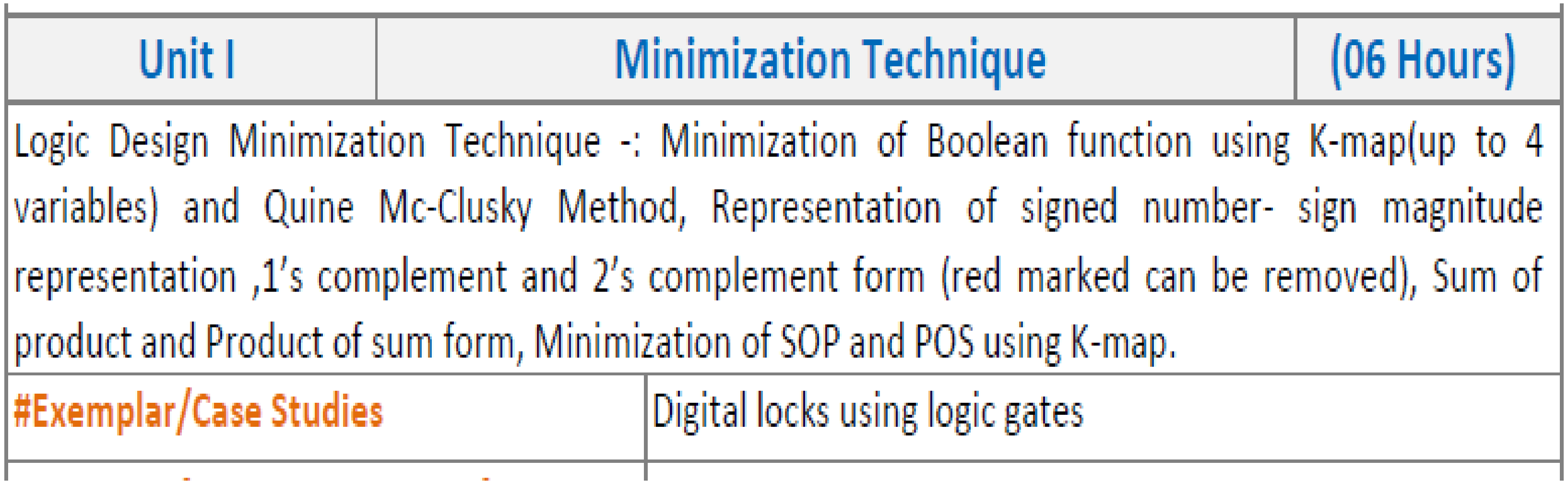
SE COMP (2019 Course) DIGITAL ELECTRONICS AND LOGIC DESIGN

TECHNIQUE TOPIC: SOP & POS FORM, 1'S & 2'S COMPLEMENT

Objectives

- To study the representation of logical function in SOP & POS form
- To find the sign & magnitude of number
- To find ones and twos complement



Unit I	Minimization Technique	(06 Hours)
Logic Design Minimization Technique -: Minimization of Boolean function using K-map(up to 4 variables) and Quine Mc-Clusky Method, Representation of signed number- sign magnitude representation ,1's complement and 2's complement form (red marked can be removed), Sum of product and Product of sum form, Minimization of SOP and POS using K-map.		
#Exemplar/Case Studies	Digital locks using logic gates	

Deriving Logical Expressions

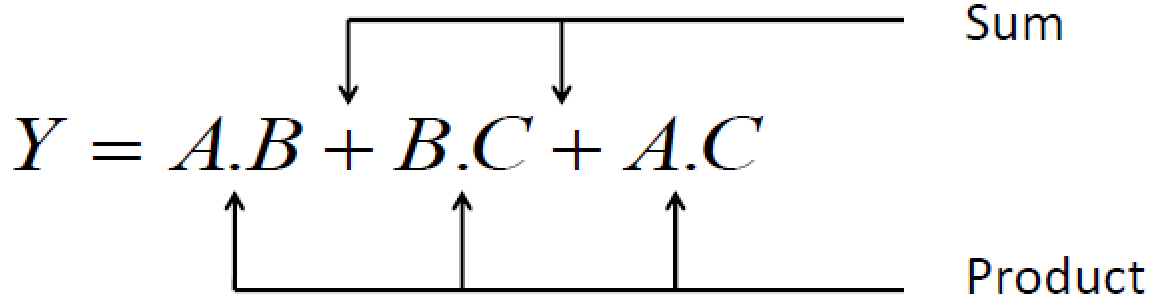
- By using Boolean laws and theorems, we can simplify the Boolean functions of digital circuits
- Sum-of-Products (SOP) Form
- Product-of-sums (POS) form
- Canonical forms
- There are two types of canonical forms:
- Sum-of-min terms or Canonical SOP
- Product-of- max terms or Canonical POS

