Course Aim

By the end of the course, students will acquire knowledge of:

number systems and coding, Boolean algebra: basic theorems, Karnaugh map, logic gates. combinational logic circuits: adder, subtractor, multiplexers, demultiplexers, decoders, encoders. Sequential logic circuits: SR latch, clocked SR Flip Flop, D-Flip Flop. Registers: Parallel-in Parallel-out, Serial-in Serial-out, Counters.

Contents

- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Encoders and Decoders
- Multiplexers and Demultiplexers
- Sequential Logic Circuits
- Registers
- Counters

Number Systems and Coding

Number system is a basis for counting various items.

The word "number" all of us immediately think that is decimal number system. With its 10 digits: 0,1,2,3,4,5,6,7,8,9 Base (b = 10).

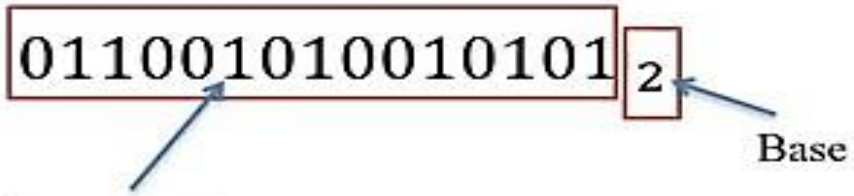
Modern computer operate with binary number system

With its 2 digits: 0,1 Here, the base is 2



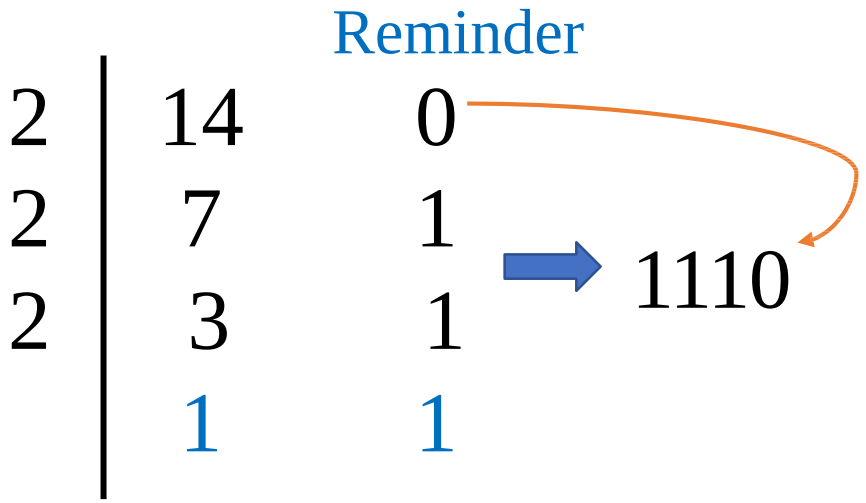
binary signal

Example:



Number Systems and Coding

Convert $(14)_{10}$ to binary number



$$(14)_{10} = (1110)_2$$

Number Systems and Coding

People don't like deal with binary numbers, due to the conversion takes more digits. In this example, takes four digits.

Therefore, they used octal number system & hexadecimal number system & Binary Coded Decimal (BCD) system to compress long strings of binary numbers. So, using these different numbering systems is easer to understand/interpret,

Octal number system with its digits: 0,1,2,3,4,5,6,7 Here, the base is 8

Hexadecimal number system with its digits: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F Here, the base is 16

Note: A,B,C,D,E,F represent 10, 11, 12, 13, 14, 15

Number Systems and Coding

In this chapter, we will learn how to convert from decimal to binary or octal or hexadecimal or BCD and vise versa.

Convert $(150)_{10}$ to octal number

		Reminder
8	150	6
8	18	2
	2	2

 226

$$(150)_{10} = (226)_8$$

Number Systems and Coding

Convert $(97)_{10}$ to hexadecimal number

		Reminder
16	97	1
	6	6

→ 61

$$(97)_{10} = (61)_{16}$$

Convert $(314)_{10}$ to hexadecimal number

			Reminder
16	314	"10"	A
16	19		3
	1		1

→ 13A

$$(314)_{10} = (13A)_{16}$$

Number Systems and Coding

Convert $(13.125)_{10}$ to binary number

		Reminder
2	13	1
2	6	0
2	3	1
	1	1

→ 1101

		Multiplication
0.25	.125	× 2
0.5	.25	× 2
1	.5	× 2

→ .001

Stop at integer number

$$(13.125)_{10} = (1101.001)_2$$

Number Systems and Coding

Convert $(145.45)_{10}$ to octal number

		Reminder
8	145	1
8	18	2
	2	2

→ 221

Multiplication

3.6	.45	× 8
4.8	.6	× 8
6.4	.8	× 8
3.2	.4	× 8
1.6	.2	× 8

→ .34631

Stop at same fraction number

$$(145.45)_{10} = (221.34631)_{8}$$

Number Systems and Coding

Convert $(394.36)_{10}$ to hexadecimal number

Reminder

16 | 394	"10"	A
16 | 24		8
1		1

➡ 18A

Multiplication

5.76	.36	× 16
12.16	.76	× 16
2.56	.16	× 16
8.96	.56	× 16
15.36	.96	× 16
5.76	.36	× 16

➡ .5C28

Stop at same fraction number
"after decreasing due to complicated"

$$(394.36)_{10} = (18A.5C28)_{16}$$

Number Systems and Coding

Convert $(1101001)_2$ to decimal number

2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	1	0	0	1

$$= 1 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 0 \times 2^4 + 1 \times 2^5 + 1 \times 2^6$$

$$= 1 + 0 + 0 + 8 + 0 + 32 + 64$$

$$= 105$$

$$\mathbf{(1101001)_2 = (105)_{10}}$$

Number Systems and Coding

Convert $(1110.101)_2$ to decimal number

2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}
1	1	1	0	.	1	0	1

$$= 1 \times 2^{-3} + 0 \times 2^{-2} + 1 \times 2^{-1} + 0 \times 2^{0} + 1 \times 2^{1} + 1 \times 2^{2} + 1 \times 2^{3}$$

$$= 0.125 + 0 + 0.5 + 0 + 2 + 4 + 8$$

$$= 14.625$$

$$\mathbf{(1110.101)_2 = (14.625)_{10}}$$

Number Systems and Coding

Convert $(2275)_8$ to decimal number

8^3	8^2	8^1	8^0
2	2	7	5

$= 5 \times 8^0 + 7 \times 8^1 + 2 \times 8^2 + 2 \times 8^3$

$= 5 + 56 + 128 + 1024$

$= 1213$

$$\mathbf{(2275)_8 = (1213)_{10}}$$

Number Systems and Coding

Convert $(125.4)_8$ to decimal number

8^2 8^1 8^0 8^{-1}

1 2 5 . 4

$= 4 \times 8^{-1} + 5 \times 8^0 + 2 \times 8^1 + 1 \times 8^2$

$= 0.5 + 5 + 16 + 64$

$= 85.5$

$$(125.4)_{10} = (85.5)_{10}$$

Number Systems and Coding

Convert $(F9B)_{16}$ to decimal number

16^2 16^1 16^0

F 9 B

$= B \times 16^0 + 9 \times 16^1 + F \times 16^2$

$= 11 \times 16^0 + 9 \times 16^1 + 15 \times 16^2$

$= 11 + 144 + 3840$

$= 3995$

$$\mathbf{(F9B)_{16} = (3995)_{10}}$$

Number Systems and Coding

Convert $(A15.C)_{16}$ to decimal number

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

A 1 5 . C

$= C \times 16^{-1} + 5 \times 16^0 + 1 \times 16^1 + A \times 16^2$

$= 12 \times 16^{-1} + 5 \times 16^0 + 1 \times 16^1 + 10 \times 16^2$

$= 0.75 + 5 + 16 + 2560$

$= 2581.75$

$$\mathbf{(A15.C)_{16} = (2581.75)_{10}}$$

Number Systems and Coding

Convert $(A39)_{16}$ to binary number

A = 10

		Reminder
2	10	0
2	5	1
2	2	0
	1	1

→ 1010

		Reminder
2	3	1
	1	1

→ 0011

		Reminder
2	9	1
2	4	0
2	2	0
	1	1

→ 1001

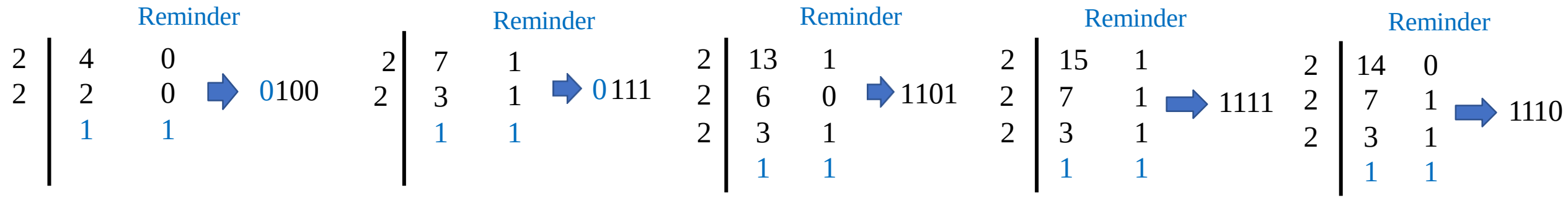
Put the numbers beside other

101000111001

$(A39)_{16} = (101000111001)_2$

Number Systems and Coding

Convert $(47D.FE)_{16}$ to binary number



101000111001

$$(47D.FE)_{16} = (010001111101.11111110)_2$$

Number Systems and Coding

Convert $(101010000101)_2$ to hexadecimal number

2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0
1	0	1	0	1	0	0	0	0	1	0	1

$= 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3$

$= 10$

$= 0 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3$

$= 8$

$= 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3$

$= 5$

$$(101010000101)_2 = (A85)_{16}$$

Number Systems and Coding

Convert $(10010.011)_2$ to hexadecimal number

2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0
0	0	0	1	0	0	1	0	0	1	1	0

$= 1 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 0 \times 2^3$

$= 1$

$= 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 0 \times 2^3$

$= 2$

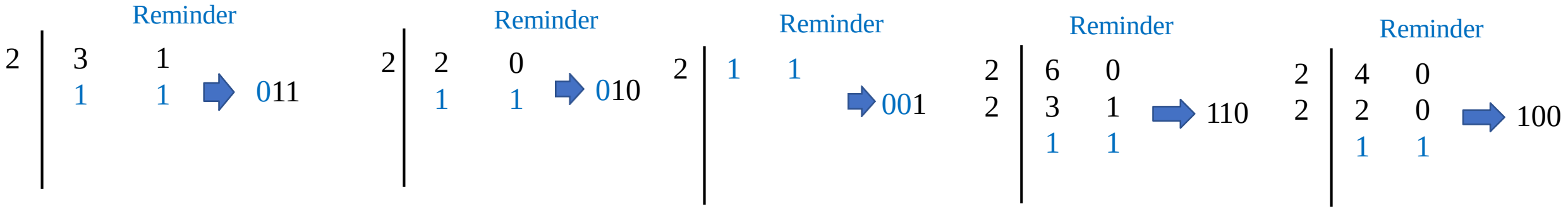
$= 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3$

$= 6$

$$(10010.011)_2 = (12.6)_{16}$$

Number Systems and Coding

Convert $(321.64)_8$ to binary number



101000111001

$$(321.64)_8 = (011010001.110100)_2$$

Number Systems and Coding

Convert $(010111.110)_2$ to octal number

2^2 2^1 2^0 | 2^2 2^1 2^0 | 2^2 2^1 2^0

0 1 0 | 1 1 1 | 1 1 0

$= 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2$ | $= 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2$ | $= 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2$

$= 2$ | $= 7$ | $= 6$

$$(010111.110)_2 = (27.6)_8$$

Number Systems and Coding

To convert from octal to hexadecimal

1- convert from octal to binary

2- then, convert from binary to hexadecimal

Convert $(670.25)_8$ to hexadecimal number

		Reminder
2	6	0
2	3	1
	1	1

$\Rightarrow 110$

		Reminder
2	7	1
2	3	1
	1	1

$\Rightarrow 111$

		Reminder
2	0	0

$\Rightarrow 000$

		Reminder
2	2	0
	1	1

$\Rightarrow 010$

		Reminder
2	5	1
2	2	0
	1	1

$\Rightarrow 101$

$$(110111000.010101)_2$$

0001 1011 1000 . 0101 0100

1 B 8 . 5 4

$$\mathbf{(670.25)_8 = (1B8.54)_{16}}$$

Number Systems and Coding

To convert from hexadecimal to octal

1- convert from hexadecimal to binary

2- then, convert from binary to octal

Convert $(AB67)_{16}$ to octal number

A	**B**	**6**	**7**
1010	1011	0110	0111

$(1010101101100111)_2$

001	010	101	101	100	111
1	2	5	5	4	7

$$\mathbf{(AB67)_{16} = (125547)_8}$$

Number Systems and Coding

Hexadecimal Digits and Its Decimal and Binary Equivalents

Hex	Decimal	Binary	Hex	Decimal	Binary
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	A	10	1010
3	3	0011	B	11	1011
4	4	0100	C	12	1100
5	5	0101	D	13	1101
6	6	0110	E	14	1110
7	7	0111	F	15	1111

Conversion Between Binary and Hexadecimal

Example:

5	C	A	← Hexadecimal
0 1 0 1	1 1 0 0	1 0 1 0	← Binary

References

[1] A.P. Godse and Dr. D. A. Godse “Design Logic Circuits Using VHDL”, Technical Publication Pune , India, First Edition, 2021.

[2] A.P. Godse and Dr. D. A. Godse “Logic Design & Computer Organization”, Technical Publication Pune , India, First Edition, 2021.

Contents

- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Multiplexers and Decoders
- Sequential Logic Circuits
- Registers
- Counters

Number Systems and Coding

Binary Coded Decimal (BCD) Numbering System

give a particular binary code for each decimal digit.

This code consists of four digits such as hexadecimal, but starts from 0000 to 1001

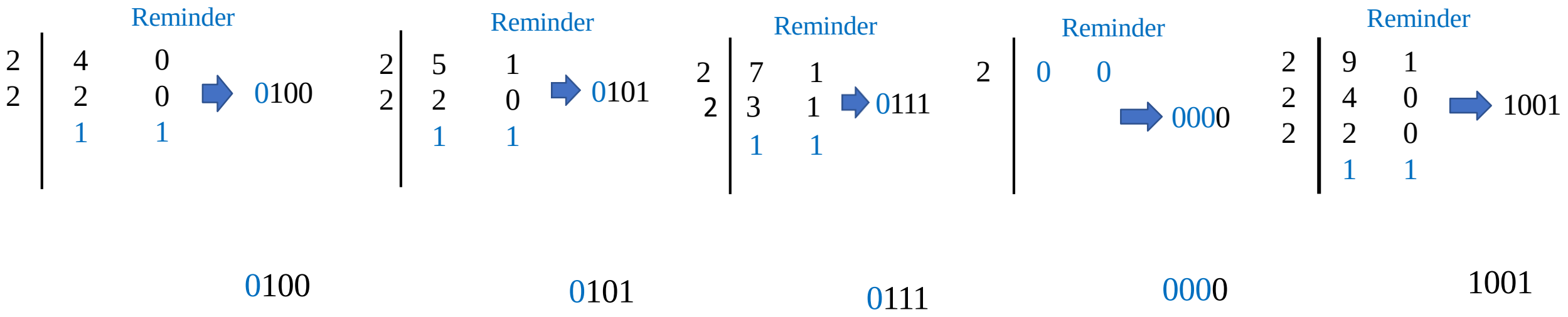
Decimal digit	BCD (8421)
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

This code is valid BCD code: 1111

No

Number Systems and Coding

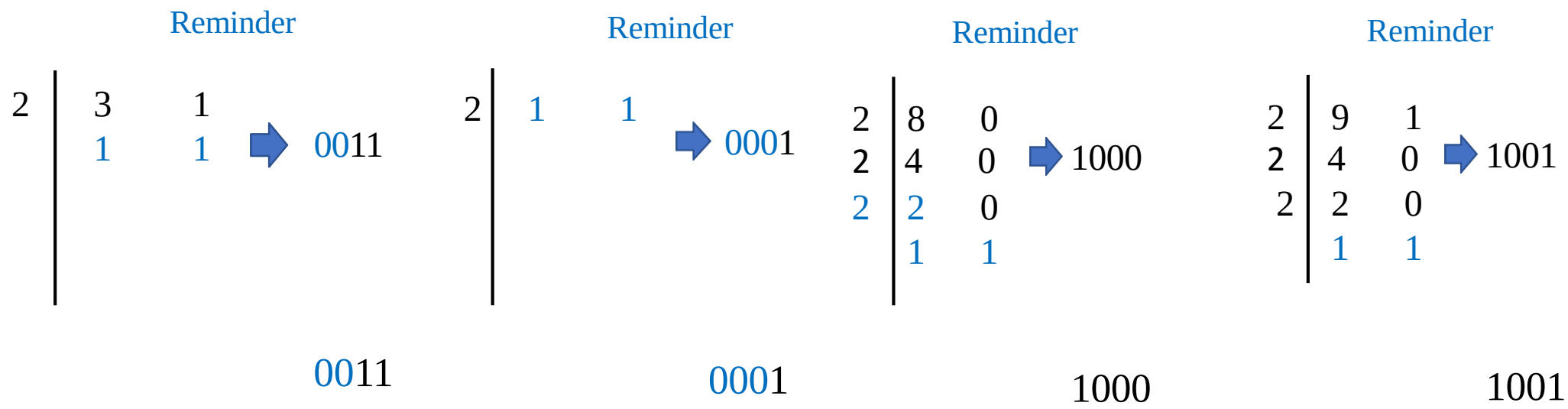
Convert $(45709)_{10}$ to BCD number



$$(45709)_{10} = (01000101011100001001)_{BCD}$$

Number Systems and Coding

Convert $(31.89)_{10}$ to BCD number



$$(31.89)_{10} = (00110001.10001001)_{BCD}$$

Number Systems and Coding

Convert $(10000110.0100)_{BCD}$ to decimal number

$2^3\ 2^2\ 2^1\ 2^0$	$2^3\ 2^2\ 2^1\ 2^0$		$2^3\ 2^2\ 2^1\ 2^0$
1000	0110	.	0100
$= 0 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3$	$= 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3$		$= 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 0 \times 2^3$
$= 0 + 0 + 0 + 8$	$= 0 + 2 + 4 + 0$		$= 0 + 4 + 0 + 0$
$= 8$	$= 6$		$= 4$

$$(10000110.0100)_2 = (86.4)_{10}$$

Number Systems and Coding

To convert from binary/octal/hexadecimal to BCD

1. convert from binary/octal/hexadecimal to decimal
2. then, convert from decimal to BCD

To convert from BCD to binary/octal/hexadecimal

1. convert from BCD to decimal
2. then, convert from decimal to binary/octal/hexadecimal

Number Systems and Coding

Convert $(0101.011000100101)_{BCD}$ to binary number

$2^3\ 2^2\ 2^1\ 2^0$ $2^3\ 2^2\ 2^1\ 2^0$. $2^3\ 2^2\ 2^1\ 2^0$ $2^3\ 2^2\ 2^1\ 2^0$

0101 0110 0010 0101

$= 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3$ $= 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3$ $= 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 0 \times 2^3$ $= 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3$

$= 1 + 0 + 4 + 0$ $= 0 + 2 + 4 + 0$ $= 0 + 2 + 0 + 0$ $= 1 + 2 + 4 + 0$

$= 5$ $= 6$ $= 2$ $= 5$

$$(5.625)_{10}$$

Reminder

2 |	5	1
2 |	2	0
	1	1

→ 101

Multiplication

1.25 |	.625	× 2	
0.5 |	.25	× 2	→ .101
1 |	.5	× 2	

Stop at integer number "one"

$$(0101.011000100101)_{BCD} = (101.101)_2$$

Number Systems and Coding

Convert $(1110.101)_2$ to BCD number

2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}
1	1	1	0	.	1	0	1

$= 1 \times 2^{-3} + 0 \times 2^{-2} + 1 \times 2^{-1} + 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 1 \times 2^3$

$= 0.125 + 0 + 0.5 + 0 + 2 + 4 + 8$

$= 14.625$

$(14.625)_{10}$

Reminder		
2 | 1	1	⇨ 0001

0001

Reminder		
2 | 4	0	⇨ 0100
2 | 2	0	
1	1	

0100

.

