Digital Systems

What is a Digital system?

- Digital Input
- Digital output
- Process digital information

Examples

- Digital Camera, Digital Clock, Digital Money, Digital Media, Computer, Calculator, Internet
- Automated washing machines, air conditioners and various parts of our car

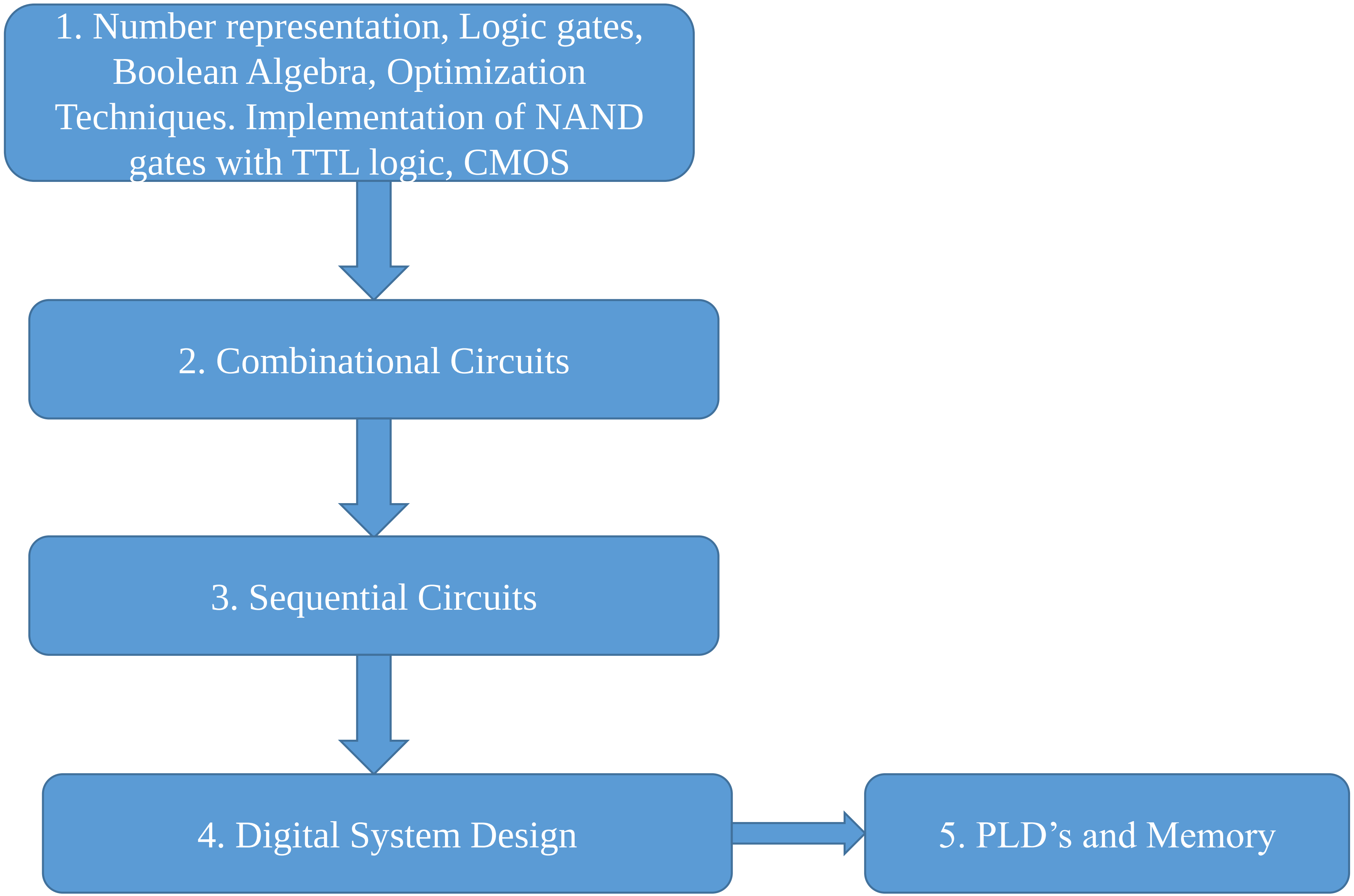
Why Digital?