Sum of Product (SOP) Form

- The sum-of-products (SOP) form is a method (or form) of simplifying the Boolean expressions of logic gates. In this SOP form of Boolean function representation, the variables are operated by **AND (product)** to form a product term and all these product terms are **ORed (summed or added)** together to get the final function
- Here the product terms are defined by using the AND operation and the sum term is defined by using OR operation.
- **Examples**
- AB + ABC + CDE
- $\overline{(AB)} + ABC + CD\overline{E}$

SOP Form:

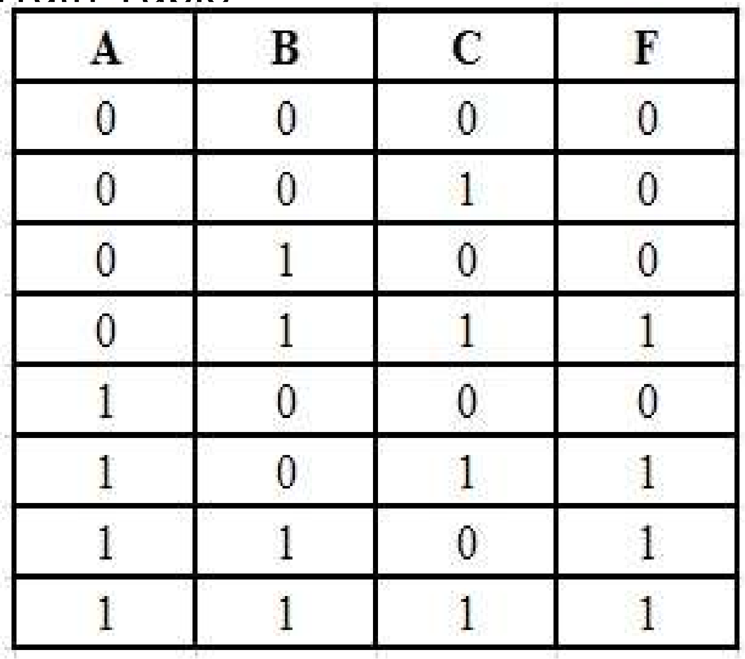
For Example, logical expression given is;



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

Ex: Boolean expression for majority function F = A’BC + AB’C + ABC ‘ + ABC

Truth Table-



A	B	C	F
0	0	0	0
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	1

Now write the input variables combination with high output. F = AB + BC + AC.

Checking

By Idempotence law, we know that

([ABC + ABC)] + ABC) = (ABC + ABC) = ABC

Now the function F = A’BC + AB’C + ABC **‘ + ABC**

= A’BC + AB’C + ABC’ + **([ABC + ABC)] + ABC)**

= (ABC + ABC ‘) + (ABC + AB’C) + (ABC + A’BC)

= AB (C + C ‘) + A (B + B’) C + (A + A’) BC

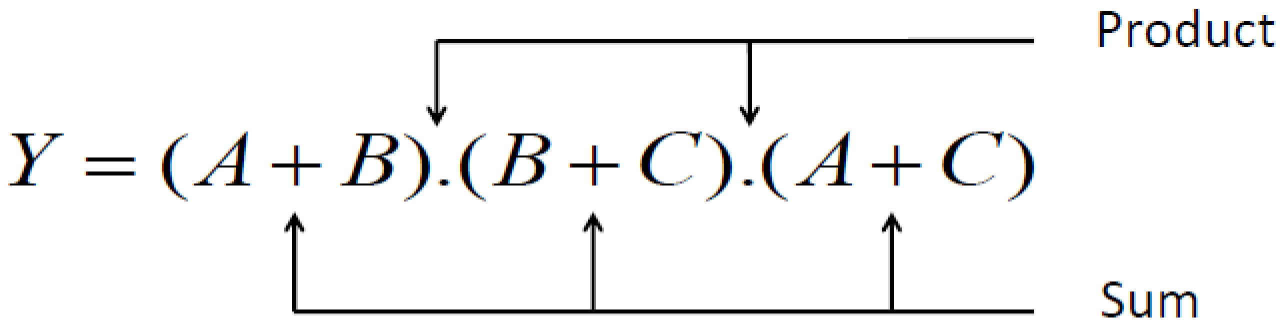
= AB + BC + AC.