Reminder		
2 | 6	0	⇨ 0110
2 | 3	1	
1	1	

0110

Reminder		
2 | 2	0	⇨ 0010
1	1	

0010

Reminder		
2 | 5	1	⇨ 0101
2 | 2	0	
1	1	

0101

$(1110.101)_2 = (00010100.011000100101)_{BCD}$

Number Systems and Coding

Arithmetic Operations

1- Adding

$$\begin{array}{r} 0 \\ + \\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ + \\ 0 \\ \hline 1 \end{array} \qquad \begin{array}{r} \text{carry} \\ (1) \\ 1 \\ + \\ 1 \\ \hline 1\ 0 \end{array} \qquad \begin{array}{r} \text{carry} \\ (1) \\ 1 \\ 1 \\ + \\ 1 \\ \hline 1\ 1 \end{array}$$

Number Systems and Coding

Arithmetic Operations

1- Adding

Examples

$$
\begin{array}{r}
\textcircled{1}\,\textcircled{1}\,\textcircled{1}\,\textcircled{1}\,\textcircled{1}\,\textcircled{1} \\
1\ 0\ 1\ 1\ 0\ 1\ 0\ 1 \\
+ \\
1\ 1\ 1\ 0\ 1\ 1\ 0\ 0 \\
\hline
1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1
\end{array}
$$

$(1011.01)_2 + (11010.1)_2$

$$
\begin{array}{r}
\textcircled{1}\,\textcircled{1}\quad\textcircled{1} \\
1\ 1\ 0\ 1\ 0.\ 1\ 0 \\
+ \\
0\ 1\ 0\ 1\ 1.\ 0\ 1 \\
\hline
1\ 0\ 0\ 1\ 0\ 1.\ 1\ 1
\end{array}
$$

$(1110.11)_2 + (11011.101)_2$

$$
\begin{array}{r}
\textcircled{1}\,\textcircled{1}\,\textcircled{1}\,\textcircled{1}\,\textcircled{1}\,\textcircled{1} \\
1\ 1\ 0\ 1\ 1.\ 1\ 0\ 1 \\
+ \\
0\ 1\ 1\ 1\ 0.\ 1\ 1\ 0 \\
\hline
1\ 0\ 1\ 0\ 1\ 0.\ 0\ 1\ 1
\end{array}
$$

Number Systems and Coding

Arithmetic Operations

Adding 395 + 496 in BCD

Solution

Firstly, convert each number to BCD

	395		0011	1001	0101
+		+			
	496		0100	1001	0110
			1000	0010	1011

If number > 9, we add to it 6 ➡ "0110"

If number carried by carry, we add to it 6 ➡ "0110"

		1000	0010	1011
+			0110	0110
		1000	1001	0001
		8	9	1

➡ $(891)_{BCD}$

Number Systems and Coding

Arithmetic Operations

2- Subtraction

0	1	1	0 (borrow: -1)
- 0	- 0	- 1	- 1
0	1	0	1

Examples

Subtract $(1011)_2$ from $(1101.1)_2$

		(0)			
	1	1	0	1.	1
-					
	1	0	1	1.	0
	0	0	1	0.	1

Number Systems and Coding

Arithmetic Operations

One's Complement: reverse one to zero & reverse zero to one

110111

One's Complement for this number → 001000

Two's Complement

110111

One's Complement for this number → 001000

\+ 1

Two's Complement for this number → 001001

Number Systems and Coding

Arithmetic Operations

Subtraction in One's Complement

Find the result of 1010 – 110 in one's complement

$$
\begin{array}{rl}
 & 1010 \\
- & 0110 \\
\hline
\end{array}
\quad \xrightarrow{\text{as it is}} \quad
\begin{array}{rl}
 & \textcircled{1}1010 \\
+ & 1001 \\
\hline
 & \textcircled{1}0011 \\
+ & \quad 1 \\
\hline
 & 0100
\end{array}
$$

Result is: + 0100

If carry is (1), we add 1

Number Systems and Coding

Arithmetic Operations

Subtraction in One's Complement

Find the result of 10000 – 10110 in one's complement

$$\begin{array}{r} 10000 \\ - \quad 10110 \\ \hline \end{array} \xrightarrow[\text{one's comp.}]{\text{as it is}} \begin{array}{r} 10000 \\ + \quad 01001 \\ \hline (0)\,11001 \end{array}$$

Result is: - 00110

If no carry, we don't add 1
and the result given in one's comp.

Number Systems and Coding

Arithmetic Operations

3- Multiplication

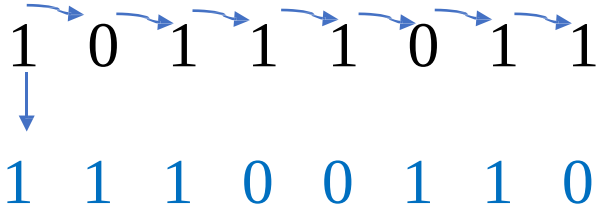
$$\begin{array}{r} 0 \\ \times\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ \times\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ \times\ 1 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ \times\ 1 \\ \hline 1 \end{array}$$

Number Systems and Coding

Arithmetic Operations

Convert binary code $(10111011)_2$ to gray code

Solution



gray code is $(11100110)_G$

Rule

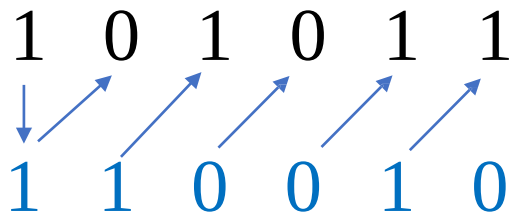
We compare
if two numbers the same, we put 0
if two numbers are different, we put 1

Number Systems and Coding

Arithmetic Operations

Convert gray code $(101011)_G$ to binary code

Solution



Rule

We compare
if two numbers the same, we put 0
if two numbers are different, we put 1

binary code is $(110010)_2$

Contents

- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Multiplexers and Decoders
- Sequential Logic Circuits
- Registers
- Counters

Boolean Algebra

In 1854, George Boole devised a systematic treatment of logic using an algebraic system, namely Boolean Algebra.

Boolean Algebra is a system of mathematical logic.

Variable: is the symbol which represent an arbitrary elements of an Boolean algebra. Any single variable or a function of several variables can have either a 1 or 0 value.

For example, in expression Y =A + BC, variables A, B and C can have either a 1 or 0 value, and function Y also can have either a 1 or 0 value; however its value depends on the value of Boolean expression.

Boolean Algebra

Basic Theorems (Rules) of Boolean Algebra

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)	$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$

Basic Theorems (Rules) of Boolean Algebra

Theorem 1 (Idempotency)	$A + A = A$	$A \cdot A = A$
Theorem 2	$\overline{A} + 1 = 1$ $A + 1 = 1$	$A \cdot 0 = 0$
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)	$A + (B + C) = (A + B) + C$	$A\,(BC) = (AB)\,C$

=A+B+C

=ABC

Basic Theorems (Rules) of Boolean Algebra

Boolean A

(1)	$A.1 = A$	$A+0 = A$
	$A.0 = 0$	$A+1 = 1$
(2)	$A.A = A$	$A+A = A$
(3)	$\overline{\overline{A}} = A$	
(4)	$A.\overline{A} = 0$	$A+\overline{A} = 1$
(5)	$AB = BA$	$A+B = B+A$

Basic Theorems (Rules) of Boolean Algebra

(6)	$(AB)C = A(BC) = ABC$	$(A+B)+C = A+(B+C) = A+B+C$
(7)	$A(B+C) = AB+AC$	$A+BC = (A+B)(A+C)$
(8)	$A(\bar{A}+B) = AB$	$A+\bar{A}B = A+B$
(9)	$A(A+B) = A$	$A+AB = A$
(10)	$(A+B)(A+\bar{B}) = A$	$AB+A\bar{B} = A$

Boolean Algebra

DeMorgan's Theorems

$$\overline{AB} = \overline{A} + \overline{B}$$ The complement of a product is equal to the sum of the complements.

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$ The complement of a sum is equal to the product of the complements.

Example

Simplify the following Boolean expression using Boolean Algebra

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

Solution :

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

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

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

Boolean Algebra

Example

Simplify the following Boolean expression using Boolean Algebra

$$\overline{A}BC\overline{D} + BC\overline{D} + B\overline{C}\,\overline{D} + B\,\overline{C}\,D$$

Solution

Distributive $= BC\overline{D}\,(\overline{A} + 1) + B\overline{C}\,\overline{D} + B\overline{C}\,D$

Theorem 2 $= BC\overline{D} + B\overline{C}\,\overline{D} + B\overline{C}\,D$

Distributive $= B\overline{D}\,(C + \overline{C}) + B\overline{C}\,D$

Complement] $= B\overline{D} + B\overline{C}\,D$

Distributive $= B\,(\overline{D} + \overline{C}\,D)$

Theorem 5 $= B\,(\overline{D} + \overline{C})$

Boolean Algebra

Example

Simplify the following Boolean expression using Boolean Algebra

$$Z = AB\,\overline{C} + A\,\overline{B}\,C + ABC + A\,\overline{B}\,\overline{C}$$

Solution

$$Z = AB\,(\overline{C} + C) + A\overline{B}\,(C + \overline{C})$$

$$Z = AB + A\overline{B}$$

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

$$Z = A \cdot 1$$

$$Z = A$$

Boolean Algebra

Example

Simplify the following Boolean expression using Boolean Algebra

$Z = AB\ C\ D + A\ \overline{B}\ C\ \overline{D} + \overline{A}\overline{B}C\overline{D} + A\ B\ C\ \overline{D} + \overline{A}\ B\ C\ D$

Solution

$Z = AB\ C\ D + A\ \overline{B}\ C\ \overline{D} + \overline{A}\overline{B}C\overline{D} + A\ B\ C\ \overline{D} + \overline{A}\ B\ C\ D + ABCD$

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

$Z = AB\ C + \overline{B}C\overline{D} + B\ C\ D$

Boolean Algebra

Report

Simplify the following Boolean expression using Boolean Algebra

$$a\bar{b}\ \bar{c} + a\bar{b}c + abc.$$

Boolean Algebra

Minterms

To express the logical functions, we use minterms

for n variables of logic functions, there are number of minterms = 2^n

For three variable, number of minterms = $2^n = 2^3 = 8$ (m_0 to m_7)

Example Find minterms for the following logic function

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

$$= m_0 + m_1 + m_3 + m_6$$

$$= \sum m_0, m_1, m_3, m_6$$

$$= \sum m(0, 1, 3, 6)$$

$\sum$ denotes **sum of product**

Variables			Minterms
A	B	C	m_i
0	0	0	$\overline{A}\,\overline{B}\,\overline{C} = m_0$
0	0	1	$\overline{A}\,\overline{B}\,C = m_1$
0	1	0	$\overline{A}\,B\,\overline{C} = m_2$
0	1	1	$\overline{A}\,B\,C = m_3$
1	0	0	$A\,\overline{B}\,\overline{C} = m_4$
1	0	1	$A\,\overline{B}\,C = m_5$
1	1	0	$A\,B\,\overline{C} = m_6$
1	1	1	$A\,B\,C = m_7$

Example

Find minterms for the following expression

$y = AB\,\overline{C} + A\,\overline{C}\,D + A\overline{B}CD + \overline{A}\,\overline{B}$

Solution

Should be all terms of the expression include all variables

$y = AB\,\overline{C}\,(D + D) + A\,\overline{C}\,D\,(B+\overline{B}) + A\overline{B}CD + A\,\overline{B}\,(C+\overline{C})\,(D + \overline{D})$

$y = AB\,\overline{C}\,D + ABC\overline{D} + A\,\overline{C}\,D\,B + A\,\overline{C}\,D\,\overline{B} + A\overline{B}CD + \overline{A}\,\overline{B}\,(CD + C\overline{D} + \overline{C}D + CD)$

$y = AB\,C\,D + AB\overline{C}\overline{D} + A\,\overline{C}\,D\,B + A\,\overline{C}\,D\,\overline{B} + A\overline{B}CD + \overline{A}\,\overline{B}\,CD + \overline{A}\,\overline{B}\,C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + A\,\overline{B}\,\overline{C}\,\overline{D})$

Remove the repeated

$y = AB\,\overline{C}\,D + AB\overline{C}\overline{D} + A\,\overline{C}\,D\,\overline{B} + A\overline{B}CD + \overline{A}\,\overline{B}\,CD + \overline{A}\,\overline{B}\,C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D})$

$y = 1101 + 1100 + 1001 + 1011 + 0011 + 0010 + 0001 + 0000$

$y = 13 + 12 + 10 + 11 + 3 + 2 + 1 + 0$

$y = \sum m_{13}, m_{12}, m_{10}, m_{11}, m_3, m_2, m_1, m_0$

Contents

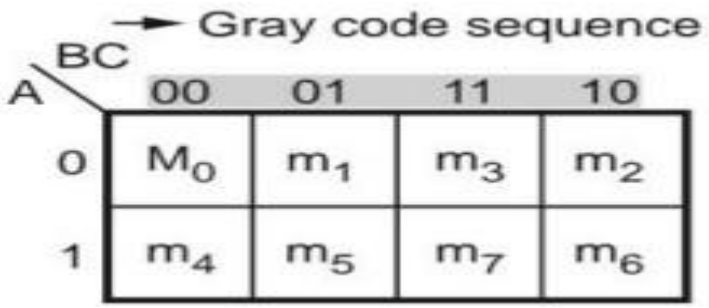
- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Multiplexers and Decoders
- Sequential Logic Circuits
- Registers
- Counters

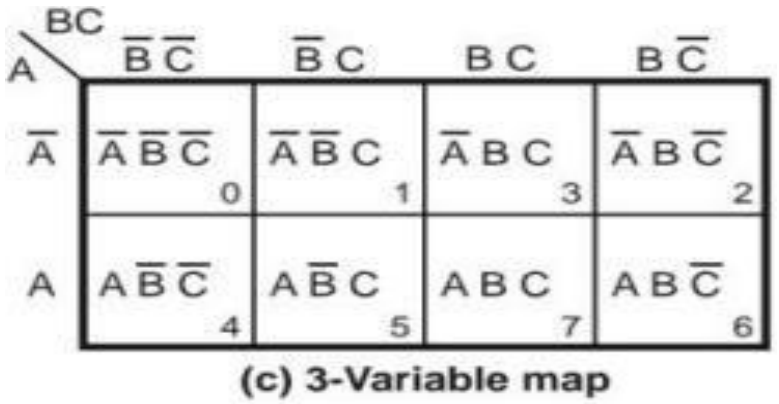
Boolean Algebra

Karnaugh Map (K-Map) Method

Used to simplify Boolean expressions rather than predicting each step. In addition, it is systematic method.

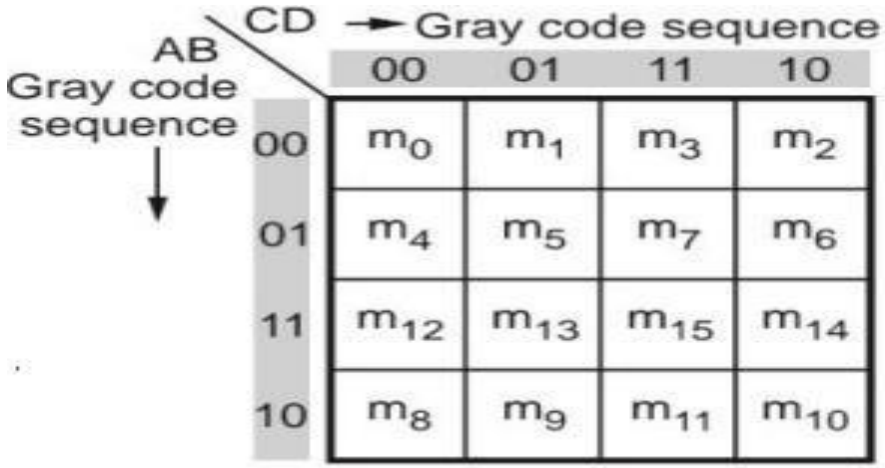
K-Map is a graphical chart contains a number of cells = 2^n, where n is the number of variables

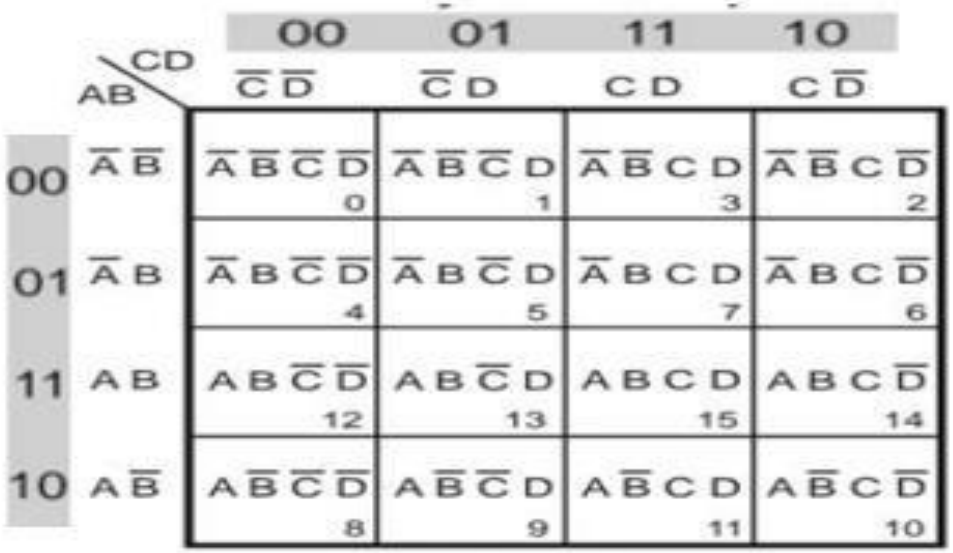




(c) 3-Variable map

product term of each cell = its row × its column





(d) 4-Variable map

Boolean Algebra

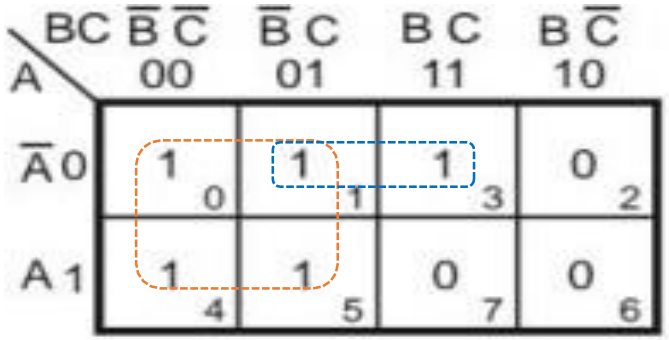
Example (1)

Simplify the following Boolean expression using Karnaugh Map

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

Solution

step 1, plot K-Map of 3-variables according to the given expression, each minterm is found in the expression replaced by 1 and the minterm which not found is 0



step 2, there are no isolated ones

step 3, combine each **Octets** of adjacent ones in a group

step 4, combine each **Pairs** of adjacent ones in a group

step 5, combine each **Quads** of adjacent ones in a group

step 6, all Ones must grouped

step 7, represent each group in a term in the expression Y

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

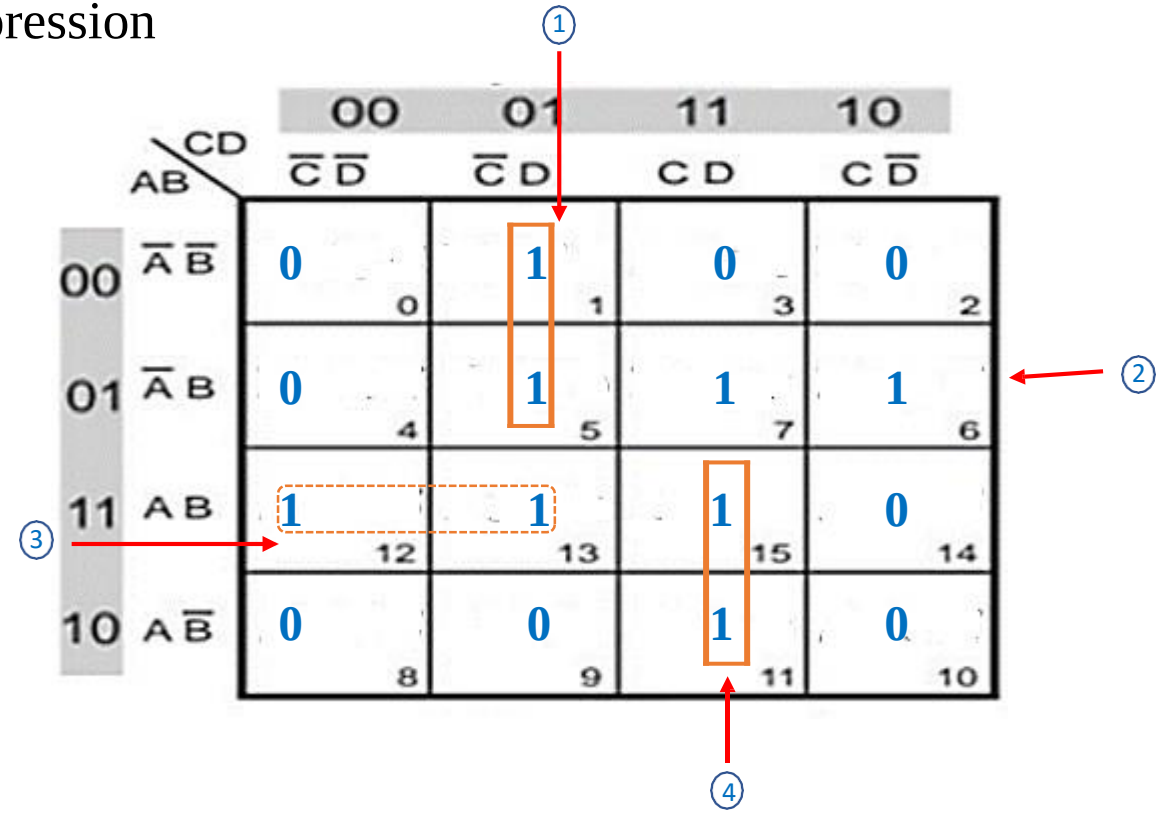
Example (2) Simplify the following Boolean expression using Karnaugh Map

$$Y = \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}D + \bar{A}BCD + \bar{A}BC\bar{D} + AB\bar{C}\bar{D} + AB\bar{C}D + ABCD + A\bar{B}CD$$