- Easy to store, copy, compress, flexibility in processing
- Error correction, encryption and various transmission options
- Immunity to noise and interference
- Inexpensive building block and analysis of data

Technology Revolution

- Steam
- Electricity
- Very Large Scale Integration (VLSI)
- Industry 4.0 (ML, AI and IoT)

Digital Logic Design



Course Learning Outcomes

- Simplify Boolean Expression, understand basic logic gates and its implementation using CMOS and TTL logic.
- Identify the basic combinational logic functions like adders, subtractors, encoders/decoders , multiplexers/demultiplexers, code converters and magnitude comparators
- Understand the several types of flipflops and sequential circuits built with flipflops
- Design of advanced digital system with sequential logic circuits.
- Implement multiple combinational logic circuits using PLD’s. Explain CPLD and FPGA.

Textbook References

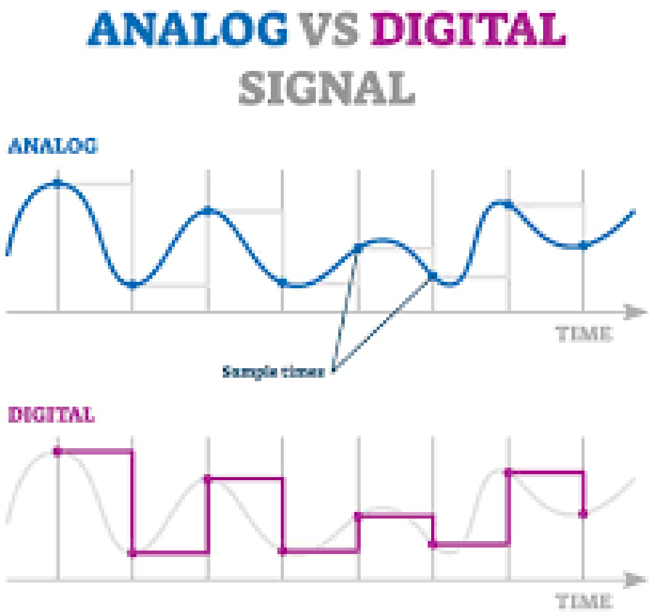
- **Digital Design by Morris Mano M and M D. Cilleti, 5th edition, Pearson publication**.
- Fundamentals of logi design By Charles H. Roth, Jr., Larry Kinney, 5th edition

Binary representation

Two stable states (True or False/ High or Low/ 1 or 0)

- 1 bit → 2 states
- 2 bits → 4 states (00, 01, 11, 10)
- 3 bits → 8 states (000, 001,010, 011, 100, 101, 110, 111)
- N bits → 2^N states

Digital signal from analog signal



Numbers with different bases

Numbers with Different Bases

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

Number-base conversion: Other formats to Decimal

General number representation

$$a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + \cdots + a_2 \cdot r^2 + a_1 \cdot r + a_0 + a_{-1} \cdot r^{-1} + a_{-2} \cdot r^{-2} + \cdots + a_{-m} \cdot r^{-m}$$

Decimal

$(7392)_{10} \rightarrow 7 \text{ x } 10^3 + 3 \text{ x } 10^2 + 9 \text{ x } 10^1 + 2 \text{ x } 10^0$

Binary

$(11010.11)_2 \rightarrow 1 \text{ x } 2^4 + 1 \text{ x } 2^3 + 0 \text{ x } 2^2 + 1 \text{ x } 2^1 + 0 \text{ x } 2^0 + 1 \text{ x } 2^{-1} + 1 \text{ x } 2^{-2} = (26.75)_{10}$

Octal

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

Hexadecimal

$(B65F)_{16} = 11 \times 16^3 + 6 \times 16^2 + 5 \times 16^1 + 15 \times 16^0 = (46{,}687)_{10}$

Decimal to binary

Integer	Remainder	
41		
20	1	
10	0	
5	0	
2	1	
1	0	
0	1	101001

$$(41.6875)_{10} = (101001.1011)_2$$

	Integer		Fraction	Coefficient
$0.6875 \times 2 =$	1	+	0.3750	$a_{-1} = 1$
$0.3750 \times 2 =$	0	+	0.7500	$a_{-2} = 0$
$0.7500 \times 2 =$	1	+	0.5000	$a_{-3} = 1$
$0.5000 \times 2 =$	1	+	0.0000	$a_{-4} = 1$

Decimal to octal

$$\begin{array}{r|l} 153 & \\ 19 & 1 \\ 2 & 3 \\ 0 & 2 = (231)_8 \end{array}$$

$$0.513 \times 8 = 4.104$$
$$0.104 \times 8 = 0.832$$
$$0.832 \times 8 = 6.656$$
$$0.656 \times 8 = 5.248$$
$$0.248 \times 8 = 1.984$$
$$0.984 \times 8 = 7.872$$

$$(153.513)_{10} = (231.406517)_8$$

$$(0.513)_{10} = (0.406517\ldots)_8$$

Binary to Octal and Hexadecimal numbers

Binary↔ Octal

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

$$\begin{array}{lccccccc}(673.124)_8 = & (110 & 111 & 011 & \cdot & 001 & 010 & 100)_2 \\ & 6 & 7 & 3 & & 1 & 2 & 4\end{array}$$