POS Form:

- The product of sums form is a method (or form) of simplifying the Boolean expressions of logic gates. In this POS form, all the variables are ORed, i.e. written as sums to form sum terms.
- All these sum terms are ANDed (multiplied) together to get the product-of-sum form. This form is exactly opposite to the SOP form. So this can also be said as “Dual of SOP form”.
- **Examples**
- $(A+B) * (A + B + C) * (C + D)$

POS Form:

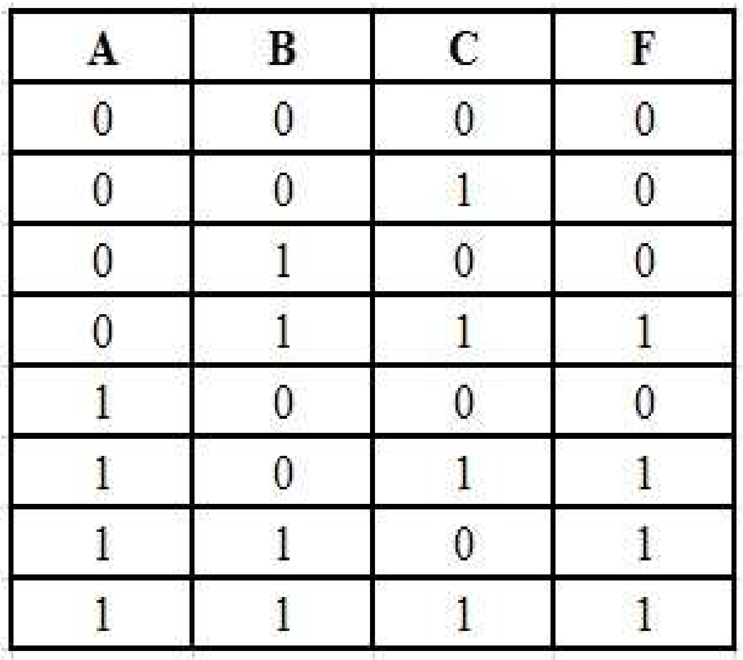
For Example, logical expression given is;



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

Boolean expression for majority function F = (A + B + C) (A + B + C ‘) (A + B’ + C) (A’ + B + C)

Truth Table-



A	B	C	F
0	0	0	0
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	1

Now write the input variables combination with high output. F = AB + BC + AC.

POS form can be obtained by

•Writing an OR term for each input combination, which produces LOW output.

•Writing the input variables if the value is 0, and write the complement of the variable if its value is 1.

•AND the OR terms to obtain the output function.

Checking

By Idempotence law, we know that

[(A + B + C) (A + B + C)] (A + B + C) = [(A + B + C)] (A + B + C) = (A + B + C)

Now the function

F = (A + B) (B + C) (A + C)

= (A + B + C) (A + B + C ‘) (A + B’ + C) (A’ + B + C)

= [(A + B + C) (A + B + C)] (A + B + C) (A + B + C ‘) (A + B’ + C) (A’ + B + C)

= [(A + B + C) (A + B + C ‘)] [(A + B + C) (A’ + B + C)] [(A + B + C) (A + B’ +C)]

= [(A + B) + (C * C ‘)] [(B + C) + (A * A’)] [(A + C) + (B * B’)]

= [(A + B) + 0] [(B + C) + 0] [(A + C) + 0] = (A + B) (B + C) (A + C)

Standard or Canonical SOP & POS Forms

✓ We can say that a logic expression is said to be in the standard (or canonical) SOP or POS form if each product term (for SOP) and sum term (for POS) consists of all the literals in their complemented or uncomplemented form.

