



Dr. Vishwanath Karad

**MIT WORLD PEACE
UNIVERSITY** | PUNE

TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Digital Electronics and Computer Architecture

SCHOOL OF COMPUTER ENGINEERING AND TECHNOLOGY

Digital Electronics and Computer Architecture

Course Objectives:

By participating in and understanding all facets of this Course a student will be able:

- 1. To learn number systems, Boolean algebra and to introduce the concepts of digital logic families.**
- 2. To develop skills for design and implementation of combinational logic circuits.**
- 3. To develop skills for design and implementation of sequential circuits**
- 4. To understand the functions, characteristics of various components of computer, instruction set, addressing mode of computer architecture and Pipeline Hazards**
- 5. To acquire the knowledge of Control unit architecture and Memory Organization**

Digital Electronics and Computer Architecture

Course Outcomes:

On completion of course, students will be able to

- 1. Identify and use the basic logic gates and solve logic equations with various reduction techniques of digital logic circuits.**
- 2. Design and implement combinational logic circuits**
- 3. Design and implement sequential logic circuits**
- 4. Recognize the functions and organization of various blocks of CPU. Describe CPU instruction characteristics used in Pipelining concept and its relevance with Processor's performance.**
- 5. Demonstrate computer architecture concepts related control unit and memory**

Syllabus

Module 1 Computer Arithmetic:

- Introduction to digital logic families, Introduction to Number System and Codes, Sign Magnitude representation, 1's and 2's complement, Binary Arithmetic: Addition, Subtraction of signed numbers, multiplication of positive numbers, Integer Arithmetic, Floating point representation and operations – IEEE standard, Boolean Algebra, Logic Gates, Minimization of logic functions using K map (SOP and POS)

Module 2 Combinational Logic Design:

- Design examples: Arithmetic circuits, Comparator, Code converters, Parity generators and checkers, Arithmetic and Logic Unit, design of adder and fast adder, look ahead carry generator, Implementation of SOP and POS using MUX, DMUX, DECODER.

Module 3 Sequential Logic Design:

- 1-bit Memory Cell, Flip flops, Conversion of flip flops, Design of ripple counters and synchronous counters, Modulus of the counter, Sequence generator, Lock out condition. Shift registers, Applications of Shift registers (ring and twisted ring counters), Moore and Mealy Models- State diagram, State Tables and Design Procedure, Sequence detector.

Syllabus

Module 4 Fundamental of Computer Architectures:

- Structure and Function, basic operational concepts of bus structures Memory locations and addresses functions. Types of computer units (CPU, Memory, I/O, System Bus), Von Neumann & Harvard architecture, Key characteristics of RISC & CISC.
- ALU: Multiplication – Block diagram, Hardware implementation of unsigned binary and signed number, Booth's Algorithm, Division Algorithms (Unsigned Binary).
- Processor organization, Register organization- user visible registers, control and status registers-Case Study 8086. Instruction Cycle- The machine cycle and Data flow. Types of operands, Types of operations, Types of instructions, Addressing modes and Instruction Formats- instruction length, allocation of bits, variable length instructions instruction formats - Case Study- 8086. Processor and Instruction Pipelining, Pipeline Performance, Pipeline Hazards - Structural, Data, Control.

Syllabus

Module 5 Control Unit and Memory System:

- Hardwired control, Micro-programmed control- micro instructions, micro-program sequencing, wide branch addressing, microinstruction with next address field, prefetching microinstructions and emulation.
- Memory hierarchy, characteristics, Cache Memory- Cache memory principles, Elements of cache design- cache address, size, mapping functions, replacement algorithms, write policy, line size, number of cache, one level and two level cache, performance characteristics of two level cache- locality & operations, main memory and secondary storage. Pentium IV cache organization

Syllabus

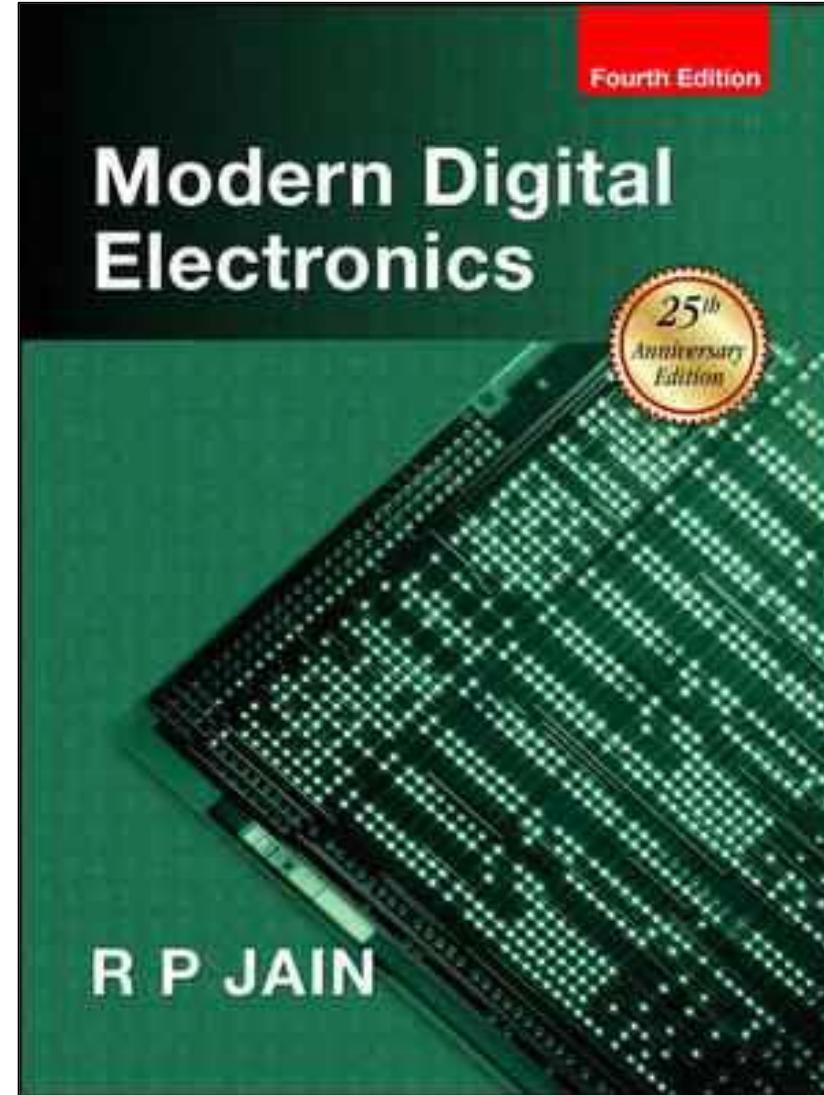
List of Assignments:

- 1. Design and Implement Code Converters using logic gates**
- 2. Design and Implement Combinational Logic Design using MUX/Decoder ICs**
- 3. Design and Implement asynchronous counter using JK- Flip flop.**
- 4. Design and Implement MOD-N asynchronous counter using JK- Flip flop /IC7490.**
- 5. Design and implement Synchronous Counter**
- 6. Design of a sequence detector**
- 7. Write an assembly language program (ALP) to display 2-digit and 4-digit hex numbers using 64- bit assembly language programming**
- 8. Write an assembly language program (ALP) to accept 2-digit and 4-digit hex numbers user input using 64- bit assembly language programming.**
- 9. Write an assembly language program (ALP) to implement addition and subtraction of 8-bit numbers (Using user input, macro and procedure).**
- 10. Write an ALP to perform basic string operations. (length and Reverse)**

Module 1 Computer Arithmetic

- **Introduction to digital logic families,**
- **Introduction to Number System and Codes,**
- **Sign Magnitude representation, 1's and 2's complement,**
- **Binary Arithmetic: Addition, Subtraction of signed numbers, multiplication of positive numbers, Integer Arithmetic,**
- **Floating point representation and operations – IEEE standard, Boolean Algebra, Logic Gates, Minimization of logic functions using K map (SOP and POS)**

Text book soft copy available



Digital and Analog Quantities

For many years, applications of digital electronics were limited to computer systems.

Digital Technology have wide range of applications

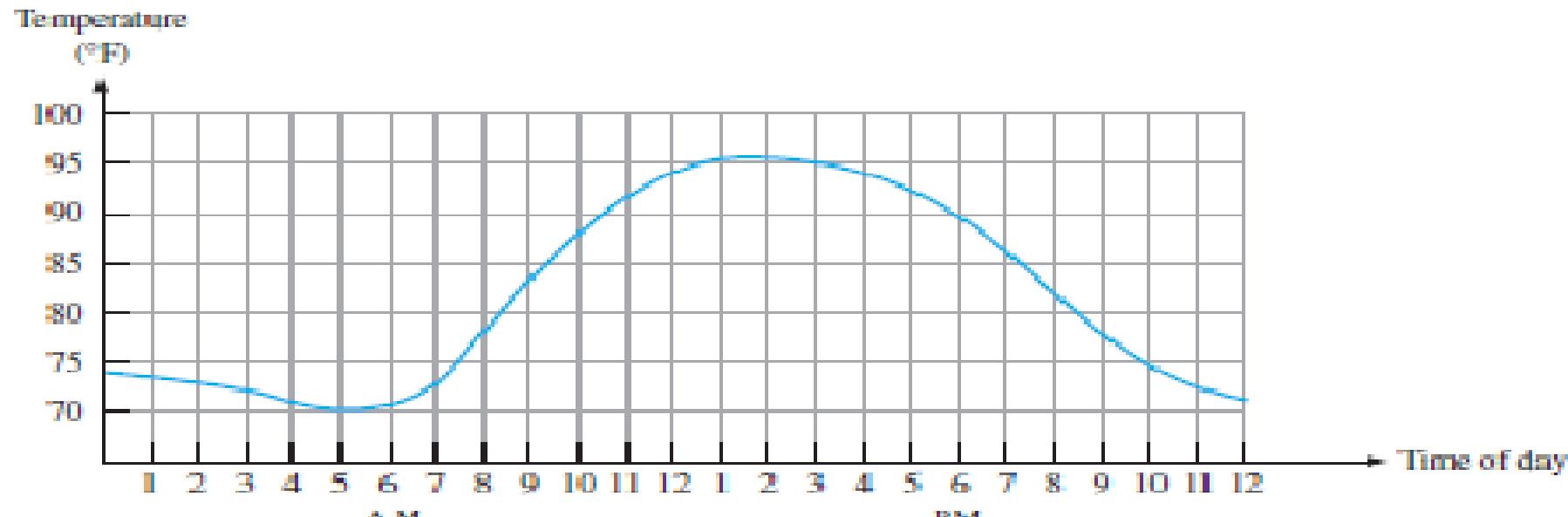
- ✓ Television
- ✓ Communication systems
- ✓ Radar
- ✓ Navigation
- ✓ Military Systems
- ✓ Medical Instrumentations
- ✓ Industrial Process Control
- ✓ Consumer electronics etc.

Electronics circuits can be divided into two broad categories

- ✓ Digital
- ✓ Analog

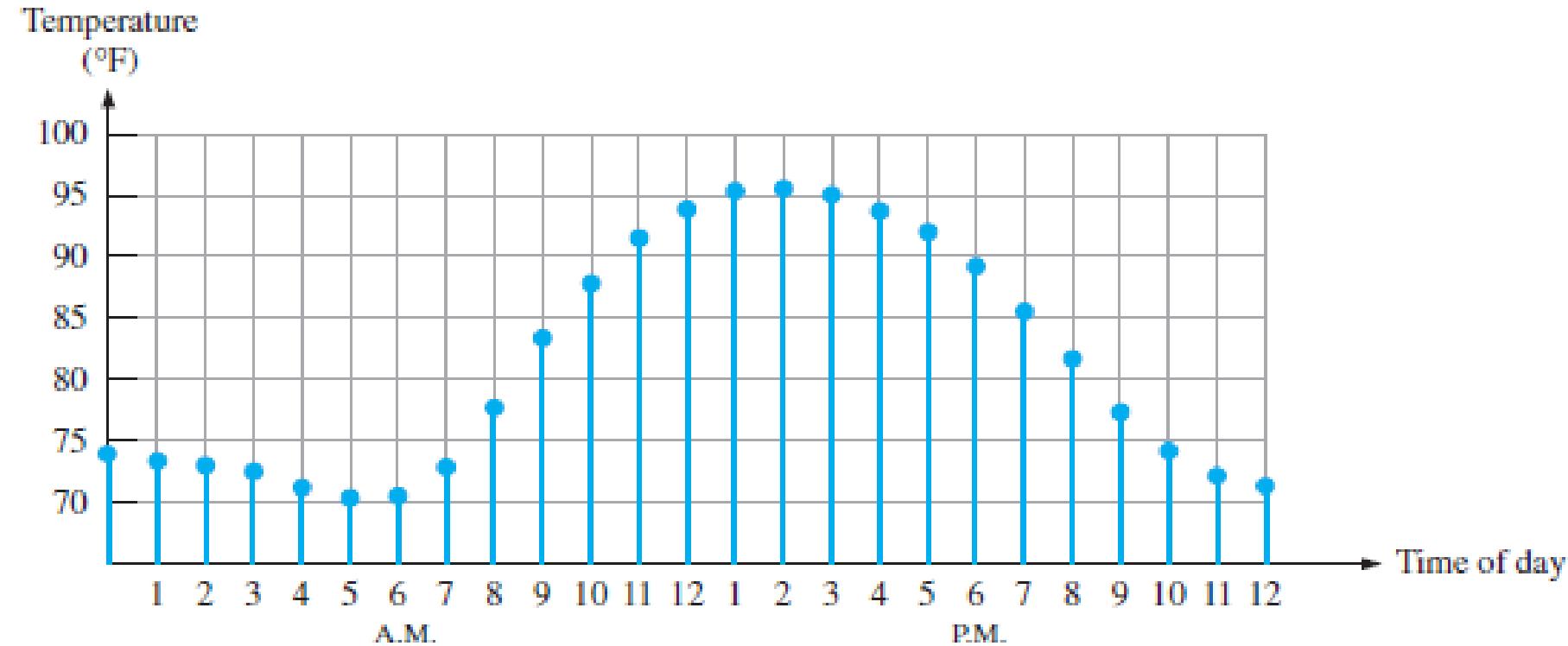
Digital and Analog Quantities

- Analog: is one having continuous values
- Digital: is one having discrete set of values
- Example: Air temperature changes over continuous range of values
- **Analog Signal**



Graph of an analog quantity (temperature versus time).

Digital Code

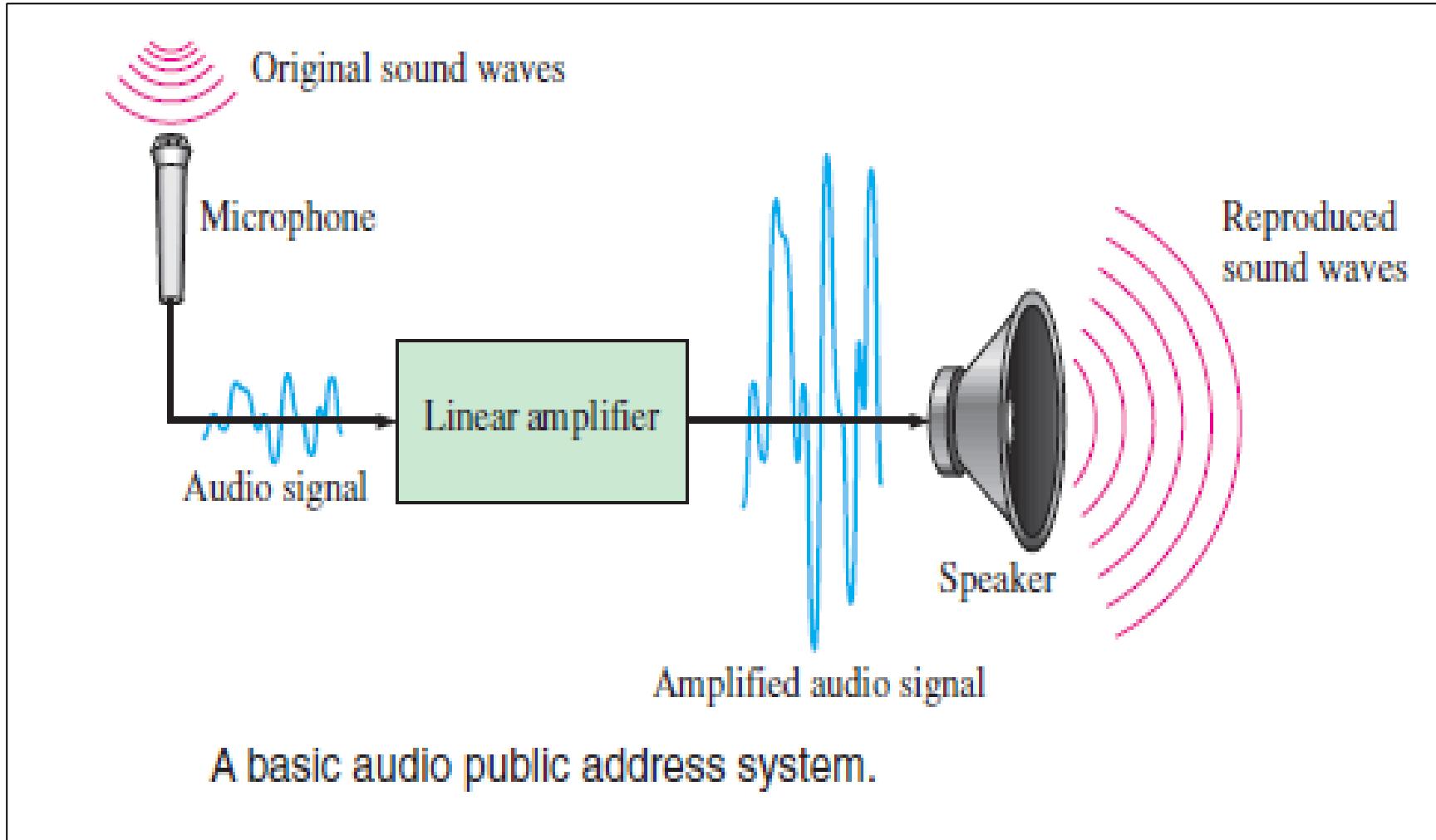


| Sampled-value representation (quantization) of the analog quantity in
Each value represented by a dot can be digitized by representing it as a digital
code that consists of a series of 1s and 0s.

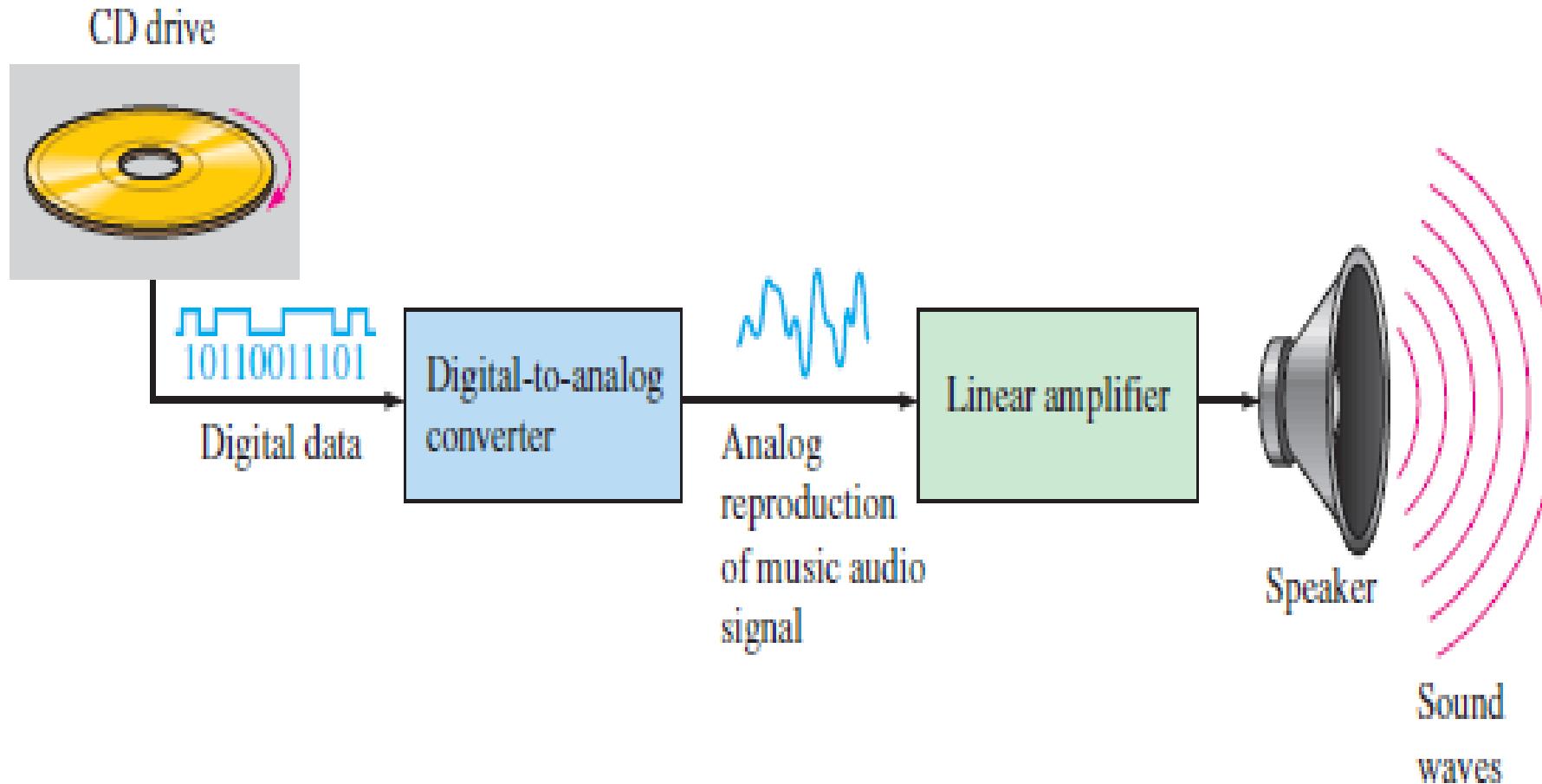
Advantages of Digital Signals

- Data transmission is more effective and reliable
- Digital data has a great advantage when storage is necessary
- Digital data is more accurate and precise
- Its ripple free or noise free

Analog Systems



A System Using Analog and Digital Methods



Basic block diagram of a CD player. Only one channel is shown.

Mechatronics

- The interdisciplinary field that comprises both mechanical and electronic components is known as mechatronics
- Both digital and analog electronics are used in the control of various mechanical systems.



(b) Robotic arm



(c) Automotive assembly line

Digital Electronics

- It uses Binary Digits 0 and 1
- Each of the two digits in the binary system, 1 and 0, is called a bit, Two different voltage levels are used to represent the two bits.
- 1 - is represented by the higher voltage, which will be referred as a HIGH,
- 0 - is represented by the lower voltage level, which will be referred as a LOW.
- This is called positive logic and will be used throughout the topic.

HIGH = 1 and LOW = 0

Number System and Binary Codes

Number System and Binary Codes

- Number systems
- Representation of unsigned and signed integers
- Fixed-point representation of real numbers
- Floating-point representation of real numbers
- Binary representation
- Codes and their conversions
- Binary arithmetic

Number Systems

■ Decimal

$$1234_{10} = (1 \times 10^3) + (2 \times 10^2) + (3 \times 10^1) + (4 \times 10^0)$$

■ Binary

$$1101_2 = (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0)$$

■ Octal

$$123_8 = (1 \times 8^2) + (2 \times 8^1) + (3 \times 8^0)$$

■ Hexadecimal

$$123_{16} = (1 \times 16^2) + (2 \times 16^1) + (3 \times 16^0)$$

here we need 16 characters – 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Binary number into Decimal

- Convert the given binary number into decimal equivalent number

$$\begin{aligned}1. \quad (100010)_2 &= 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 0 \times 2^3 + 0 \times 2^4 + 1 \times 2^5 \\&= 0 + 2 + 0 + 0 + 0 + 32 \\&= (34)_{10}\end{aligned}$$

$$\begin{aligned}2. \quad (0.1011)_2 &= 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} \\&= 0.5 + 0 + 0.125 + 0.0625 \\&= (0.6875)_{10}\end{aligned}$$

$$\begin{aligned}3. \quad (10101.011)_2 \\ \text{Integer part: } (10101)_2 &= 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 \\&= 1 + 0 + 4 + 0 + 16 = (21)_{10}\end{aligned}$$

$$\begin{aligned}\text{Decimal part: } (0.011)_2 &= 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\&= (0.375)_{10} \\&= (21.375)_{10}\end{aligned}$$

Hexadecimal Number System

| Hexadecimal number system | Equivalent binary number |
|---------------------------|--------------------------|
| | 8 4 2 1 (weights) |
| 0 | 0 0 0 0 |
| 1 | 0 0 0 1 |
| 2 | 0 0 1 0 |
| 3 | 0 0 1 1 |
| 4 | 0 1 0 0 |
| 5 | 0 1 0 1 |
| 6 | 0 1 1 0 |
| 7 | 0 1 1 1 |
| 8 | 1 0 0 0 |
| 9 | 1 0 0 1 |
| 10 | 1 0 1 0 = A |
| 11 | 1 0 1 1 = B |
| 12 | 1 1 0 0 = C |
| 13 | 1 1 0 1 = D |
| 14 | 1 1 1 0 = E |
| 15 | 1 1 1 1 = F |

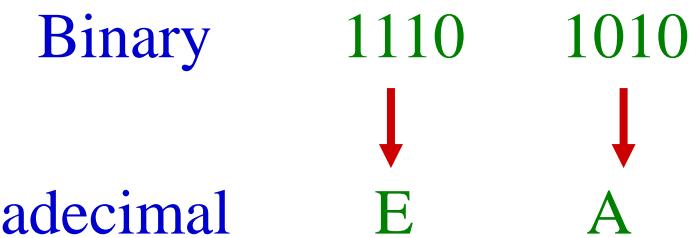
Hexadecimal Number System:

The hexadecimal number system has base 16. It has 16 distinct digit symbols. It uses the digits 0-9 & letters A,B,C,D,E,F as 16 digit symbols.