Binary↔ Hexadecimal

$$\begin{array}{cccccccl}(10 & 1100 & 0110 & 1011 & \cdot & 1111 & 0010)_2 & = (2C6B.F2)_{16} \\ 2 & C & 6 & B & & F & 2 & \end{array}$$

$$\begin{array}{lccccc}(306.D)_{16} = & (0011 & 0000 & 0110 & \cdot & 1101)_2 \\ & 3 & 0 & 6 & & D\end{array}$$

Binary Codes

Table 1.5
Four Different Binary Codes for the Decimal Digits

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

ASCII Codes

American Standard Code for Information Interchange (ASCII)

<table>
<tr><th></th><th colspan="8">$b_7b_6b_5$</th></tr>
<tr><th>$b_4b_3b_2b_1$</th><th>000</th><th>001</th><th>010</th><th>011</th><th>100</th><th>101</th><th>110</th><th>111</th></tr>
<tr><td>0000</td><td>NUL</td><td>DLE</td><td>SP</td><td>0</td><td>@</td><td>P</td><td>`</td><td>p</td></tr>
<tr><td>0001</td><td>SOH</td><td>DC1</td><td>!</td><td>1</td><td>A</td><td>Q</td><td>a</td><td>q</td></tr>
<tr><td>0010</td><td>STX</td><td>DC2</td><td>“</td><td>2</td><td>B</td><td>R</td><td>b</td><td>r</td></tr>
<tr><td>0011</td><td>ETX</td><td>DC3</td><td>#</td><td>3</td><td>C</td><td>S</td><td>c</td><td>s</td></tr>
<tr><td>0100</td><td>EOT</td><td>DC4</td><td>$</td><td>4</td><td>D</td><td>T</td><td>d</td><td>t</td></tr>
<tr><td>0101</td><td>ENQ</td><td>NAK</td><td>%</td><td>5</td><td>E</td><td>U</td><td>e</td><td>u</td></tr>
<tr><td>0110</td><td>ACK</td><td>SYN</td><td>&</td><td>6</td><td>F</td><td>V</td><td>f</td><td>v</td></tr>
<tr><td>0111</td><td>BEL</td><td>ETB</td><td>‘</td><td>7</td><td>G</td><td>W</td><td>g</td><td>w</td></tr>
<tr><td>1000</td><td>BS</td><td>CAN</td><td>(</td><td>8</td><td>H</td><td>X</td><td>h</td><td>x</td></tr>
<tr><td>1001</td><td>HT</td><td>EM</td><td>)</td><td>9</td><td>I</td><td>Y</td><td>i</td><td>y</td></tr>
<tr><td>1010</td><td>LF</td><td>SUB</td><td>*</td><td>:</td><td>J</td><td>Z</td><td>j</td><td>z</td></tr>
<tr><td>1011</td><td>VT</td><td>ESC</td><td>+</td><td>;</td><td>K</td><td>[</td><td>k</td><td>{</td></tr>
<tr><td>1100</td><td>FF</td><td>FS</td><td>,</td><td><</td><td>L</td><td>\</td><td>l</td><td>|</td></tr>
<tr><td>1101</td><td>CR</td><td>GS</td><td>–</td><td>=</td><td>M</td><td>]</td><td>m</td><td>}</td></tr>
<tr><td>1110</td><td>SO</td><td>RS</td><td>.</td><td>></td><td>N</td><td>∧</td><td>n</td><td>~</td></tr>
<tr><td>1111</td><td>SI</td><td>US</td><td>/</td><td>?</td><td>O</td><td>–</td><td>o</td><td>DEL</td></tr>
</table>

Binary Arithmetic

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = 0$ and carry 1 to the next column

$0 - 0 = 0$

$0 - 1 = 1$ and borrow 1 from the next column

$1 - 0 = 1$

$1 - 1 = 0$

Problems:

1. Add $(1101)_2 + (1001)_2$

2. Subtract $(111001)_2 - (1011)_2$

3. Multiply $(1101)_2$ and $(101)_2$

Binary-Coded Decimal (BCD) Arithmetic simplification

Table 1.4
Binary-Coded Decimal (BCD)