Solution

step 1, plot K-Map of 4-variables according to the given expression

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$ (00)	0 (0)	1 (1)	0 (3)	0 (2)
$\bar{A}B$ (01)	0 (4)	1 (5)	1 (7)	1 (6)
AB (11)	1 (12)	1 (13)	1 (15)	0 (14)
$A\bar{B}$ (10)	0 (8)	0 (9)	1 (11)	0 (10)



step 2, there are no isolated ones

step 3, combine each **Octets** of adjacent ones in a group

step 4, combine each **Pairs** of adjacent ones in a group

step 5, combine each **Quads** of adjacent ones in a group

step 6, all Ones must grouped

step 7, represent each group in a term in the expression Y

$$Y = \bar{A}\bar{C}D + \bar{A}BC + AB\bar{C} + ACD$$

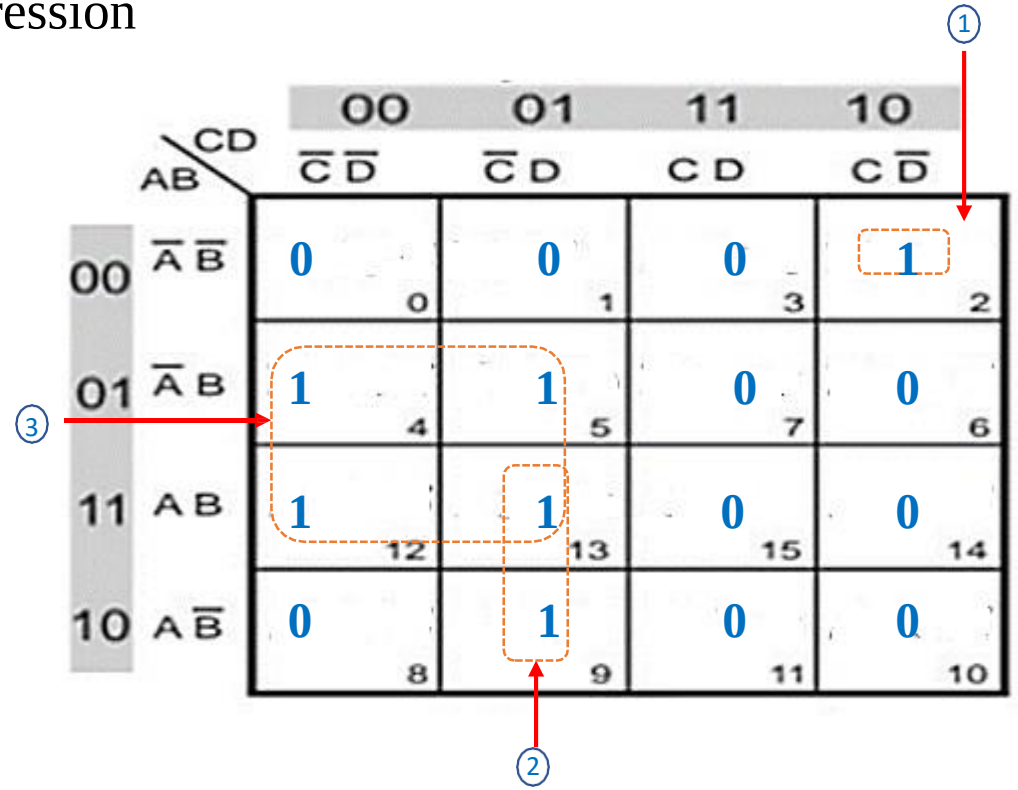
Example (3) Simplify the following Boolean expression using Karnaugh Map

$$Y = \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + AB\overline{C}\overline{D} + AB\overline{C}D + A\overline{B}\overline{C}D + \overline{A}\,\overline{B}\,CD$$

Solution

step 1, plot K-Map of 4-variables According to the given expression

AB \ CD	$\overline{C}\,\overline{D}$ (00)	$\overline{C}D$ (01)	CD (11)	$C\overline{D}$ (10)
$\overline{A}\,\overline{B}$ (00)	0 (0)	0 (1)	0 (3)	1 (2)
$\overline{A}B$ (01)	1 (4)	1 (5)	0 (7)	0 (6)
AB (11)	1 (12)	1 (13)	0 (15)	0 (14)
$A\overline{B}$ (10)	0 (8)	1 (9)	0 (11)	0 (10)



step 2, group isolated ones

step 3, combine each **Octets** of adjacent ones in a group

step 3, combine each **Pairs** of adjacent ones in a group

step 5, combine each **Quads** of adjacent ones in a group

step 6, all Ones must grouped

step 7, represent each group in a term in the expression Y

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

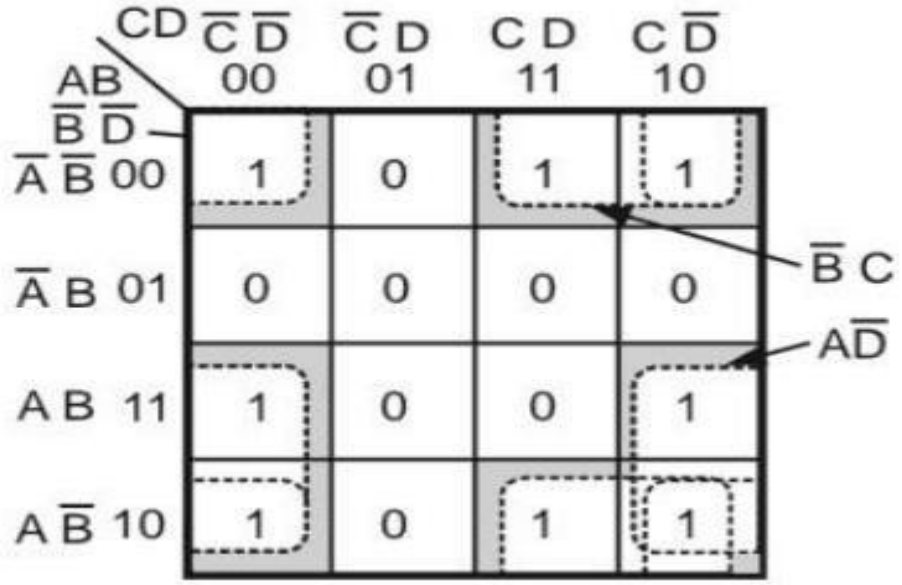
Example (4) Simplify the following Boolean expression using Karnaugh Map

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

Solution

step 1, plot K-Map of 4-variables According to the given expression

AB \ CD	$\overline{C}\,\overline{D}$ (00)	$\overline{C}D$ (01)	CD (11)	$C\overline{D}$ (10)
$\overline{A}\,\overline{B}$ (00)	1 (0)	0 (1)	1 (3)	1 (2)
$\overline{A}B$ (01)	0 (4)	0 (5)	0 (7)	0 (6)
AB (11)	1 (12)	0 (13)	0 (15)	1 (14)
$A\overline{B}$ (10)	1 (8)	0 (9)	1 (11)	1 (10)



step 2, There are no isolated ones

step 3, combine each **Octets** of adjacent ones in a group

step 5, combine each **Quads** of adjacent ones in a group

step 4, there are no **Pairs** of adjacent ones in a group

step 6, all Ones must grouped

step 7, represent each group in a term in the expression Y

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

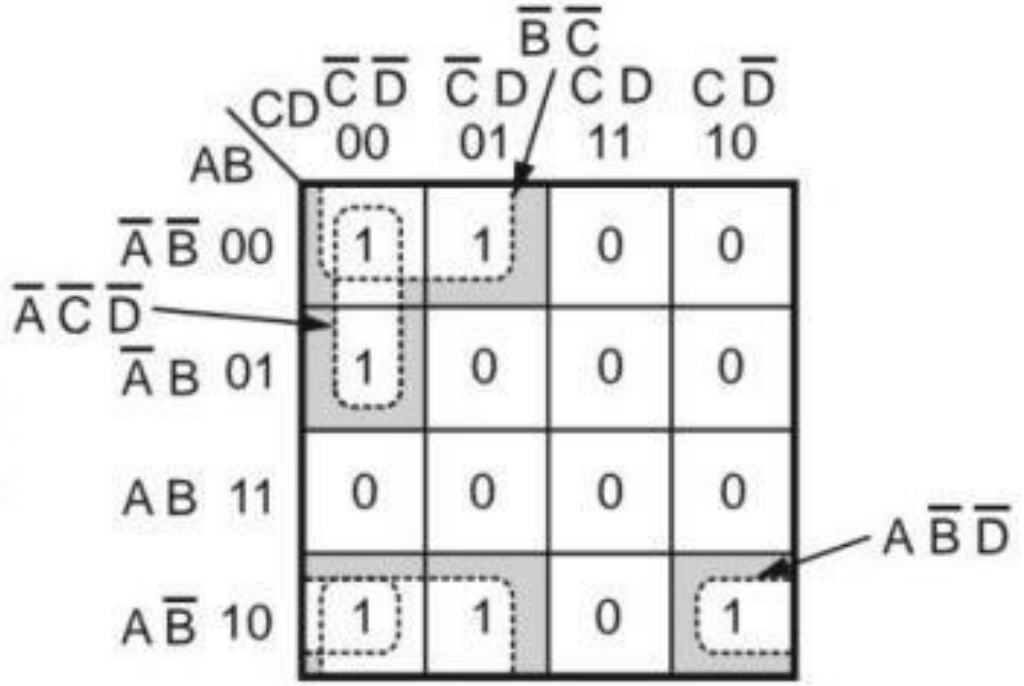
Example (5) Reduce the following function using K-map technique

$$f\,(A,\ B,\ C,\ D) = \sum \mathrm{m}\,(0,\ 1,\ 4,\ 8,\ 9,\ 10).$$

Solution

step 1, plot K-Map of 4-variables According to the given expression

AB \ CD	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	1 (0)	1 (1)	0 (3)	0 (2)
$\bar{A}B$ 01	1 (4)	0 (5)	0 (7)	0 (6)
AB 11	0 (12)	0 (13)	0 (15)	0 (14)
$A\bar{B}$ 10	1 (8)	1 (9)	0 (11)	1 (10)



step 2, There are no isolated ones

step 3, combine each **Octets** of adjacent ones in a group

step 5, combine each **Quads** of adjacent ones in a group

step 4, there are no **Pairs** of adjacent ones in a group

step 6, all Ones must grouped

step 7, represent each group in a term in the expression Y

$$f\,(A,\ B,\ C,\ D) = \bar{A}\bar{C}\bar{D} + A\bar{B}\bar{D} + \bar{B}\bar{C}$$

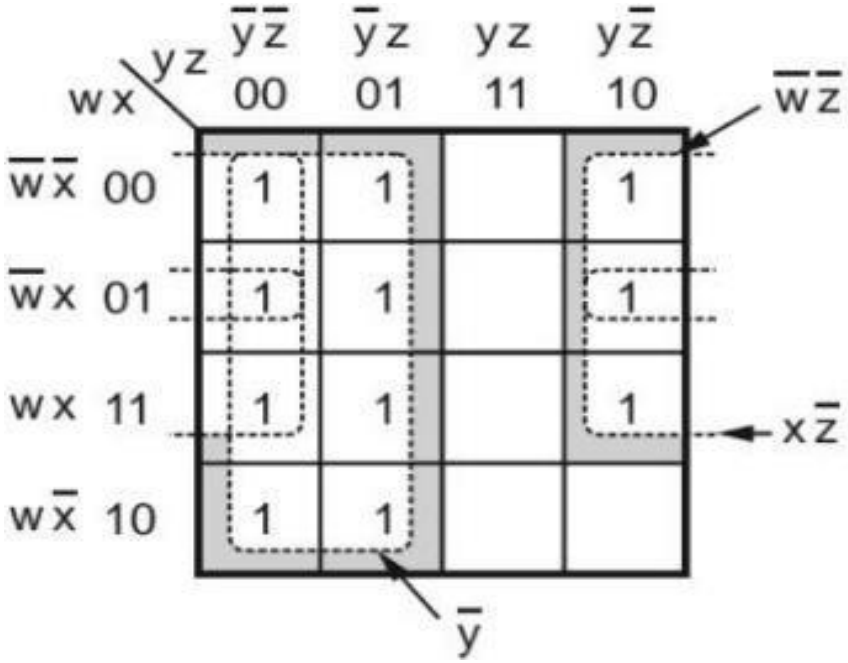
Example (6)/Assignment Plot the following Boolean function on a Karnaugh map and simplify it.

$$F(w,x,y,z) = \Sigma\,(0,1,2,4,5,6,8,9,12,13,14)$$

Solution

step 1, plot K-Map of 4-variables According to the given expression

AB \ CD	00 ($\bar{C}\bar{D}$)	01 ($\bar{C}D$)	11 (CD)	10 ($C\bar{D}$)
00 ($\bar{A}\bar{B}$)	1 (0)	1 (1)	0 (3)	1 (2)
01 ($\bar{A}B$)	1 (4)	1 (5)	0 (7)	1 (6)
11 (AB)	1 (12)	1 (13)	0 (15)	1 (14)
10 ($A\bar{B}$)	1 (8)	1 (9)	0 (11)	0 (10)



$$\therefore f(w, x, y, z) = \bar{y} + x\bar{z} + \bar{w}\bar{z}$$

Contents

- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Multiplexers and Decoders
- Sequential Logic Circuits
- Registers
- Counters

Boolean Algebra

Logic Gates

Circuits performs a particular logical function, these gates deal with a number of binary inputs.

Basic Logic Gates → NOT & AND & OR

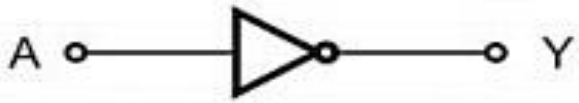
Derived Logic Gates → NAND & NOR

NOT Gate

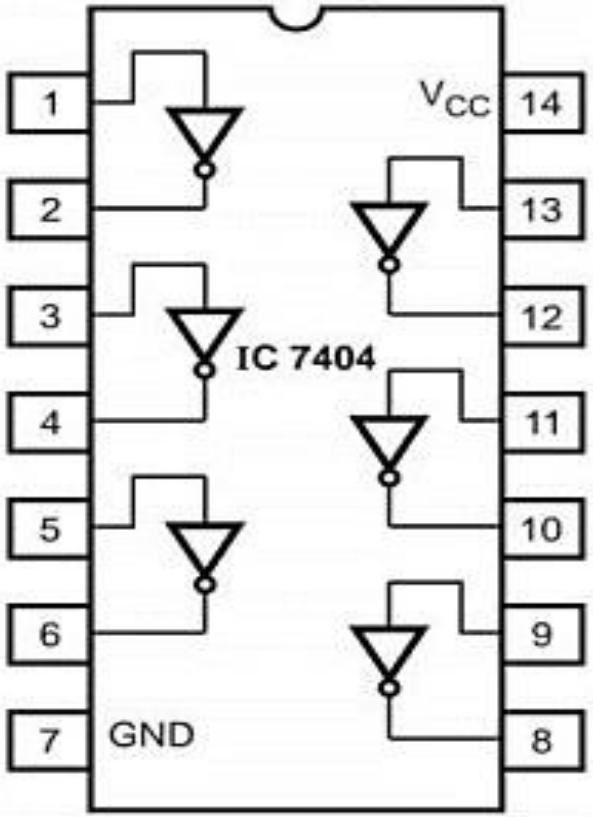
Boolean Algebra

NOT gate (Inverter) : The output is a complement of input.

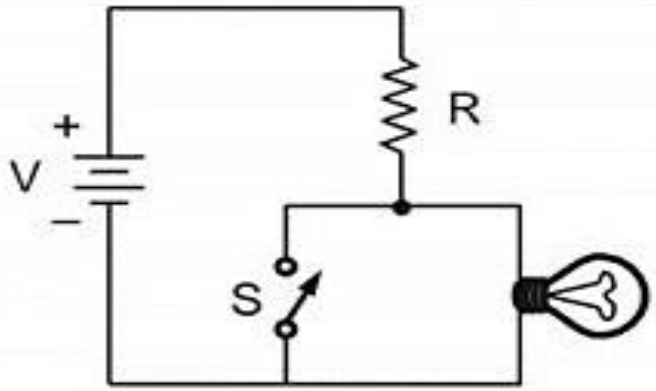
Logic Diagram (Symbol)



Pin Diagram



Switch Equivalent



Input	Output
Switch open (Low)	Lamp ON (High)
Switch close (High)	Lamp OFF (Low)

Application :

Used to complement (invert) digital signal.

Truth Table

Input	Output
A	Y
0	1
1	0

Boolean Expression

$$Y = \overline{A}$$

AND Gate

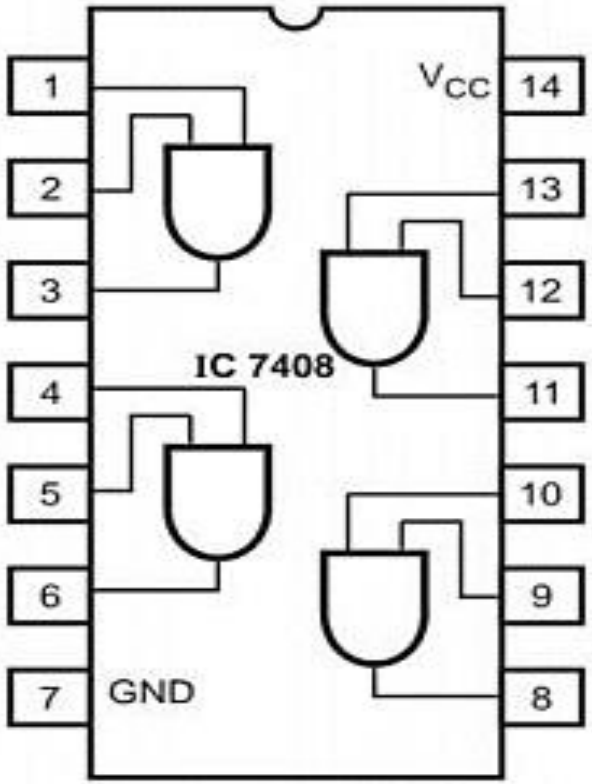
Boolean Algebra

AND gate : The output is high only when all inputs are high.

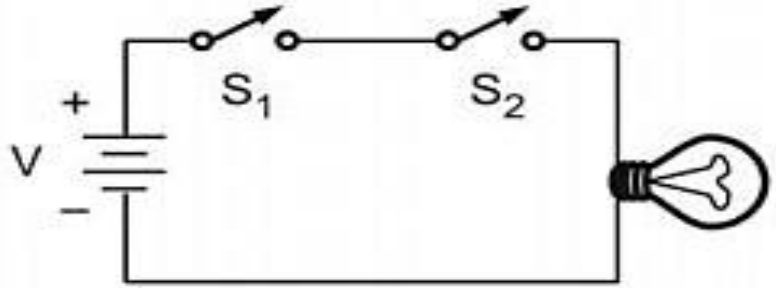
Logic Diagram (Symbol)



Pin Diagram



Switch Equivalent



Input		Output
S_1	S_2	
Open (Low)	Open (Low)	Lamp OFF (Low)
Open (Low)	Close (High)	Lamp OFF (Low)
Close (High)	Open (Low)	Lamp OFF (Low)
Close (High)	Close (High)	Lamp ON (High)

Application : Used to implement logical AND operation.