- **Hexadecimal to Binary Conversion**



- **Binary to Hexadecimal Conversion**



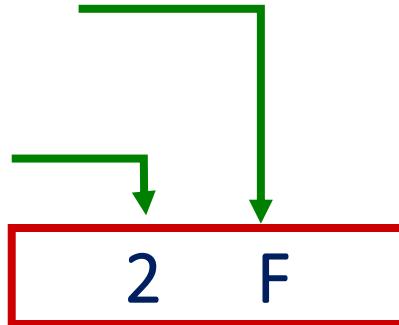
Hexadecimal Number System

- DECIMAL TO HEXADECIMAL CONVERSION

Divide by 16 Process

Decimal # $47 \div 16 = 2$ remainder 15

$2 \div 16 = 0$ remainder 2



HEXADECIMAL NUMBER INTO DECIMAL NUMBER SYSTEM

$$\begin{aligned}
 1. (2B8D.E2)_{16} &= 2 \times 16^3 + 11 \times 16^2 + 8 \times 16^1 + 13 \times 16^0 + 14 \times 16^{-1} + 2 \times 16^{-2} \\
 &= 8192 + 2816 + 128 + 13 + 0.875 + 0.0078125 \\
 &= (11149.88281)_{10}
 \end{aligned}$$

$$(269)_{16} = ?$$

Answer: $(617)_{10}$

Octal Number System

Octal number system into Binary number system:

$(7423.245)_8$

| | | | | | | | |
|-----|-----|-----|-----|---|-----|-----|-----|
| 7 | 4 | 2 | 3 | . | 2 | 4 | 5 |
| 111 | 100 | 010 | 011 | . | 010 | 100 | 101 |

Conversion from Binary number system to Octal number system:

$(11101101110.11111)_2$

| | | | | | | |
|-----|-----|-----|-----|---|-----|-----|
| 011 | 101 | 101 | 110 | . | 111 | 110 |
| 3 | 5 | 5 | 6 | . | 7 | 6 |

A number system that uses 8 digit (0-7) is called an octal number system. It has base 8.

| Octal Numbers | Binary Equivalent number |
|---------------|--------------------------|
| | 4 2 1 (weights) |
| 0 | 0 0 0 |
| 1 | 0 0 1 |
| 2 | 0 1 0 |
| 3 | 0 1 1 |
| 4 | 1 0 0 |
| 5 | 1 0 1 |
| 6 | 1 1 0 |
| 7 | 1 1 1 |

Octal Number System

- Octal to Decimal Conversion
- Multiplying each Octal digit by its positional weight

$$\begin{aligned}
 (314)_8 &= 3 \times 8^3 + 3 \times 8^2 + 1 \times 8^1 + 4 \times 8^0 \\
 &= 192 + 8 + 4 \\
 &= (204)_{10}
 \end{aligned}$$

- Decimal to Octal Conversion
- Repeated division or double dabble method

$$(1375)_{10} = (2537)_8$$

remainder

| | | |
|---|-------------|----------|
| 8 | 1375 | |
| 8 | 171 | 7 |
| 8 | 21 | 3 |
| | 2 | 5 |
| | | 2 |



MSB

- **Hexadecimal number system to Octal number system:**

First write down the 4 bit binary equivalent of hexadecimal digit and then rearranging into group of three bit each.

$$(2AB.9)_{16} = (?)_8$$

| | |
|-------------------------------|--|
| Hexadecimal equivalent | 2 A B . 9 0010 1010 1011 . 1001 |
| Octal equivalent | <u>001</u> <u>010</u> <u>101</u> <u>011</u> . <u>100</u> <u>100</u> 1 2 5 3 . 4 4 |

Convert $(1375)_8 = ()_{16}$

$$1375 = 001\ 011\ 111\ 101$$

Now 001011111101

$$0010\ 1111\ 1101$$



$$\begin{matrix} 2 & 15 & 13 \end{matrix}$$

$$\begin{matrix} 2 & F & D \end{matrix}$$

TEST



- Convert Hexadecimal number **A6** to Binary

1010 0110 (Binary)

- Convert Hexadecimal number **16** to Decimal

22 (Decimal)

- Convert Decimal **63** to Hexadecimal

3F (Hexadecimal)

- Convert Decimal **41.915** to Binary

101001.11101

- Convert hexadecimal **7FD6** to Octal

77726

Takeaway

- Digital computers are implemented via logic circuits and thus represent *all* numbers in binary (base 2).
- We (humans) often write numbers as decimal and hexadecimal for convenience, so need to be able to convert to binary and back (to understand what computer is doing!).

Representation of Unsigned and Signed Integers

- Are the negative numbers just numbers with a minus sign in the front? This is probably true...but there are issues to represent negative numbers in computing systems
- Common schemas:
 - Sign-magnitude
 - Complementary representations:
 - 1's complement
 - *2's complement – most common & important*

One's complement

- For example, in 8-bit one's complement,
- positive 3 is: 00000011
- Negative 3 is: 11111100
- In one's complement, as with signed magnitude, negative values are indicated by a 1 in the high order bit.
- Complement systems are useful because they eliminate the need for subtraction. The difference of two values is found by adding the minuend to the complement of the subtrahend.

Two's Complement Representation

- **Step 1:** First complement all the bits (that is find one's complement) Make all 1s as 0s and all 0s as 1s
- **Step 2:** Then perform increment by 1 Add 0001 b

Signed Numbers

- **Range of Values**

Total combinations = 2^n

2's complement form:

- (2^{n-1}) to + $(2^{n-1} - 1)$

Range for 8 bit number:

$$n = 8$$

$$-(2^{8-1}) = -2^7 = -128 \quad \text{minimum}$$

$$+(2^{8-1}) - 1 = +2^7 - 1 = +127 \quad \text{maximum}$$

Total combination of numbers is $2^8 = 256$.

WHAT IS A BINARY CODE ?

- In the coding, when **numbers, letters or words** are represented by a **specific group of symbols**, it is said that the number, letter or word is being encoded.
- The group of symbols is called as a code. The digital data is represented, stored and transmitted as group of binary bits. This group is also called as **binary code**. The binary code is represented by the number as well as alphanumeric letter.

Classification of Binary Codes

- **Weighted Codes**
- **Non-Weighted Codes**
- **Reflective Codes**
- **Sequential Codes**
- **Alphanumeric Codes**
- **Error Detecting and Correcting Codes**

CODE

- Weighted and Non-Weighted Code
- In weighted code, each digit represents specific weight.
 - Binary code has a weight 8 4 2 1
- In non-weighted code weight is not assigned to position.
 - Excess 3 and Gray Codes.

Type of Codes

- **Binary - follows 8,4,2,1**
 - Example – decimal 12 – Binary 1100
- **BCD – Binary Coded Decimal** – each digit of decimal number is represented by a group of bits. Follows 8,4,2,1.
 - Example – decimal 12 – BCD 0001 0010
- **Excess -3 Code** – derived from BCD code. Excess -3 is calculated by adding 3 to each BCD code. It is non weighted and sequential code.
 - Example BCD 12 – 0001 0010 its Excess -3 code is – 0100 0101
- **Gray Code**

Codes and their conversions

- **Gray Code**

- Gray code is a code where only one bit changes at a time while traversing from 0 to any decimal number in sequence.
- Gray Code is nonweighted and cyclic code
- Not arithmetic code
- Optical shaft encoders for disk to reduce errors

Binary to Gray Code Conversion

- 1. Copy the MSB bit as it is**
- 2. Perform EX-OR operation between previous and current bit and write the result**
- 3. Repeat the step 2 for all bits**

Example – Binary 1011

1 , (1 EX-OR 0) , (0 EX-OR 1), (1 EX-OR 1)
1, 1, 1, 0

Binary to Gray Code Conversion

Step1. The left most code digit is the same as the left most binary code

| | | | | | |
|---|---|---|---|---|--------|
| 1 | 0 | 1 | 1 | 0 | Binary |
| ↓ | | | | | |
| 1 | | | | | Gray |

Step2. Add the left most binary code bit to the adjacent one.

| | | | | | | |
|---|---|---|---|---|---|--------|
| 1 | + | 0 | 1 | 1 | 0 | Binary |
| | | | ↓ | | | |
| 1 | | 1 | | | | Gray |

Step3. Add the next adjacent pair..

| | | | | | | |
|---|---|---|---|---|---|--------|
| 1 | 0 | + | 1 | 1 | 0 | Binary |
| | | | ↓ | | | |
| 1 | 1 | | 1 | | | Gray |

Step4. Add the next adjacent pair.

| | | | | | | |
|---|---|---|---|---|---|--------|
| 1 | 0 | 1 | + | 1 | 0 | Binary |
| | | | ↓ | | | |
| 1 | 1 | 1 | | 0 | | Gray |

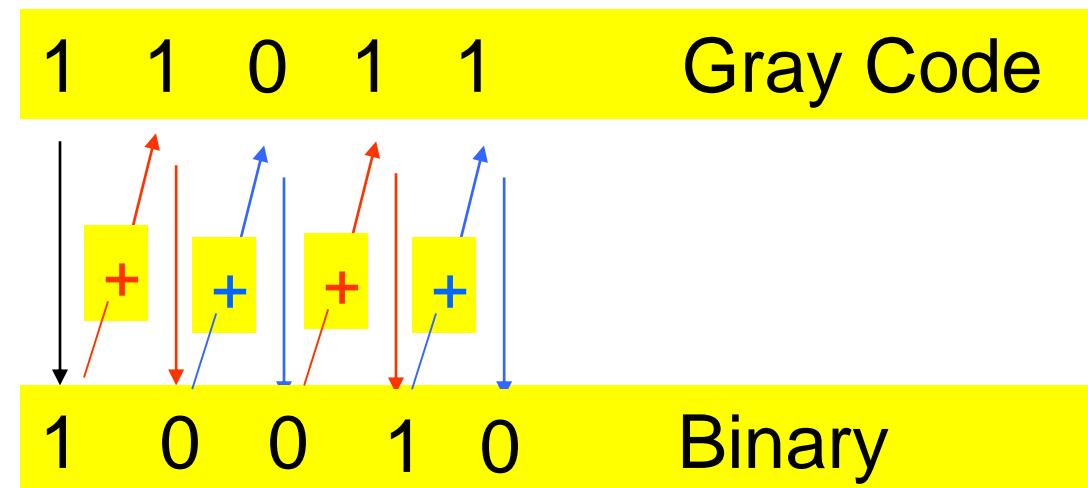
Step5. Add the next adjacent pair.

| | | | | | | |
|---|---|---|---|---|---|--------|
| 1 | 0 | 1 | 1 | + | 0 | Binary |
| | | | ↓ | | | |
| 1 | 1 | 1 | | 1 | | Gray |

Gary to Binary Code Conversion

- MSB of binary is set to the MSB of Gray code
- Going from left to right, add the binary bit to the next Gray code bit. Discard carries.
- Example. Convert 11011 to binary

MSB of Gray Code is 1, so set the MSB of the binary number to 1.



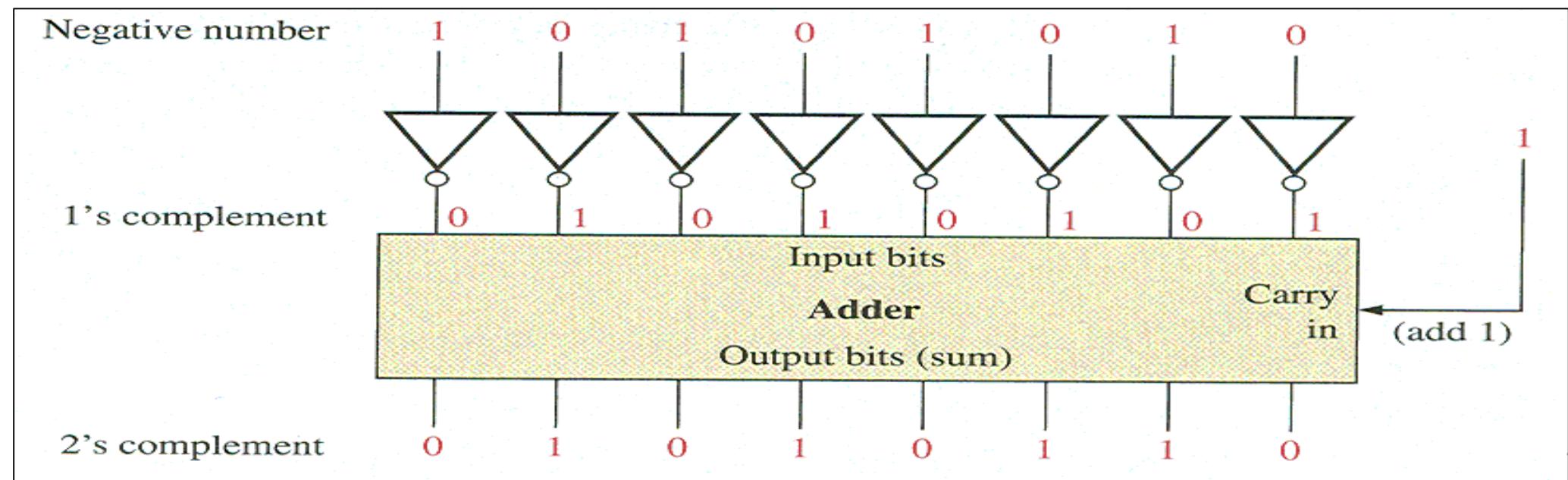
4 Bit Binary To Gray Code Conversion:

| Decimal Value | INPUT (BINARY CODE) | | | | OUTPUT (GRAY CODE) | | | |
|---------------|---------------------|----------------|----------------|----------------|--------------------|----------------|----------------|----------------|
| | B ₃ | B ₂ | B ₁ | B ₀ | G ₃ | G ₂ | G ₁ | G ₀ |
| 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 |
| 10 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 11 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 12 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 13 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 15 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

Sign Magnitude
1's Complement
2's Complement

Terminology: One's-Complement and Two's-Complement

- The ***one's-complement*** of a binary number is the binary number that you get when you invert each bit.
- Example: What is the one's-complement of 10101010?
- The ***two's-complement*** of a binary number is the binary number that you get when you invert each bit and then add 1 to the result.
- Example: What is the two's-complement of 10101010?



Twos Complement Representation

- ◆ Negative number: Bitwise complement **plus one**

$$\diamond 0011 \equiv 3_{10}$$

$$\diamond 1101 \equiv -3_{10}$$

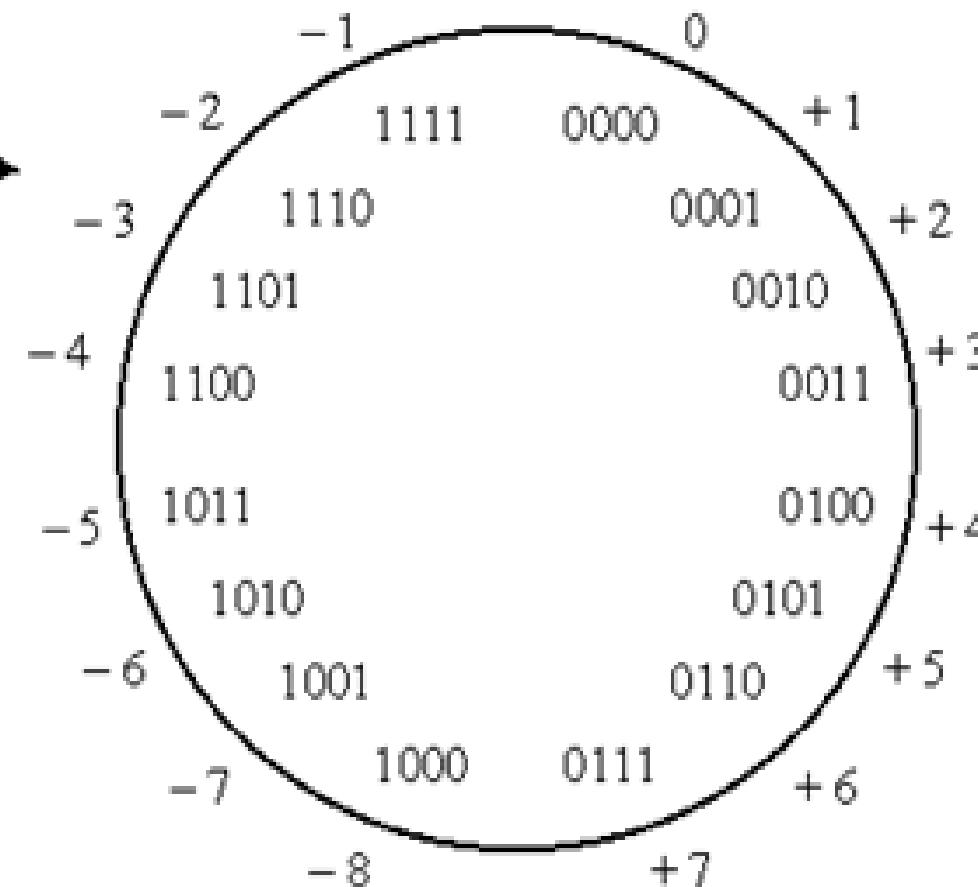
- ◆ Number wheel

- ◆ Only one zero!

- ◆ msb is the sign digit

$$\diamond 0 \equiv \text{positive}$$

$$\diamond 1 \equiv \text{negative}$$



Binary Arithmetic

Binary addition:-

| A | B | Sum | Carry |
|---|---|-----|-------|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

- If addition is unsinged –keep on adding from LSB to MSB
- In case of Signed Numbers – Sign bit and Concept of Sign bit extension need to be used.

Integer Arithmetic: Addition

+7 = 0111

2's complement of +7= 1001

+4 = 0100

2's complement of +4= 1100

+1 = 0001

2's complement of +1= 1111

+6 = 0110

2's complement of +6= 1010

OVERFLOW RULE:

If two numbers are added, and they are both positive or both negative, then overflow occurs if and only if the result has the opposite sign.

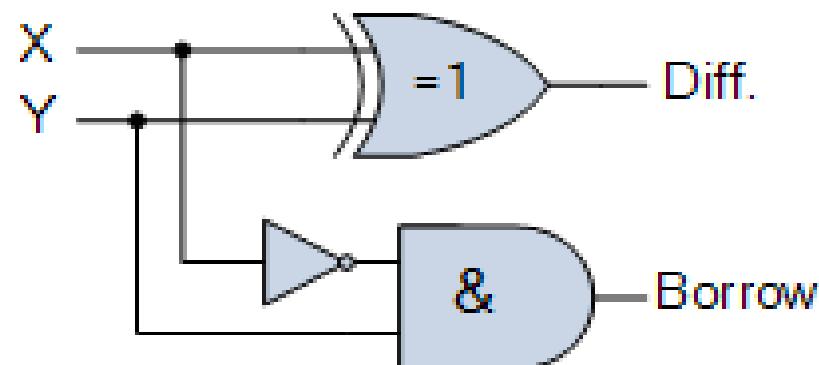
If A and B are of opposite sign, then overflow cannot occur.

| | |
|--|---|
| $\begin{array}{r} 1001 = -7 \\ +0101 = 5 \\ \hline 1110 = -2 \\ \textbf{0010} \end{array}$ (a) $(-7) + (+5)$ | $\begin{array}{r} 1100 = -4 \\ +0100 = 4 \\ \hline 10000 = 0 \end{array}$ (b) $(-4) + (+4)$ |
| $\begin{array}{r} 0011 = 3 \\ +0100 = 4 \\ \hline 0111 = 7 \end{array}$ (c) $(+3) + (+4)$ | $\begin{array}{r} 1100 = -4 \\ +1111 = -1 \\ \hline 11011 = -5 \\ \textbf{0101} \end{array}$ (d) $(-4) + (-1)$ |
| $\begin{array}{r} 0101 = 5 \\ +0100 = 4 \\ \hline 1001 = \text{Overflow} \end{array}$ (e) $(+5) + (+4)$ | $\begin{array}{r} 1001 = -7 \\ +1010 = -6 \\ \hline 10011 = \text{Overflow} \\ \textbf{1101} \end{array}$ (f) $(-7) + (-6)$ |

Binary Subtraction

| Case | A - B | Subtract | Borrow |
|------|-------|----------|--------|
| 1 | 0 - 0 | 0 | 0 |
| 2 | 1 - 0 | 1 | 0 |
| 3 | 1 - 1 | 0 | 0 |
| 4 | 0 - 1 | 1 | 1 |

| Symbol | | Truth Table | |
|--------|---|-------------|--------|
| Y | X | DIFFERENCE | BORROW |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |



Binary Subtraction

1. Negate a number (by taking its 2's-complement).
2. Add binary numbers.

Subtraction

| | |
|--|---|
| $ \begin{array}{r} 0010 = 2 \\ +1001 = -7 \\ \hline 1011 = -5 \end{array} $ | $ \begin{array}{r} 0101 = 5 \\ +1110 = -2 \\ \hline 10011 = 3 \end{array} $ |
| $ \begin{array}{l} (a) M = 2 = 0010 \\ S = 7 = 0111 \\ -S = 1001 \end{array} $ | $ \begin{array}{l} (b) M = 5 = 0101 \\ S = 2 = 0010 \\ -S = 1110 \end{array} $ |
| $ \begin{array}{r} 1011 = -5 \\ +1110 = -2 \\ \hline 11001 = -7 \end{array} $ | $ \begin{array}{r} 0101 = 5 \\ +0010 = 2 \\ \hline 0111 = 7 \end{array} $ |
| $ \begin{array}{l} (c) M = -5 = 1011 \\ S = 2 = 0010 \\ -S = 1110 \end{array} $ | $ \begin{array}{l} (d) M = 5 = 0101 \\ S = -2 = 1110 \\ -S = 0010 \end{array} $ |
| $ \begin{array}{r} 0111 = 7 \\ +0111 = 7 \\ \hline 1110 = \text{Overflow} \end{array} $ | $ \begin{array}{r} 1010 = -6 \\ +1100 = -4 \\ \hline 10110 = \text{Overflow} \end{array} $ |
| $ \begin{array}{l} (e) M = 7 = 0111 \\ S = -7 = 1001 \\ -S = 0111 \end{array} $ | $ \begin{array}{l} (f) M = -6 = 1010 \\ S = 4 = 0100 \\ -S = 1100 \end{array} $ |

SUBTRACTION RULE: $a - b = a + (-b)$

To subtract one number (**Subtrahend**) from another (**Minuend**), take the two's complement (**negation**) of the subtrahend and add it to the minuend.

Multiplication (binary)

It's interesting to note that binary multiplication is a sequence of shifts and adds of the first term (depending on the bits in the second term).

$$\begin{array}{r} 1101 \\ \times 1011 \\ \hline 1101 \\ 11010 \\ + 1101000 \\ \hline 10001111 \end{array}$$

110100 is missing here because the corresponding bit in the second terms is 0.

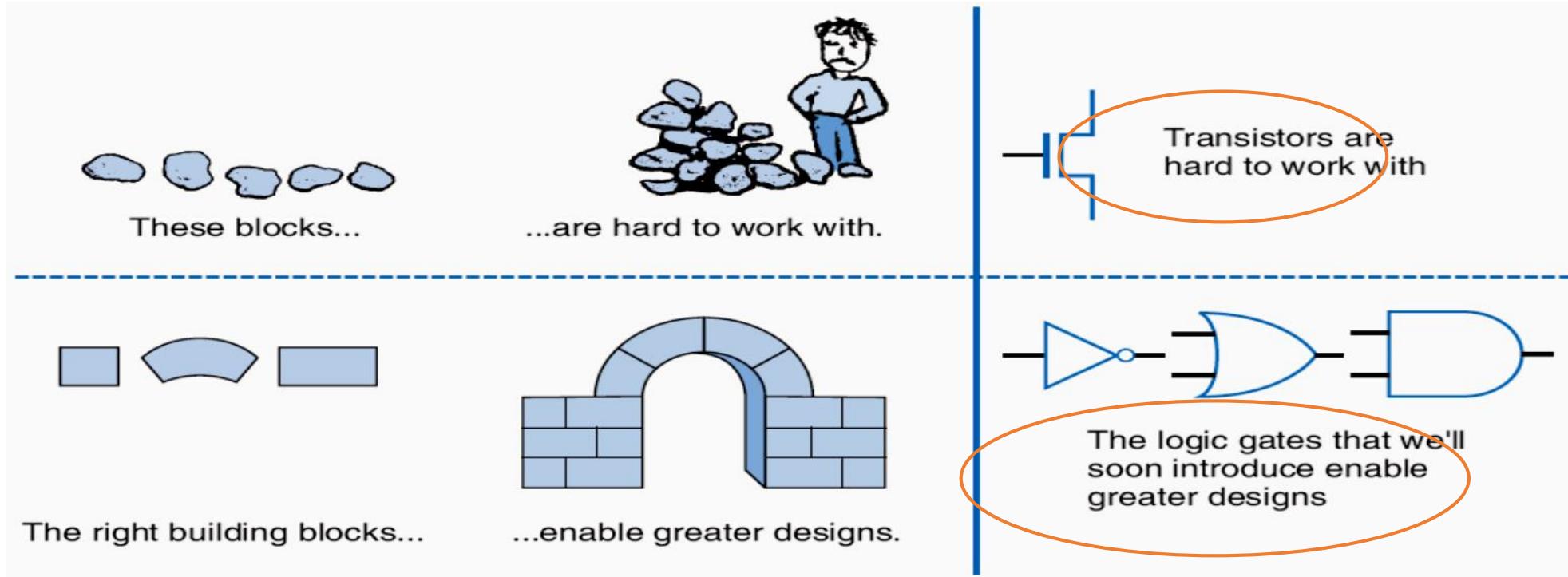
Binary Division

Divisor → 1001 Quotient ← 1101
Dividend ← 1110101

$$\begin{array}{r} 1101 \\ \hline 1001) 1110101 \\ \underline{1001} \\ 1011 \\ \underline{1001} \\ 001001 \\ \underline{1001} \\ 0000 \end{array}$$

Boolean Logic Gates (Building Blocks for Digital Circuits)

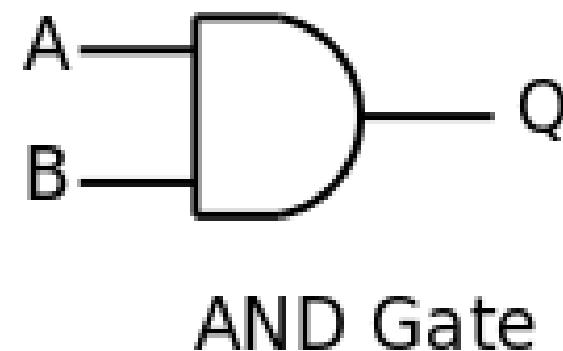
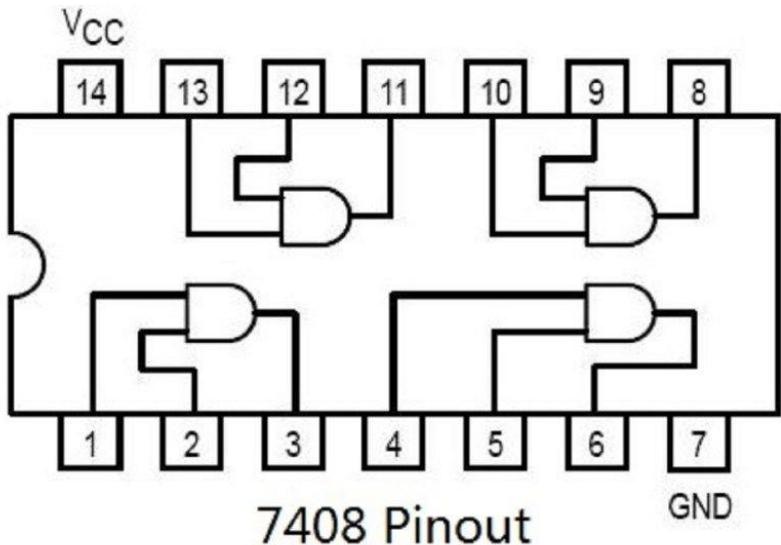
(Because Switches are Hard to Work With)



“Logic gates” are better digital circuit building blocks than switches (transistors), Why?...