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

- each group of 4 bits representing one decimal digit.
- the binary combinations 1010 through 1111 are not used and have no meaning in BCD

Example:

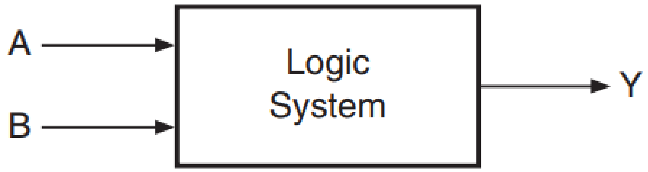
$(185)_{10} = (0001\ 1000\ 0101)_{BCD} = (10111001)_2$

Decimal	BCD
10	0001 0000
11	0001 0001
99	1001 1001
100	0001 0000 0000
101	0001 0000 0001
487	0100 1000 0111

Logic Gates

- Truth table

- all possible combinations of input binary variables and the corresponding outputs of a logic system.
- The logic system output can be found from the logic expression, often referred to as the Boolean expression, that relates the output with the inputs of that very logic system.



(a)

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

(b)

Figure 4.1 Two-input logic system.

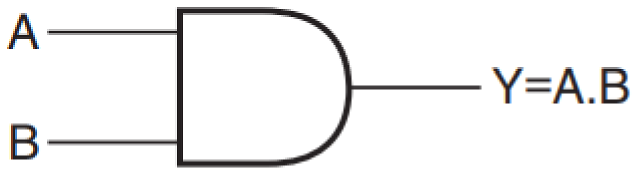
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Figure 4.2 Truth table of a three-input logic system

For n input variables, 2^n input possible combinations

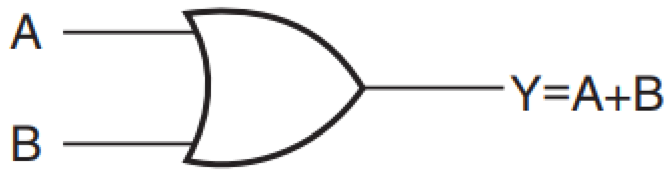
Basic Logic gates

- **Logic gates**
  - Basic building block of any digital system
  - Implemented with electronic circuits
  - Three basic logic gates (**AND, OR** and **NOT**)
  - Used to construct any logic circuit for given logic expression



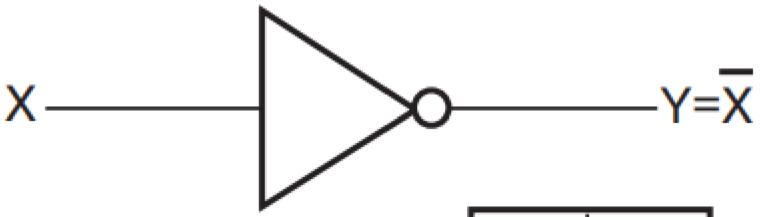
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Figure 4.7 Two-input AND gate.



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Figure 4.3 Two-input OR gate.

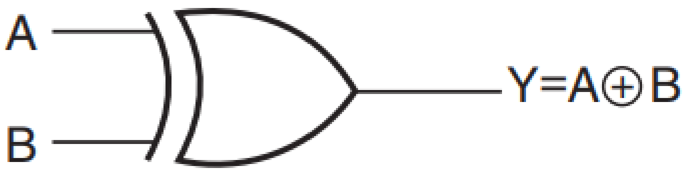


X	Y
0	1
1	0

Figure 4.10 (a) Circuit symbol of a NOT circuit and (b) the truth table of a NOT circuit.

EXCLUSIVE-OR (EX-OR) and EXCLUSIVE-NOR (EX-NOR) Logic gates

- **EX-OR**
  - Output is 1 when the inputs are unlike
  - Output is 0 when the inputs are like



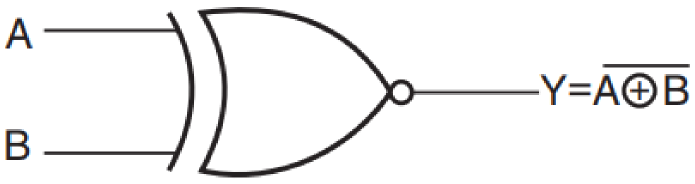
(a)

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

(b)

$$Y = (A \oplus B) = \overline{A}B + A\overline{B}$$

- **EX-NOR**
  - Output is 0 when the inputs are unlike
  - Output is 1 when the inputs are like



(a)

