'
2, 3, 6, 7	0	–	1	–	A'C

Therefore, F = ABC' + B'D' + A'C.

Example: Simplify the given function using tabular method.
$F(A, B, C, D) = \Sigma m\ (0, 2, 3, 6,7)+d\ (5, 8, 10, 11, 15)$

Solution: The given function is $F(A, B, C, D) = \Sigma m\ (0, 2, 3, 6,7)+d\ (5, 8, 10, 11, 15)$

Step: 1 The minterms of the function are represened in binary form. The binary represented are grouped into a number of sections interms of the number of 1's index as shown in Table

Minterms	*Binary ABCD*	*No. of 1's*	*Minterms Group*	*Index*	*Binary ABCD*
m_0	0000	0	m_0	0	0000
m_2	0010	1	m_2		0010
m_3	00 11	2	m_8	1	1000
m_5	0101	2	m_3		0011
m_6	0110	2	m_5	2	0101
m_7	0111	3	m_6		0110
m_8	1000	1	m_{10}		1010
m_{10}	1010	2	m_7	3	0111
m_{11}	1011	3	m_{11}		1011
m_{15}	1111	4	m_{15}	4	1111

Step: 2 Compare each binary term with every term in the adjacent next higher category. If they differ only by one position put a check mark and copy the term into the next column with (–) in the place where the variable is unmatched, which is shown in next Table

Minterms	*Binary ABCD*	*No. of 1's*
m_0	0000	0
m_2	0010	1
m_3	0011	2
m_5	0101	2
m_6	0110	2
m_7	0111	3
m_8	1000	1
m_{10}	1010	2
m_{11}	1011	3
m_{15}	1111	4

Minterms Group	*Index*	*Binary ABCD*
m_0	0	0000✔
m_2	1	0010✔
m_8		1000✔
m_3	2	0011✔
m_5		0101✔
m_6		0110✔
m_{10}		1010✔
m_7	3	0111✔
m_{11}		1011✔
m_{15}	4	1111✔

Minterm Group	*Binary A B C D*
0, 2	0 0 – 0
0, 8	– 0 0 0
2, 3	0 0 1 –
2, 6	0 – 1 0
2, 10	– 0 1 0
8, 10	1 0 – 0
3, 7	0 – 1 1
3, 11	– 0 1 1
5, 7	0 1 – 1
6, 7	0 1 1 –
10, 11	1 0 1 –
7, 15	– 1 1 1
11, 15	1 – 1 1

Step: 3 Apply same process to the resultant column of above Table and continue until no further elimination of literals. This is shown in below Table.

Minterm Group	A	B	C	D	
0, 2	0	0	–	0	✔
0, 8	–	0	0	0	✔
2, 3	0	0	1	–	✔
2, 6	0	–	1	0	✔
2, 10	–	0	1	0	✔
8, 10	1	0	–	0	✔
3, 7	0	–	1	1	✔
3, 11	–	0	1	1	✔
5, 7	0	1	–	1	PI
6, 7	0	1	1	–	✔
10, 11	1	0	1	–	✔
7, 15	–	1	1	1	✔
11, 15	1	–	1	1	✔

Minterm Group	A	B	C	D	
0, 2, 8, 10	–	0	–	0	PI
0, 8, 2, 10	–	0	–	0	PI Eliminated
2, 3, 6, 7	0	–	1	–	PI
2, 3 10, 11	–	0	1	–	PI
2, 6, 3, 7	0	–	1	–	PI Eliminated
2, 10, 3, 11	–	0	1	–	PI Eliminated
3, 7, 11, 15	–	–	1	1	PI
3, 11, 7, 15	–	–	1	1	PI Eliminated

Step: 4 All the terms which remain unchecked are PIs. Moreover, one of two same combinations is eliminated.

Step: 5 Now we prepare a PI chart to determine EPIs as follows shown in below Table

Prime Implicants	*Minterms*				
	0	*2*	*3*	*6*	*7*
(5, 7)					×
(0, 2, 8, 10) *	×	×			
(2, 3, 6, 7) *		×	×	×	×
(2, 3, 10, 11)		×	×		
(3, 7, 11, 15)			×		×
	✔			✔	

Step: 6 The minterms have been covered by EPIs

EPIs.	*Binary representation*				*Variable Representation*
	A	*B*	*C*	*D*	
0, 2, 8, 10	–	0	–	0	B′D′
2, 3, 6, 7	0	–	1	–	A′C

Therefore, F = B' D' + A'C.