Logic Gates

- **AND Gate**

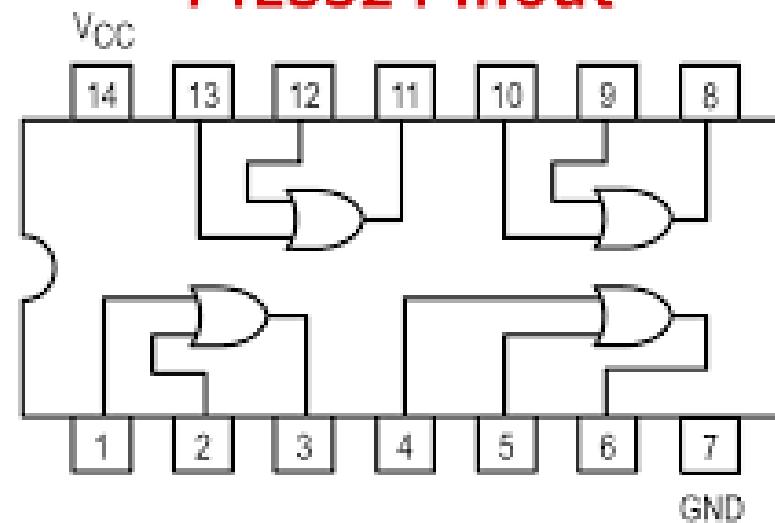


| A | B | Q |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

OR Gate

- OR Gate

74LS32 Pinout



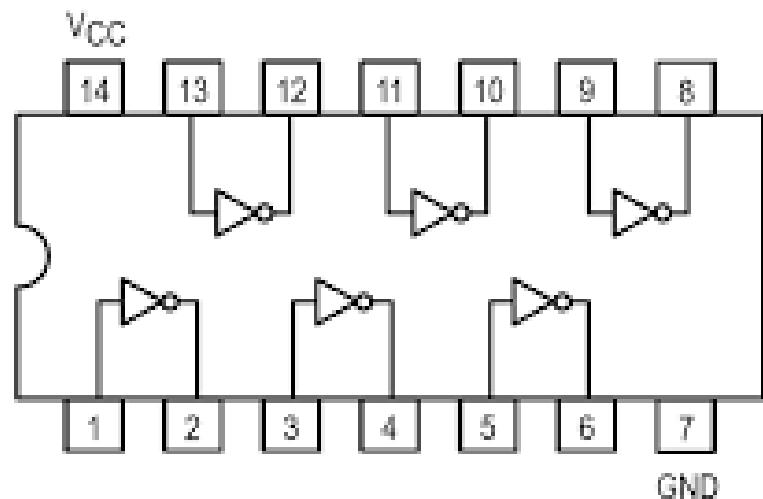
CIRCUITS DIY
PROJECTS & TUTORIALS

| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

NOT gate

- NOT Gate

74LS04 Pinout

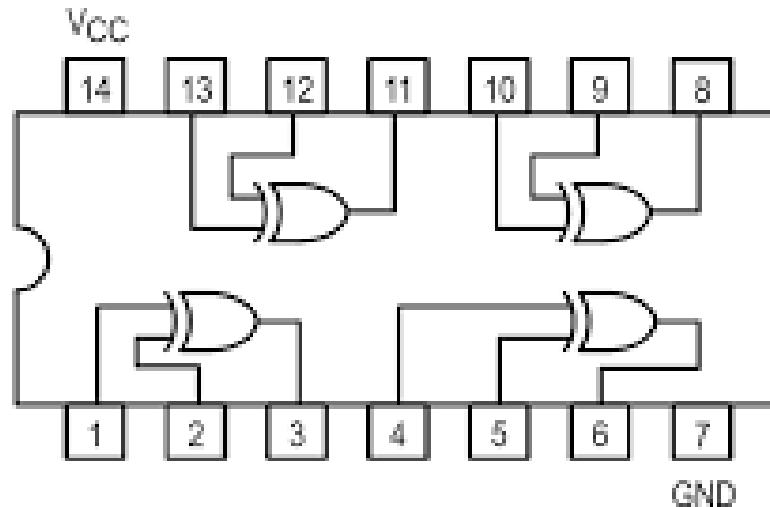


| Input | Output |
|-------|--------|
| A | Y |
| 0 | 1 |
| 1 | 0 |

EX-OR

-

74LS86 Pinout

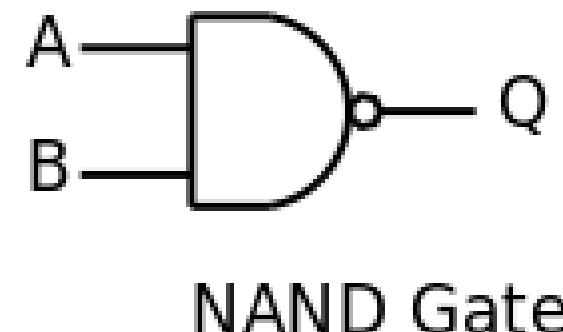
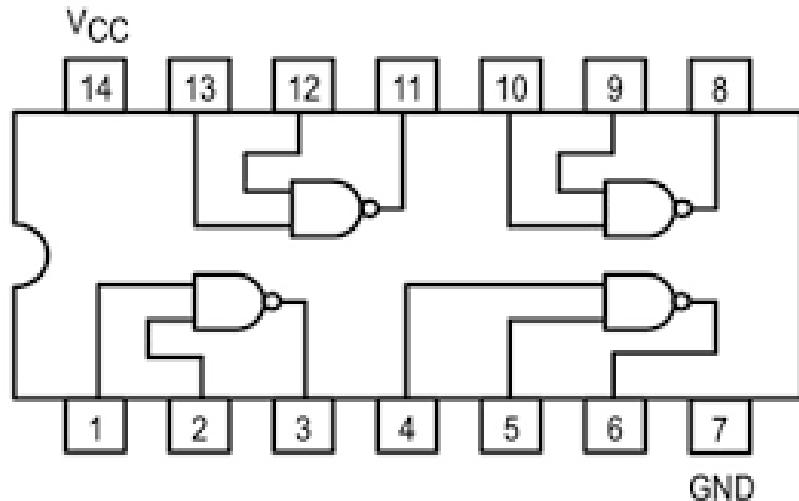


CIRCUITS DIY
BY
SARAH ELECTRONICS

| Inputs | | Output |
|--------|---|--------|
| A | B | X |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

NAND Gate

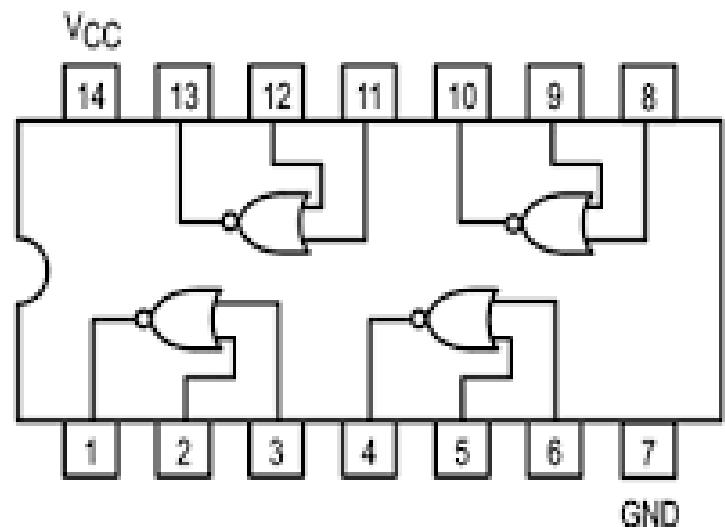
- NAND Gate IC 7400



| A | B | Q |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

NOR Gate

- NOR Gate IC 7402



| A | B | $\overline{A+B}$ |
|---|---|------------------|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

Boolean Algebra Terminology

English Mathematician George Boole develop rules for manipulation of binary variables.

- Variable - an arbitrary element of a Boolean Algebra is known as Variable. It can have either value 0 or 1.
- Constant – it can have fixed value as 0 or 1
- Complement – is represented with the “bar over the letter,”
- Literal – each occurrence of a variable in Boolean Function in complemented or complemented way is called as a Literal
- Boolean Function- Boolean expressions are constructed by connecting variable, constant in Boolean operations which, altogether describes Boolean function.

Boolean Algebra - Properties

- **Closure Property –**
 - With respect to + Operator – When two binary elements are operated with + operator the result is a unique binary element.
 - With respect to . Operator – When two binary elements are operated with . (dot) operator the result is a unique binary element.
- **Identity Property**
- **Cumulative Property**
- **Distributive Property**
- **Complement Property**
- **Idempotency Property (Sameness)**
- **Absorption Property**
- **Involution Property**

| Postulates | (a) | (b) |
|----------------------------|---|-----------------------------------|
| Postulate 1 | Result of each operation is either 0 or 1 $\in B$ | |
| Postulate 2 (Identity) | $A + 0 = A$ | $A \cdot 1 = A$ |
| Postulate 3 (Commutative) | $A + B = B + A$ | $AB = BA$ |
| Postulate 4 (Distributive) | $A(B + C) = AB + AC$ | $A + BC = (A + B)(A + C)$ |
| Postulate 5 (Complement) | $A + \overline{A} = 1$ | $A \cdot \overline{A} = 0$ |
| Theorems | (a) | (b) |
| Theorem 1 (Idempotency) | $A + A = A$ | $A \cdot A = A$ |
| Theorem 2 | $A + 1 = 1$ | $A \cdot 0 = 0$ |
| Theorem 3 (Involution) | | $\overline{\overline{A}} = A$ |
| Theorem 4 (Absorption) | $A + AB = A$ | $A(A + B) = A$ |
| Theorem 5 | $A + \overline{A}B = A + B$ | $A \cdot (\overline{A} + B) = AB$ |
| Theorem 6 (Associative) | $A + (B + C) = (A + B) + C$ | $A(BC) = (AB)C$ |

Representations of Logic Functions

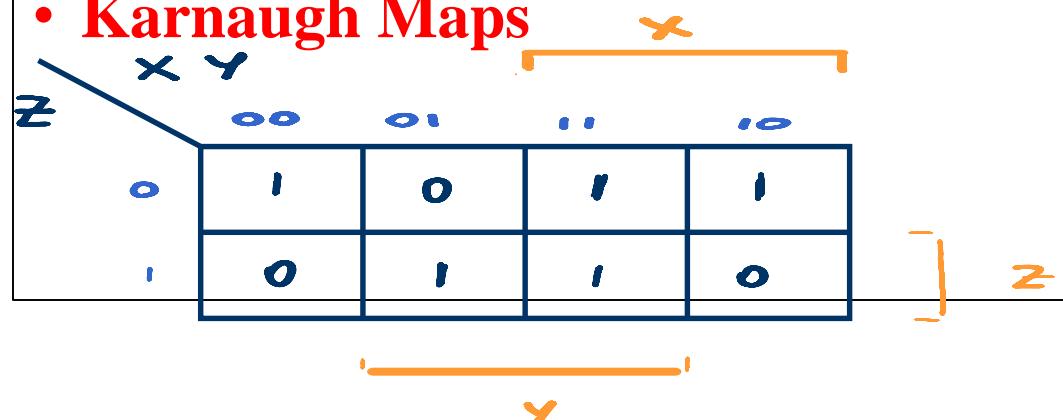
- **Truth Tables**

Table that shows the output value for each combination of the inputs.

The truth table for an n-variable (n-inputs) logic function has 2^n rows.

- **Algebraic expressions**

- **Karnaugh Maps**



| Row | X | Y | Z | F |
|-----|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |

$$F = X \cdot \bar{Z} + \bar{Y} \cdot \bar{Z} + Y \cdot Z$$

Common Terminology

- **Literal**

The appearance of a variable or its complement

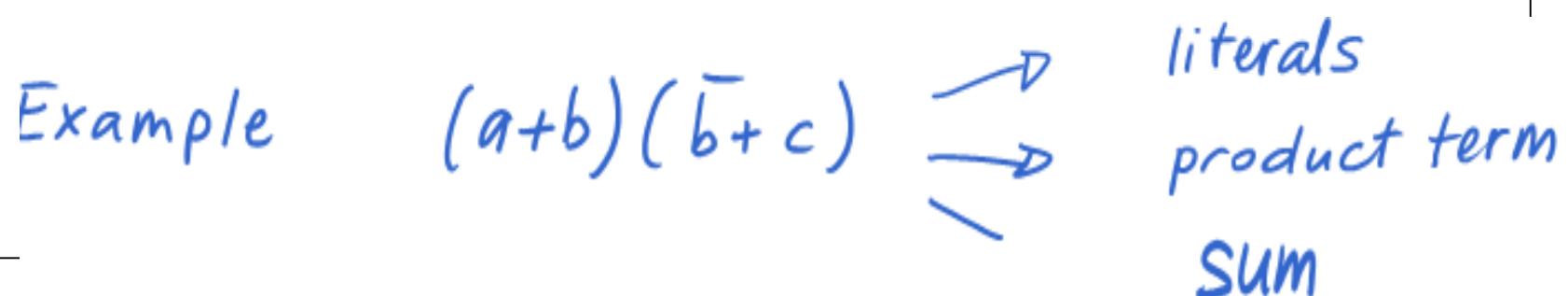
- **Product Term**

One or more literals connected by AND operators

- **Sum Term**

One or more literals connected by OR operators

Example $(a+b)(\bar{b}+c)$



literals
product term
sum

Common Terminology(cnt'd)

Standard form of Boolean Expressions

1. **Sum-of-Products (SOP)** - is called a '**minterm**'
 - a. first the product (AND) terms are formed then these are summed (OR)
 - b. eg: ABC + DEF + GHI
2. **Product-of-sum form (POS)** - is called a '**maxterm**'
 - a. first the sum (OR) terms are formed then the products are taken (AND)
 - b. eg: (A+B+C) (D+E+F) (G+H+I)
3. **Canonical form**
 - a. Canonical form is not efficient but sometimes useful in analysis and design
 - b. In an expression in canonical form, every variable appears in every term
 - c. eg: $f(A, B, C, D) = ABCD + \overline{ABC}D + A\overline{B}CD$

SOP & POS

SOP Form Representation

- Consider Following Truth Table

| | A | B | C | F |
|------|---|---|---|---|
| m0 0 | 0 | 0 | 0 | 0 |
| m1 1 | 0 | 0 | 1 | 0 |
| m2 2 | 0 | 1 | 0 | 1 |
| m3 3 | 0 | 1 | 1 | 0 |
| m4 4 | 1 | 0 | 0 | 1 |
| m5 5 | 1 | 0 | 1 | 1 |
| m6 6 | 1 | 1 | 0 | 1 |
| m7 7 | 1 | 1 | 1 | 1 |

Low → F = $\overline{A} \cdot \overline{B} \cdot \overline{C}$

High → F = $A \cdot B \cdot C$

In SOP Form always remember

$$\begin{aligned} 1 &= A \\ 0 &= \overline{A} \end{aligned}$$

$$F = \overline{ABC} + A\overline{BC} + \overline{AB}C + ABC + A\overline{B}\overline{C}$$

Practice Examples(Contd..)

--- Distribution law: $X + YZ = (X+Y)(X+Z)$ ---

- In Example ---- $(B' + B)(B'+A)$

- $F(A, B, C) = \sum m (0, 4, 6, 1, 5, 7)$

$$\begin{aligned}
 F &= \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC \\
 &= \bar{B}\bar{C}(\bar{A}+A) + AB (\bar{C}+C) + \bar{B}C(\bar{A}+A) \\
 &= \bar{B}\bar{C}(1) + AB(1) + \bar{B}C(1) \\
 &= \bar{B}\bar{C} + AB + \bar{B}C \\
 &= \bar{B}(\bar{C}+C) + AB \\
 &= \bar{B} + AB \\
 &= \bar{B} + A
 \end{aligned}$$



Values of logic variables and logic functions are in binary form(1 and 0).

Any logic function can be expressed in the forms of

- 1. Sum of Products form (SOP)**

$$Y = AB + AC' + BC \text{-----SOP}$$

Product terms: AB, AC', BC [AND Gate Z = XY]

Here Summation Terms : 2 terms [OR Gate Z = X + Y]

ORing of ANDs

Values of logic variables and logic functions are in binary form(1 and 0).

Any logic function can be expressed in the forms of

2. Product of Sums form (POS)

$$Y = (A + C)(B + C') \text{-----POS}$$

Sum Terms: $A+C$, $B+C'$ [OR Gate $Z = X + Y$]

Here Product Terms: 1 [AND Gate $Z = XY$]

ANDing of ORs

SOP and POS

Canonical Form of Expression and Summation form (SOP) /Product form (POS) of expression

Standard Representations for Logic Functions

Canonical Normal Form

- Standard form for Boolean expression that provide a unique signature
- If each term in SOP and POS forms contains all the literals then these are known as canonical SOP and POS respectively.
- Each individual term in canonical SOP form is called as minterm (m₀, m₁, m₂.....)
- Each individual term in canonical POS form is called as maxterm (M₀, M₁, M₂.....)

Canonical or Conventional form

- Consider the following expression

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



$$W=f(A,B,C) = A\overline{B}C + \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}\overline{C}$$

Sum of Product (SOP)

- It is a Disjunctive Normal form or Minterm Expansion

$$AB + CD$$

(A AND B) **OR** (C AND D)

i.e. **OR of AND terms**

Summation Form of Expression --SOP

- In this case we substitute a “1” for all positive variables & a “0” for all inverted
- Ex: $A = 1$ and $A' = 0$; $B = 1$ and $B' = 0$ and $C = 1$ and $C' = 0$

- E.g. $F(A,B,C) = A\bar{B}C + \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}\bar{B}\bar{C}$ Minterms
 $= 101 + 000 + 010 + 001$
 $= 5 + 0 + 2 + 1$
 $= \sum m(5,0,2,1)$
 $= \sum m(0,1,2,5)$

| A | B | C | Decimal Value | Minterm |
|---|---|---|---------------|---------|
| 0 | 0 | 0 | 0 | m0 |
| 0 | 0 | 1 | 1 | m1 |
| 0 | 1 | 0 | 2 | m2 |
| 0 | 1 | 1 | 3 | m3 |
| 1 | 0 | 0 | 4 | m4 |
| 1 | 0 | 1 | 5 | m5 |
| 1 | 1 | 0 | 6 | m5 |
| 1 | 1 | 1 | 7 | m6 |

Canonical Form and Summation form

Conversion Process – Canonical form to summation form

- It takes 3 steps:

1. Each **minterm is converted into it's binary equivalent.**

$$\begin{aligned} \text{E.g. } F(A,B,C) &= A\bar{B}C + \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}\bar{B}\bar{C} \\ &= 101 + 000 + 010 + 001 \end{aligned}$$

2. Each minterm is converted from **binary to decimal.**

$$\begin{aligned} &= 101 + 000 + 010 + 001 \\ &= 5 + 0 + 2 + 1 \\ &= \Sigma m(5,0,2,1) \end{aligned}$$

3. Finally the **min terms are placed in ascending order**

$$= \Sigma m(0,1,2,5)$$

Advantages of Summation form

- Useful for implementation in Programmable Logic
- Very long & complex expressions can be easily described
- Useful for implementation with hardware devices



MIT-WPU

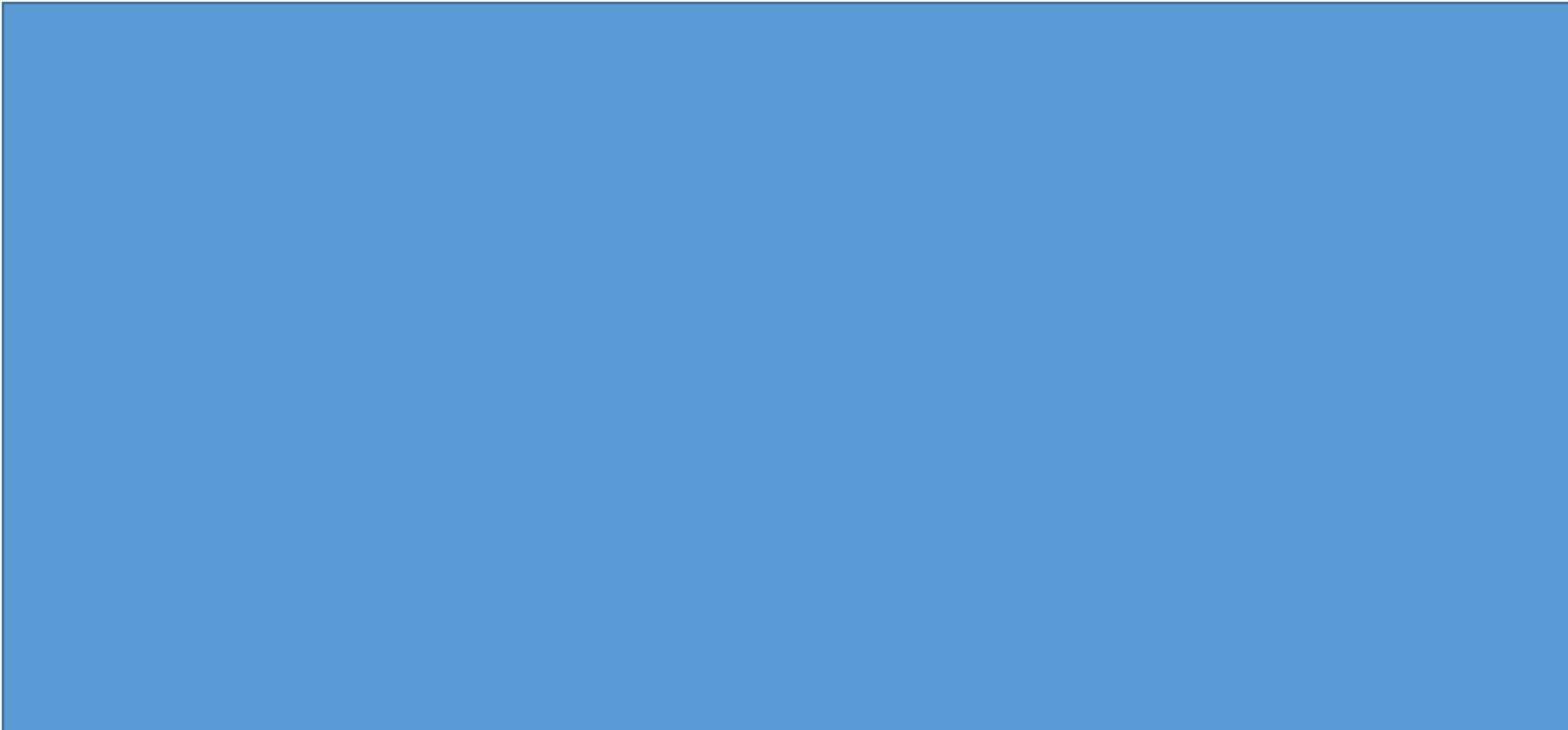
॥ विद्यानन्तर्दृशं ध्रुवा ॥

Practice Examples

$$\begin{aligned}1. \quad F(A,B,C) &= AB + BC \\&= AB(C+\bar{C}) + BC(A+\bar{A}) \\&= ABC + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}BC \\&= ABC + A\bar{B}\bar{C} + \bar{A}BC \\&= 111 + 110 + 011 \\&= \sum m(7,6,3) \\&= \sum m(3,6,7)\end{aligned}$$

Conversion from Conventional to Summation Form

- E.g. $X = f(A, B, C, D) = \bar{A}CD + ABC\bar{C} + \bar{C}\bar{D}$



Conversion from Conventional to Summation Form

- E.g. $X = f(A,B,C,D) = \bar{A}CD + ABC\bar{C} + \bar{C}\bar{D}$

$$= \bar{A}CD(B+\bar{B}) + ABC\bar{C}(D+\bar{D}) + \bar{C}\bar{D} (A+A)(B+\bar{B})$$

$$= \bar{A}BCD + \bar{A}\bar{B}CD + AB\bar{C}D + AB\bar{C}\bar{D} + \\ \bar{C}\bar{D}(\bar{A}\bar{B} + A\bar{B} + \bar{A}B + AB)$$

$$= \bar{A}BCD + \bar{A}\bar{B}\bar{C}D + AB\bar{C}D + AB\bar{C}\bar{D} + \\ \bar{A}\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}\bar{D} + AB\bar{C}\bar{D}$$

$$= \sum m (7, 3, 13, 12, 0, 8, 4, 12)$$

$$= \sum m (0, 3, 4, 7, 8, 12, 13)$$

Conversion from Conventional to Summation Form

- E.g. $X = f(A, B, C, D) = \overline{A}CD + ABC\bar{C} + \overline{C}\overline{D}$
 $= \overline{A}CD(B+\overline{B}) + ABC\bar{C}(D+\overline{D}) + \overline{C}\overline{D}(A+A)(B+\overline{B})$
 $= \overline{A}BCD + \overline{A}\overline{B}CD + AB\overline{C}D + AB\overline{C}\overline{D} +$
 $\quad \overline{C}\overline{D}(\overline{AB}+\overline{AB}+\overline{AB}+AB)$
 $= \overline{A}BCD + \overline{A}\overline{B}CD + AB\overline{C}D + AB\overline{C}\overline{D} +$
 $\quad \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}CD + \overline{A}\overline{B}CD + ABC\overline{D}$
 $= \sum m (7, 3, 13, 12, 0, 8, 4, 12)$
 $= \sum m (0, 3, 4, 7, 8, 12, 13)$

- Consider Following Truth Table

| | A | B | C | F |
|------|---|---|---|---|
| m0 0 | 0 | 0 | 0 | 0 |
| m1 1 | 0 | 0 | 1 | 0 |
| m2 2 | 0 | 1 | 0 | 1 |
| m3 3 | 0 | 1 | 1 | 0 |
| m4 4 | 1 | 0 | 0 | 1 |
| m5 5 | 1 | 0 | 1 | 1 |
| m6 6 | 1 | 1 | 0 | 1 |
| m7 7 | 1 | 1 | 1 | 1 |

Low → F = A.B.C

High → F = A.B.C

In SOP Form always remember

$$1 = A$$

$$0 = \bar{A}$$

$$F = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + AB\bar{C} + ABC$$

Practice Examples(Contd..)