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

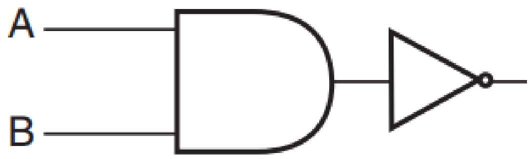
(b)

$$Y = (\overline{A \oplus B}) = (A.B + \overline{A}.\overline{B})$$

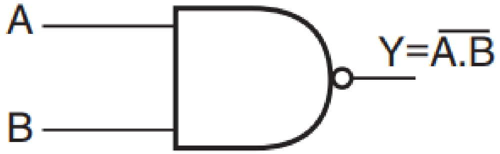
Universal Gates (NAND and NOR)

NAND (AND NOT Gate)

$$Y = \overline{A.B}$$



(a)

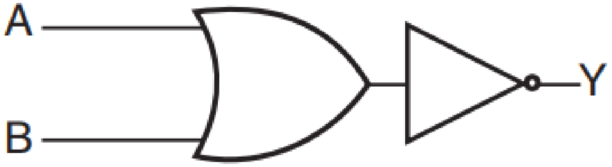


(b)

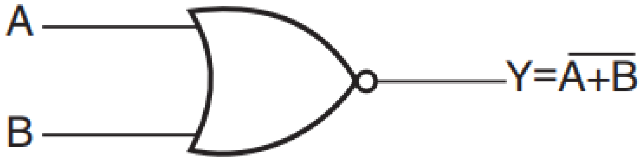
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR (OR NOT Gate)

$$Y = \overline{(A+B)}$$



(a)



(b)

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Boolean Algebra

- Set of symbols and set of rules to manipulate these symbols
- The letter symbols can take on either of two values: 0 and 1.
- Value assigned to a variable has logical significance
- '.' represents AND operation and '+' represents OR operation
- Variable are different symbols in a Boolean expression
- Variable can occur as such and as well as in complement form in an expression
- Each occurrence of a variable or its complement in an expression is called literal

$$\overline{A}+A.B+A.\overline{C}+\overline{A}.B.C$$

$$(\overline{P}+Q).(R+\overline{S}).(P+\overline{Q}+R)$$

Boolean Algebra - Postulates

S.No.	Name of the Postulates	Postulate Equation
1	Law of Identity	$A + 0 = 0 + A = A$ $A . 1 = 1 . A = A$
2	Commutative Law	$(A + B) = (B + A)$ $(A . B) = (B . A)$
3	Distributive Law	$A . (B + C) = (A . B) + (A . C)$ $A + (B . C) = (A + B) . (A + C)$
4	Associative Law	$A + (B + C) = (A + B) + C$ $(A . B) . C = A . (B . C)$
5	Complement Law	$A + A' = 1$ $A . A' = 0$

Boolean Algebra - Theorems

DUALITY

- Dual of any Boolean function can be obtained by
    - Interchanging AND with OR and OR with AND
    - 0’s with 1’s and 1’s with 0’s
- For every Boolean function, there will be a corresponding Dual function.

Group1	Group2
x + 0 = x	x.1 = x
x + 1 = 1	x.0 = 0
x + x = x	x.x = x
x + x’ = 1	x.x’ = 0
x + y = y + x	x.y = y.x
x + $y + z$ = $x + y$ + z	x. $y . z$ = $x . y$ .z
x. $y + z$ = x.y + x.z	x + $y . z$ = $x + y$. $x + z$

Boolean Algebra - Theorems

COMPLEMENT

- Complement of any Boolean function can be obtained by
  - Complementing each literal
  - Interchanging AND with OR and OR with AND
  - 0's with 1's and 1's with 0's
- For every Boolean function F, there will be a complement function $\bar{F}$.

Boolean Algebra - Theorems

COMPLEMENT

- Complement of any Boolean function can be obtained by
  - Complementing each literal
  - Interchanging AND with OR and OR with AND
  - 0's with 1's and 1's with 0's
- For every Boolean function F, there will be a complement function $\bar{F}$.
- Example: F = x + y + z then $\bar{F} = \overline{(X + Y + Z)} = \bar{X}.\bar{Y}.\bar{Z}$