Truth Table

Input		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Boolean Expression

$$Y = A \cdot B$$

OR Gate

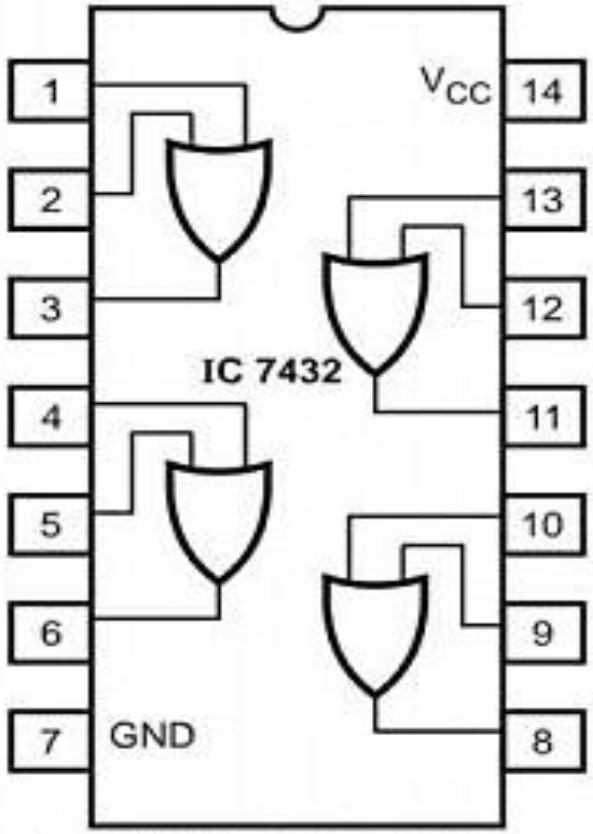
Boolean Algebra

OR gate : The output is high when one or more inputs are high.

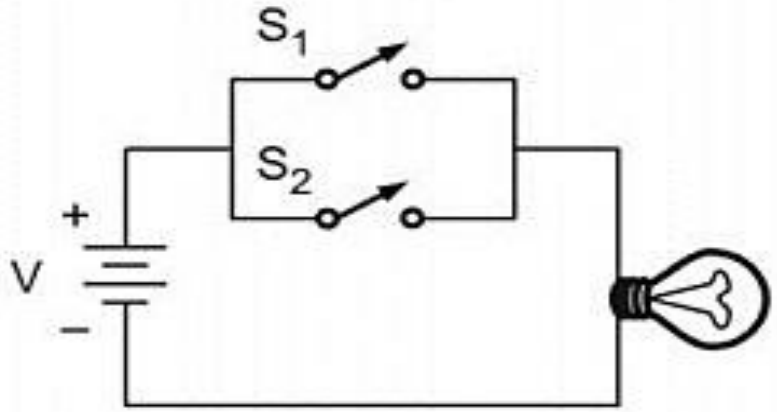
Logic Diagram (Symbol)



Pin Diagram



Switch Equivalent



Input		Output
S_1	S_2	
Open (Low)	Open (Low)	Lamp OFF (Low)
Open (Low)	Close (High)	Lamp ON (High)
Close (High)	Open (Low)	Lamp ON (High)
Close (High)	Close (High)	Lamp ON (High)

Application : Used to implement logical OR operation.

Truth Table

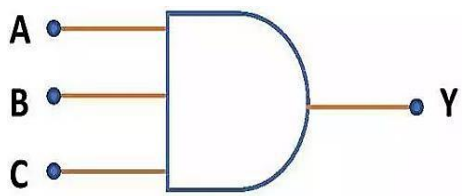
Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

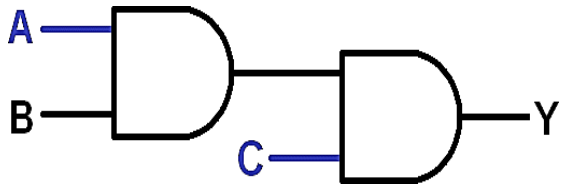
Boolean Expression

$$Y = A + B$$

Boolean Algebra

AND Gate with Three Inputs

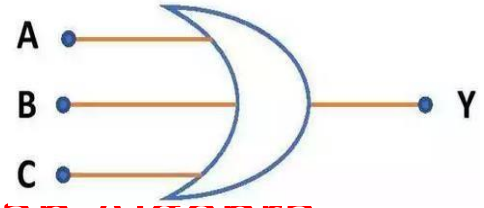


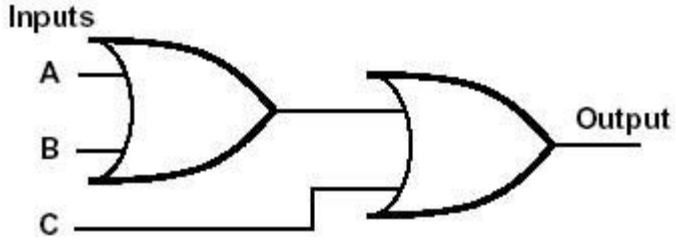


TRUTH TABLE

INPUTS			OUTPUT
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

OR Gate with Three Inputs





Boolean Algebra

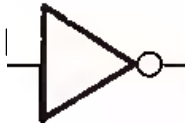
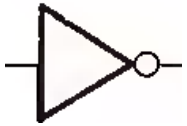
Three input OR gate :

Inputs			Output
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

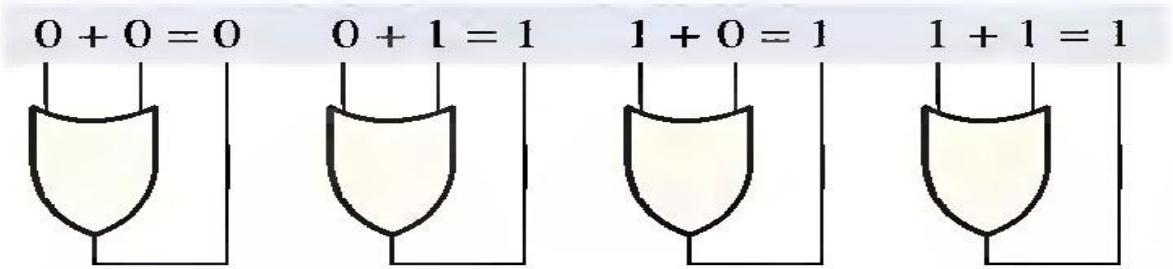
Boolean Algebra

Basic Logic Gates

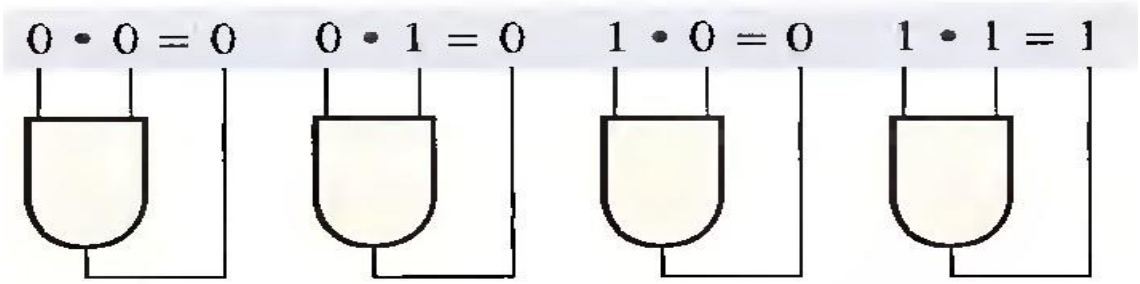
- NOT Gate

(Inve 1 1  0

- OR Gate (Boolean Addition)

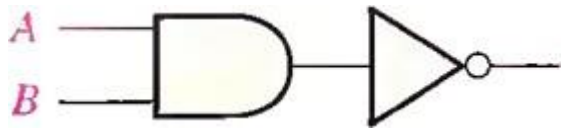


- AND Gate (Boolean Multiplication)



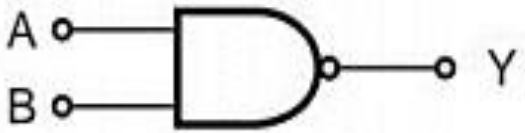
Boolean Algebra

NAND Gate



NAND gate : The output is high only when one or more inputs are low.

Symbol

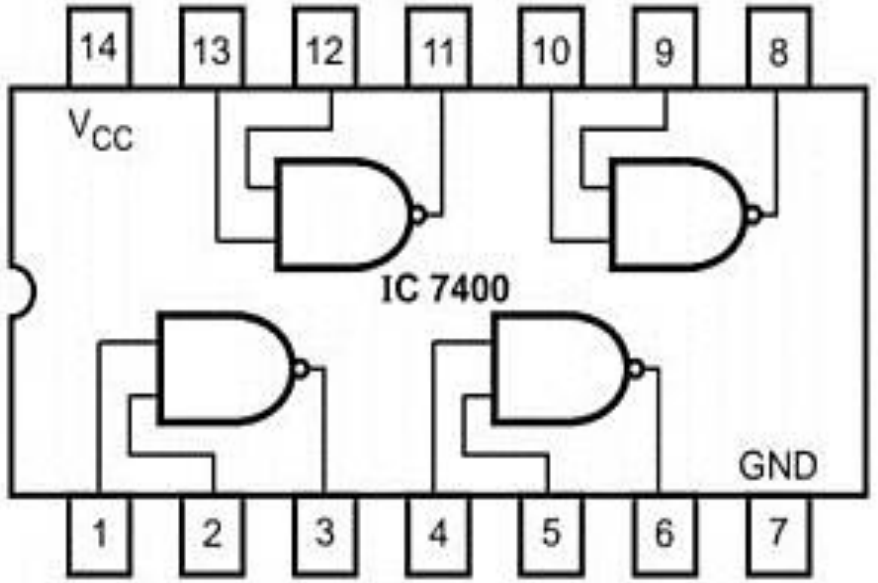


Application : It can be used to implement any digital circuit.

Boolean Expression

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

Pin Diagram

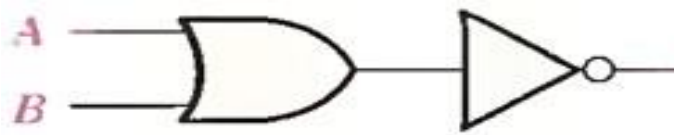


Truth Table

Input		Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Boolean Algebra

NOR Gate



NOR Gate : The output is high when all the inputs are low.

Symbol

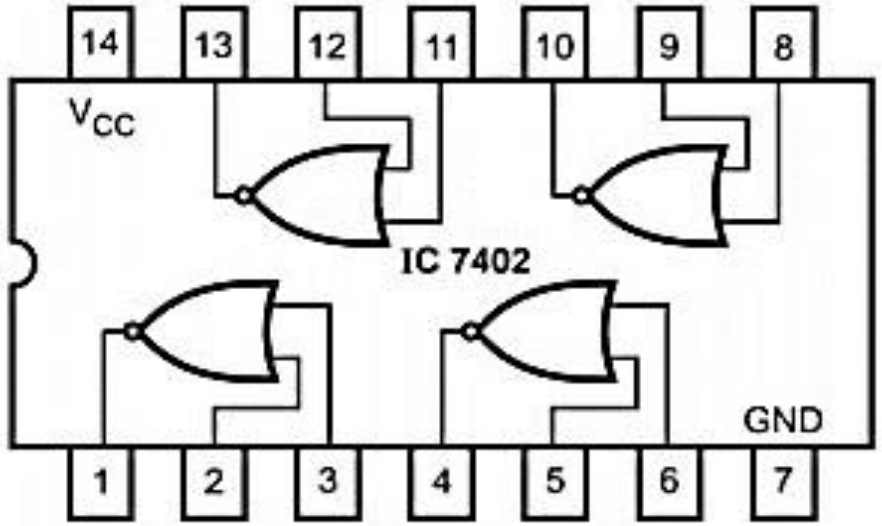


Application : It can be used to implement any digital circuit.

Boolean Expression

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

Pin Diagram



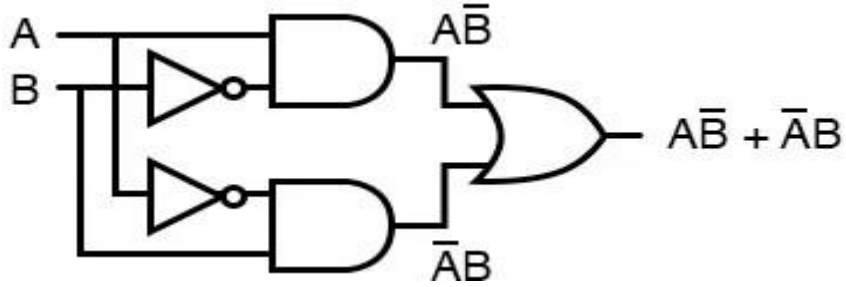
Truth Table

Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

NAND and NOR are known Universal gates since any logic function can be implemented by them.

Exclusive OR (EX-OR) Gate

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



Exclusive OR (EX-OR) gate : The output is high only when odd number of inputs are high.

Symbol

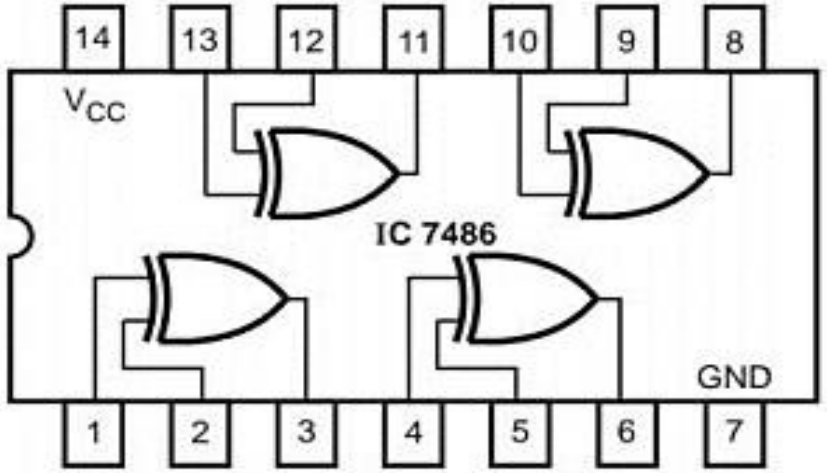


Application : It is used to implement magnitude comparator, gray code converter, adder/subtractor circuits, parity generator, modulo-2 adder, etc.

Boolean Expression

$$Y = A \oplus B$$

Pin Diagram



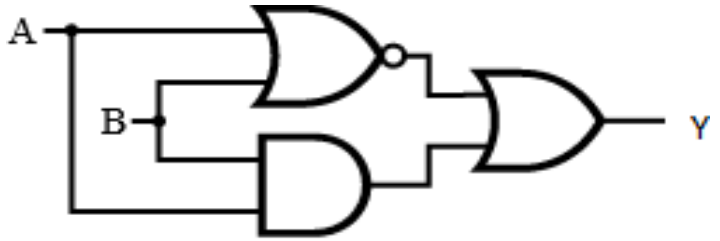
Truth Table

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Boolean Algebra

Exclusive NOR (EX-NOR) Gate

Iis an XOR gate followed by a NOT gate



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

Exclusive NOR (EX-NOR) gate : The output is high only when even number of inputs are high.

Symbol

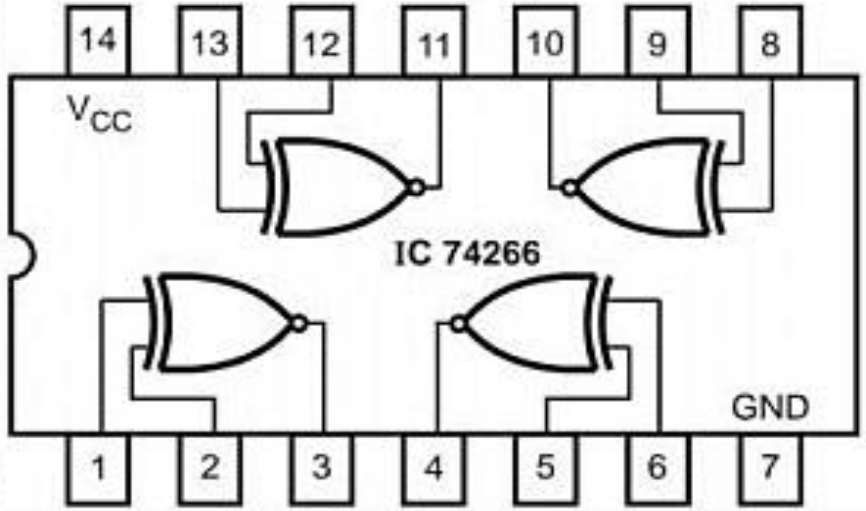


Applications : It is used to implement even parity generator, comparator, even parity checker, etc.

Boolean Expression

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

Pin Diagram



Truth Table

Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

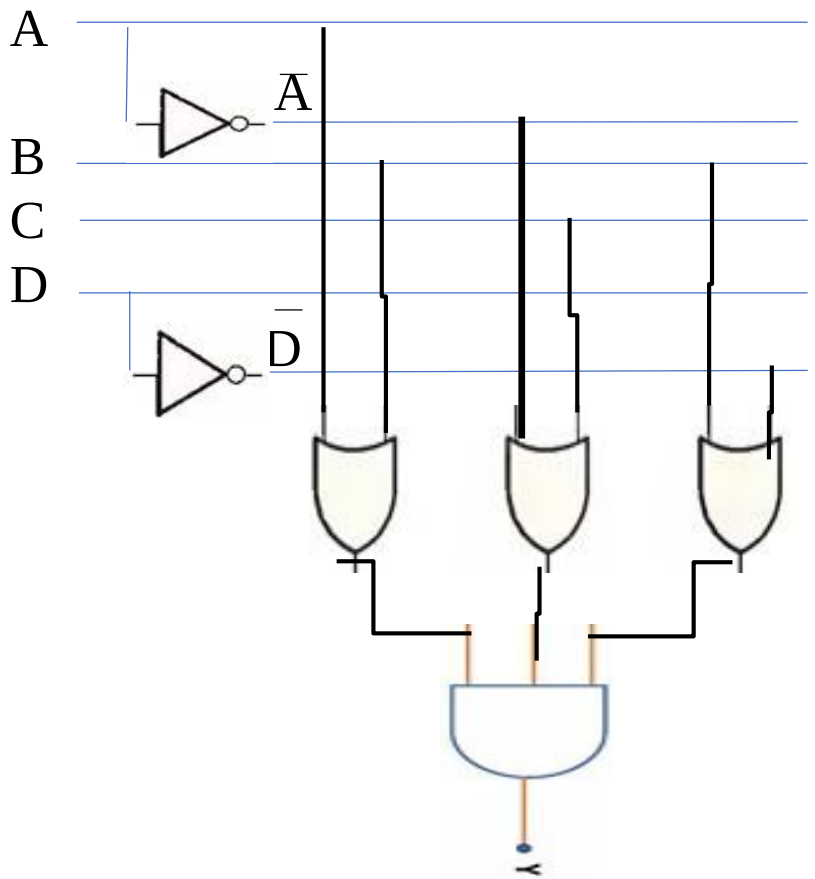
Boolean Algebra

The simplified Boolean expression can be implemented using basic logic gates.

Example

Implement logic circuit for the following expression $Y = (A+B)(\bar{A}+C)(B+\bar{D})$

Solution



Boolean Algebra

Three switches are used to operate a lamp, the lamp is ON when at least two switches are switched ON. Realize a logic circuit for this lamp

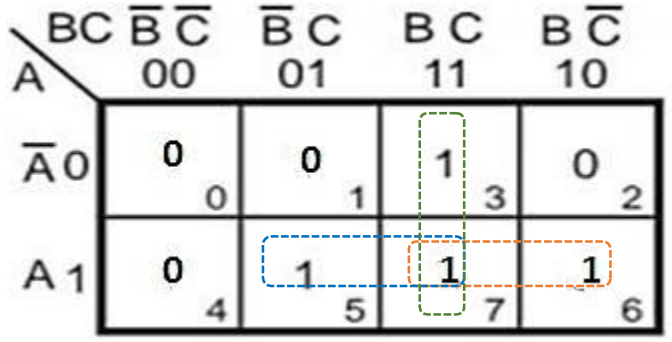
Solution

Let the three switches are A, B, and C and the lamp is Y

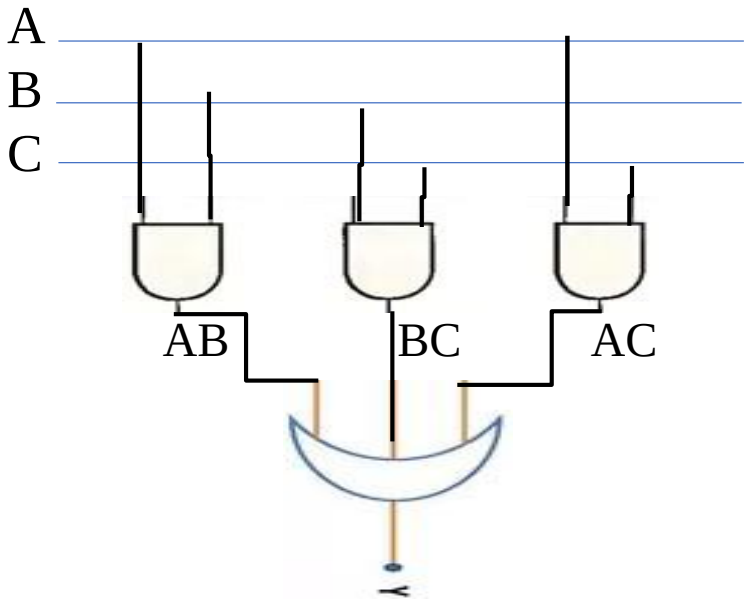
Truth Table

A	B	C	Y	
0	0	0	0	m_0
0	0	1	0	m_1
0	1	0	0	m_2
0	1	1	1	m_3
1	0	0	0	m_4
1	0	1	1	m_5
1	1	0	1	m_6
1	1	1	1	m_7

$Y = \sum m_3 , m_5 , m_6 , m_7$



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



Contents

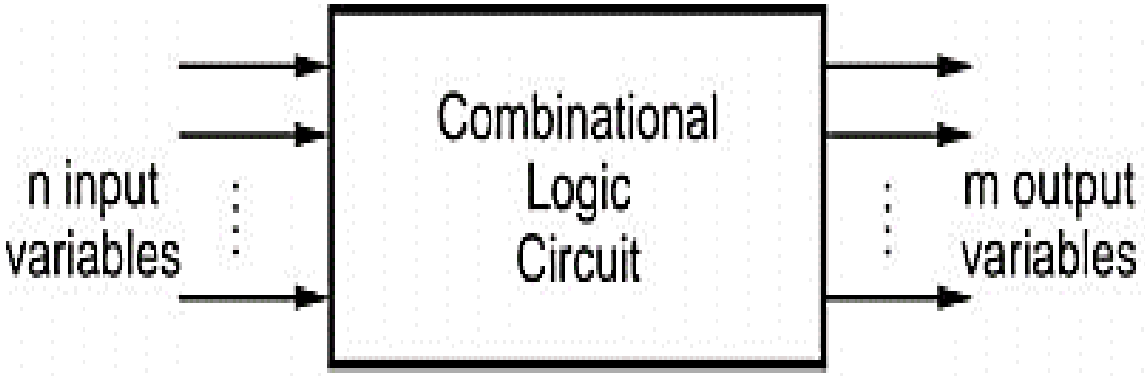
- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Multiplexers and Decoders
- Sequential Logic Circuits
- Registers
- Counters

Combinational Logic Circuits

Combinational Logic Circuit is a circuit consists of logic gates connected together to produce a specified output for certain combinations of input variables, with no storage involved.