{Law of distribution: $A+BC = (A+B)(A+C)$
 so $B' + AB = (B' + A)(B' + B) = B' + A$ }

- $F(A, B, C) = \sum m (0, 4, 6, 1, 5, 7)$

$$F = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + ABC\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC$$

$$= \bar{B}\bar{C}(\bar{A}+A) + AB (\bar{C}+C) + \bar{B}C(\bar{A}+A)$$

$$= \bar{B}\bar{C}(1) + AB(1) + \bar{B}C(1)$$

$$= \bar{B}\bar{C} + AB + \bar{B}C$$

$$= \bar{B}(\bar{C}+C) + AB$$

$$= \bar{B} + AB$$

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

| A | B | C | Decimal Value | Minterm |
|---|---|---|---------------|---------|
| 0 | 0 | 0 | 0 | m0 |
| 0 | 0 | 1 | 1 | m1 |
| 0 | 1 | 0 | 2 | m2 |
| 0 | 1 | 1 | 3 | m3 |
| 1 | 0 | 0 | 4 | m4 |
| 1 | 0 | 1 | 5 | m5 |
| 1 | 1 | 0 | 6 | m5 |
| 1 | 1 | 1 | 7 | m6 |

Practice Examples(Contd..)

- $F(A, B, C) = \sum m (0, 4, 6, 1, 5, 7)$
$$\begin{aligned} F &= \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + AB\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC \\ &= \bar{B}\bar{C}(\bar{A}+A) + AB (\bar{C}+C) + \bar{B}C(\bar{A}+A) \\ &= \bar{B}\bar{C}(1) + AB(1) + \bar{B}C(1) \\ &= \bar{B}\bar{C} + AB + \bar{B}C \\ &= \bar{B}(\bar{C}+C) + AB \\ &= \bar{B} + AB \\ &= \bar{B} + A \end{aligned}$$

Practice Examples(Contd..)

010, 100, 101, 110, 111

- $F(A, B, C) = \sum m (2, 4, 5, 6, 7)$

Practice Examples(Contd..)

$$A'X + A = (A+A')(A+X) = A+X = A + BC'$$

- $F(A, B, C) = \sum m (2, 4, 5, 6, 7)$

$$\begin{aligned}
 F &= \overline{A}B\overline{C} + A\overline{B}\overline{C} + A\overline{B}C + AB\overline{C} + ABC \\
 &= \overline{A}B\overline{C} + A\overline{B}(\overline{C}+C) + AB (\overline{C}+C) \\
 &= \overline{A}B\overline{C} + A\overline{B}(1) + AB(1) \\
 &= \overline{A}B\overline{C} + A\overline{B} + AB \\
 &= \overline{A}B\overline{C} + A(\overline{B}+B) \\
 &= \overline{A}B\overline{C} + A \quad \xrightarrow{\text{if } B\overline{C} = x} \\
 &= X + A \quad \xrightarrow{\text{if } \overline{A}x = x} \\
 &= A + X \\
 &= A + BC
 \end{aligned}$$

Practice Examples(Contd..)

CSE203

DIGITAL ELECTRONICS AND LOGIC DESIGN

- Simplify the Expression

$$Y(A\bar{B}) = \sum m(0,2,3)$$

Practice Examples(Contd..)

CSE203

 $m_0 = A'B'$
 DIGITAL ELECTRONICS AND LOGIC DESIGN

 $m_1 = A'B$
 $m_2 = AB'$
 $m_3 = AB$

- Simplify the Expression

$$Y(A,B) = \sum m(0,2,3)$$

$$= m_0 + m_2 + m_3$$

$$= \bar{A}\bar{B} + A\bar{B} + AB$$

$$= (\bar{A}+A)\bar{B} + AB$$

$$= \bar{B} + AB \leftarrow \boxed{\bar{A} + AB = \bar{A} + B}$$

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

| A | B | $Y = \bar{B} + A$ |
|---|---|-------------------|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

solve

- $A + A'B + AB$ ans: $A+B$

- $AB' + A'B + AB + A'B'$ ans: 1

Product of Sum (POS)

- It is a **Conjunctive Normal form or Maxterm Expansion**
- E.g.

$$(A+B) (C+D)$$

- **(A OR B) AND (C OR D)**

i.e. **AND of OR terms**

POS Contd..

| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

A+B+C
 A+B+C'
 A+B'+C'

In POS Form always remember

$1 = \bar{A}$
 $0 = A$

| SOP | POS |
|-------------------------|-------------------------|
| $0 \rightarrow \bar{A}$ | $0 \rightarrow A$ |
| $1 \rightarrow A$ | $1 \rightarrow \bar{A}$ |

POS (Contd..)

- Why $0 = A$ & $1 = \bar{A}$
- E.g. $Y = (A+B+C) (A+B+\bar{C}) (A+\bar{B}+\bar{C})$

$$\bar{Y} = \overline{ABC} + \overline{AB}\bar{C} + \overline{A}\overline{BC}$$

$$\bar{Y} = [\overline{ABC} + \overline{AB}\bar{C} + \overline{A}\overline{BC}] \leftarrow \text{Taking complement of both sides}$$

$$Y = (\overline{ABC})' \cdot (\overline{AB}\bar{C})' \cdot (\overline{A}\overline{BC})'$$

$$Y = (A+B+C) \cdot (A+B+\bar{C}) \cdot (A+\bar{B}+\bar{C})$$

Using De-Morgan's Law
 $(A+B) = A' \cdot B'$

Again by Using De-Morgan's Law

But it is complicated to use demorgan's law every time,
 so it will be easy to use following convention

$0 \rightarrow A$
 $1 \rightarrow \bar{A}$

POS (Contd..)

- $$\begin{aligned}
 Y &= (\underline{A+B+C}) \ (\underline{A+B+\bar{C}}) \ (A+\bar{B}+\bar{C}) \\
 &= (X + CC\bar{C}) \ (A+\bar{B}+\bar{C}) \leftarrow \\
 &= (X+0) \ (A+\bar{B}+\bar{C}) \\
 &= X \cdot (A+\bar{B}+\bar{C}) \\
 &= (A+B) \ (A+\bar{B}+\bar{C}) \\
 &= A+B(\bar{B}+\bar{C}) \\
 &= A + B\bar{B} + BC \\
 &= A + BC\bar{C} \quad \leftarrow \\
 &= (A+B) \ (A+\bar{C})
 \end{aligned}$$

From Distributive law
 $(X + C)(X + C) = X + CC$

From Distributive law
 $A + BC = (A + B)(A + C)$

Practice Example

- For given truth table minimize POS expression

| | A | B | Y |
|----|---|---|---|
| M0 | 0 | 0 | 1 |
| M1 | 0 | 1 | 0 |
| M2 | 1 | 0 | 1 |
| M3 | 1 | 1 | 0 |

Practice Example

- For given truth table minimize POS expression

| | A | B | Y |
|----|---|---|---|
| M0 | 0 | 0 | 1 |
| M1 | 0 | 1 | 0 |
| M2 | 1 | 0 | 1 |
| M3 | 1 | 1 | 0 |

$$\begin{aligned}
 Y &= (A + \bar{B})(\bar{A} + \bar{B}) \\
 &= \bar{B} + A\bar{A} \\
 &= \bar{B} + 0 \\
 Y &= \bar{B}
 \end{aligned}$$

From Distributive law
 $(A + B)(A + C) = A + BC$

Practice Example (Contd..)

- **POS Representation**

$$Y = \pi(M1, M3)$$

$$Y = \pi M(1,3)$$

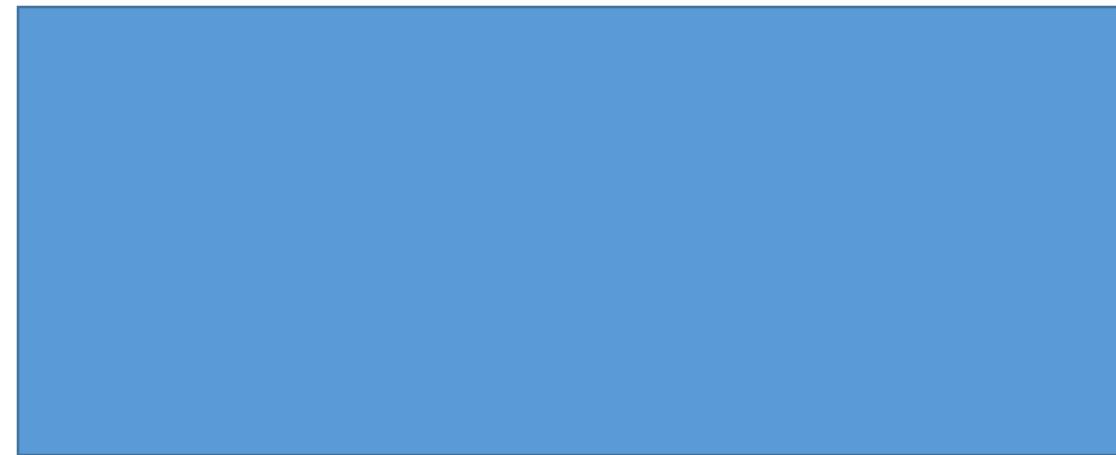
- **SOP Representation**

$$Y = \sum m(0,2)$$

Practice Example (Contd..)

- For following truth table find minterm & maxterm i.e. SOP & POS forms

| | A | B | C | F |
|------|---|---|---|---|
| m0 0 | 0 | 0 | 0 | 1 |
| M1 1 | 0 | 0 | 1 | 0 |
| m2 2 | 0 | 1 | 0 | 1 |
| m3 3 | 0 | 1 | 1 | 1 |
| M4 4 | 1 | 0 | 0 | 0 |
| M5 5 | 1 | 0 | 1 | 0 |
| m6 6 | 1 | 1 | 0 | 1 |
| m7 7 | 1 | 1 | 1 | 1 |



Practice Example (Contd..)

- For following truth table find summation & product forms i.e. SOP & POS form

| | A | B | C | F |
|------|---|---|---|---|
| m0 0 | 0 | 0 | 0 | 1 |
| M1 1 | 0 | 0 | 1 | 0 |
| m2 2 | 0 | 1 | 0 | 1 |
| m3 3 | 0 | 1 | 1 | 1 |
| M4 4 | 1 | 0 | 0 | 0 |
| M5 5 | 1 | 0 | 1 | 0 |
| m6 6 | 1 | 1 | 0 | 1 |
| m7 7 | 1 | 1 | 1 | 1 |

$$Y(A,B,C) = \sum m(0,2,3,6,7)$$

$$Y(A,B,C) = \pi M(1,4,5)$$

Practice Example (Contd..)

- $Y(A, B, C) = \sum m(0, 2, 3, 6, 7)$

Practice Example (Contd..)

- $Y(A, B, C) = \sum m(0, 2, 3, 6, 7)$

$$\begin{aligned}
 Y &= \overline{\overline{A}}\overline{B}\overline{C} + \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + A\overline{B}\overline{C} + ABC \\
 &= \overline{\overline{A}}\overline{B}\overline{C} + \overline{A}B(\overline{C}+C) + AB(\overline{C}+C) \\
 &= \overline{\overline{A}}\overline{B}\overline{C} + \overline{A}B + AB \\
 &= \overline{\overline{A}}\overline{B}C + (\overline{A}+\overline{A})B \\
 &= \overline{\overline{A}}\overline{B}C + B \\
 &= X \cdot \overline{B} + B \quad \leftarrow \text{Let } X = \overline{A}\overline{C} \\
 &= X + B \\
 &= \overline{A}\overline{C} + B
 \end{aligned}$$

Practice Example (Contd..)

- $\text{Y(A,B,C)} = \pi M(1,4,5)$

Practice Example (Contd..)

- $Y(A,B,C) = \pi M(1,4,5)$

$$\pi M(001,100,101)$$

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

Consider $A'+B = X$ and $C = Y$ and $C' = Z$

$$= (A+B+C')(A' + B + CC') ----- (A+B)(A+C) = A+BC — Distributive Law$$

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

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

$$= B + A \cdot A' + A' C'$$

$$= B + A' C'$$

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

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

Product of Sums form (POS)

$$Y = (A + C)(B + C') \text{-----POS}$$

Sum Terms: $A+C$, $B+C'$ [OR Gate $Z = X + Y$]

Here Product Terms: 1 [AND Gate $Z = XY$]

ANDing of ORs

- It is a **Conjunctive Normal form or Maxterm Expansion**

Example:

$$(A+B) (C+D)$$

- $(A \text{ OR } B) \text{ AND } (C \text{ OR } D)$

POS - Maxterms / SOP - Minterms

Truth table describing a digital system/Logic circuit

| | A | B | C | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 1 |
| 3 | 0 | 1 | 1 | 0 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |

$$A+B+C$$

$$A+B+C'$$

$$A+B'+C'$$

In POS Form always remember

$$1 = \bar{A}$$

$$0 = A$$

POS Form:

$$Y(A,B,C) = \pi(M_0, M_1, M_3)$$

$$Y(A,B,C) = \pi M(0,1,3)$$

| SOP | POS |
|-------------------------|-------------------------|
| $0 \rightarrow \bar{A}$ | $0 \rightarrow A$ |
| $1 \rightarrow A$ | $1 \rightarrow \bar{A}$ |

SOP Form:

$$Y(A,B,C) = \sum(m_2, m_4, m_5, m_6, m_7)$$

$$Y(A,B,C) = \sum m(2,4,5,6,7)$$

POS form simplification

$$X + YZ = (X+Y) (X+Z)$$

$$(A+B) (A+B'+C')$$

- $$\begin{aligned} Y &= (\cancel{A+B+C}) \ (\cancel{A+B+C}) \ (\cancel{A+\bar{B}+\bar{C}}) \\ &= (X + C\bar{C}) \ (A+\bar{B}+\bar{C}) \leftarrow \\ &= (X+0) \ (A+\bar{B}+\bar{C}) \\ &= X \cdot (A+\bar{B}+\bar{C}) \\ &= (A+B) \ (A+\bar{B}+\bar{C}) \\ &= A+B(\bar{B}+\bar{C}) \\ &= A + B\bar{B} + B\bar{C} \\ &= A + B\bar{C} \leftarrow \\ &= (A+B) \ (A+\bar{C}) \end{aligned}$$

From Distributive law
 $(X + C)(X + \bar{C}) = X + C\bar{C}$

From Distributive law
 $A + BC = (A + B)(A + C)$

| | A | B | C | F | |
|---|---|---|---|---|-----------|
| 0 | 0 | 0 | 0 | 0 | $A+B+C$ |
| 1 | 0 | 0 | 1 | 0 | $A+B+C'$ |
| 2 | 0 | 1 | 0 | 1 | |
| 3 | 0 | 1 | 1 | 0 | $A+B'+C'$ |
| 4 | 1 | 0 | 0 | 1 | |
| 5 | 1 | 0 | 1 | 1 | |
| 6 | 1 | 1 | 0 | 1 | |
| 7 | 1 | 1 | 1 | 1 | |



MIT-WPU

॥ विद्यानन्तर्दृशं ध्रुवा ॥

Practice Example

POS Representation

$$Y = \pi(M_1, M_3)$$

$$Y = \pi M(1,3)$$

- For given truth table minimize POS expression

| | A | B | Y |
|----|---|---|---|
| M0 | 0 | 0 | 1 |
| M1 | 0 | 1 | 0 |
| M2 | 1 | 0 | 1 |
| M3 | 1 | 1 | 0 |

$$\begin{aligned} Y &= (A + \bar{B})(\bar{A} + \bar{B}) \\ &= \bar{B} + A\bar{A} \\ &= \bar{B} + 0 \\ Y &= \bar{B} \end{aligned}$$

From Distributive law
 $(A + B)(A + C) = A + BC$

SOP Representation

$$Y = \sum m(0,2)$$

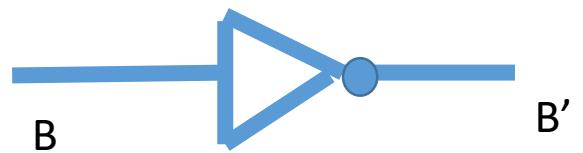
SOP FORM

$$Y = \sum m(0,2)$$

$$Y = A'B' + AB'$$

$$Y = B'(A' + A)$$

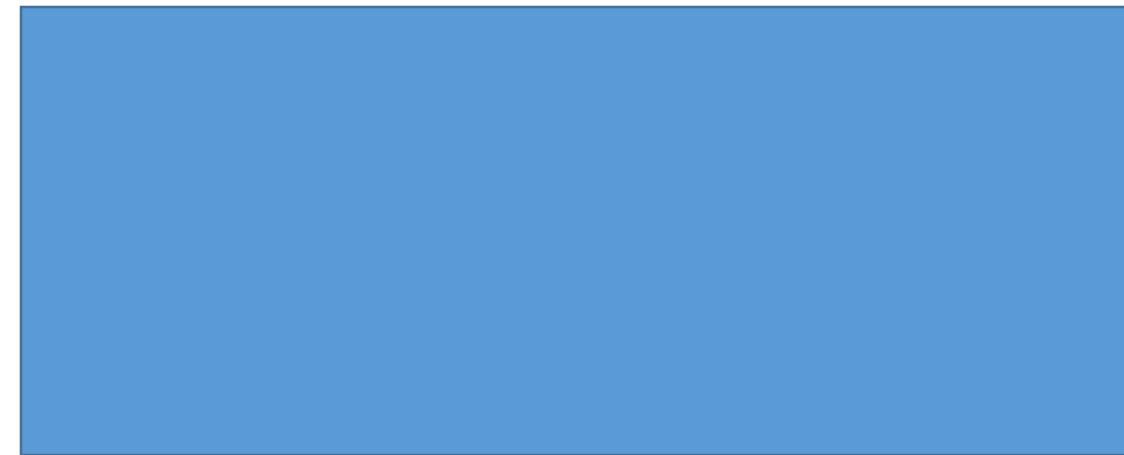
$$Y = B'$$



Practice Example (Contd..)

- For following truth table find minterm & maxterm i.e. SOP & POS forms

| | A | B | C | F |
|-------------|---|---|---|---|
| m0 0 | 0 | 0 | 0 | 1 |
| M1 1 | 0 | 0 | 1 | 0 |
| m2 2 | 0 | 1 | 0 | 1 |
| m3 3 | 0 | 1 | 1 | 1 |
| M4 4 | 1 | 0 | 0 | 0 |
| M5 5 | 1 | 0 | 1 | 0 |
| m6 6 | 1 | 1 | 0 | 1 |
| m7 7 | 1 | 1 | 1 | 1 |



Practice Example (Contd..)

- For following truth table find summation & product forms i.e. SOP & POS form

| | A | B | C | F |
|------|---|---|---|---|
| m0 0 | 0 | 0 | 0 | 1 |
| M1 1 | 0 | 0 | 1 | 0 |
| m2 2 | 0 | 1 | 0 | 1 |
| m3 3 | 0 | 1 | 1 | 1 |
| M4 4 | 1 | 0 | 0 | 0 |
| M5 5 | 1 | 0 | 1 | 0 |
| m6 6 | 1 | 1 | 0 | 1 |
| m7 7 | 1 | 1 | 1 | 1 |

$$Y(A,B,C) = \sum m(0,2,3,6,7)$$

$$Y(A,B,C) = \pi M(1,4,5)$$

Practice Example (Contd..)

- $Y(A, B, C) = \sum m(0, 2, 3, 6, 7)$

Practice Example (Contd..)

- $Y(A, B, C) = \sum m(0, 2, 3, 6, 7)$

$$\begin{aligned}
 Y &= \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + AB\overline{C} + ABC \\
 &= \overline{A}\overline{B}\overline{C} + \overline{A}B(\overline{C}+C) + AB(\overline{C}+C) \\
 &= \overline{A}\overline{B}\overline{C} + \overline{A}B + AB \\
 &= \overline{A}\overline{B}\overline{C} + (\overline{A}+\overline{A})B \\
 &= \overline{A}\overline{B}\overline{C} + B \\
 &= X \cdot \overline{B} + B \quad \leftarrow \text{Let } X = \overline{A}\overline{C} \\
 &= X + B \\
 &= \overline{A}\overline{C} + B
 \end{aligned}$$

Practice Example (Contd..)

- $\text{Y(A,B,C)} = \pi M(1,4,5)$

Practice Example (Contd..)