DEMORGAN's THEOREM

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

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

Boolean Algebra - Theorems

	Postulate / Theorem	Dual
1. Identity Element	$x + 0 = x$	$x . 1 = x$
2. Complementation	$x + x' = 1$	$x . x' = 0$
3. Idem potency	$x + x = x$	$x . x = x$
4. Null Law	$x + 1 = 1$	$x . 0 = 0$
5. Involution	$(x')' = x$	-
6. Commutative	$x + y = y + x$	$x y = y x$
7. Associative	$x + (y + z) = (x + y) + z$	$x(yz) = (x y) z$
8. Distributive	$x + yz = (x + y)(x + z)$	$x(y + z) = xy + xz$
9. De Morgan	$(x + y)' = x' y'$	$(xy)' = x' + y'$
10. Absorption	$x + xy = x$	$x(x + y) = x$
11. Simplification	$x + x'y = x + y$	$x(x' + y) = xy$
12. Consensus	$xy + x'z + yz = xy + x'z$	$(x+y)(x'+z)(y+z) = (x+y)(x'+z)$

Boolean Algebra Simplification

PRACTICE

1. Simplify F = A(A'+C)(A'B+C)(A'BC +C')
2. Simplify Y = (A+B)(A' (B' + C')) ' + A' (B+C)
3. Find the Dual of AB ' + BC ' + CD '
4. Find the Complement [(AB ' + C ')D+E ']F

Sum of products (∑) Boolean Expression - Minterms

SOP

1. Contains the sum of different terms with each term being a single literal or product of more than one literal

$$Y = \overline{A}\,.\overline{B}.\overline{C} + \overline{A}.B.C + A.B.\overline{C} + A.\overline{B}.C$$

Table 6.5 truth table of boolean expression of equation 6.33.

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Product of sum (∏) Boolean Expression - Minterms

POS Contains the product of different terms with each term being a single literal or a sum of more than one literal

$$Y = (A+B+\overline{C}).(A+\overline{B}+C).(\overline{A}+B+C).(\overline{A}+\overline{B}+\overline{C})$$

Table 6.5 truth table of boolean expression of equation 6.33.

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Canonical form

MINTERMS (AND Terms)

- Each term contains all the variables either in true or complement form
- An expanded form of Boolean expression, where each term contains all Boolean variables in their true or complemented form, is also known as the canonical form of the expression.

$$A.\overline{B} + B.\overline{C} + A.B.\overline{C} + \overline{A}.C$$

It is a three-variable expression. Expanded versions of different minterms can be written as follows:

- $A.\overline{B} = A.\overline{B}.(C + \overline{C}) = A.\overline{B}.C + A.\overline{B}.\overline{C}.$
- $B.\overline{C} = B.\overline{C}.(A + \overline{A}) = B.\overline{C}.A + B.\overline{C}.\overline{A}.$
- $A.B.\overline{C}$ is a complete term and has no missing variable.
- $\overline{A}.C = \overline{A}.C.(B + \overline{B}) = \overline{A}.C.B + \overline{A}.C.\overline{B}.$

The expanded sum-of-products expression is therefore given by

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

Canonical form

- **MAXTERMS (OR Terms)**
- An expanded form of Boolean expression, where each term contains all Boolean variables in their true or complemented form, is also known as the canonical form of the expression.

$$(\overline{A}+B).(\overline{A}+B+\overline{C}+\overline{D})$$

It is four-variable expression with A, B, C and D being the four variables. $\overline{A}+B$ in this case expands to $(\overline{A}+B+C+D).(\overline{A}+B+C+\overline{D}).(\overline{A}+B+\overline{C}+D).(\overline{A}+B+\overline{C}+\overline{D})$.

The expanded product-of-sums expression is therefore given by

$$(\overline{A}+B+C+D).(\overline{A}+B+C+\overline{D}).(\overline{A}+B+\overline{C}+D).(\overline{A}+B+\overline{C}+\overline{D}).(\overline{A}+B+\overline{C}+\overline{D})$$

$$=(\overline{A}+B+C+D).(\overline{A}+B+C+\overline{D}).(\overline{A}+B+\overline{C}+D).(\overline{A}+B+\overline{C}+\overline{D})$$

Minterms and Maxterms

Variable			Minterm		Maxterm	
x	y	z	Term	Designation	Term	Designation
0	0	0	x'y'z'	m_0	x+y+z	M_0
0	0	1	x'y'z	m_1	x+y+z'	M_1
0	1	0	x'yz'	m_2	x+y'+z	M_2
0	1	1	x'yz	m_3	x+y'+z'	M_3
1	0	0	xy'z'	m_4	x'+y+z	M_4
1	0	1	xy'z	m_5	x'+y+z'	M_5
1	1	0	xyz'	m_6	x'+y'+z	M_6
1	1	1	xyz	m_7	x'+y'+z'	M_7

Example 6.8

For a Boolean function $f(A,B)=\sum 0,2$, *prove that* $f(A,B)=\prod 1,3$ *and* $f'(A,B)=\sum 1,3=\prod 0,2$.