Standard SOP

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

Each product term consists all the literals

Standard POS

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

Each sum term consists all the literals

Examples:

Sr. No.	Expression	Type
1	$Y = AB + AB\overline{C} + \overline{A}BC$	Non Standard SOP
2	$Y = AB + A\overline{B} + \overline{A}\overline{B}$	Standard SOP
3	$Y = (\overline{A} + B).(A + \overline{B}).(\overline{A} + \overline{B})$	Standard POS
4	$Y = (\overline{A} + B).(A + \overline{B} + C)$	Non Standard POS

Conversion of SOP form to Standard SOP

Procedure:

- 1.Write down all the terms.
- 2.If one or more variables are missing in any product term, expand the term by multiplying it with the sum of each one of the missing variable and its complement .
- 3.Drop out the redundant terms

Example 1

Convert given expression into its standard SOP form $Y = AB + A\overline{C} + BC$

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

Missing literal is A

Missing literal is B

Missing literal is C

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

Term formed by ORing of missing literal & its complement

Example 1 (Cont.)

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

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

$$Y = \underline{ABC} + \underline{AB\overline{C}} + \underline{AB\overline{C}} + A\overline{B}\,\overline{C} + \underline{ABC} + \overline{A}BC$$

$$Y = \underline{ABC + AB\overline{C} + A\overline{B}\,\overline{C} + \overline{A}BC}$$

Standard SOP form
Each product term consists all the literals

Example 2

- Convert the expression into standard form:

$$Y = A + BC + ABC$$

Conversion of POS form to Standard POS

Procedure:

- 1.Write down all the terms.
- 2.If one or more variables are missing in any sum term, expand the term by adding the products of each one of the missing variable and its complement .
- 3.Drop out the redundant terms

Example 3

Convert given expression into its standard POS Form

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

Missing literal is A

Missing literal is B

Missing literal is C

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

Term formed by ANDing of missing literal & its complement

Example 3 (Cont.)