Simplify the following expression

$$Y(A,B,C) = \sum m(0,2,4,6) \quad \text{Ans: } Y = C'$$

$$Y(A,B) = \pi M(0,1,2) \quad \text{Ans: } Y = AB$$

Karnaugh Map K-MAP

Introduction

- Expression reduction through basic rules of Boolean algebra is time consuming for higher variables.
- Solution is K-Map - graphical method
- A **Karnaugh map (K-Map)** is a pictorial method used to minimize Boolean expression without having to use Boolean algebra theorems & equation manipulation
- It is a special version of truth table
- Using K-Map, expressions with 2 to 4 variables are easily minimized

2 – Variables K- Map

A. SOP: -

| | \bar{B} | B |
|-----------|------------------|------------|
| \bar{A} | $\bar{A}\bar{B}$ | $\bar{A}B$ |
| A | $A\bar{B}$ | AB |

B. POS: -

| | B | \bar{B} |
|-----------|-------------|-------------------|
| \bar{A} | $A+B$ | $A+\bar{B}$ |
| A | $\bar{A}+B$ | $\bar{A}+\bar{B}$ |

A. SOP: -

| | \bar{B} | B |
|-----------|-----------|---|
| \bar{A} | 1 | 1 |
| A | 2 | 3 |

3 – Variables K- Map

| | | BC | 00 | 01 | 11 | 10 | |
|--|--|----|----|----|----|----|---|
| | | A | 0 | 0 | 1 | 3 | 2 |
| | | A | 1 | 4 | 5 | 7 | 6 |
| | | | 00 | 01 | 11 | 10 | |
| | | | 0 | 1 | 3 | 2 | |
| | | | 1 | 4 | 5 | 6 | |

| | | BC | 00 | 01 | 11 | 10 | |
|--|--|----|----|----------|---------|--------|---------|
| | | A | 0 | $A'B'C'$ | $A'B'C$ | $A'BC$ | $A'BC'$ |
| | | A | 1 | $AB'C'$ | $AB'C$ | ABC | ABC' |
| | | | 00 | 01 | 11 | 10 | |
| | | | 0 | $A'B'C'$ | $A'B'C$ | $A'BC$ | $A'BC'$ |
| | | | 1 | $AB'C'$ | $AB'C$ | ABC | ABC' |

SOP Form

| | | BC | 00 | 01 | 11 | 10 | |
|--|--|----|----|----------|-----------|------------|-----------|
| | | A | 0 | $A+B+C$ | $A+B+C'$ | $A+B'+C'$ | $A+B'+C$ |
| | | A | 1 | $A'+B+C$ | $A'+B+C'$ | $A'+B'+C'$ | $A'+B'+C$ |
| | | | 00 | 01 | 11 | 10 | |
| | | | 0 | $A+B+C$ | $A+B+C'$ | $A+B'+C'$ | $A+B'+C$ |
| | | | 1 | $A'+B+C$ | $A'+B+C'$ | $A'+B'+C'$ | $A'+B'+C$ |

POS Form

4 – Variables K- Map

| | | CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|----|
| | | AB | 00 | 01 | 11 | 10 |
| AB | 00 | 0 | 1 | 3 | 2 | |
| | 01 | 4 | 5 | 7 | 6 | |
| | 11 | 12 | 13 | 15 | 14 | |
| | 10 | 8 | 9 | 11 | 10 | |

| | | CD | 00 | 01 | 11 | 10 | |
|----|----|---------|--------|----------|---------|--------|---------|
| | | AB | 00 | A'B'C'D' | A'B'C'D | A'B'CD | A'B'CD' |
| AB | 00 | A'BC'D' | A'BC'D | A'BCD | A'BCD' | | |
| | 01 | | | | | | |
| | 11 | ABC'D' | ABC'D | ABCD | ABCD' | | |
| | 10 | AB'C'D' | AB'C'D | AB'CD | AB'CD' | | |

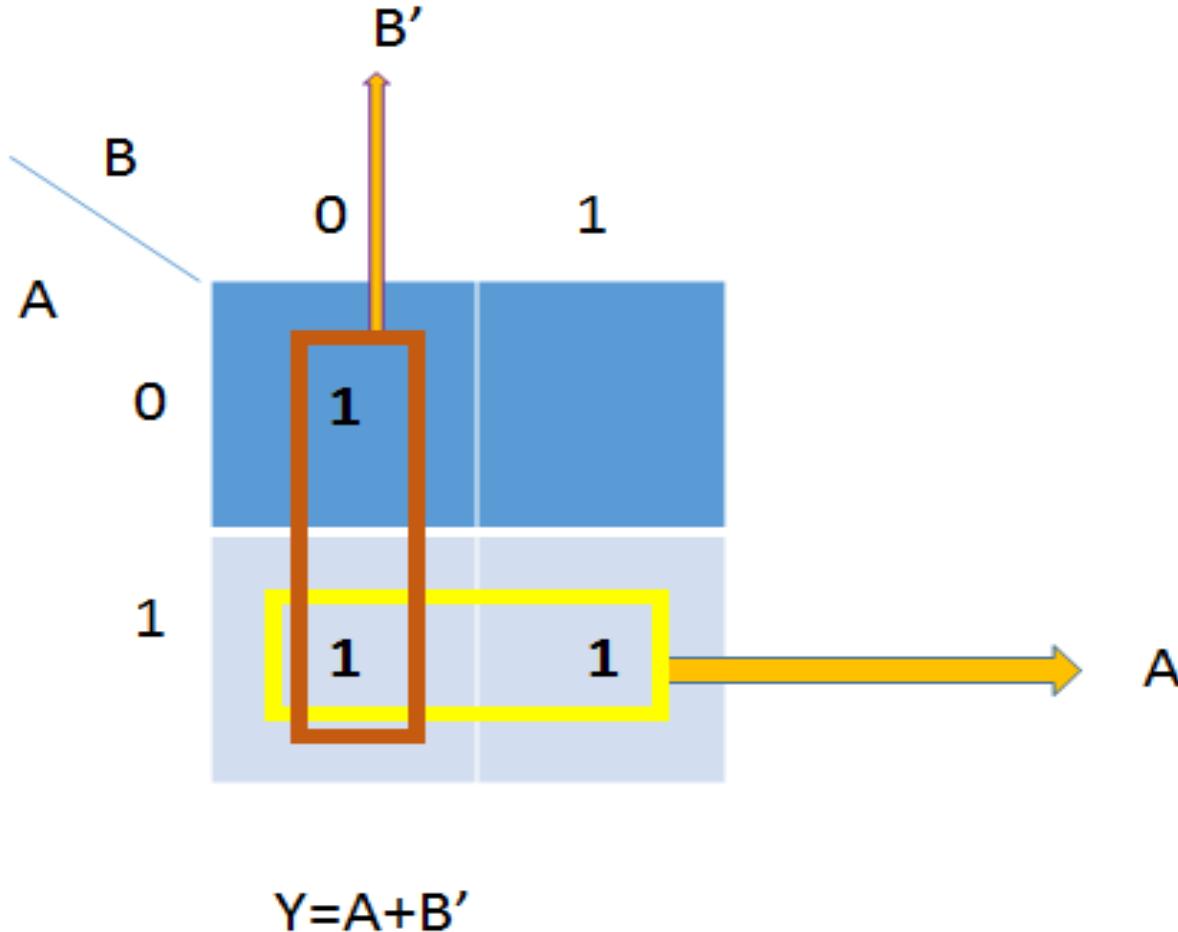
| | | CD | 00 | 01 | 11 | 10 | |
|----|----|-----------|------------|-------------|------------|-----------|----------|
| | | AB | 00 | A+B+C+D | A+B+C+D' | A+B+C'+D' | A+B+C'+D |
| AB | 00 | A+B'+C+D | A+B'+C+D' | A+B'+C'+D' | A+B'+C'+D | | |
| | 01 | | | | | | |
| | 11 | A'+B'+C+D | A'+B'+C+D' | A'+B'+C'+D' | A'+B'+C'+D | | |
| | 10 | A'+B+C+D | A'+B+C+D' | A'+B+C'+D' | A'+B+C'+D | | |

SOP Form

POS Form

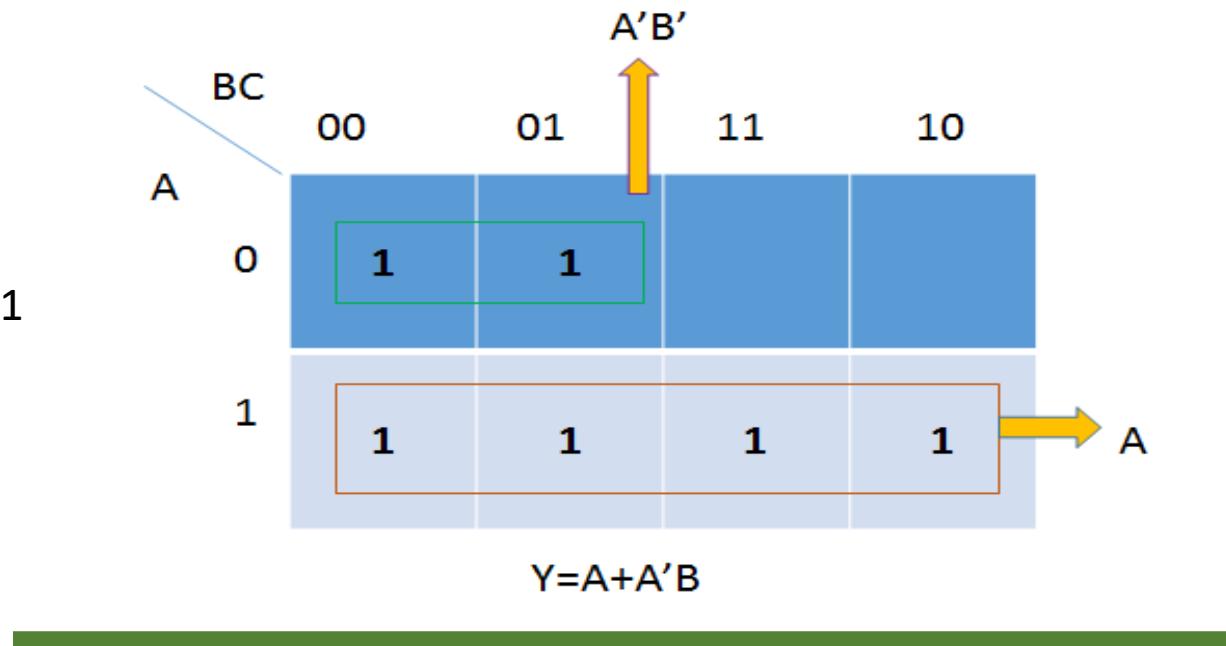
Practice Examples

| | |
|----|----|
| m0 | m1 |
| m2 | m3 |

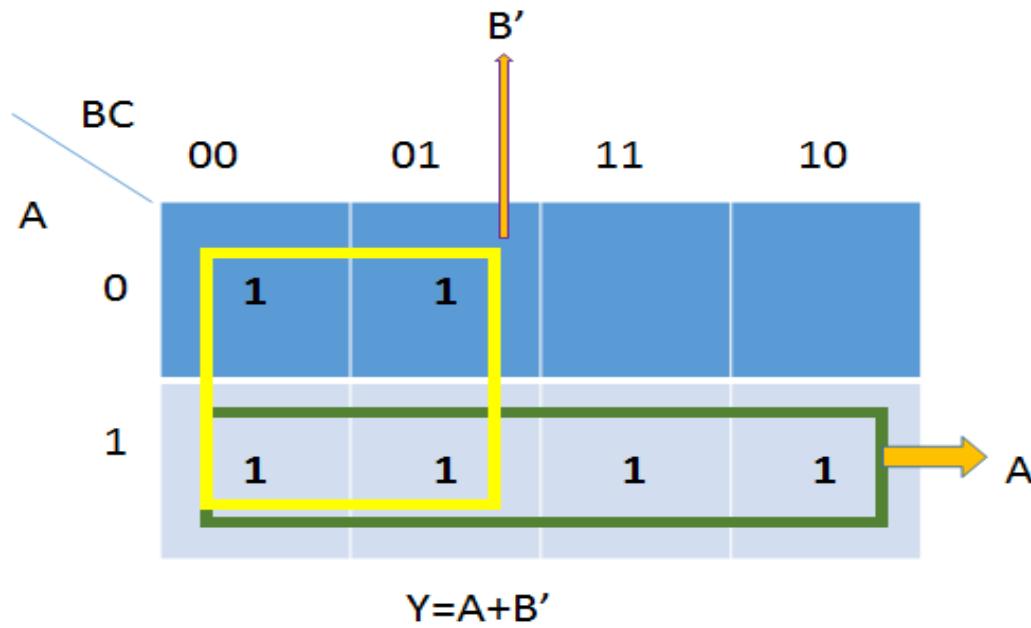


$$\begin{aligned}
 Y(AB) &= \sum m(0, 2, 3) \\
 &= m0 + m2 + m3 \\
 &= \bar{A}\bar{B} + A\bar{B} + AB \\
 &= (\bar{A}+A)\bar{B} + AB \\
 &= \bar{B} + AB \leftarrow \bar{A} + AB = \bar{A} + B \\
 &= \bar{B} + A
 \end{aligned}$$

Way 1



Way 2



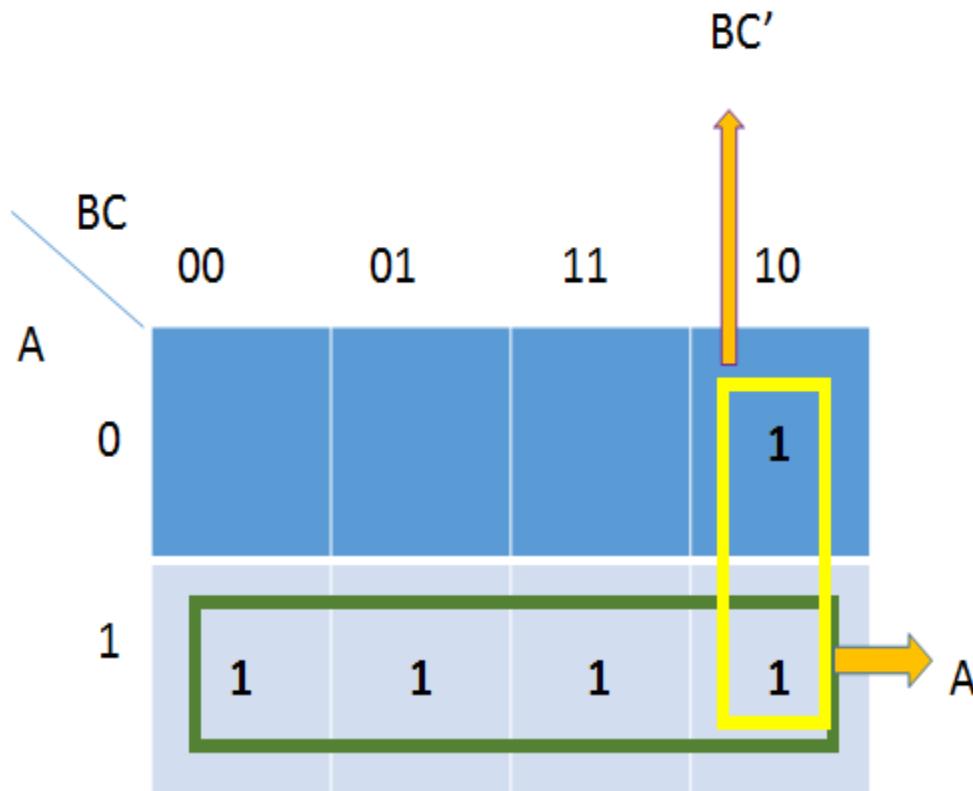
Practice Examples(Contd..)

- $F(A, B, C) = \sum m (0, 4, 6, 1, 5, 7)$

$$\begin{aligned}
 F &= \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + AB\bar{C} + \bar{A}\bar{B}C + A\bar{B}C + ABC \\
 &= \bar{B}\bar{C}(\bar{A}+A) + AB(\bar{C}+C) + \bar{B}C(\bar{A}+A) \\
 &= \bar{B}\bar{C}(1) + AB(1) + \bar{B}C(1) \\
 &= \bar{B}\bar{C} + AB + \bar{B}C \\
 &= \bar{B}(\bar{C}+C) + AB \\
 &= \bar{B} + AB \\
 &= \bar{B} + A
 \end{aligned}$$

| | | | | |
|----|----|----|----|----|
| BC | 00 | 01 | 11 | 10 |
| A | 0 | 1 | 3 | 2 |
| 1 | 4 | 5 | 7 | 6 |

Practice example



$$Y = A + BC'$$

- $F(A,B,C) = \sum m(2,4,5,6,7)$

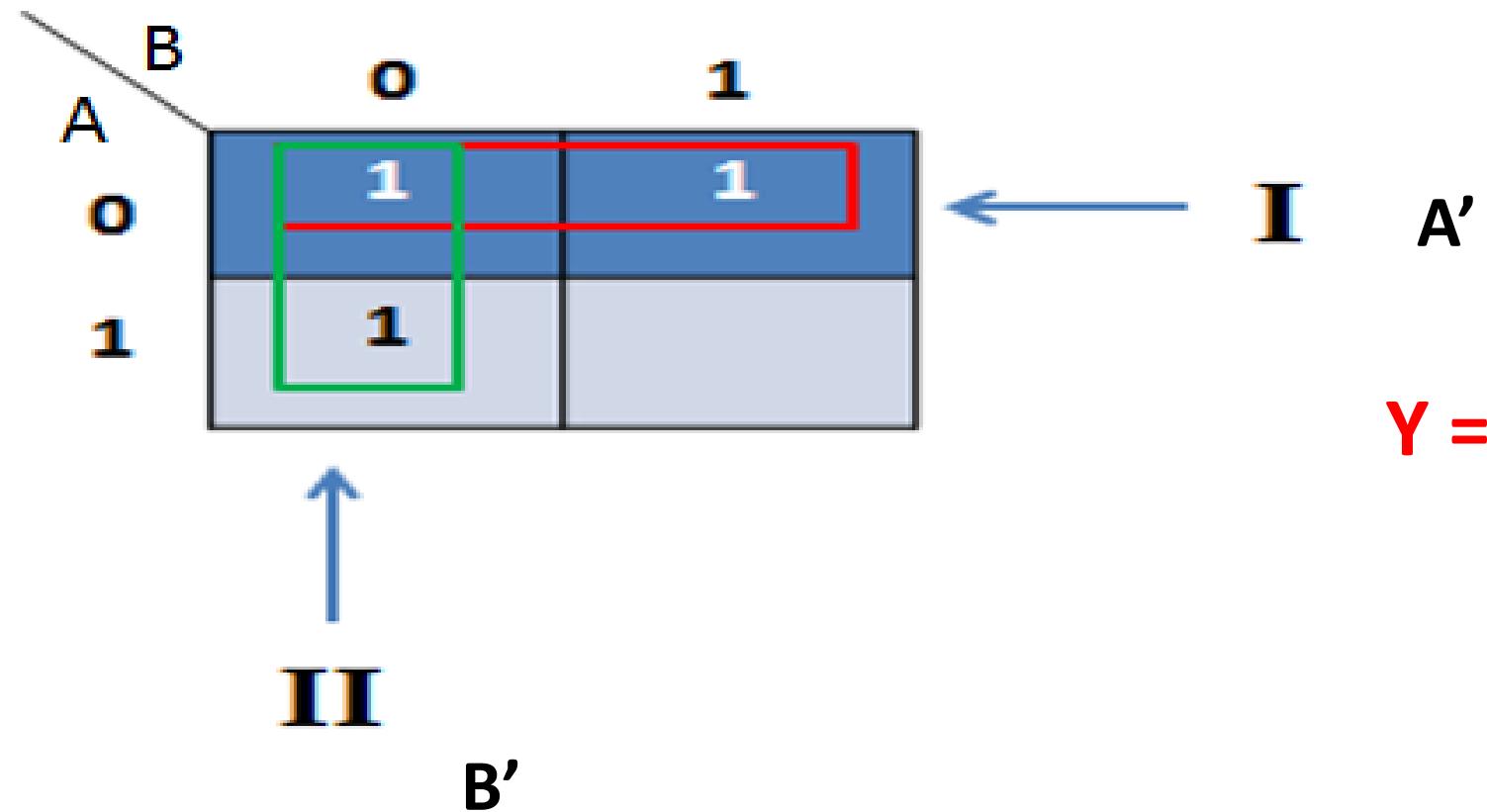
$$\begin{aligned}
 F &= \overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + \overline{A}\overline{B}C + AB\overline{C} + ABC \\
 &= \overline{A}\overline{B}\overline{C} + AB(\overline{C} + C) + AB(C + \overline{C}) \\
 &= \overline{A}\overline{B}\overline{C} + AB(1) + AB(1) \\
 &= \overline{A}\overline{B}\overline{C} + AB \\
 &= \overline{A}\overline{B}\overline{C} + A(\overline{B} + B) \\
 &= \overline{A}\overline{B}\overline{C} + A \quad \text{if } \overline{B}\overline{C} = X \\
 &= X + A \quad \text{if } \overline{A}X = X \\
 &= A + X \\
 F &= A + BC'
 \end{aligned}$$

| | | | | | |
|---|----|----|----|----|----|
| | BC | 00 | 01 | 11 | 10 |
| A | 0 | 0 | 1 | 3 | 2 |
| 1 | 4 | 5 | 7 | 6 | |

Practice Example

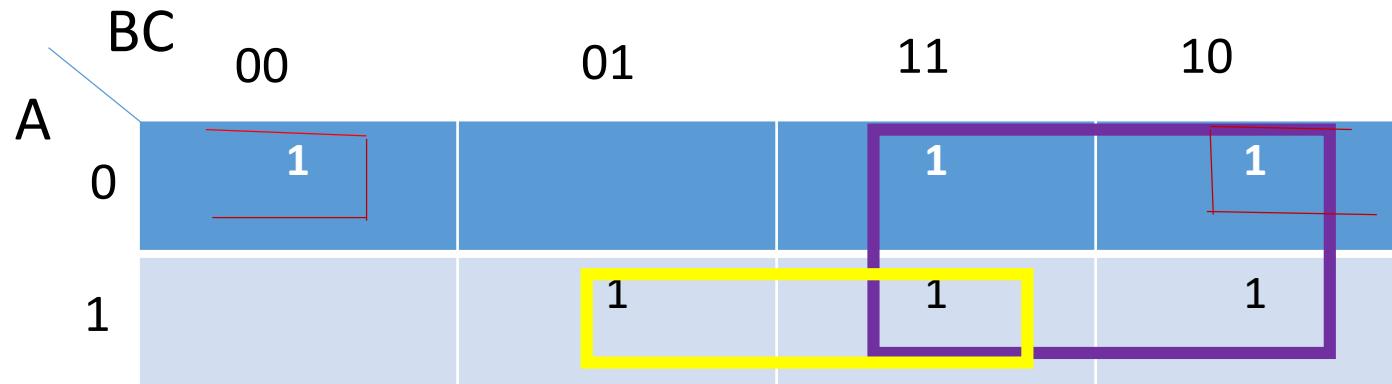
$$F(A,B) = \bar{A}\bar{B} + A\bar{B} + \bar{A}B$$

Minterms: 0,2,1



Example (Contd..)

- $Z = f(A, B, C) = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + \bar{A}BC + ABC + A\bar{B}C$

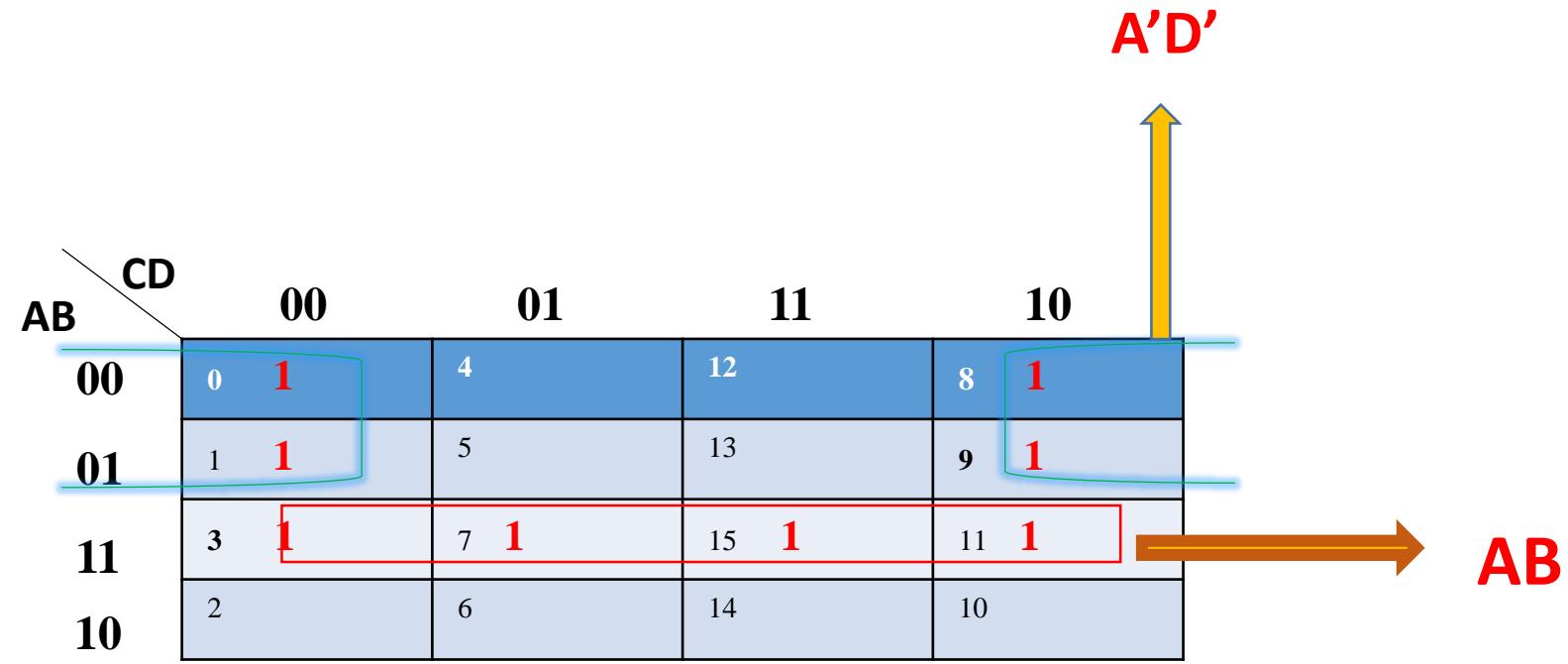


$$Z = A'C' + AC + B$$

Minterms: 0,2,6,3,7,5

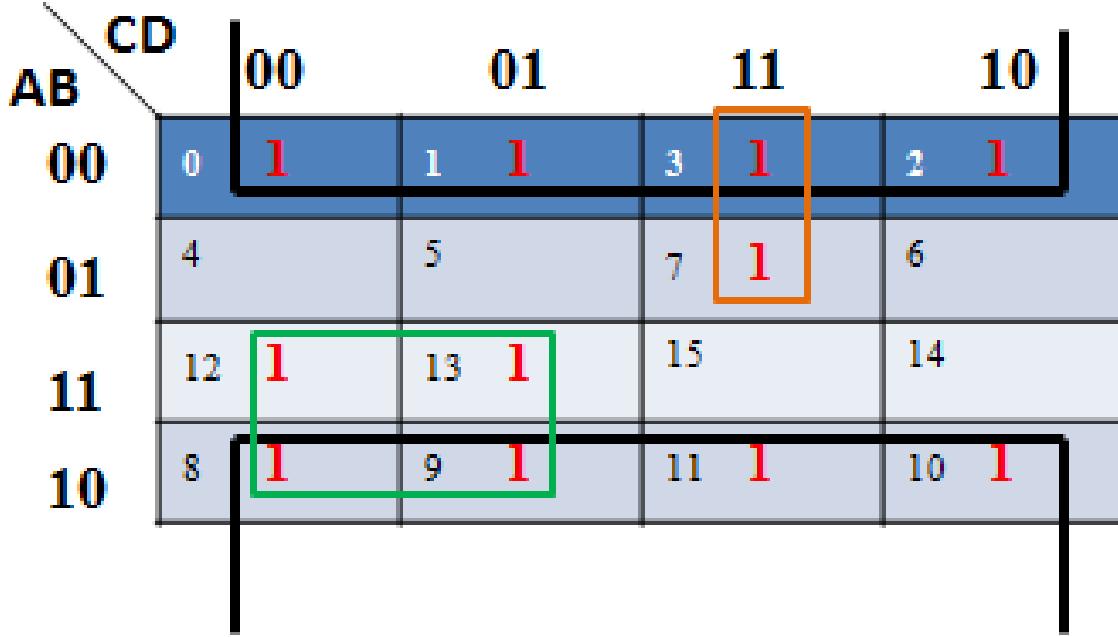
| | | BC | 00 | 01 | 11 | 10 |
|----|---|----|----|----|----|----|
| | | A | 0 | 1 | 3 | 2 |
| BC | A | 0 | 0 | 1 | 3 | 2 |
| | | 1 | 4 | 5 | 7 | 6 |

Four Adjacent ones



$$Y = A'D' + AB$$

Practice Example

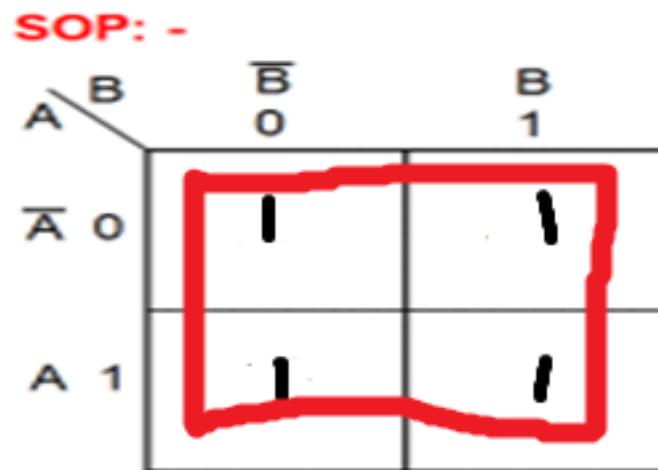


$$Y = \bar{B} + A\bar{C} + \bar{A}CD$$

| A | B | C | D | Y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

Example

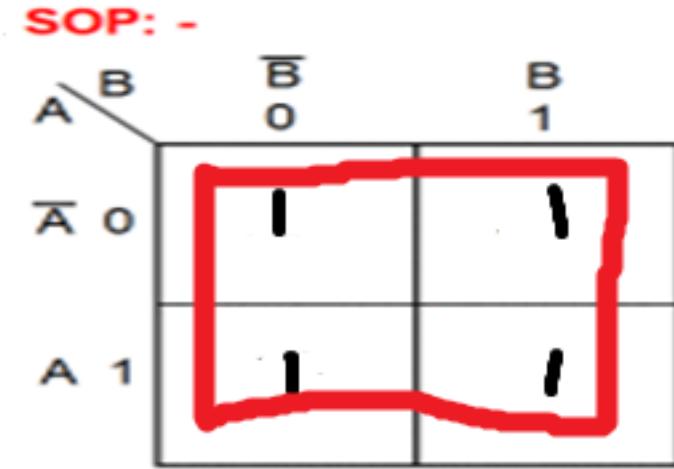
What is the solution of k map if all entries are 1?. We want SOP form solution.