- A combinational circuit consists of input variables, logic gates, and output variables.

Combinational circuits don't memorize old values of data. The outputs at any time instant T, are determined by the values of the inputs at the same time instant T.

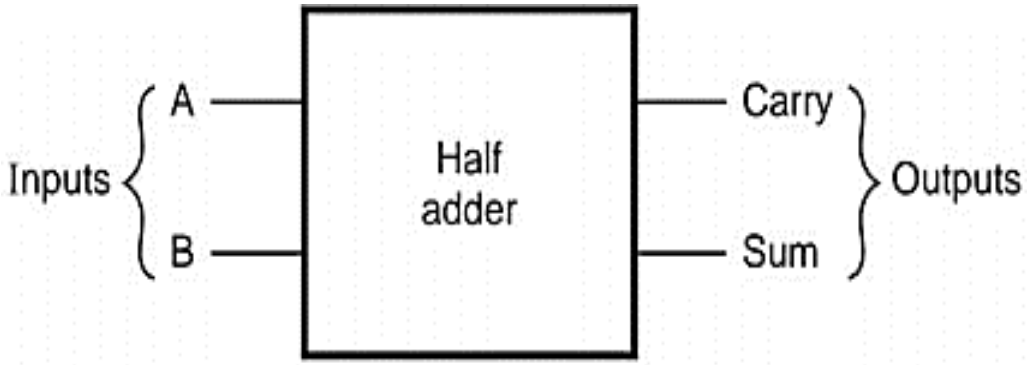


Digital computers perform various arithmetic operations, such as addition which implemented/designed using adder circuits.

Combinational Logic Circuits

Adder Circuits: half adder & Full adder

1- Half Adder: a circuit used to add two bits together and needs two binary inputs and has two outputs.



Logic Symbol

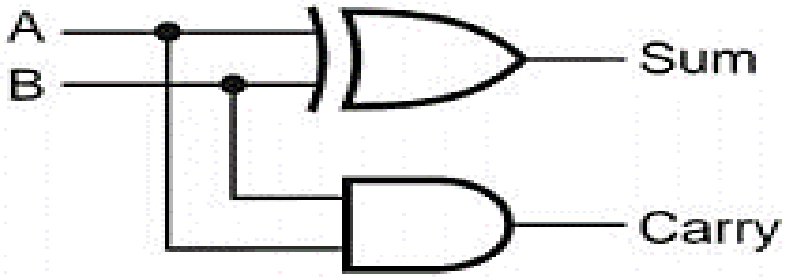
Truth Table of Half Adder

Inputs		Outputs	
A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Design or Logic Diagram of Half Adder

Boolean expression for Sum: $S = A \oplus B = \bar{A}B + A\bar{B}$

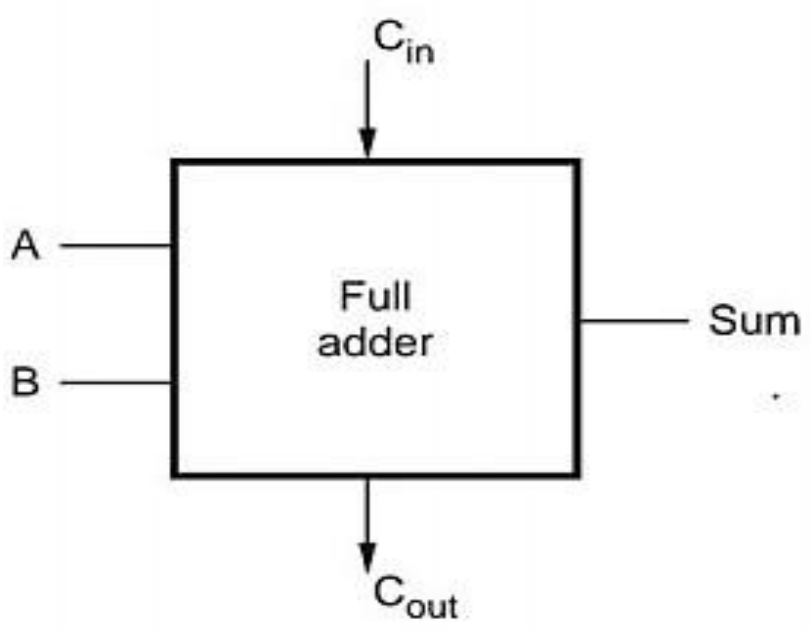
Boolean expression for Carry: $C = AB$



Logic Diagram

Combinational Logic Circuits

2- Full Adder: needs three binary inputs



Logic Symbol

Truth Table of Full Adder

Inputs			Outputs	
A	B	C_{in}	Carry	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Combinational Logic Circuits

Full Adder

For carry (C_{out})

A \ BC_{in}	00	01	11	10
0	0	0	1	0
1	0	1	1	1

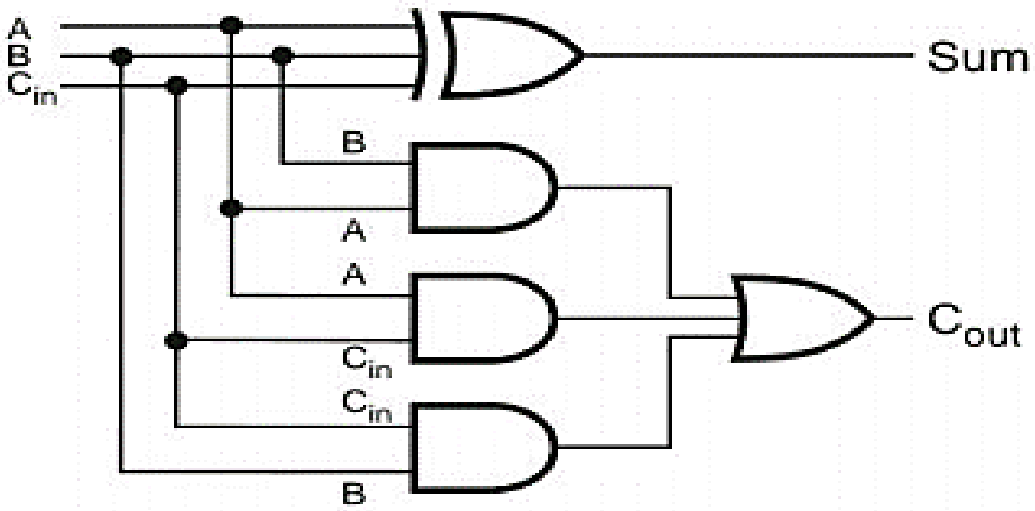
$$C_{out} = AB + A\,C_{in} + B\,C_{in}$$

For sum

A \ BC_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$Sum = \overline{A}\,\overline{B}C_{in} + \overline{A}B\overline{C}_{in} + A\overline{B}\,\overline{C}_{in} + ABC_{in}$$

Design or Logic Diagram of Full Adder

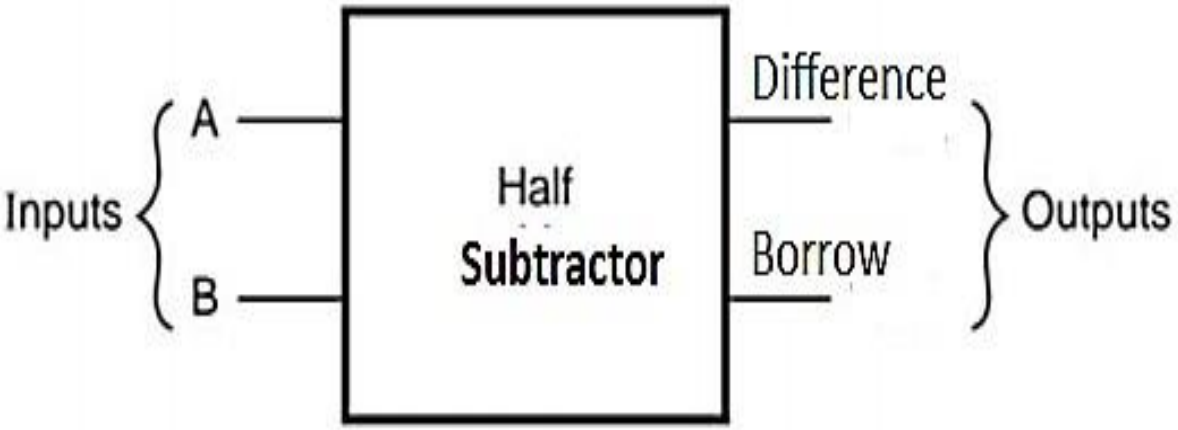


$$\begin{aligned} Sum &= \overline{A}\,\overline{B}\,C_{in} + \overline{A}\,B\,\overline{C}_{in} + A\,\overline{B}\,\overline{C}_{in} + A\,B\,C_{in} \\ &= C_{in}\,(\overline{A}\,\overline{B} + AB) + \overline{C}_{in}\,(\overline{A}\,B + A\,\overline{B}) \\ &= C_{in}\,(A \odot B) + \overline{C}_{in}\,(A \oplus B) \\ &= C_{in}\,\overline{(A \oplus B)} + \overline{C}_{in}\,(A \oplus B) \\ &= C_{in} \oplus (A \oplus B) \end{aligned}$$

Combinational Logic Circuits

Subtractor

Half Subtractor: needs two binary inputs



Logic Symbol

Truth Table of Half Subtractor

Inputs		Outputs	
A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Combinational Logic Circuits

Half Subtractor

For difference

A \ B	0	1
0	0	1
1	1	0

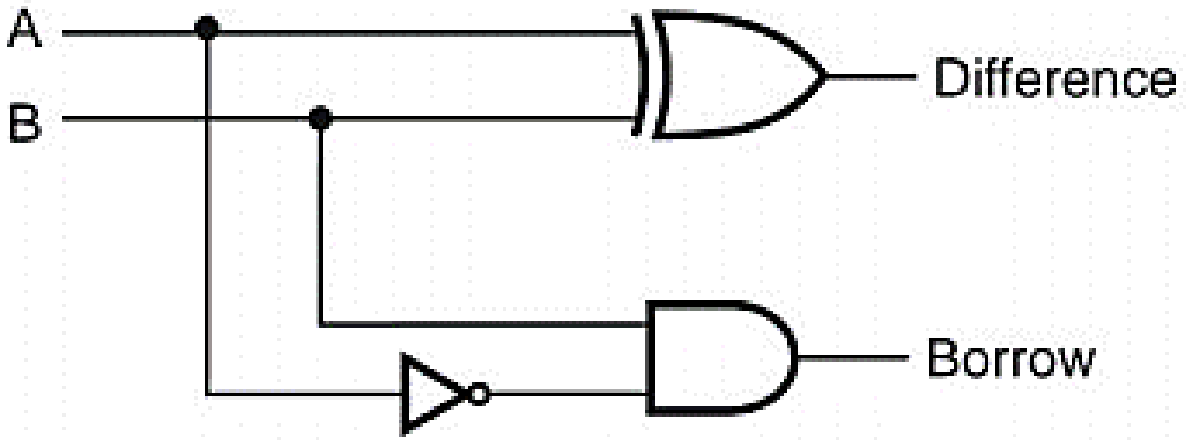
$$\text{Difference} = A\overline{B} + \overline{A}B$$
$$= A \oplus B$$

For borrow

A \ B	0	1
0	0	1
1	0	0

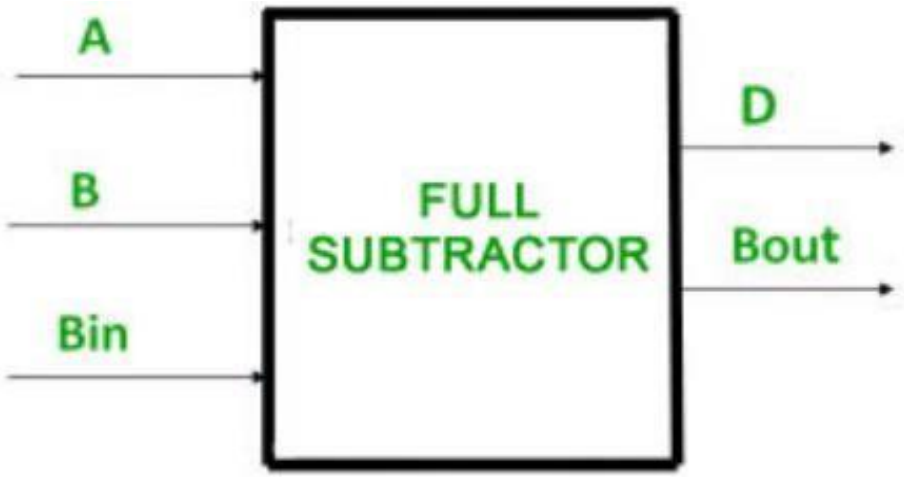
$$\text{Borrow} = \overline{A}B$$

Design or Logic Diagram of Half Subtractor



Combinational Logic Circuits

Full Subtractor



Logic Symbol

Truth Table of Full Subtractor

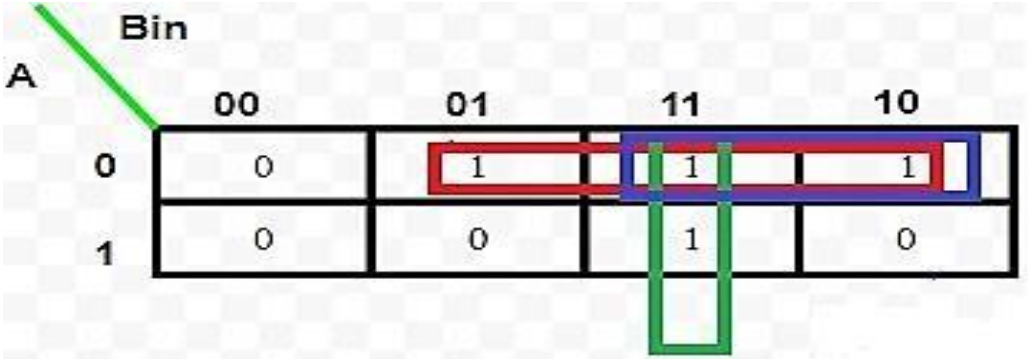
INPUT			OUTPUT	
A	B	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Combinational Logic Circuits

Full Subtractor

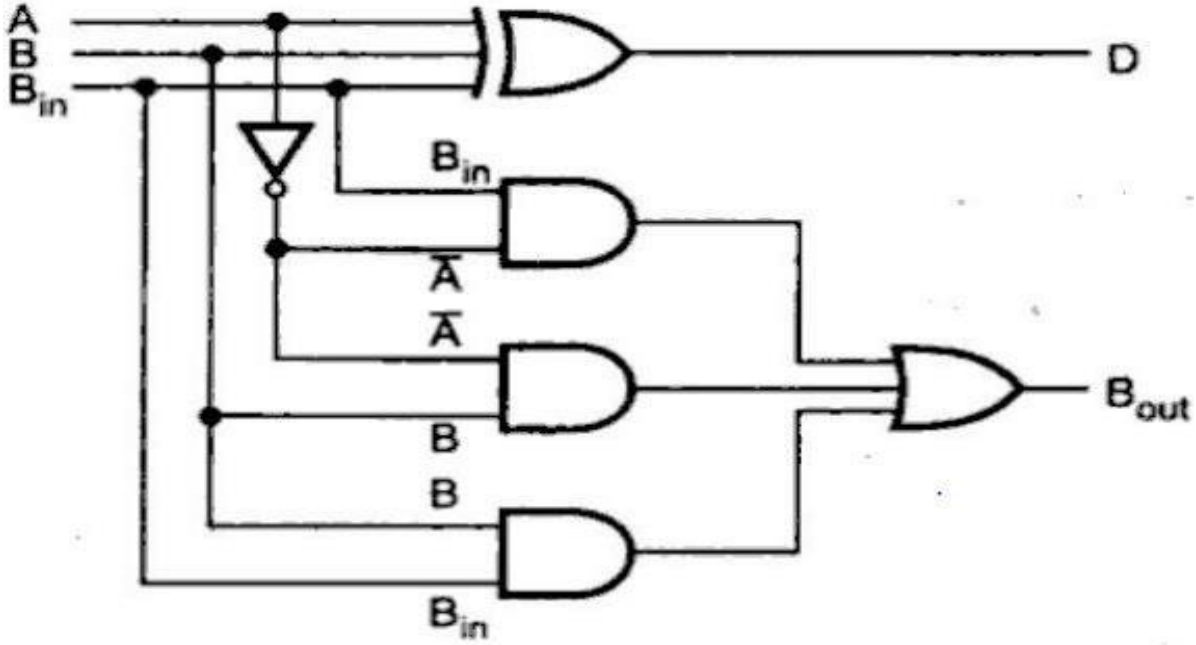
A \ Bin	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$D = \bar{A}\,\bar{B}\,B_{in} + \bar{A}\,B\,\bar{B}_{in} + A\,\bar{B}\,\bar{B}_{in} + A\,B\,B_{in}$$



$$Bout = A'Bin + A'B + BBin$$

Design or Logic Diagram of Full Subtractor



$$D = \bar{A}\,\bar{B}\,B_{in} + \bar{A}\,B\,\bar{B}_{in} + A\,\bar{B}\,\bar{B}_{in} + A\,B\,B_{in}$$

$$= B_{in}\,(\bar{A}\,\bar{B} + AB) + \bar{B}_{in}\,(\bar{A}\,B + A\,\bar{B})$$

$$= B_{in}\,(A \odot B) + \bar{B}_{in}\,(A \oplus B)$$

$$= B_{in}\,(\overline{A \oplus B}) + \bar{B}_{in}\,(A \oplus B)$$

$$= B_{in} \oplus (A \oplus B)$$

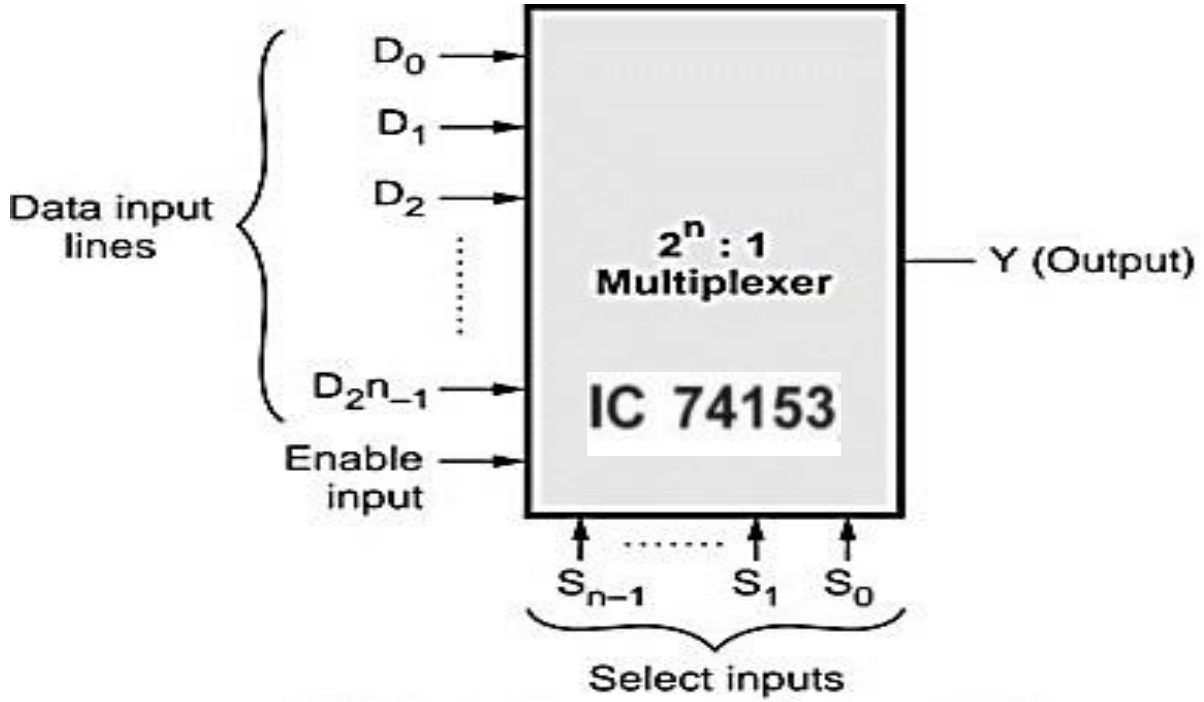
Combinational Logic Circuits

Multiplexer (many to one)

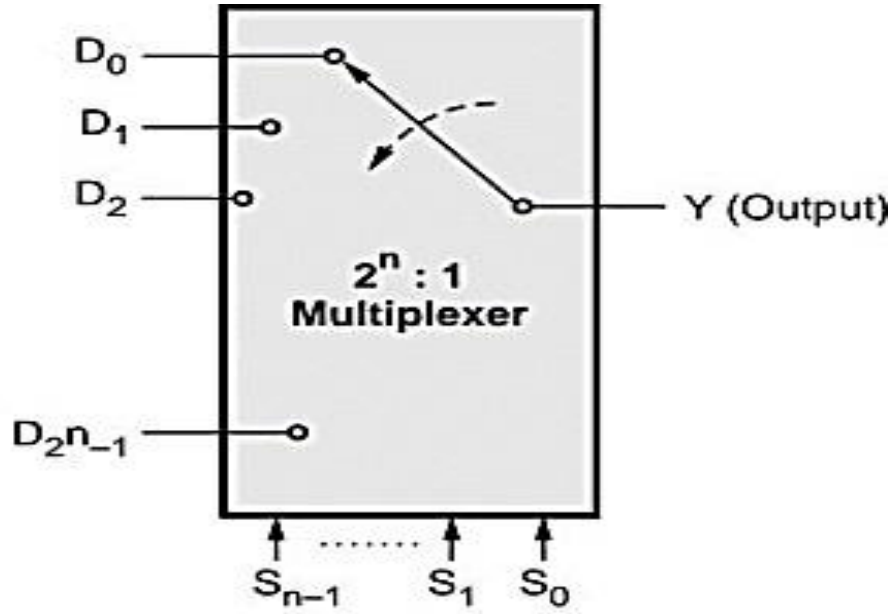
In digital systems, need a **multiplexer circuit** used to select single data line from multi data-input lines, and the data from the selected data line should be available on the output.