Solution

- $f(A,B)=\sum 0,2=\overline{A}.\overline{B}+A.\overline{B}=\overline{B}.(A+\overline{A})=\overline{B}$.
- Now, $\prod 1,3=(A+\overline{B}).(\overline{A}+\overline{B})=A.\overline{A}+A.\overline{B}+\overline{B}.\overline{A}+\overline{B}.\overline{B}=A.\overline{B}+\overline{A}.\overline{B}+\overline{B}=\overline{B}$.
- Now, $\sum 1,3=\overline{A}.B+A.B=B.(\overline{A}+A)=B$.
  and $\prod 0,2=(A+B).(\overline{A}+B)=A.\overline{A}+A.B+B.\overline{A}+B.B=A.B+\overline{A}.B+B=B$.
- Therefore, $\sum 1,3=\prod 0,2$.
- Also, $f(A,B)=\overline{B}$.
- Therefore, $f'(A,B)=B$ or $f'(A,B)=\sum 1,3=\prod 0,2$.

Karnaugh Map method

Draw K map

- An n-variable K Map has 2^nsquares with each input allotted to a square

MINTERMS or STANDARD PRODUCT:

- 1 is placed for all all those squares for which the output is ‘1’
- ‘0’ is placed in all those squares for which the output is ‘0’. 0s are omitted for simplicity
- An ‘X’ is placed in squares corresponding to ‘don’t care’ conditions.

MAXTERMS or STANDURD SUM:

- 1 is placed for all all those squares for which the output is ‘0’
- ‘0’ is placed in all those squares for which the output is ‘1’. 0s are omitted for simplicity
- An ‘X’ is placed in squares corresponding to ‘don’t care’ conditions.

Karnaugh Map styles

- only one bit changes in value from one adjacent column to the next

	$\bar{B}$	B
$\bar{A}$		
A		

(a)

	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$				
A				

(a)

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				
$\bar{A}B$				
AB				
$A\bar{B}$				

$x \backslash y$	0	1 (y)
0	m_0 $x'y'$	m_1 $x'y$
(x) 1	m_2 xy'	m_3 xy

(b)

$x \backslash yz$	00	01 (z)	11 (y, z)	10 (y)
0	m_0 $x'y'z'$	m_1 $x'y'z$	m_3 $x'yz$	m_2 $x'yz'$
(x) 1	m_4 $xy'z'$	m_5 $xy'z$	m_7 xyz	m_6 xyz'

(b)

$wx \backslash yz$	00	01 (z)	11 (y, z)	10 (y)
00	m_0 $w'x'y'z'$	m_1 $w'x'y'z$	m_3 $w'x'yz$	m_2 $w'x'yz'$
01 (x)	m_4 $w'xy'z'$	m_5 $w'xy'z$	m_7 $w'xyz$	m_6 $w'xyz'$
11 (w, x)	m_{12} $wxy'z'$	m_{13} $wxy'z$	m_{15} $wxyz$	m_{14} $wxyz'$
10 (w)	m_8 $wx'y'z'$	m_9 $wx'y'z$	m_{11} $wx'yz$	m_{10} $wx'yz'$

(b)

Karnaugh Map Reduction rules

Form groups of 1's

- An n-variable K Map has 2^n squares with each input allotted to a square

1. Each square containing a '1' must be considered at least once, although it can be considered as often as desired.
2. The objective should be to account for all the marked squares in the minimum number of groups.
3. The number of squares in a group must always be a power of 2, i.e. groups can have 1, 2, 4 8, 16, squares.
4. Each group should be as large as possible, which means that a square should not be accounted for by itself if it can be accounted for by a group of two squares; a group of two squares should not be made if the involved squares can be included in a group of four squares and so on.
5. 'Don't care' entries can be used in accounting for all of 1-squares to make optimum groups. They are marked 'X' in the corresponding squares. It is, however, not necessary to account for all 'don't care' entries. Only such entries that can be used to advantage should be used.