What is the solution of k map if all entries are 1?. We want SOP form solution.

| | | BC | 00 | 01 | 11 | 10 |
|---|---|----|----|----|----|----|
| | | A | 0 | 1 | 3 | 2 |
| | | 0 | 0 | 1 | 3 | 2 |
| A | B | 1 | 4 | 5 | 7 | 6 |
| | | 0 | 0 | 1 | 3 | 2 |

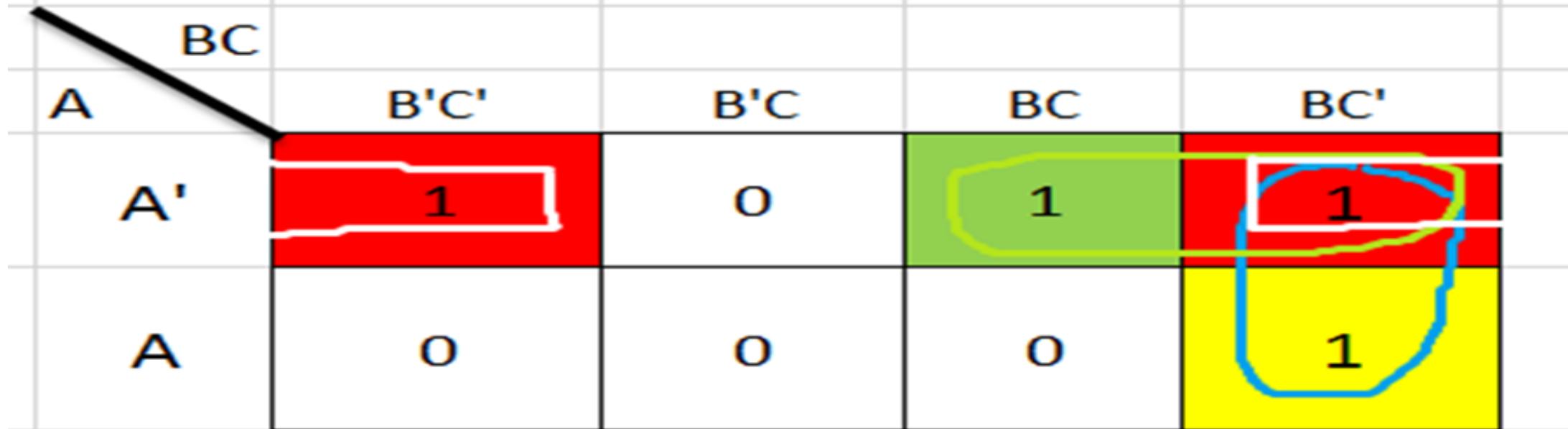
| | | CD | 00 | 01 | 11 | 10 | |
|----|----|----|----|----|----|----|---|
| | | AB | 00 | 0 | 1 | 3 | 2 |
| | | 00 | 0 | 1 | 3 | 2 | |
| AB | CD | 01 | 4 | 5 | 7 | 6 | |
| | | 01 | 4 | 5 | 7 | 6 | |
| AB | CD | 11 | 12 | 13 | 15 | 14 | |
| | | 11 | 12 | 13 | 15 | 14 | |
| AB | CD | 10 | 8 | 9 | 11 | 10 | |
| | | 10 | 8 | 9 | 11 | 10 | |



$$\begin{aligned}
 Y(A, B) &= A'B' + A'B + AB' + AB \\
 &= A'(B' + B) + A B' + B \\
 &= A' + A \\
 &= 1
 \end{aligned}$$

Folding K map in horizontal direction

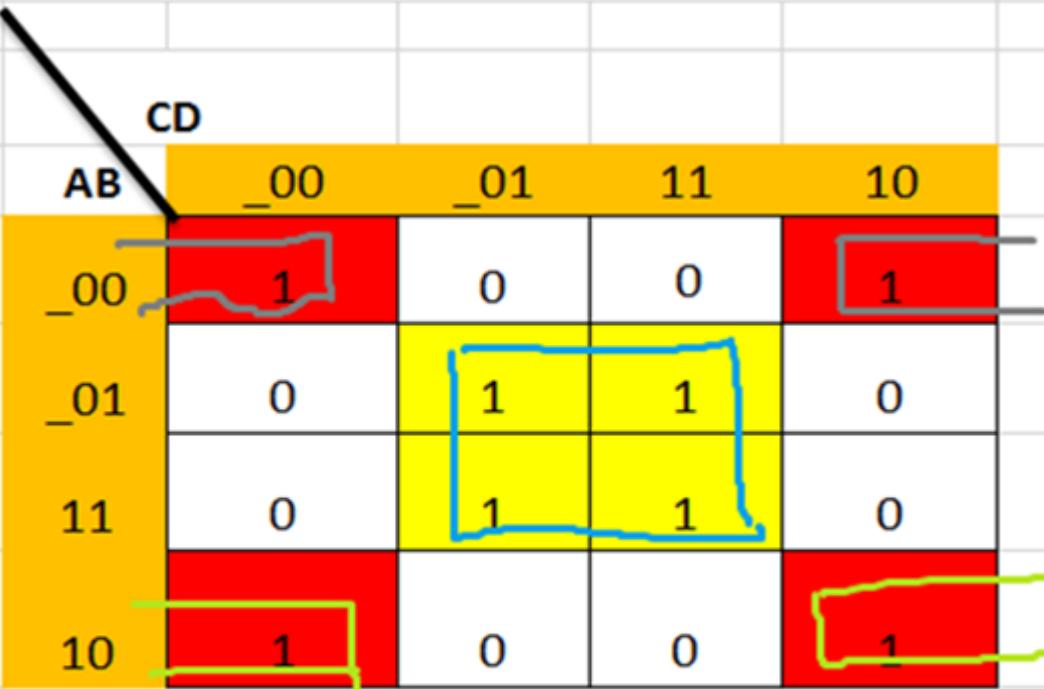
$$Y = \sum m(0, 2, 3, 6)$$



$$Y = \underline{\text{BC}'} + \underline{\text{A}'\text{B}} + \underline{\text{A}'\text{C}'}$$

Folding K map only in horizontal direction

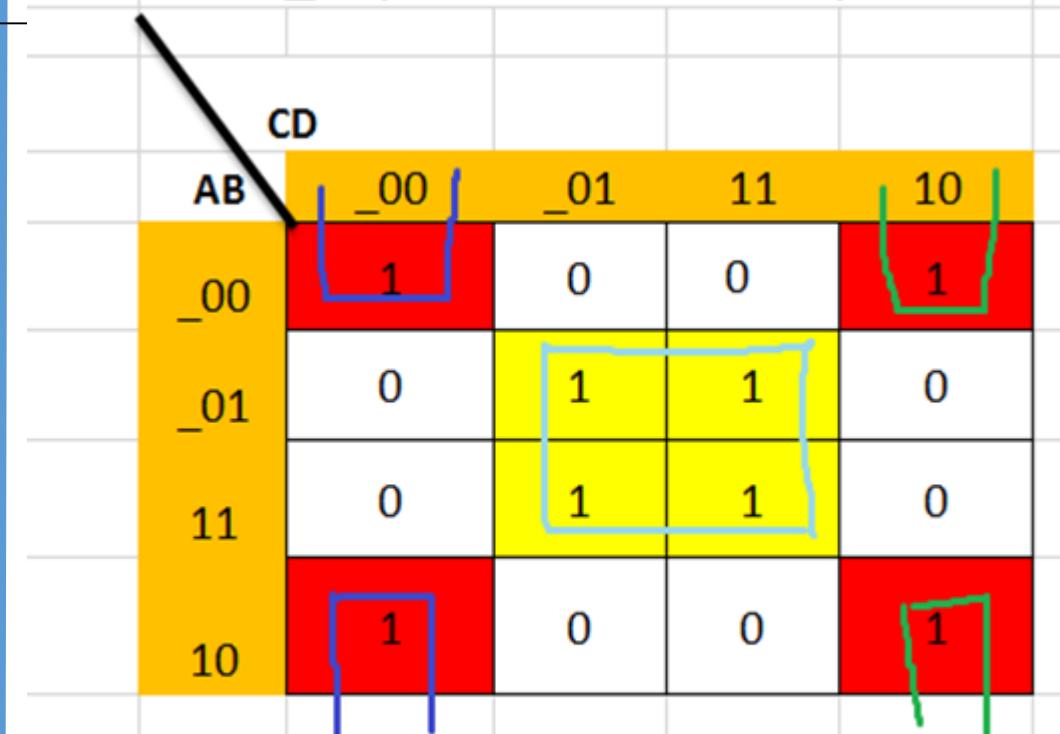
$$Y = \sum m(0,1,3,6,10,11,14)$$



$$Y = A'B'D' + \underline{AB'D'} + \underline{BD}$$

Folding K map only in Vertical direction

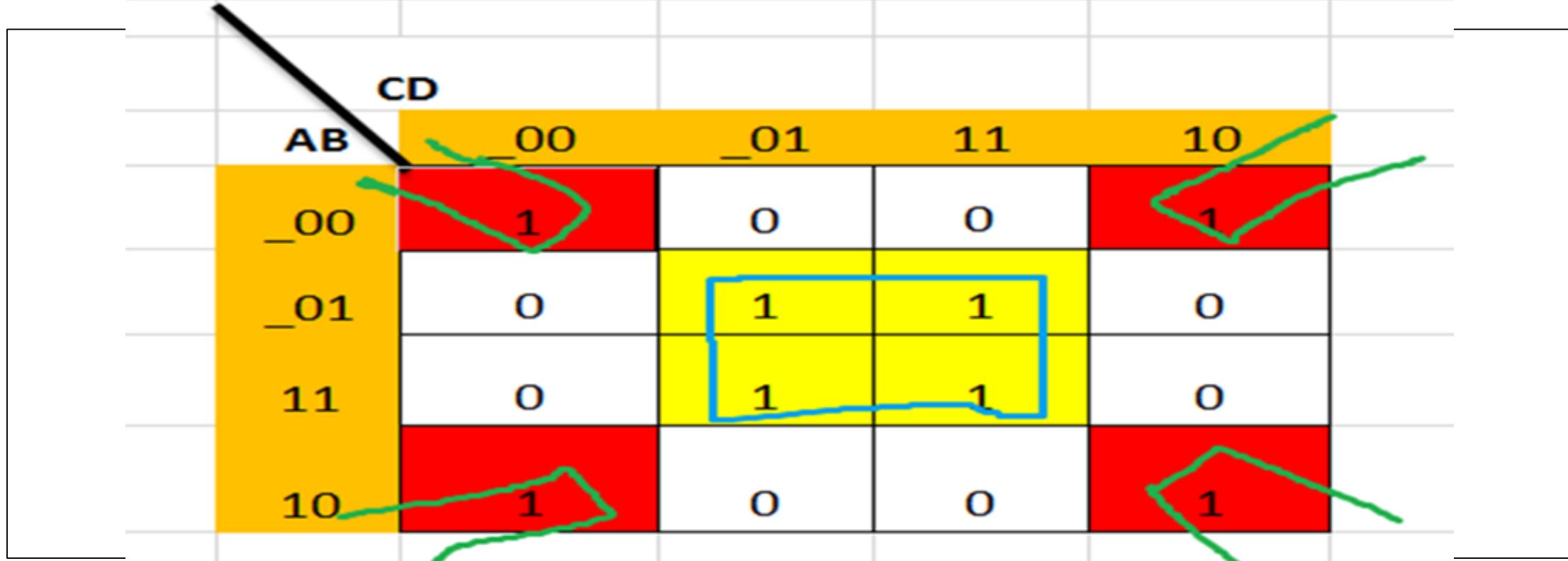
$$Y = \sum m(0,1,3,6,10,11,14)$$



$$Y = \underline{B'C'D'} + B'CD' + BD$$

Folding K map in both horizontal as well as vertical direction

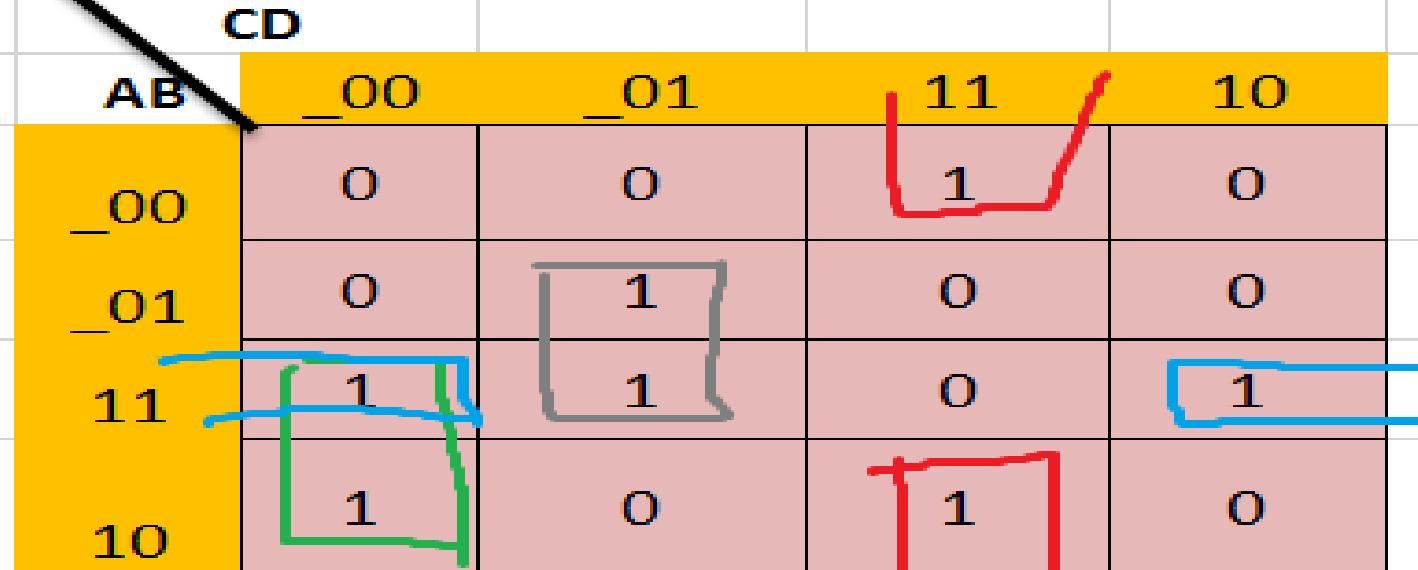
$$Y = \sum m(0, 1, 3, 6, 10, 11, 14)$$



$$\underline{Y=B'D'} + \underline{BD}$$

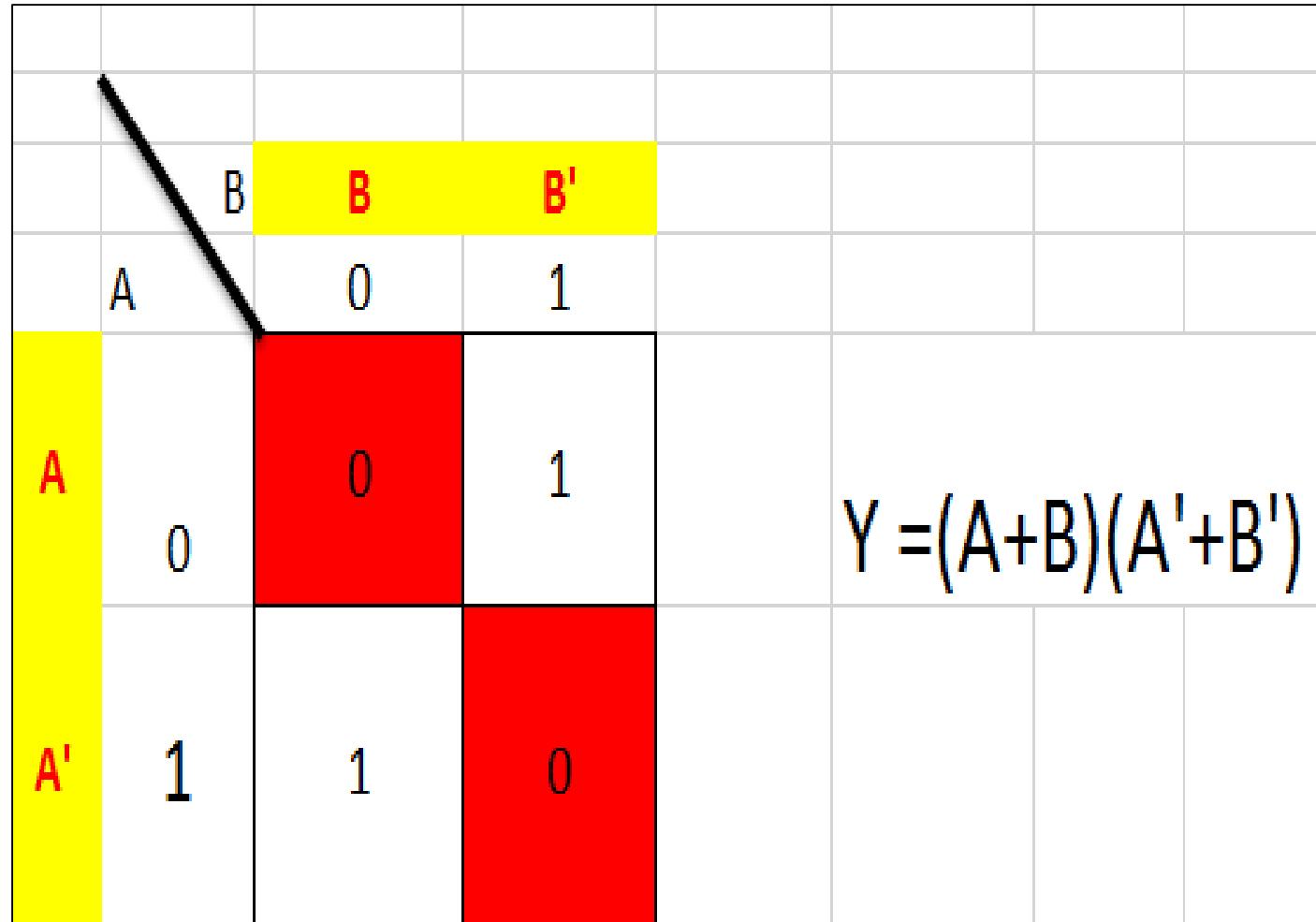
Folding K map - AN EXAMPLE

$$Y = \sum m(3, 5, 8, 11, 12, 13, 14)$$

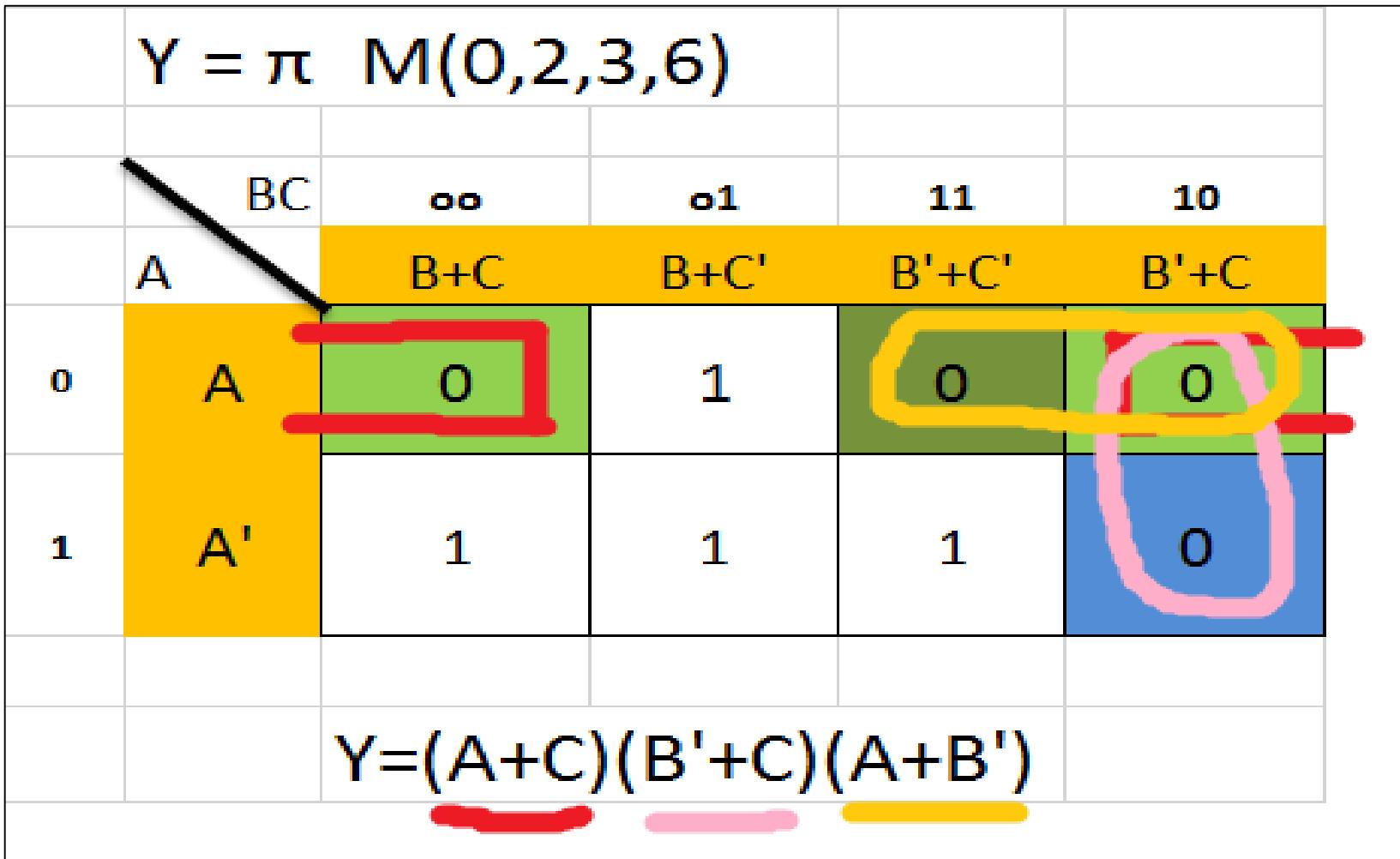


$$Y = \underline{AC'D'} + \underline{BC'D} + \underline{B'CD} + \underline{ABD'}$$

K map for POS form Boolean expressions



POS form – k map simplification



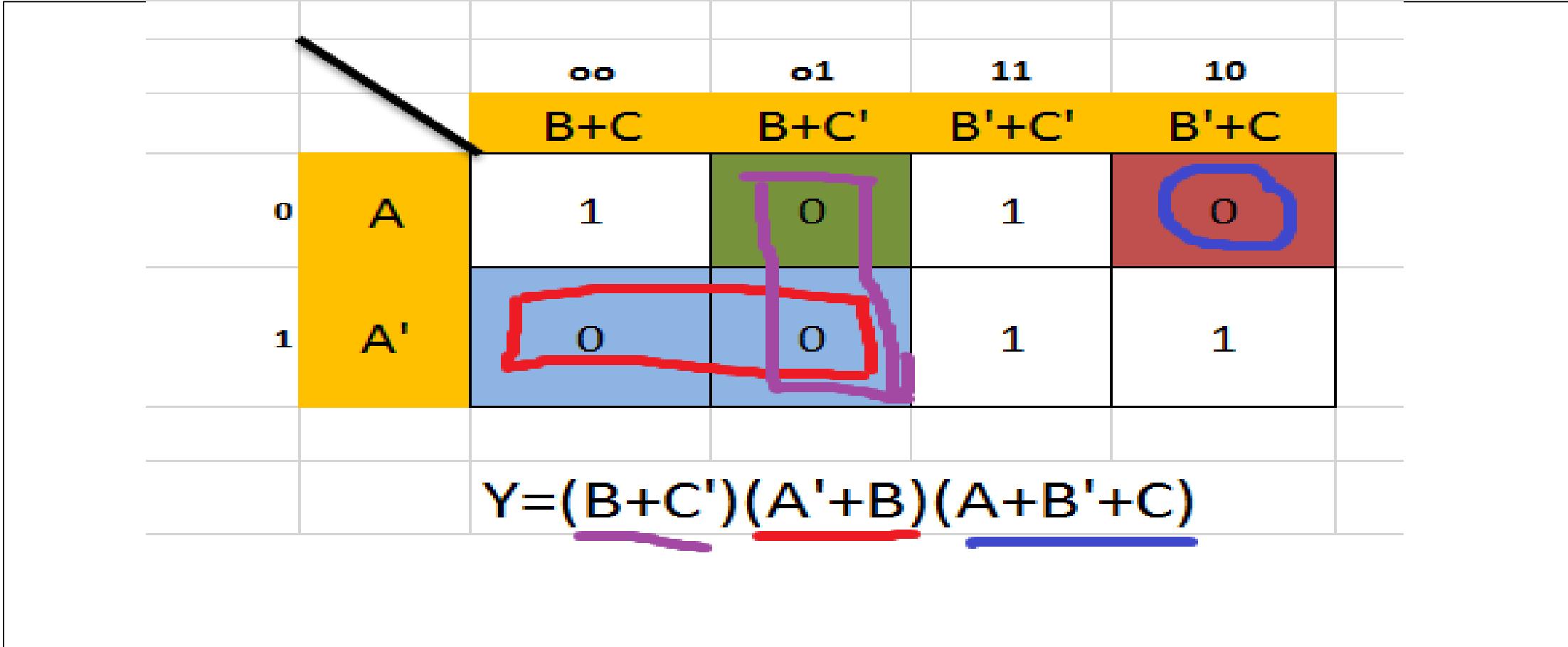
POS form – k map simplification

| | | $Y = \pi M(0,1,3,6,10,11,14)$ | | | | | |
|----|-------|-------------------------------|-----|------|-------|------|--|
| | | CD | 00 | 01 | 11 | 10 | |
| | | AB | C+D | C+D' | C'+D' | C'+D | |
| 00 | A+B | 0 | 0 | 0 | 1 | | |
| 01 | A+B' | 1 | 1 | 1 | 0 | | |
| 11 | A'+B' | 1 | 1 | 1 | 0 | | |
| 10 | A'+B | 1 | 1 | 0 | 0 | | |