The selection of a particular input line is controlled by selection lines (S) "Signals". Known **Data Selector** S "used as control"

There are 2^n input lines and n selection lines whose bit combinations determine which input is selected.



(a) Block diagram of 2^n : 1 multiplexer



(b) Equivalent circuit

Combinational Logic Circuits

Multiplexer

Example (1)

Design an 2 to 1 multiplexer circuit

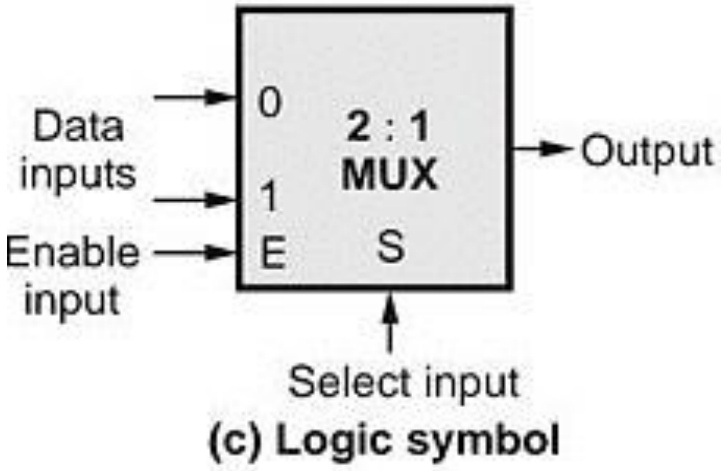
Solution

Number of inputs = 2 and number of outputs =1

Number of inputs = $2^{\text{no of control bits}}$

$2 = 2^{\text{no of control bits}}$

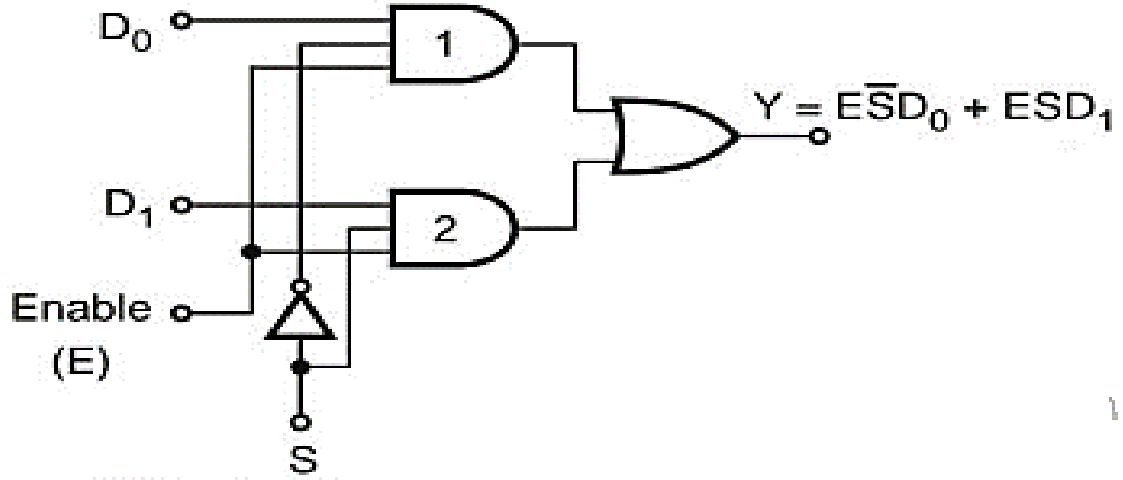
$2^{1} = 2^{\text{no of control bits}}$



(c) Logic symbol

E	S	Y
1	0	D_0
1	1	D_1
0	X	0

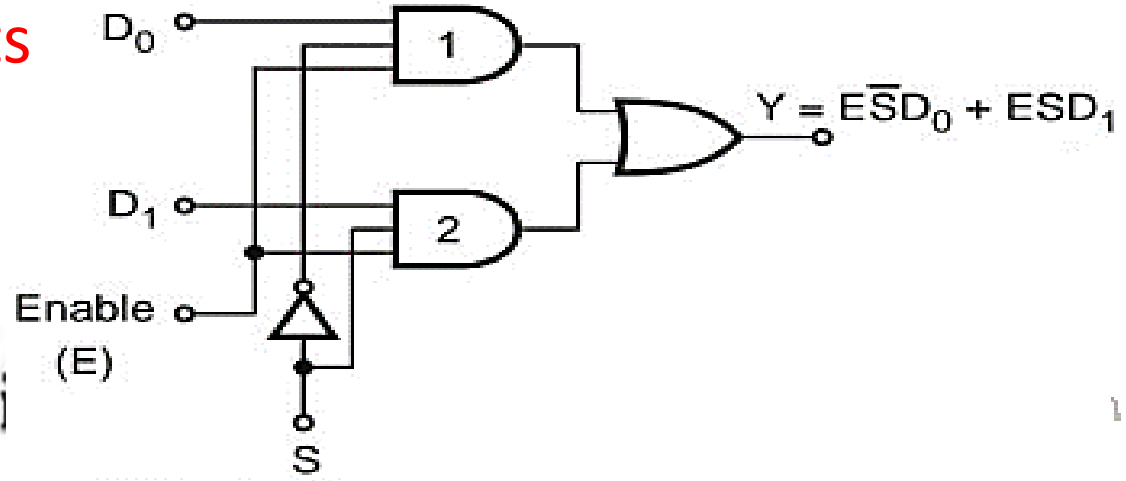
(b) Function table



(a) Logic diagram

Combinational Logic Circuits

Multiplexer



(a) Logic diagram

Working of 2 to 1 multiplexer circuit

- When E = 0, output is 0, i.e. Y = 0 irrespective of any input condi
- When E = 1, the circuit works as follows :

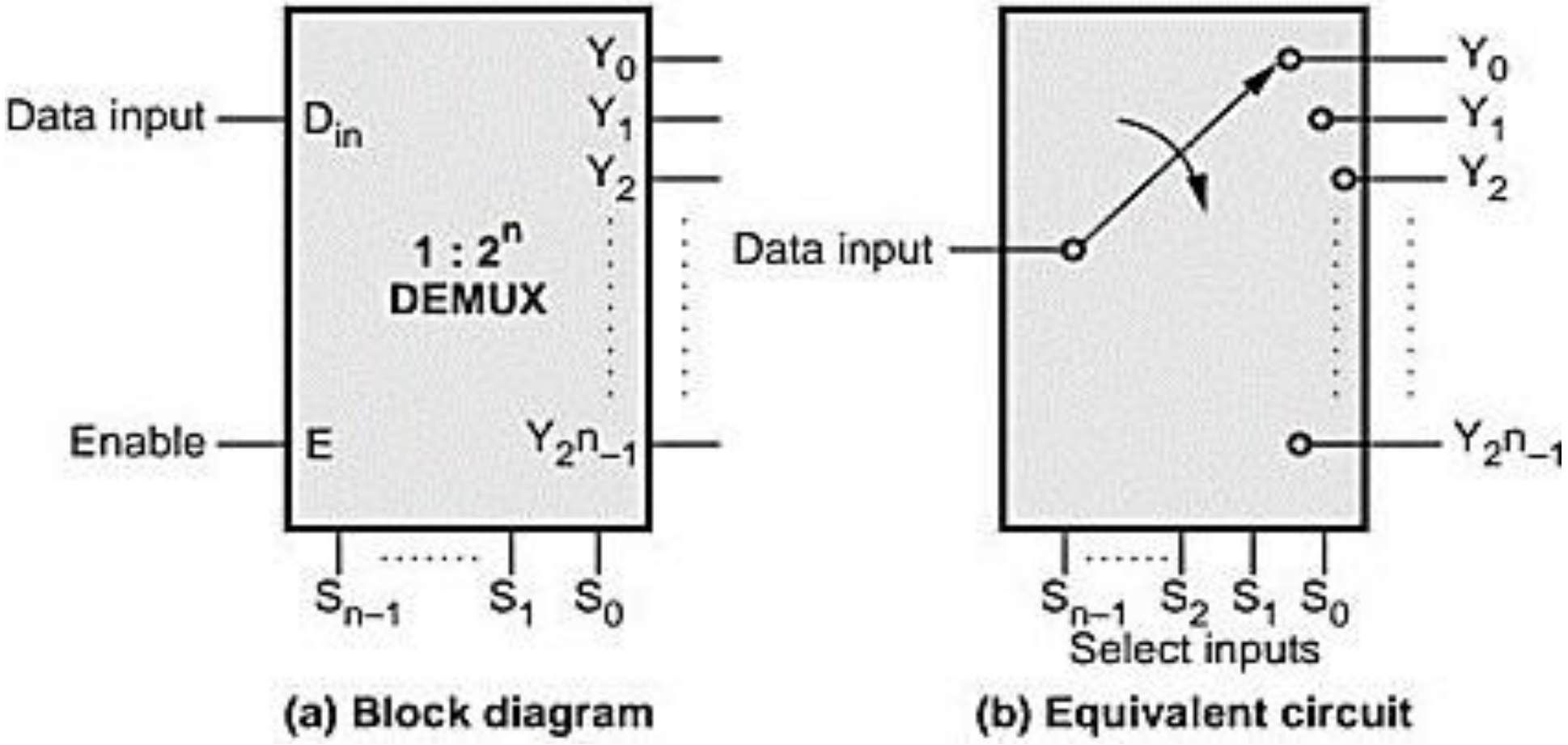
a. When S = 0, the inverted S, that is 1 gets applied as second input to first AND gate. Since S is applied directly as input to second AND gate; its output goes zero irrespective of first input. Since the second input of first AND gate is 1, its output is equal to its first input, that is D_0. Hence $Y = D_0$.

b. Exactly opposite is the case when S = 1. In this case, second AND gate output is equal to its first input D_1 and first AND gate output is 0. Hence $Y = D_1$. Both these cases are summarized in truth table shown in Fig. 5.1.2 (b).

Combinational Logic Circuits

Demultiplexer (One to many)

It is a circuit that receives information on a single line and transmits this information on one of 2^n possible output lines.



Combinational Logic Circuits

Demultiplexer

Example (2)

Design an 1 to 4 demultiplexer circuit

Solution

Number of inputs = 1 , number of outputs = 4

Number of control bits = $\log_2 4 = 2$

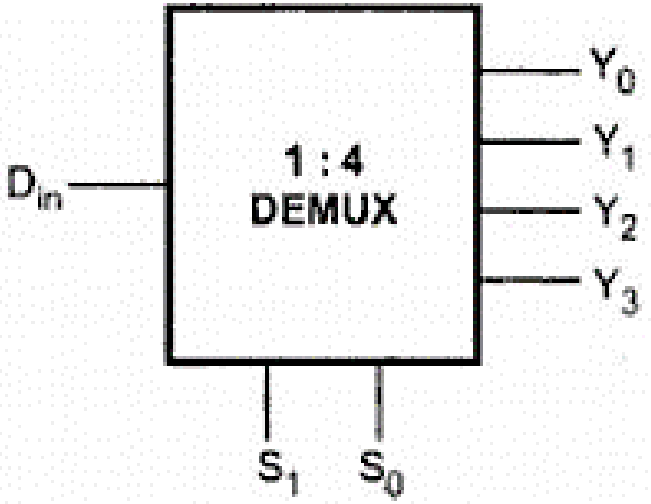
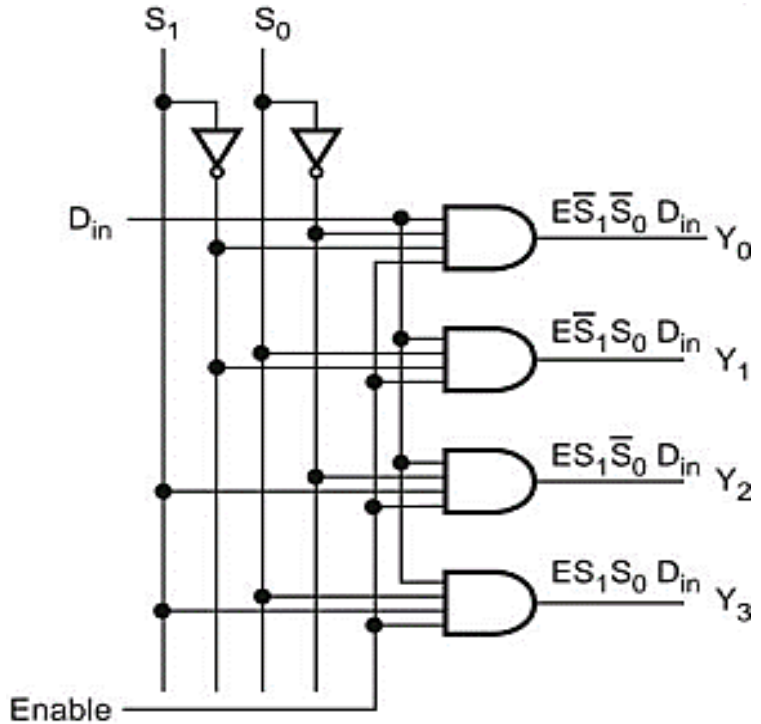


Fig. 2.17 (b) Logic symbol

Enable (E)	S_1	S_0	D_{in}	Y_0	Y_1	Y_2	Y_3	
0	X	X	X	0	0	0	0	
1	0	0	0	0	0	0	0	D_{in} is routed to Y_0 : $Y_0 = E\,\overline{S}_1\,\overline{S}_0\,D_{in}$
1	0	0	1	1	0	0	0	
1	0	1	0	0	0	0	0	D_{in} is routed to Y_1 : $Y_1 = E\,\overline{S}_1\,S_0\,D_{in}$
1	0	1	1	0	1	0	0	
1	1	0	0	0	0	0	0	D_{in} is routed to Y_2 : $Y_2 = E\,S_1\,\overline{S}_0\,D_{in}$
1	1	0	1	0	0	1	0	
1	1	1	0	0	0	0	0	D_{in} is routed to Y_3 : $Y_3 = E\,S_1\,S_0\,D_{in}$
1	1	1	1	0	0	0	1	

(b) Function table for 1 : 4 demultiplexer



(a) Logic diagram

Contents

- Number Systems and Coding
- Boolean Algebra
- Combinational Logic Circuits
- Multiplexers and Decoders
- Sequential Logic Circuits
- Registers
- Counters

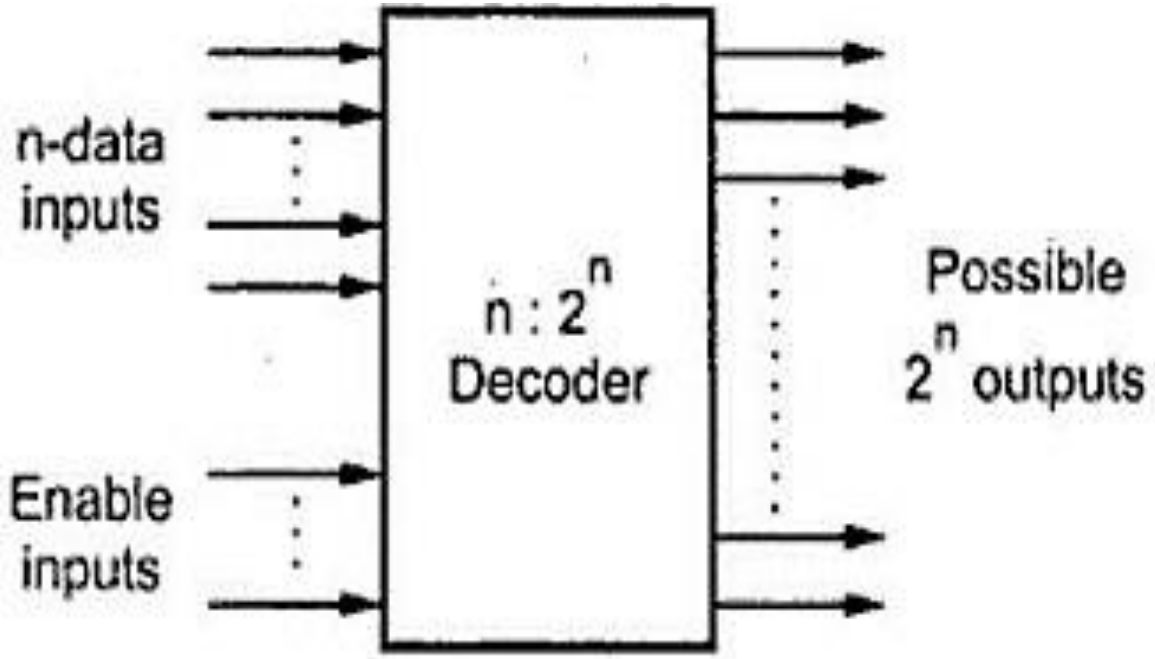
Combinational Logic Circuits

Decoder (many to many)

It is a multiple-input, multiple-output logic circuit used to convert coded inputs into coded outputs where the input and output codes are different.