Example: Simplify the given function using tabular method:
F (A, B, C, D, E, F, G) = Σ m(20, 28, 38, 39, 52, 60, 102, 103, 127)

Solution: The given function is **F (A, B, C, D, E, F, G) = Σ m(20, 28, 38, 39, 52, 60, 102, 103, 127)**

Minterms	*Binary ABCDEFG*	*No. of 1's*
m_{20}	0010100	2
m_{28}	0011100	3
m_{38}	0100110	3
m_{39}	0100111	4
m_{52}	0110100	3
m_{60}	0111100	4
m_{102}	1100110	4
m_{103}	1100111	5
m_{127}	1111111	7

Minterms Group	*Index*	*Binary ABCDEFG*
m_{20}	2	0010100 ✔
m_{28}	3	0011100 ✔
m_{38}		0100110 ✔
m_{52}		0110100 ✔
m_{39}	4	0100111 ✔
m_{60}		0111100 ✔
m_{102}		1100110 ✔
m_{103}	5	1100111 ✔
m_{127}	7	1111111 PI

Minterms Group	*Binary* A	B	C	D	E	F	G
20, 28	0	0	1	–	1	0	0
20, 52	0	–	1	0	1	0	0
28, 60	0	–	1	1	1	0	0
38, 39	0	1	0	0	1	1	–
38, 102	–	1	0	0	1	1	0
52, 60	0	1	1	–	1	0	0
39, 103	–	1	0	0	1	1	1
102, 103	1	1	0	0	1	1	–

Minterms	Binary ABCDEFG	No. of 1's
m_{20}	0010100	2
m_{28}	0011100	3
m_{38}	0100110	3
m_{39}	0100111	4
m_{52}	0110100	3
m_{60}	0111100	4
m_{102}	1100110	4
m_{103}	1100111	5
m_{127}	1111111	7

Minterms Group	Index	Binary ABCDEFG
m_{20}	2	0010100 ✔
m_{28}		0011100 ✔
m_{38}	3	0100110 ✔
m_{52}		0110100 ✔
m_{39}	4	0100111 ✔
m_{60}		0111100 ✔
m_{102}		1100110 ✔
m_{103}	5	1100111 ✔
m_{127}	7	1111111 PI

Minterms Group	Binary A	B	C	D	E	F	G	
20, 28	0	0	1	–	1	0	0	✔
20, 52	0	–	1	0	1	0	0	✔
28, 60	0	–	1	1	1	0	0	✔
38, 39	0	1	0	0	1	1	–	✔
38, 102	–	1	0	0	1	1	0	✔
52, 60	0	1	1	–	1	0	0	✔
39, 103	–	1	0	0	1	1	1	✔
102, 103	1	1	0	0	1	1	–	✔

Mintesms Group	Binary A	B	C	D	E	F	G
20, 28, 52, 60	0	–	1	–	1	0	0
20, 52, 28, 60	0	–	1	–	1	0	0
38, 39 102, 103	–	1	0	0	1	1	–
38, 102, 39, 103	–	1	0	0	1	1	–

Now we prepare a PI chart to determine EPIs

All the minterms have been covered by EPIs

Prime Implicants		Minterms 20	28	38	39	52	60	102	103	127
127	*									×
(20, 28, 52, 60)	*	×	×			×	×			
(38, 39, 102, 103)	*			×	×			×	×	
		✔	✔	✔	✔	✔	✔	✔	✔	✔

EPIs.	Binary representation A	B	C	D	E	F	G	Variable Representation
127	1	1	1	1	1	1	1	ABCDEFG
20, 28, 52, 60	0	–	1	–	1	0	0	A′CEF′G′
38, 39 102, 103	–	1	0	0	1	1	–	BC′D′EF

Therefore, F (A, B, C, D, E, F, G) = ABCDEFG + A'CEF'G' + BC'D'EF

THANK YOU