Y = (A+B+C) (A+B+D') (B'+C'+D) (A'+B+C')

POS form – k map simplification

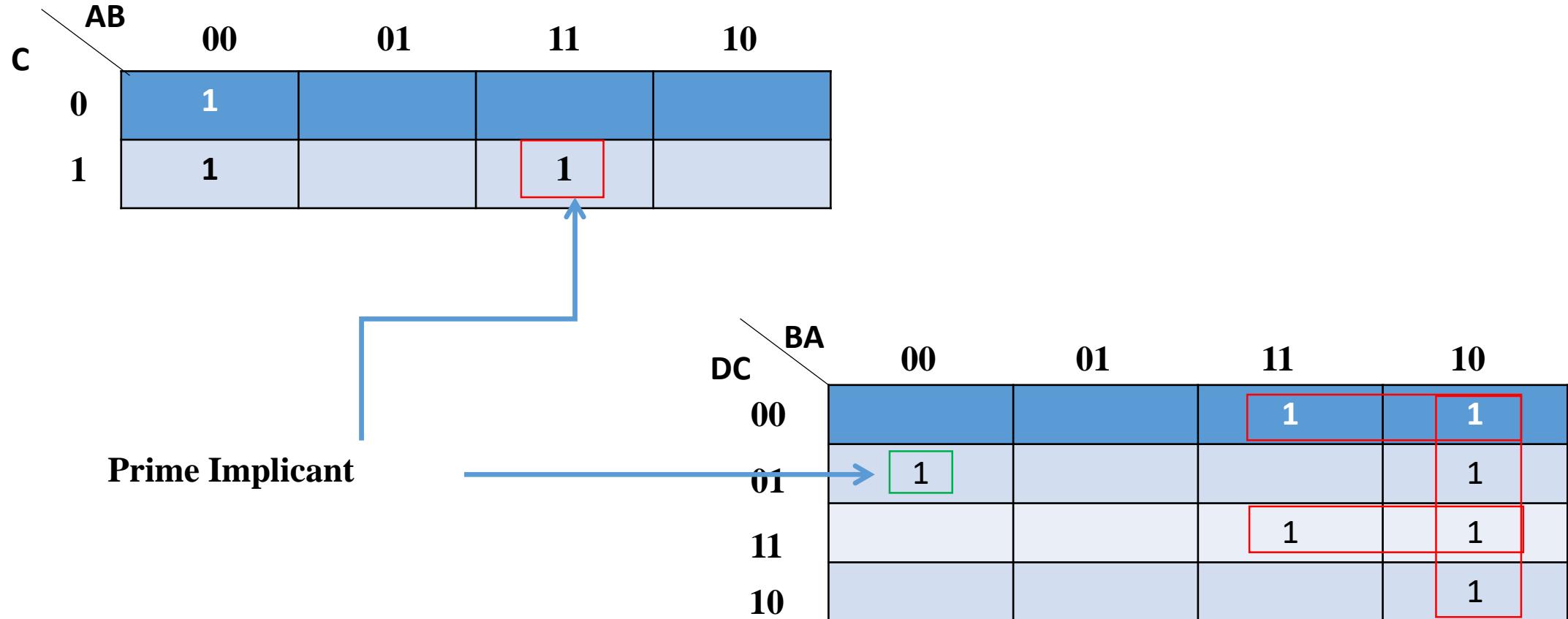
$$Y = \pi M(1,2,4,5)$$



Prime Implicant

- The process of simplification involves grouping of Minterms & identifying Prime Implicants (PI) & Essential Prime Implicants (EPI)
- Prime implicant is a group of ones (minterms) that cannot be combined with any other ones (minterms) or groups
- EPI is PI, in which one or more minterms are unique
 - i.e. it contain atleast one minterm which is not contained in any other PI

Prime Implicant (Contd..)



Don't Care Condition (**X**)

- The K-Maps are simplified using either 1's or 0's, therefore we make the entries in the map for either 0 or 1
- The cell which do not contain 1 are assumed to contain 0 & vice versa
- This is not always true, since there are cases in which certain combinations of I/P variables do not matter

Don't Care Condition (X)

॥ विद्यानन्तर्धावं ध्रुवा ॥

- In such a situation the designer has a flexibility & it is left to him whether to assume 0 or 1 as O/P for each of these combinations
- This condition is known as **Don't Care Condition** and can be represented on K-Map as a X mark in the corresponding cell
- The X mark in a cell may be assumed to be a 1 or 0 depending upon which one leads to a simpler expression

Don't Care Condition (X)

-

| A \ BC | 00 | 01 | 11 | 10 |
|--------|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | * | * |

| A \ BC | 00 | 01 | 11 | 10 |
|--------|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | * | * |

Without Considering don't care conditions.

$$Y = AB'C$$

Assuming don't care as 1.

$$Y = AC$$

Example

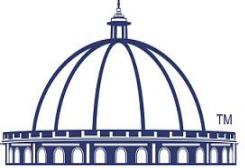
- $f(A,B,C,D) = \sum m(1,3,7,11,15) + d(0,2,5)$

Example

- $f(A, B, C, D) = \sum m(1, 3, 7, 11, 15) + d(0, 2, 5)$

| | | CD | 00 | 01 | 11 | 10 |
|----|----|------------|------------|-------------|------------|----|
| | | AB | 00 | 01 | 11 | 10 |
| AB | 00 | 0 X | 1 1 | 3 1 | 2 X | |
| | 01 | 4 | 5 X | 7 1 | | 6 |
| | 11 | 12 | 13 | 15 1 | | 14 |
| | 10 | 8 | 9 | 11 1 | | 10 |

$$A'D + CD$$



BCD to Excess-3 Code conversion

| | Decimal | BCD Code | | | | Ex-3 Code | | | |
|---------|---------|----------|----|----|----|-----------|----|----|--|
| | B3 | B2 | B1 | B0 | E3 | E2 | E1 | E0 | |
| Valid | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | |
| | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | |
| | 2 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | |
| | 3 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | |
| | 4 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | |
| | 5 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | |
| | 6 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | |
| | 7 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | |
| | 8 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | |
| | 9 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | |
| Invalid | 10 | 1 | 0 | 1 | 0 | X | X | X | |
| | 11 | 1 | 0 | 1 | 1 | X | X | X | |
| | 12 | 1 | 1 | 0 | 0 | X | X | X | |
| | 13 | 1 | 1 | 0 | 1 | X | X | X | |
| | 14 | 1 | 1 | 1 | 0 | X | X | X | |
| | 15 | 1 | 1 | 1 | 1 | X | X | X | |

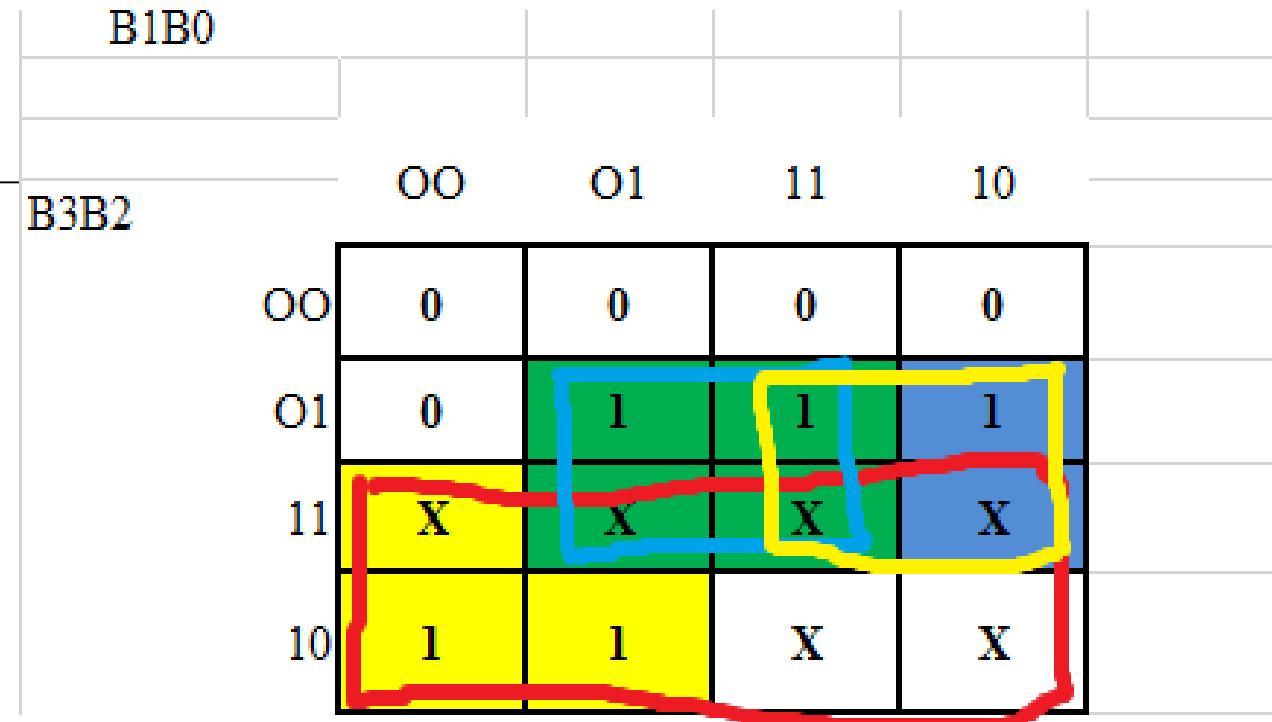


MIT-WPU

॥ विद्यानन्तर्धावं ध्रुवा ॥

BCD to Excess-3 Code conversion

| Decimal | BCD Code | | | | Ex-3 Code | | | |
|---------|----------|----|----|----|-----------|----|----|----|
| | B3 | B2 | B1 | B0 | E3 | E2 | E1 | E0 |
| 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 | 0 |
| 10 | 1 | 0 | 1 | 0 | X | X | X | X |
| 11 | 1 | 0 | 1 | 1 | X | X | X | X |
| 12 | 1 | 1 | 0 | 0 | X | X | X | X |
| 13 | 1 | 1 | 0 | 1 | X | X | X | X |
| 14 | 1 | 1 | 1 | 0 | X | X | X | X |
| 15 | 1 | 1 | 1 | 1 | X | X | X | X |



$$E3 = B3 + B2B0 + B2B1$$

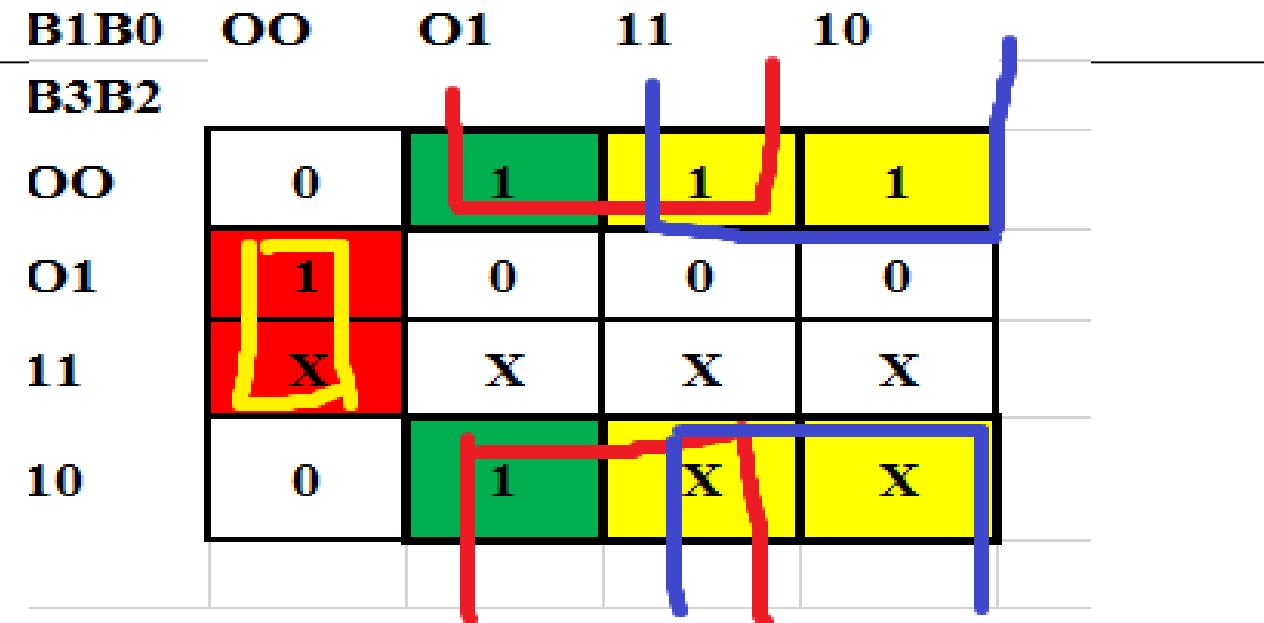


MIT-WPU

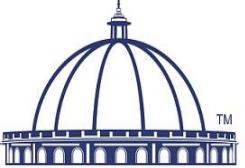
॥ विद्यानन्तर्धावं ध्रुवा ॥

BCD to Excess-3 Code conversion

| | Decimal | BCD Code | | | | Ex-3 Code | | | |
|---------|---------|----------|----|----|----|-----------|----|----|----|
| | | B3 | B2 | B1 | B0 | E3 | E2 | E1 | E0 |
| Valid | 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 | 0 |
| Invalid | 10 | 1 | 0 | 1 | 0 | X | X | X | X |
| | 11 | 1 | 0 | 1 | 1 | X | X | X | X |
| | 12 | 1 | 1 | 0 | 0 | X | X | X | X |
| | 13 | 1 | 1 | 0 | 1 | X | X | X | X |
| | 14 | 1 | 1 | 1 | 0 | X | X | X | X |
| | 15 | 1 | 1 | 1 | 1 | X | X | X | X |



$$E2 = B2'B0 + B2'B1 + B2B1'B0'$$



MIT-WPU

॥ विद्यानन्तर्धावं ध्रुवा ॥

| | | 00 | 01 | 11 | 10 |
|----|----|------|------|----|----|
| | | B1B0 | B3B2 | | |
| 00 | 01 | 0 | 0 | 0 | 0 |
| | | 0 | 1 | 1 | 1 |
| 11 | 10 | X | X | X | X |
| | | 1 | 1 | X | X |

$$X_3 = B_3 + B_2 (B_0 + B_1)$$

| | | 00 | 01 | 11 | 10 |
|----|----|------|------|----|----|
| | | B1B0 | B3B2 | | |
| 00 | 01 | 0 | 1 | 1 | 1 |
| | | 1 | 0 | 0 | 0 |
| 11 | 10 | X | X | X | X |
| | | 0 | 1 | X | X |

$$X_2 = \bar{B}_2 B_1 \bar{B}_0 + \bar{B}_2 B_0 + B_2 \bar{B}_1$$

| | | 00 | 01 | 11 | 10 |
|----|----|------|------|----|----|
| | | B1B0 | B3B2 | | |
| 00 | 01 | 1 | 0 | 1 | 0 |
| | | 1 | 0 | 1 | 0 |
| 11 | 10 | X | X | X | X |
| | | 1 | 0 | X | X |

$$X_1 = \bar{B}_0 B_1 + B_0 B_1$$

| | | 00 | 01 | 11 | 10 |
|----|----|------|------|----|----|
| | | B1B0 | B3B2 | | |
| 00 | 01 | 1 | 0 | 0 | 1 |
| | | 1 | 0 | 0 | 1 |
| 11 | 10 | X | X | X | X |
| | | 1 | 0 | X | X |

$$X_0 = \bar{B}_0$$

Change in position of variable (Important to observe)

$$Y(A,B,C,D) = \sum m(3, 5, 8, 11, 12, 13, 14)$$

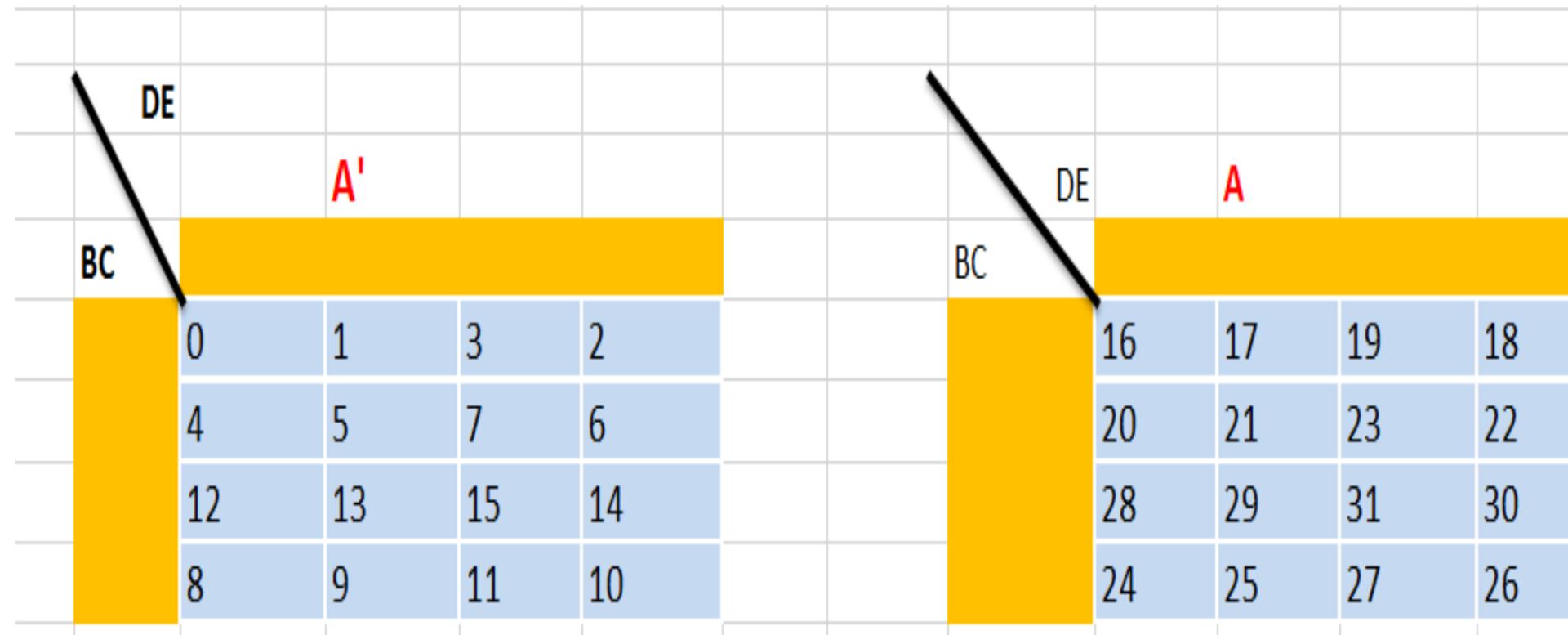
| | | AB | | | | |
|----|----|----|----|----|----|----|
| | | CD | 00 | 01 | 11 | 10 |
| CD | 00 | 0 | 4 | 12 | 8 | |
| | 01 | 1 | 5 | 13 | 9 | |
| | 11 | 3 | 7 | 15 | 11 | |
| | 10 | 2 | 6 | 14 | 10 | |

$$Y(A,B,C,D) = \sum m(3, 5, 8, 11, 12, 13, 14)$$

| | | CD | | | | |
|----|----|----|----|----|----|----|
| | | AB | 00 | 01 | 11 | 10 |
| AB | 00 | 0 | 1 | 3 | 2 | |
| | 01 | 4 | 5 | 7 | 6 | |
| | 11 | 12 | 13 | 15 | 14 | |
| | 10 | 8 | 9 | 11 | 10 | |

5 variable Kmap (Not in syllabus)

$$f(A,B,C,D,E) = Y = \sum m(0,1,3,6,10,11,14,16,20,25,28,29,31)$$



Logic Families

Introduction

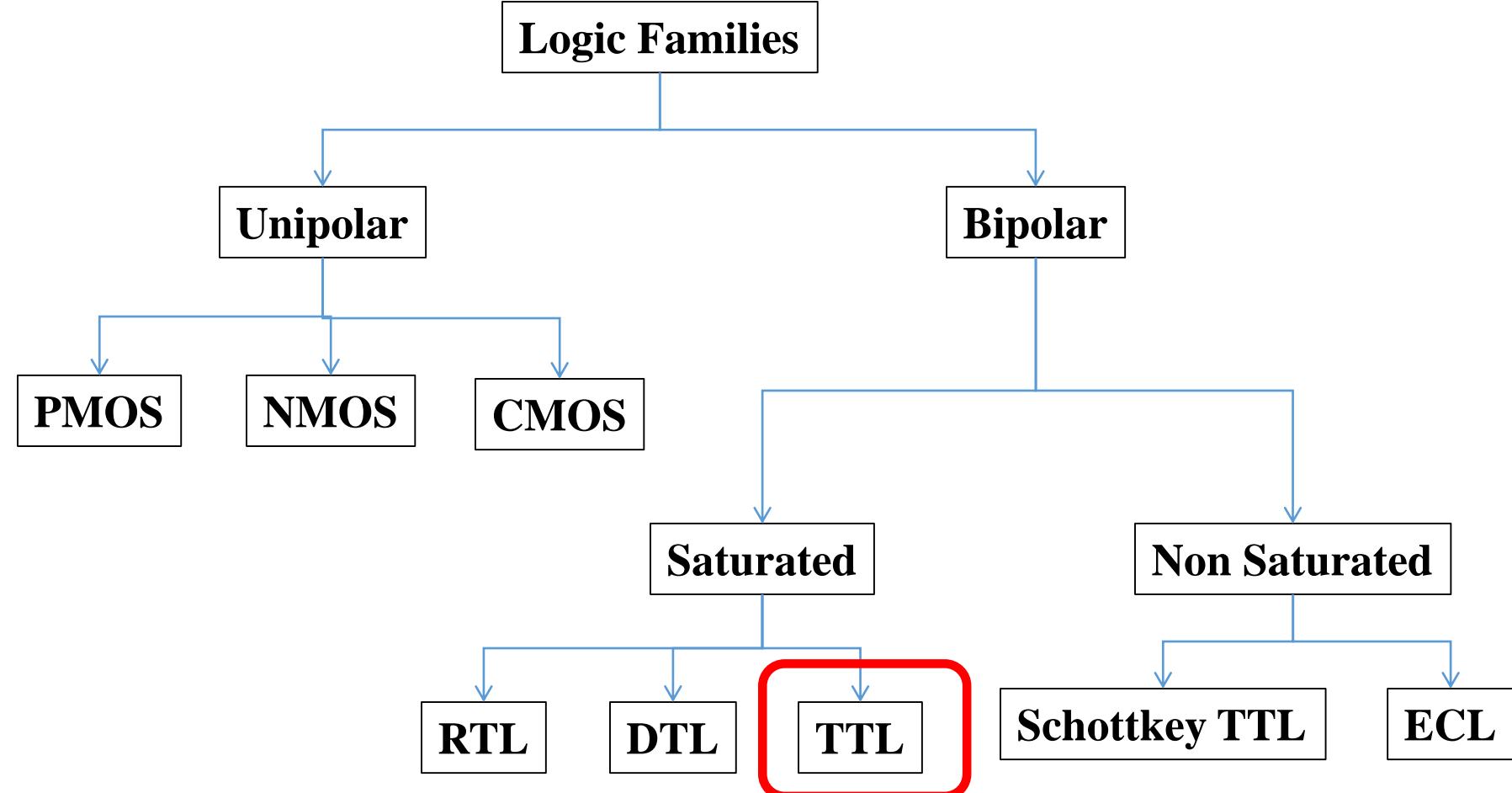
- **What is logic families?**

- It is a group of compatible ICs with the same logic levels & supply voltages for performing various logic functions have been fabricated using a specific circuit configuration

- **Significance of logic families**

- First and most important is simplicity in design
- It reduces the complexity in design
- Provide compatibility in digital circuit
- Easy to expand

Classification of Logic Families



Digital Logic Families

- Digital integrated circuits are classified not only by their complexity or logical operation, but also by the specific circuit technology to which they belong.
- A group of compatible ICs with same logic levels and supply voltage for performing various logic functions have been fabricated using a specific circuit configuration which is referred to as a *logic families*.
- The electronic components used in the construction of the basic circuit are usually used as the name of the technology. The following are the most popular:
 - RTL resistor-transistor logic (obsolete)
 - DTL diode-transistor logic (obsolete)
 - TTL transistor-transistor logic (widespread, standard)
 - ECL emitter-coupled logic (high speed)
 - MOS PMOS, NMOS metal-oxide semiconductor (high component density)
 - CMOS complementary metal-oxide semiconductor (low power consumption)