Number of inputs = number of control bits = n

Number of outputs = 2^n



Combinational Logic Circuits

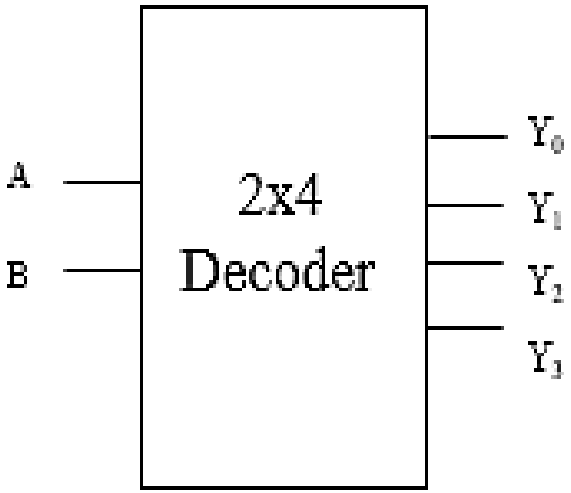
Decoder

Example (1)

Design an 2 to 4 decoder circuit

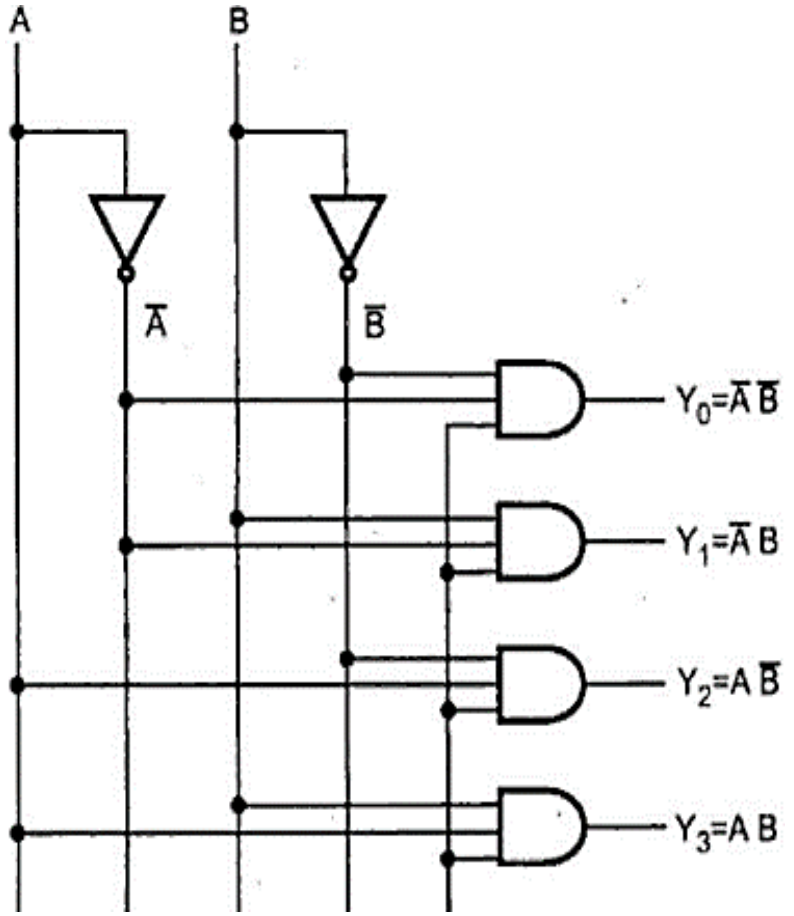
Solution

Number of inputs = number of control bits = n = 2



Inputs			Outputs			
EN	A	B	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Table 2.8 Truth table for a 2 to 4 decoder



2 to 4 decoder circuit

As shown in the truth table, if enable input is 1 (EN = 1), one, and only one, of the outputs Y_0 to Y_3, is active for a given input. The output Y_0 is active, i.e. $Y_0 = 1$ when inputs A = B = 0, the output Y_1 is active when inputs A = 0 and B = 1. If enable input is 0, i.e. EN = 0, then all the outputs are 0.

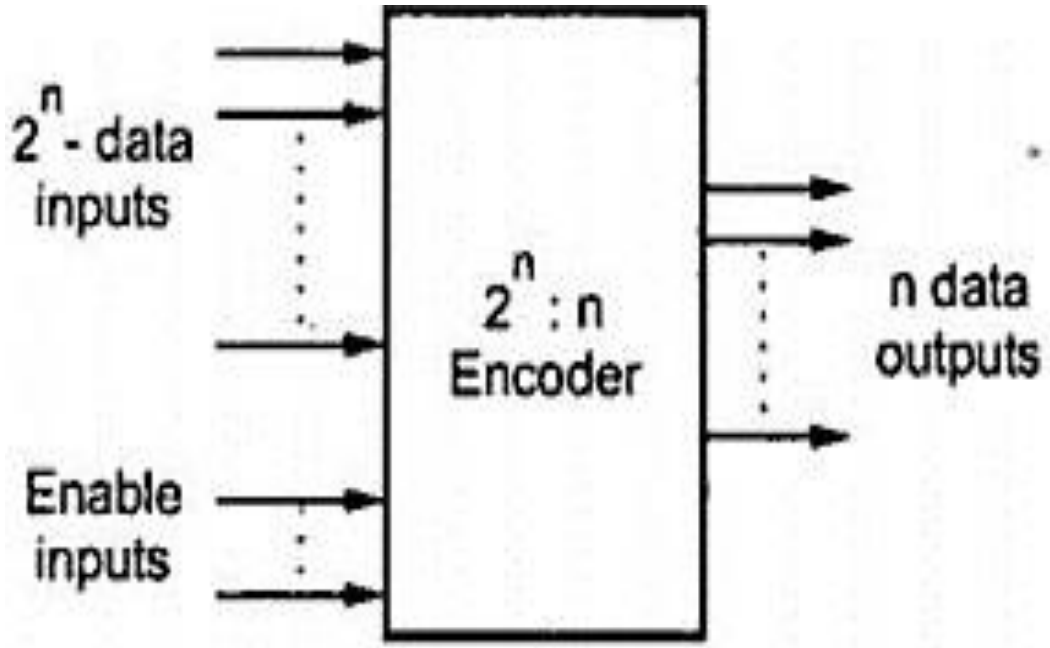
Encoder (many to many)

It is a digital circuit used to perform the inverse operation of the decoder. An encoder has 2^n (or fewer) input lines and n output lines.

In encoder, the output lines generate the binary code corresponding to the input value.

Number of outputs = number of control bits = n

Number of inputs = 2^n



Combinational Logic Circuits

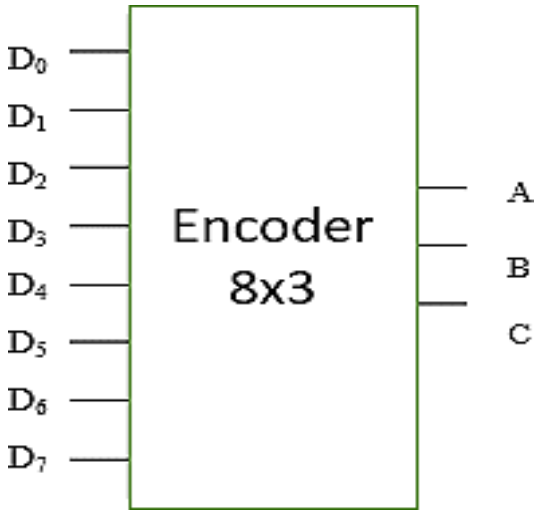
Encoder

Example (2)

Design an 8 to 3 encoder circuit

Solution

Number of inputs = number of control bits = n = 8

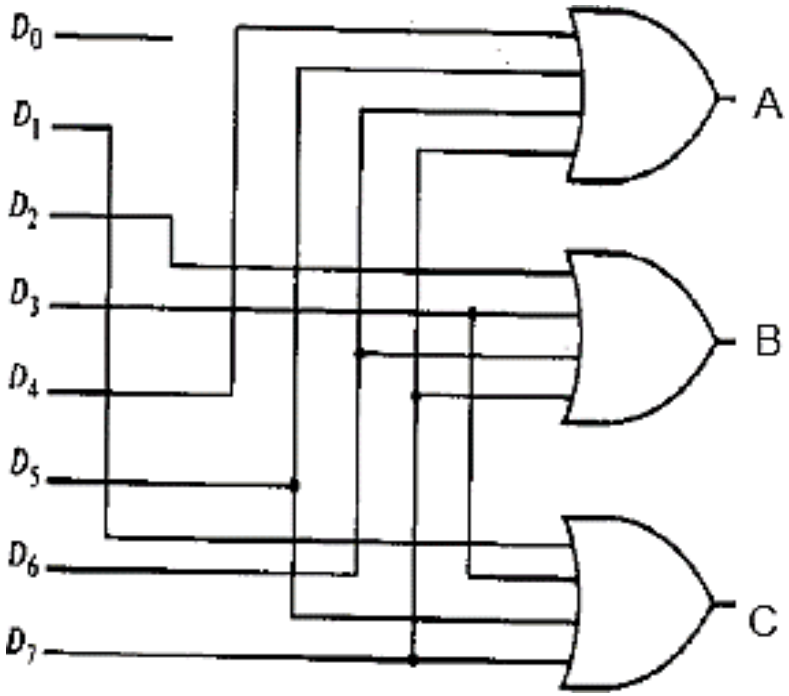


D0	D1	D2	D3	D4	D5	D6	D7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

$A = D_4 + D_5 + D_6 + D_7$

$B = D_2 + D_3 + D_6 + D_7$

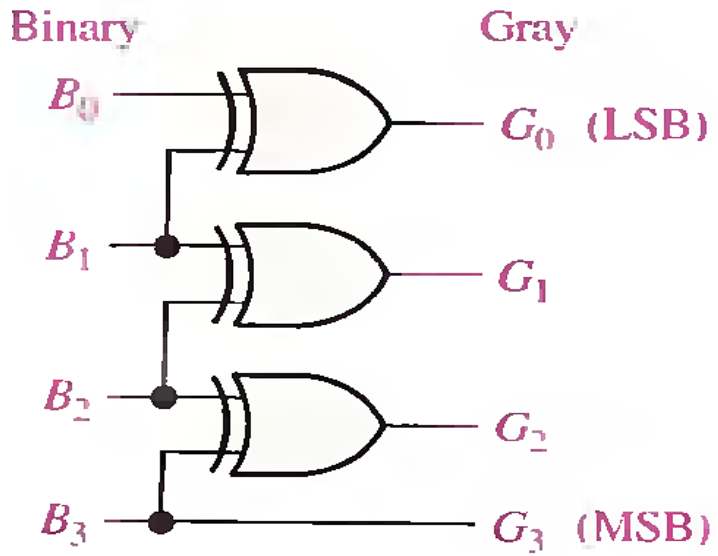
$C = D_1 + D_3 + D_5 + D_7$



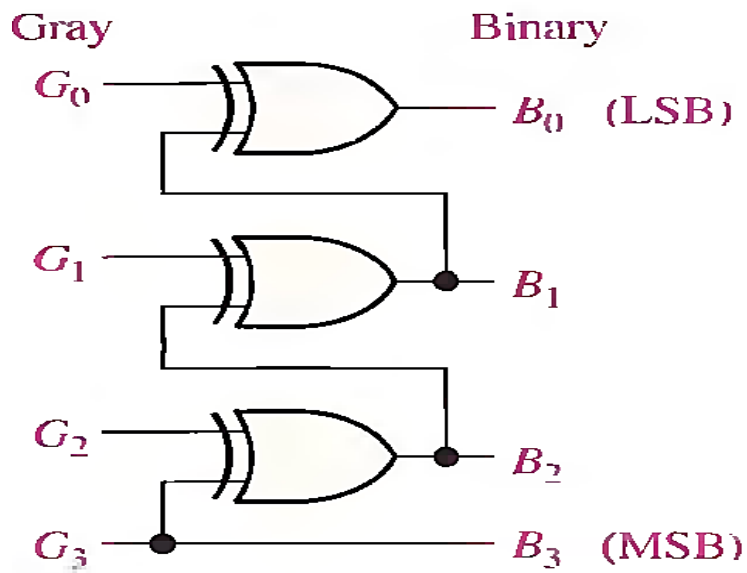
8 to 3 encoder circuit

Combinational Logic Circuits

Binary to Gray Converter



Gray to Binary Converter

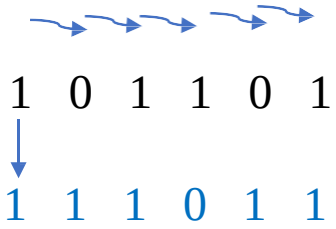


Combinational Logic Circuits

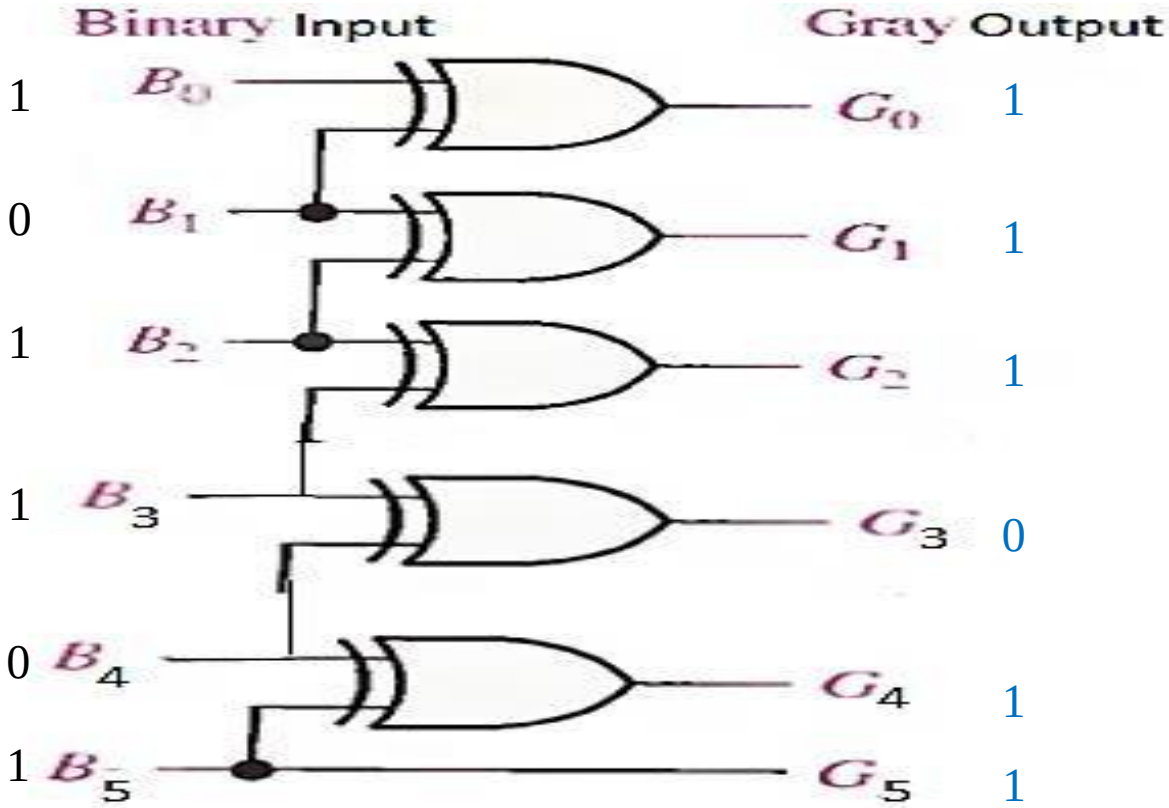
Example (3)

Design 6 bit binary $(101101)_b$ to gray

Solution



gray code is $(111011)_G$

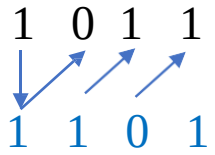


Combinational Logic Circuits

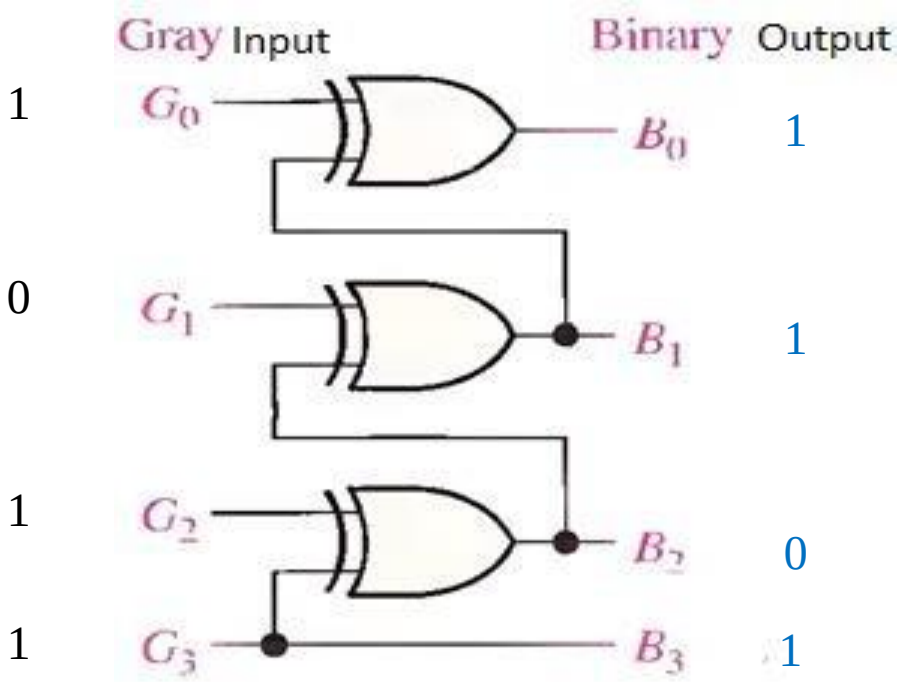
Example (4)

Design 4 bit gray $(1011)_G$ to binary

Solution



binary code is $(1101)_b$

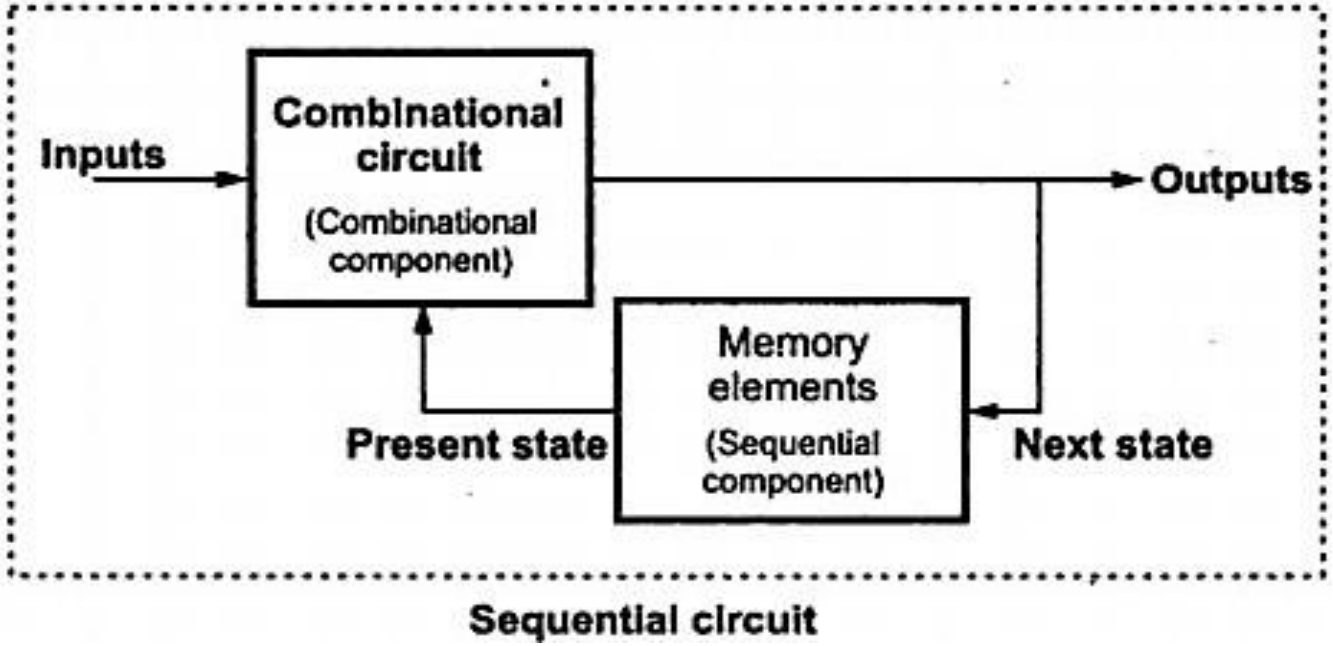


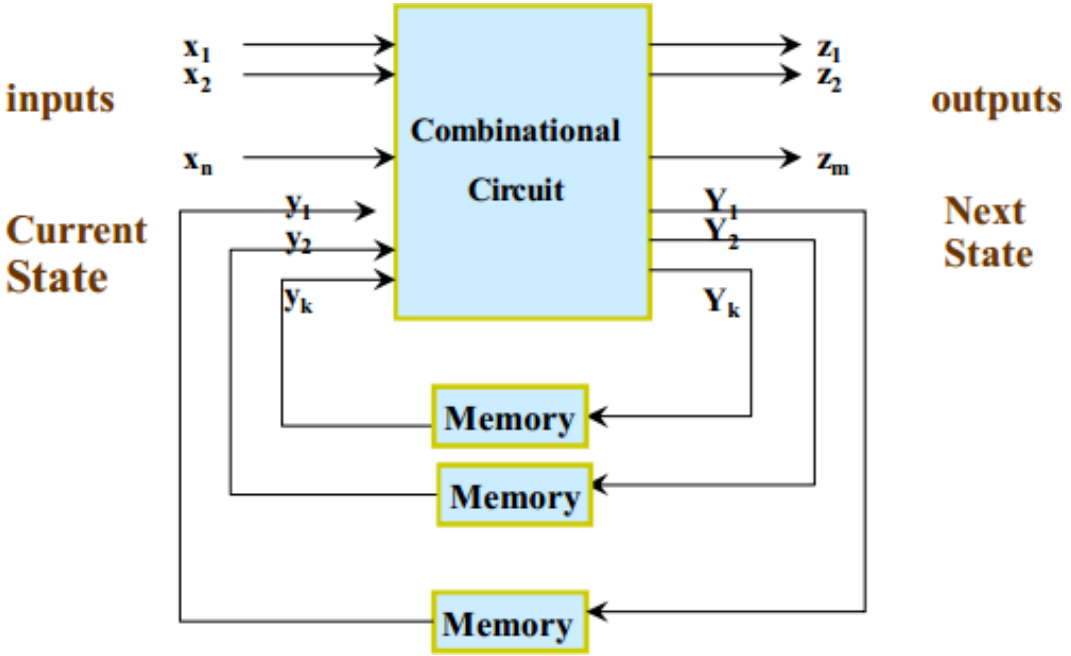
Sequential Logic Circuits

Digital circuits are divided into two categories combinational logic circuits and sequential circuits.