Karnaugh Map Example

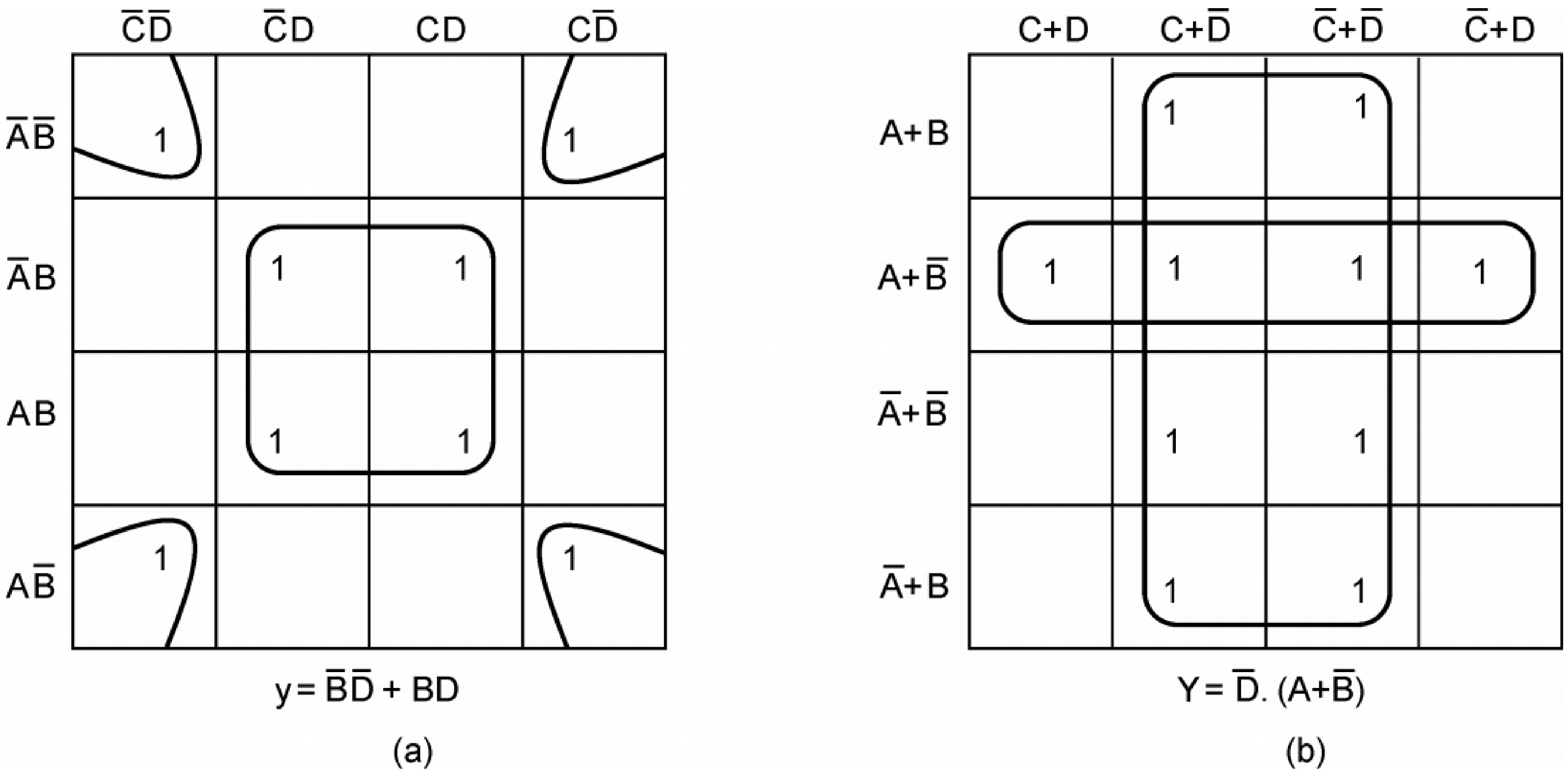


Figure 6.13 Group formation in minterm and maxterm Karnaugh maps.

Problem

PRACTICE

1. **Express the Boolean function F = A+B'C+AD as sum of minterms**
2. **Express the Boolean function F = x'y + xz product of maxterms**
3. **Simplify the given function using K-Map**
    1. **$F(x,y) = \sum(1,2,3)$**
    2. **$F(x,y,z) = \sum(1,2,3,5,7)$**
    3. **$F(w,x,y,z) = \sum(0,1,3,8,9,10,11,12,13,14,15)$**

K-Map Problems

1. $F(A, B, C, D) = \Sigma m(0, 1, 3, 5, 7, 8, 9, 11, 13, 15)$
2. $F(A, B, C, D) = \Sigma m(1, 3, 4, 6, 8, 9, 11, 13, 15) + \Sigma d(0, 2, 14)$
3. $F(A, B, C) = \Sigma m(0, 1, 6, 7) + \Sigma d(3, 5)$
4. $F(A, B, C) = \Sigma m(1, 2, 5, 7) + \Sigma d(0, 4, 6)$
5. $F(A, B, C, D) = \Sigma m(0, 2, 8, 10, 14) + \Sigma d(5, 15)$