Example Logic Families

- General comparison of three commonly available logic families.

| Acronym | Full name | Advantages | Disadvantages |
|---------|---|--|--|
| CMOS | Complementary metal-oxide-semiconductor | Lowest power consumption. Most common logic family- used in all microcomputer chips today. | Easily damaged by static discharge and voltage spikes. |
| TTL | Transistor-transistor logic | Earliest developed. Most rugged – least susceptible to electrical damage. | Consumes more power than CMOS – not suitable for battery operated devices. |
| ECL | Emitter-coupled logic | Fastest available logic family | Consumes more power than CMOS. Requires extreme care in wiring. |

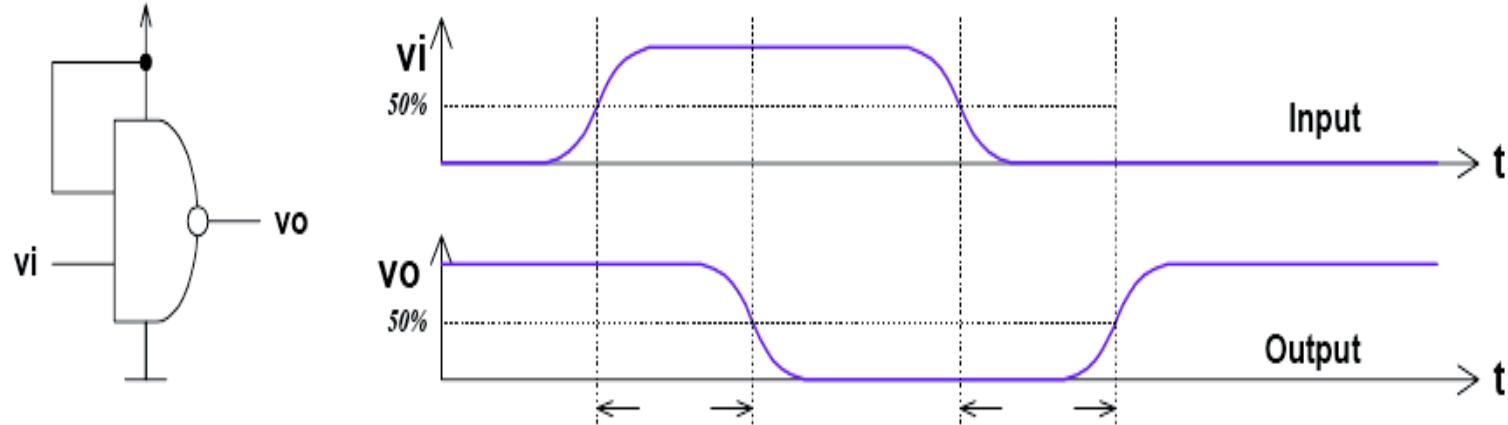
the most important to understand

Characteristics of Digital ICs

- Speed of Operation
- Power Dissipation
- Figure of Merit
- Fan Out
- Fan In
- Current and Voltage Parameters
- Noise Immunity
- Power Supply Requirements
- Operating Temperature

Characteristics of Digital ICs

- Speed of Operation/ Propagation Delay time



- TPD,HL–input-to-output propagation delay from HI to LO output
TPD,LH–input-to-output propagation delay from LO to HI output
- Propagation delay is average of TPD, HL and TPD, LH

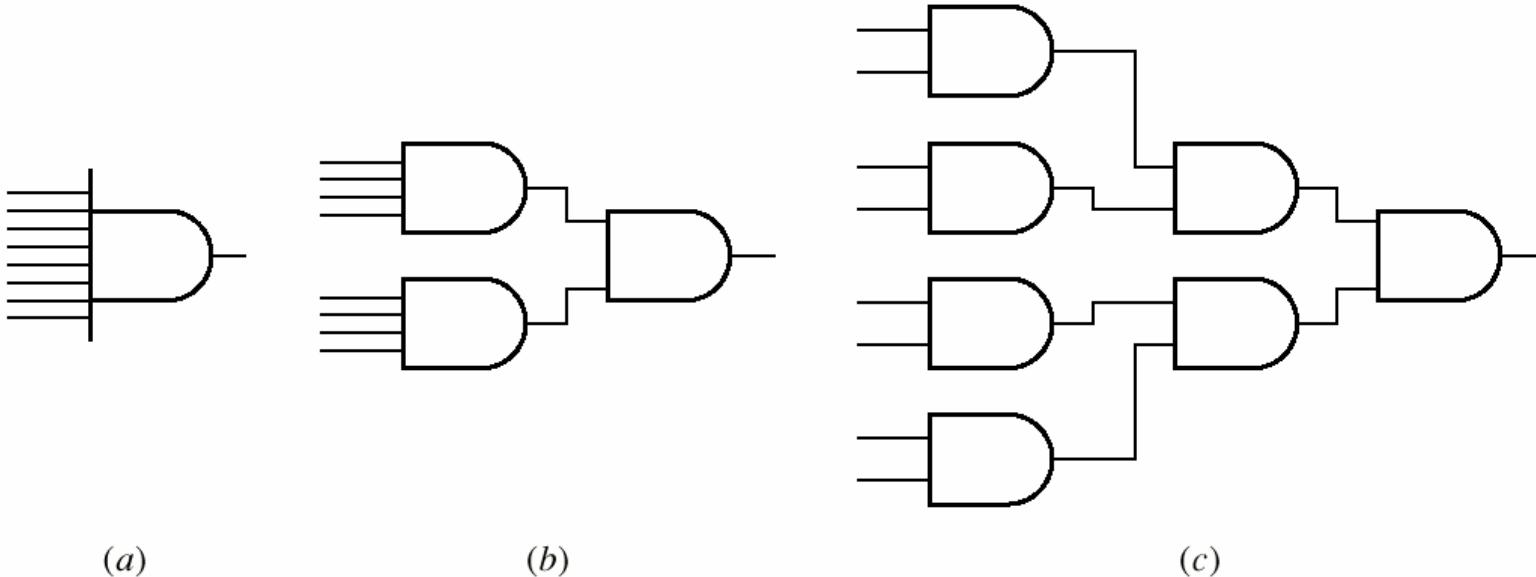
Characteristics of Digital ICs

Propagation Delay

Propagation Delay - Causes

Time to switch the transistor

Time to charge or discharge the load capacitance



Characteristics of Digital ICs

Power Dissipation

- It is wastage of power across the circuit components.
- It should be as low as possible.
- It is measured in milliwatts

Characteristics of Digital ICs

- **Figure of Merit**

Figure of Merit = Propagation Delay (ns) * Power Dissipation (mw) in pj

ns – nano seconds

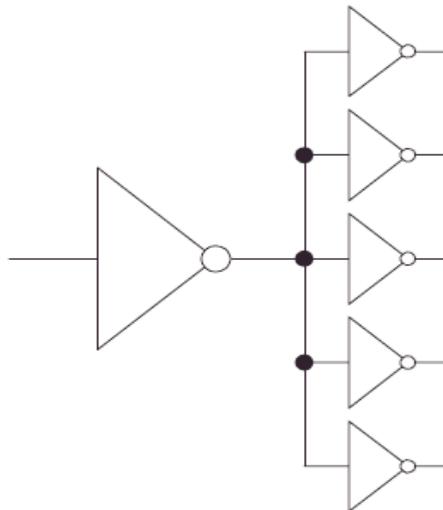
mw – milliwatts

pj – pico joules

- **Speed-power product: TPD × Pavg**

Characteristics of Digital ICs

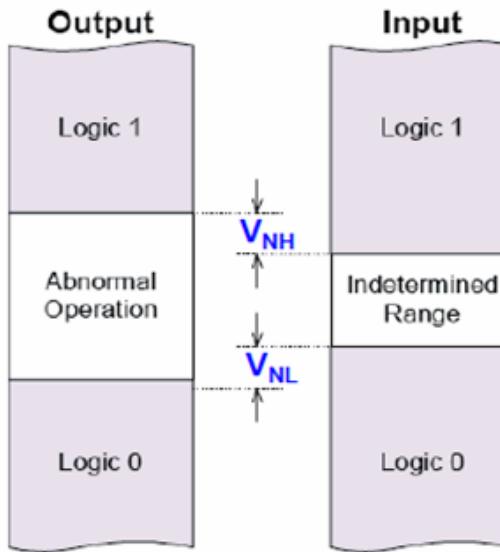
- **Fan Out**
- It is capability of a logic gate to drive maximum no. of similar gates.
- High fan out is desirable



- **Fan In**

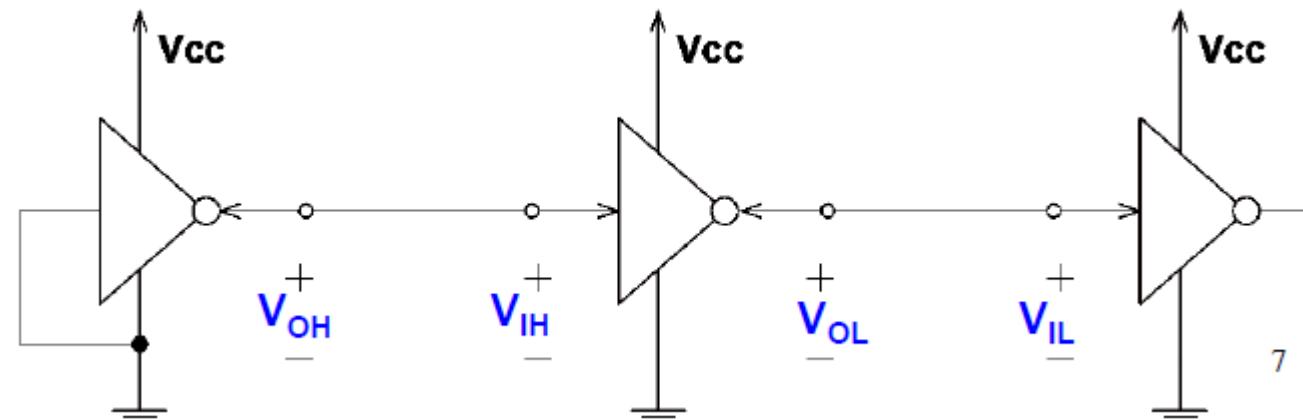
- It is total no. of inputs connected to any gate.
- Two input gate – Fan in =2
- Four input gate – Fan in =4

Characteristics of Digital ICs



Voltage Parameters

- **VOH(min)** – The minimum voltage level at an output in the logical “1” state under defined load conditions
- **VOL(max)** – The maximum voltage level at an output in the logical “0” state under defined load conditions
- **VIH(min)** – The minimum voltage required at an input to be recognized as “1” logical state
- **VIL(max)** – The maximum voltage required at an input that still will be recognized as “0” logical state



Floating-Point Representation

- The signed magnitude, 1's complement, and 2's complement representations are not useful in scientific or business applications that deal with real number values over a wide range.
- Floating-point representation solves this problem.
- Capable of representing very large and very small numbers without an increase in the number of bits

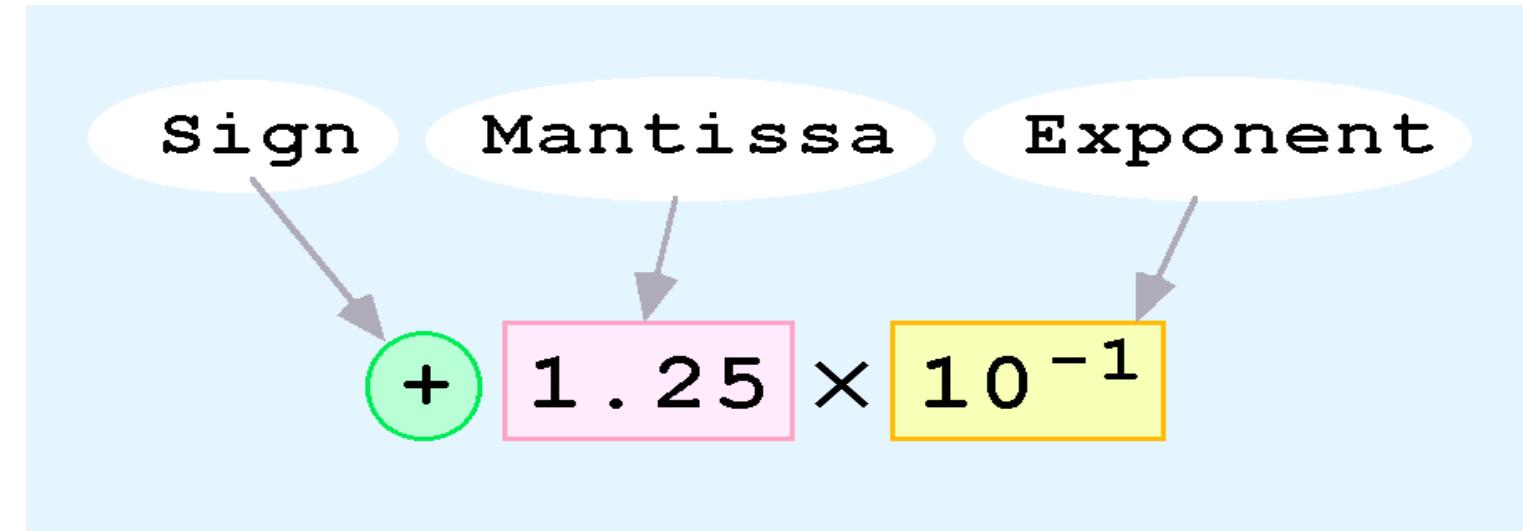
Signed Numbers

- **Floating-point numbers**
- **Three Parts**
 - **Sign**
 - **Mantissa** represents magnitude of number
 - **Exponent** represents number of places that binary point is to be moved
- **Three forms (IEEE 754)**
 - Single-precision (32 bits) **float**
 - Double-precision (64 bits) **double**
 - Extended-precision (80 bits) **long double**
 - Also have Quadruple and Quadruple extended!

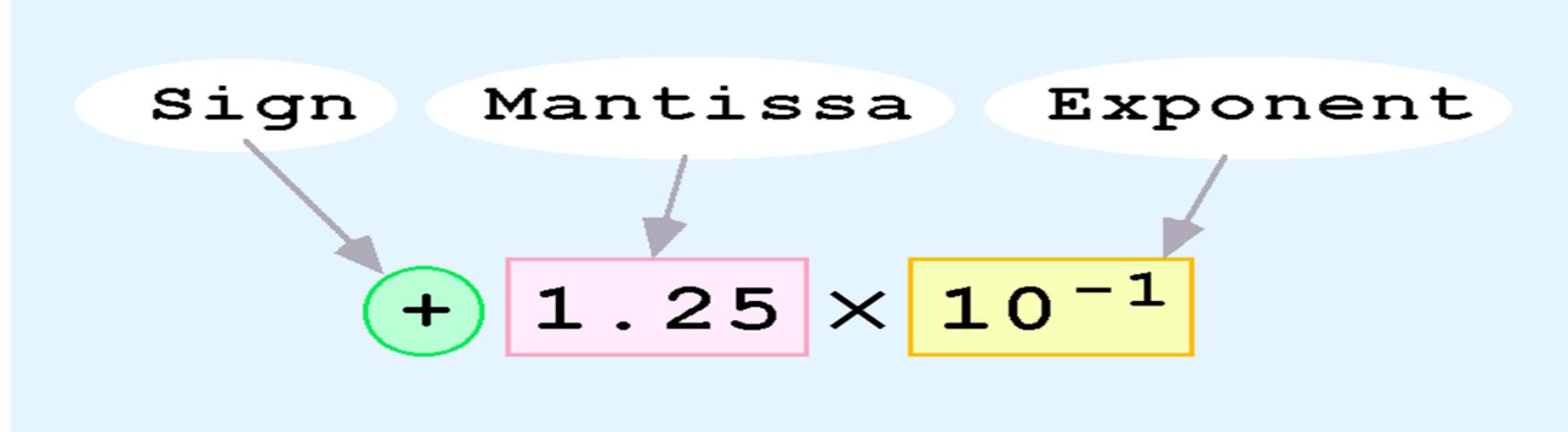
Floating-Point Representation

- Computers use a form of scientific notation for floating-point representation

Numbers written in scientific notation have three components:



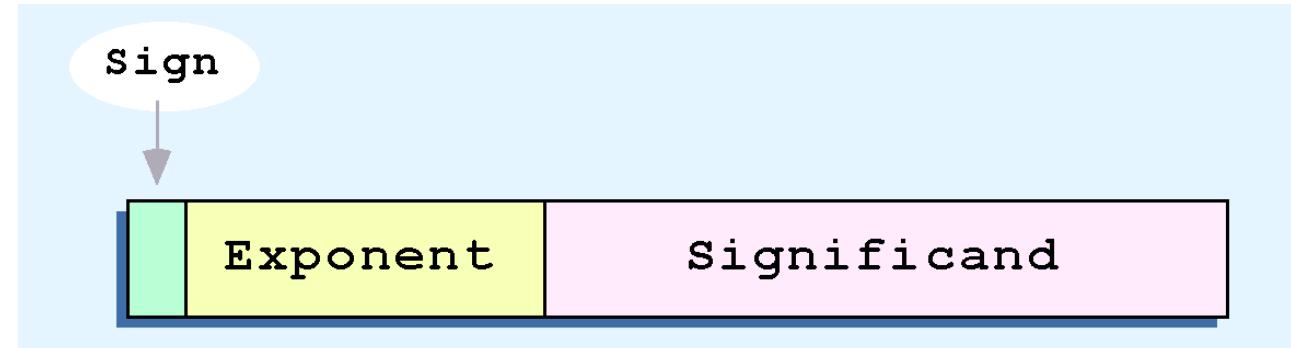
Floating-Point Representation



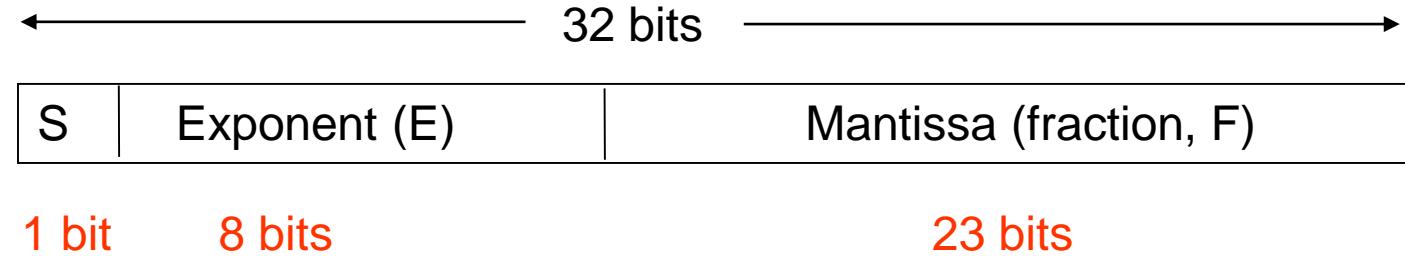
- The one-bit sign field is the sign of the stored value.
- The size of the exponent field, determines the range of values that can be represented.
- The size of the Mantissa/significand determines the precision of the representation.

Floating-Point Representation

- The IEEE-754 *single precision* floating point standard uses an 8-bit exponent and a 23-bit Mantissa/significand.
- The IEEE-754 *double precision* standard uses an 11-bit exponent and a 52-bit significand

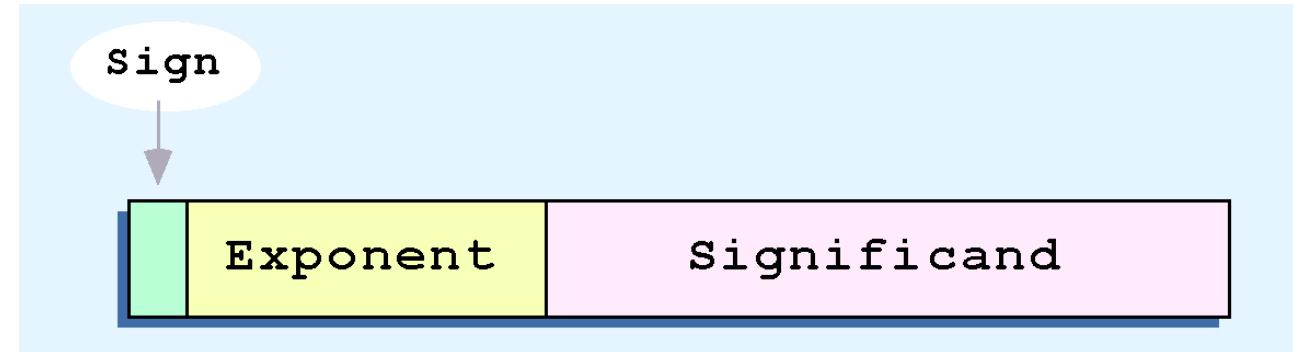


Single Precision



- IEEE 754 standard
 - Mantissa (F) has hidden bit so actually has 24 bits. Gives 7 significant figures.
 - 1st bit in mantissa is always a one
 - Exponent (E) is biased by 127 called Excess-127 Notation
 - Add 127 to exponent so easier to compare
 - Range of exponents is -126 to +128
 - Sign (S) bit tells whether number is negative or positive

Floating-Point Representation



- The significand of a floating-point number is always preceded by an implied binary point.
- Thus, the significand always contains a fractional binary value.
- The exponent indicates the power of 2 to which the significand is raised.

Single Precision

- Example: Convert 5777_{10} to Floating Point
- 1st, convert to binary using divide by 2 method

- $5777_{10} = 1011010010001_2$
- Positive number, so sign bit (S) equals 0.
- 2nd, count number of places to move binary point
 $1011010010001_2 = 1.011010010001 \times 2^{12}$
 Add 127 to 12 = $139_{10} = 10001011_2$
- Mantissa is fractional part, 011010010001
- Finally, put everything together

Fill in with trailing zeroes

| | | |
|---|----------|------------------------------|
| S | E | F |
| 0 | 10001011 | 0110100100010000000000000000 |

Ex 1: Find the IEEE FP representation of 40.15625

Step 1.

- Compute the binary equivalent of the whole part and the fractional part. (i.e. convert 40 and .15625 to their binary equivalents)
- $40.15625_{10} = 101000.00101_2$

Step 2.

- Normalize the number by moving the decimal point to the right of the leftmost one.
- $101000.00101 = 1.0100000101 \times 2^5$

Step 3.

- Convert the exponent to a biased exponent

$$127 + 5 = 132$$

- And convert biased exponent to 8-bit unsigned binary:

$$132_{10} = 10000100_2$$

Step 4.

- Store the results from steps 1-3:
- Sign Exponent Mantissa
- (from step 3) (from step 2)

► 0

10000100

010000010100000000000000

- The smallest magnitude of a normalized number in single precession is
 $\pm 0000\ 0001\ 000\ 0000\ 0000\ 0000\ 0000\ 0000$, whose value is 1.0×2^{-126} .
- The largest magnitude of a normalized number in single precession is
 $\pm 1111\ 1110\ 111\ 1111\ 1111\ 1111\ 1111\ 1111$, whose value is
 $1.99999988 \times 2^{127} \approx 3.403 \times 10^{38}$.

TEST

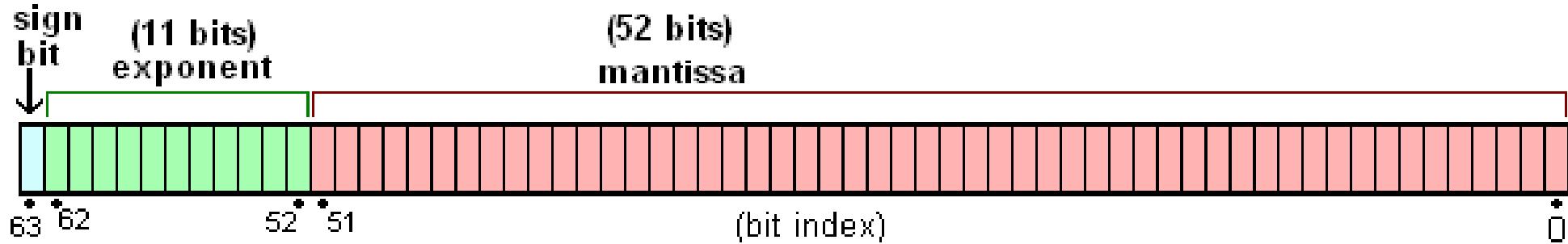
Ex 2: Find the IEEE FP representation of -24.75

1

10000011

1000110..0

Double Precision



- Exponent has 11 bits so uses Excess-1023 Notation
- Mantissa has 53 bits (one hidden)
- 53 bits gives 16 significant figures

IEEE standard to Decimal Floating Point Conversion.

Ex 1: Convert the following 32-bit binary number to its decimal floating point equivalent:

Sign

1

Exponent

01111101

Mantissa

010..0

IEEE standard to Decimal Floating Point Conversion..

Step 1: Extract the biased exponent and unbias it

$$\text{Biased exponent} = 01111101_2 = 125_{10}$$

$$\text{Unbiased Exponent: } 125 - 127 = -2$$

IEEE standard to Decimal Floating Point Conversion..

Step 2: Write Normalized number in the form:

1 . _____ $\times 2^{---$

Mantissa

Exponent

For our number:

-1.01 $\times 2^{-2}$

IEEE standard to Decimal Floating Point Conversion.

Step 3: Denormalize the binary number from step 2 (i.e. move the decimal and get rid of ($x 2^n$) part):

-0.0101₂ (negative exponent – move left)

Step 4: Convert binary number to the FP equivalent (i.e. Add all column values with 1s in them)

$$-0.0101_2 = - (0.25 + 0.0625)$$

$$= -0.3125_{10}$$

IEEE standard to Decimal Floating Point Conversion.

Ex 2: Convert the following 32 bit binary number to its decimal floating point equivalent:

| <u>Sign</u> | <u>Exponent</u> | <u>Mantissa</u> |
|-------------|-----------------|-----------------|
| 0 | 10000011 | 10011000..0 |