In combinational circuits, the output at any instance of time **depend on the inputs present** at this instance of time i.e., there is **no memory** in these circuits.

In sequential circuits, the output at a particular instant of time depend on the **present inputs and previous output** i.e., these are elements used to store past information. **These elements are known as memory**.

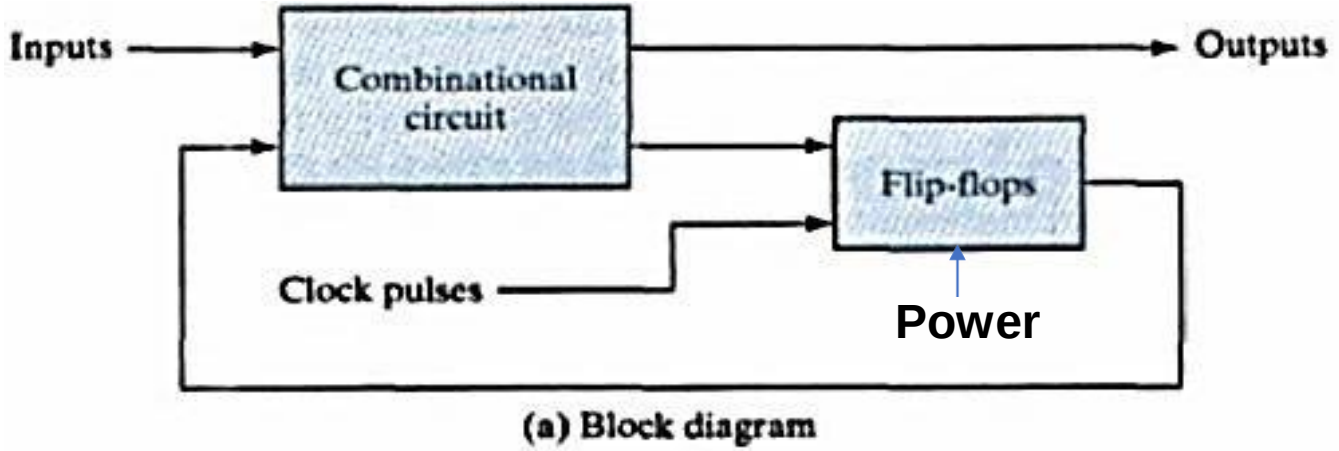




Sequential Logic Circuits

There are two types of sequential Logic circuits namely **Synchronous** and **Asynchronous**.

In synchronous sequential circuits, the memory elements are clocked elements such as Flip-Flop



(a) Block diagram

(b) Timing diagram of clock pulses

Flip-Flop and Latches : circuits used to store 1-bit of data " 0" or "1".

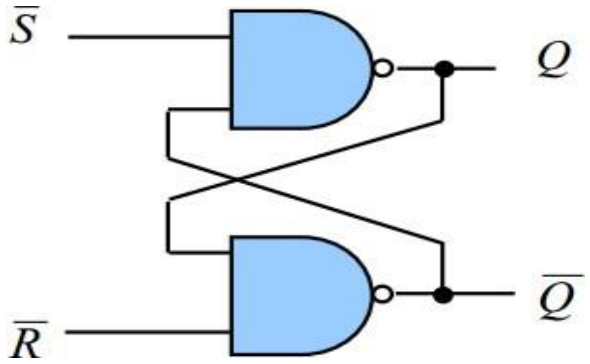
Flip-Flops and Latches can be designed using two NOR gates or two NAND gates, currently, all gates of these circuits packaged in one digital IC.

Sequential Logic Circuits

Asynchronous Sequential Circuits

Latches

- SR Latch



المدخلات		الخرج	وضع التشغيل (Mode of Operation)
$\overline{S}$	$\overline{R}$	Q	
0	0	?	وضع الخطر أو الوضع غير المسموح به Invalid condition
0	1	1	الوضع الفعال Latch SETS
1	0	0	الوضع غير الفعال Latch RESETS
1	1	Q_0	وضع الإمساك (عدم التغير) No Change

Synchronous Sequential Circuits

Flip-Flops

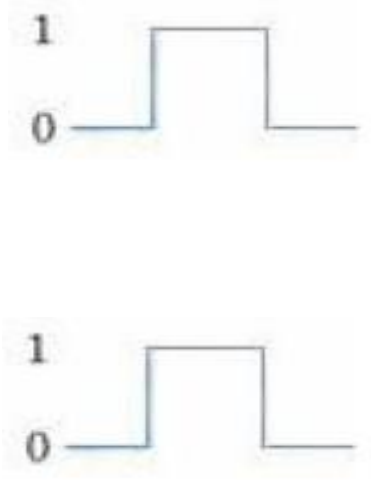
- Clocked SR Flip-Flop (control input of SR Flip-Flop)

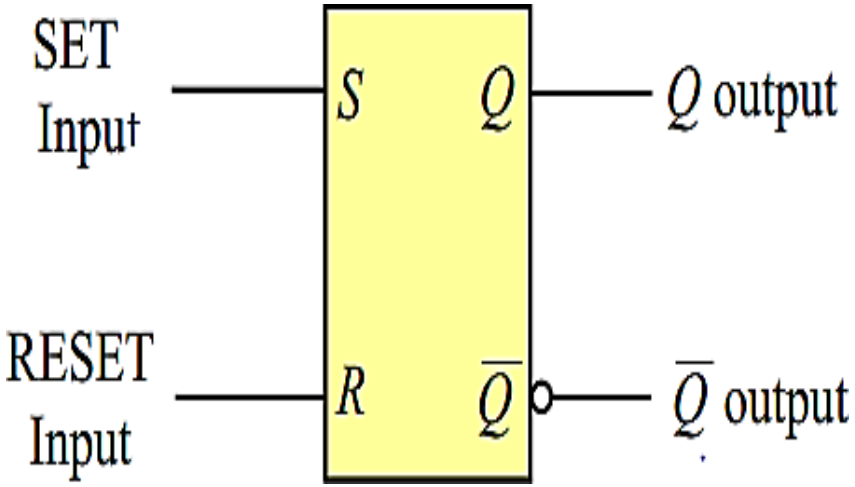
- D Flip-Flop

Sequential Logic Circuits

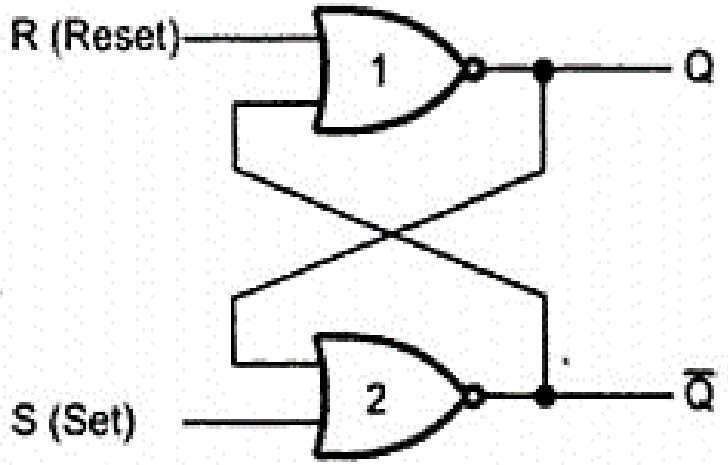
SR Latch

This circuit used to store one bit of data " 0" or "1".





Logic Symbol of
SR Latch Circuit



SR Latch Circuit implementation

SR Latch

Sequential Logic Circuits

Truth Table of SR Latch

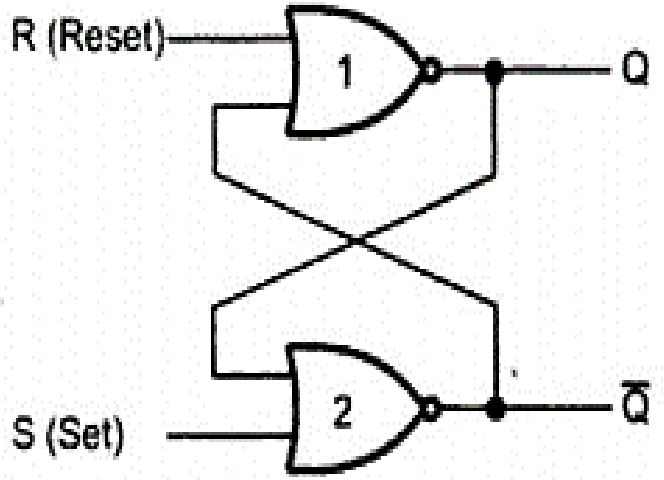
المداخل S	R	الخرج Q	¤	وضع التشغيل لدائرة الماسك	(Mode of Operation)
0	0	Q_0	¤	وضع الإمساك (عدم التغير)	No Change
0	1	0	1	وضع غير الفعال	Latch RESETS
1	0	1	0	وضع الفعال	Latch SETS
1	1	?	x	وضع الحظر أو وضع غير مسموح به	Invalid condition

Unknown output

X: prohibited (forbidden)

Q: Present State

$\overline{Q}$: Next State



SR Latch Circuit implementatio

So, we should not use this condition to avoid instability for the circuit when operate.

Operation of the SR Latch Circuit

Case 1 : if $S = R = 0$ at the same time, the state of the SR latch not change (the output values don't change).

Case 2 : if $S = 0$ and $R = 1$ Here, if R is changed to become 1 after occurring case 1, the output Q=0 “the latch saves 0” for temporary time.

Case 3 : $S = 1$ and $R = 0$ Here, if R is changed again to 0 with change S to 1 and after occurring case 2, the output Q=1.

Case 4 : $S = 1$ and $R = 1$ We don’t allow to use case 4, due to the output state unknown.

Sequential Logic Circuits

Latches

The output state “Q” changes with changing the inputs

Disadvantage of the Latch

This circuit causes a delay to transfer the data through a system

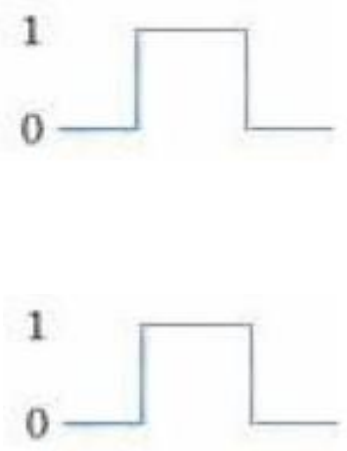
i. e. 1 0101010

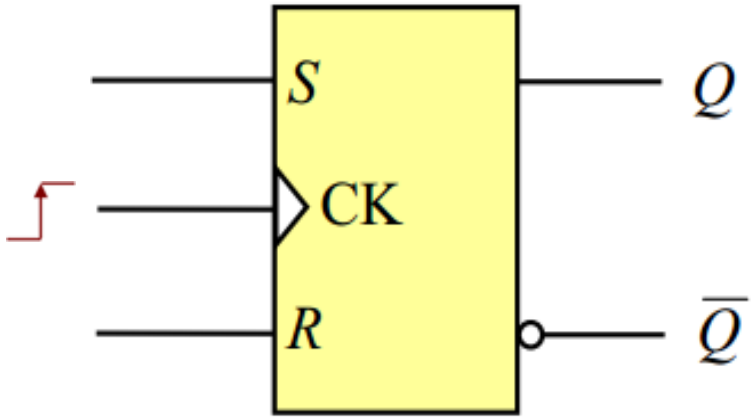
Sequential Logic Circuits

Clocked SR Flip-Flop

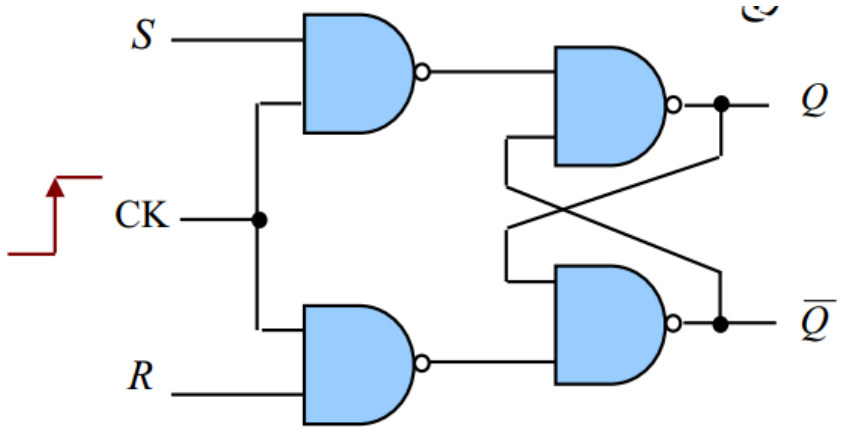
used to store one bit of data " 0" or "1".

It operates according to timing. This time represented by a clock pulse (CK).





Logic Symbol of Clocked SR Flip-Flop



Clocked SR Flip-Flop Circuit implementati

The external clock pulse is applied to the clock input of the Flip-Flop.

Two NAND gates are connected to the main Latch to get the **Synchronous** characteristic.

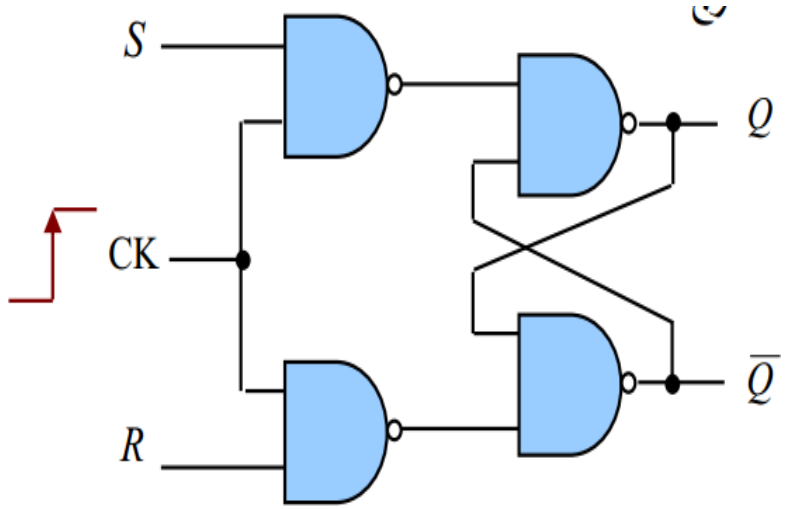
Clocked SR Flip-Flop

Sequential Logic Circuits

Truth Table of SR Flip-Flop

المداخل			الخرج	وضع التشغيل	(Mode of Operation)
S	R	CK	Q		
0	0	X	Q_0	وضع الإمساك (عدم التغير)	No Change
0	1	↑	0	الوضع غير الفعال	Latch RESETS
1	0	↑	1	الوضع الفعال	Latch SETS
1	1	↑	?	وضع الخطر أو وضع غير مسموح به	Invalid condition

Unknown output



Clocked SR Flip-Flop Circuit implementation

Operation of the Clocked SR Flip-Flop Circuit

Case 1 : if $S = R = 0$ Apply clock pulse, the output state remain as it is before reaching of the clock pulse “output don’t change”.

Case 2 : if $S = 0$ and $R = 1$ and clock pulse jumps from 0 to 1, the output Q=0.

Case 3 : $S = 1$ and $R = 0$ and clock pulse jumps from 0 to 1, the output Q=1.

Case 4 : $S = 1$ and $R = 1$ We don’t use this case due to the output state unknown.

Sequential Logic Circuits

Example (1)

Draw the timing diagram for the output Q of the SR Flip-Flop, if input pulses for S, R, CK in Fig. 1. Let that the Flip-Flop gives output Q= 0 before arriving the first clock pulse.

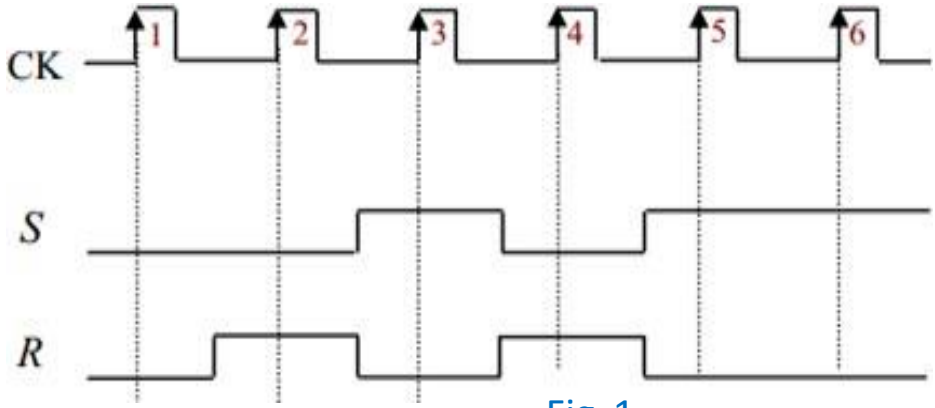


Fig. 1

Solution

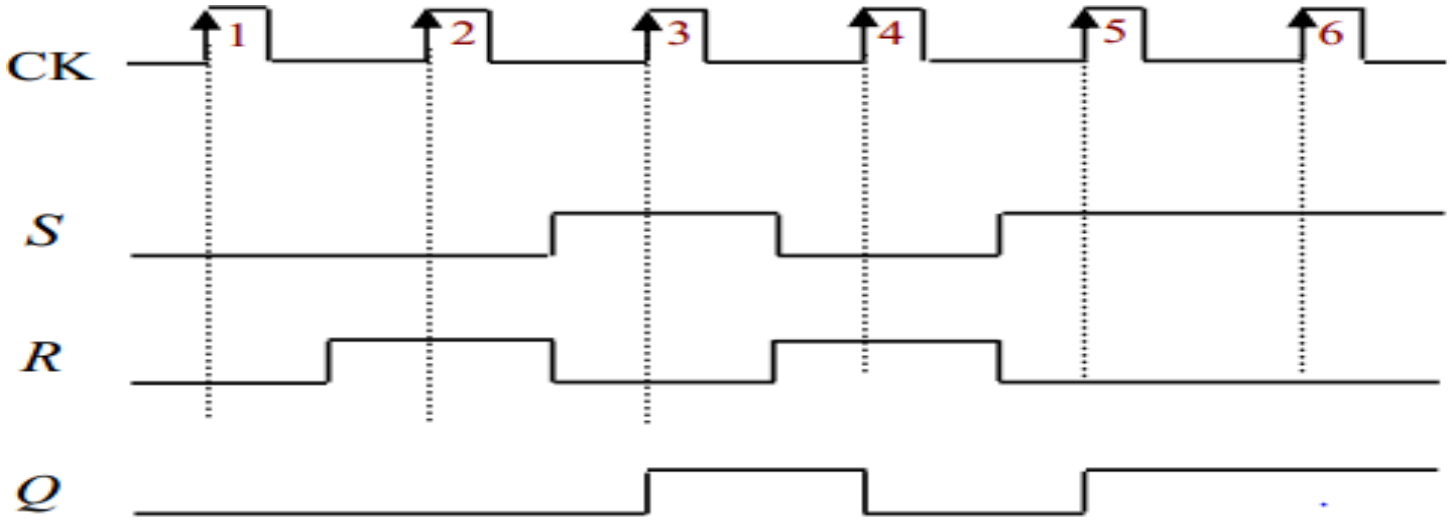
1-At arriving the first clock pulse, S=0, R=0, therefore, the output doesn't change Q=0

2- At arriving the second clock pulse, S=0, R=1, therefore, the output remains as it is Q=0 (Reset) 3- At arriving the third clock pulse, S=1, R=0, therefore, the output changes to Q=1 (Set)

4- At arriving the fourth clock pulse, S=0, R=1, therefore, the output Q=0 (Reset) 5- At arriving the fifth clock pulse, S=1, R=0, therefore, the output Q=1 (Set)

6- At arriving the six clock pulse, S=1, R=0, therefore, the output remains Q=1

المداخل			الخرج
S	R	CK	Q
0	0	X	Q_o
0	1	↑	0
1	0	↑	1
1	1	↑	?



Timing Diagram