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

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

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

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

Standard POS form
Each sum term consists all the literals

Example 4

- Convert the given expression into standard form

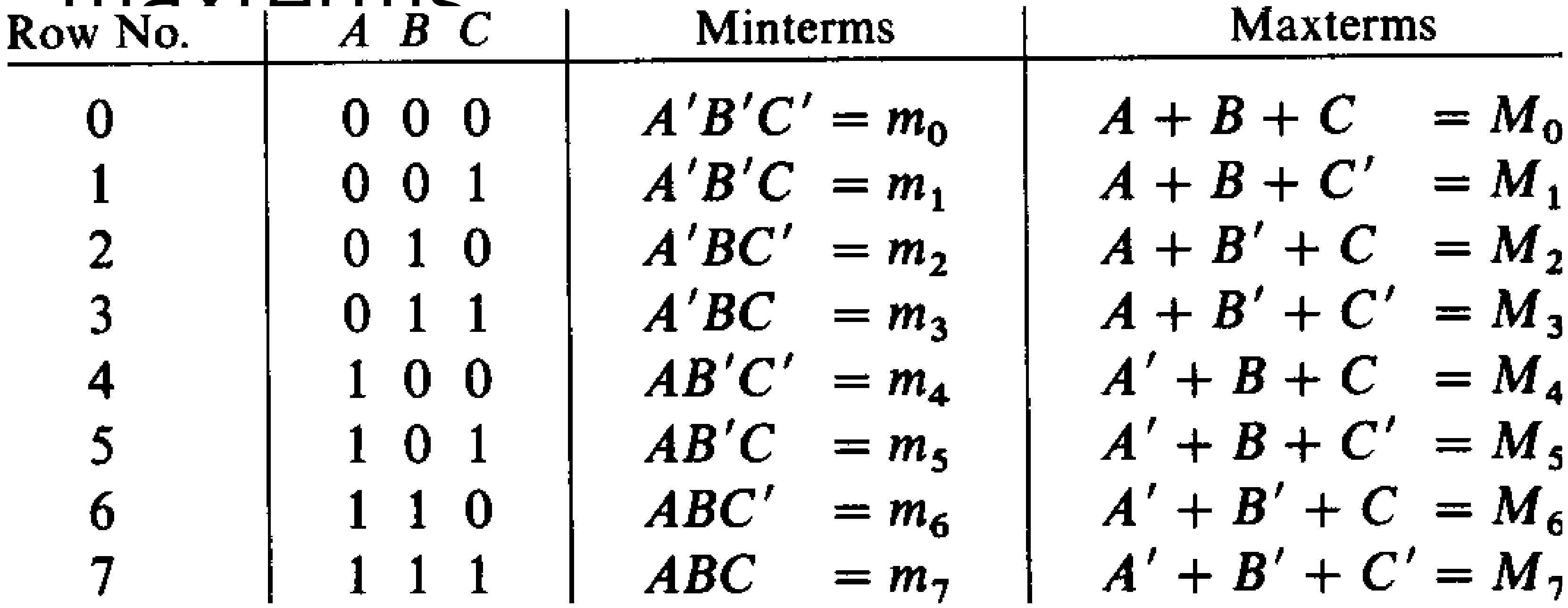
$$Y = (A + B).(A + \overline{C})$$

Concept of Minterm and Maxterm

- Minterm: Each individual term in the standard SOP form is called as “Minterm”.
- Maxterm: Each individual term in the standard POS form is called as “Maxterm”.

Minterms & Maxterms (Cont.)

- Truth tables, minterms, and maxterms



Row No.	A B C	Minterms	Maxterms
0	0 0 0	$A'B'C' = m_0$	$A + B + C = M_0$
1	0 0 1	$A'B'C = m_1$	$A + B + C' = M_1$
2	0 1 0	$A'BC' = m_2$	$A + B' + C = M_2$
3	0 1 1	$A'BC = m_3$	$A + B' + C' = M_3$
4	1 0 0	$AB'C' = m_4$	$A' + B + C = M_4$
5	1 0 1	$AB'C = m_5$	$A' + B + C' = M_5$
6	1 1 0	$ABC' = m_6$	$A' + B' + C = M_6$
7	1 1 1	$ABC = m_7$	$A' + B' + C' = M_7$

SOP **POS**

Minterms and maxterms for a 3-variable logic function,

Minter & Maxterms (Cont.)

- Each minterm is represented by m_i where i=0,1,2,3,.......,2^{n-1}
- Each maxterm is represented by M_i where i=0,1,2,3,.......,2^{n-1}
- If ‘n’ number of variables forms the function, then number of minterms or maxterms will be 2^n
- i.e. for 3 variables function f(A,B,C), the number of minterms or maxterms are $2^3=8$

Minter & Maxterms (Cont.)

- 3-input majority function

A	B	C	F
0	0	0	0
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	1

- SOP logical expression
- Four product terms
  - Because there are 4 rows with a 1 output

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

Minter & Maxterms (Cont.)

- 3-input majority function

A	B	C	F
0	0	0	0
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	1

- POS logical expression
- Four sum terms
  - Because there are 4 rows with a 0 output

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

Representation of Logical expression using minterm

$$Y = \underbrace{ABC}_{m_7} + \underbrace{\bar{A}BC}_{m_3} + \underbrace{A\bar{B}\bar{C}}_{m_4} + \underbrace{A\bar{B}C}_{m_5}$$

$\leftarrow$ **Logical Expression**

$\leftarrow$ **Corresponding minterms**

$$Y = m_7 + m_3 + m_4 + m_5$$

$$Y = \Sigma m(3,4,5,7)$$

OR

$$Y = f(A,B,C) = \Sigma m(3,4,5,7)$$

where $\sum$ denotes sum of products

Representation of Logical expression using maxterm

$$Y = \underbrace{(A+\overline{B}+C)}_{M_2}.\underbrace{(A+B+C)}_{M_0}.\underbrace{(\overline{A}+\overline{B}+C)}_{M_6}$$

← Logical Expression

← Corresponding maxterms

$$Y = M_2.M_0.M_6$$

$$Y = \Pi M(0,2,6)$$

OR

$$Y = f(A,B,C) = \Pi M(0,2,6)$$

where $\prod$ denotes product of sum

Conversion from SOP to POS & Vice versa

- The relationship between the expressions using minterms and maxterms is complementary.

- We can exploit this complementary relationship to write the expressions in terms of maxterms if the expression in terms of minterms is known and vice versa

Conversion from SOP to POS & Vice versa

- For example, if a SOP expression for 4 variable is given by,

$$Y = \Sigma m(0,1,3,5,6,7,11,12,15)$$

- Then we can get the equivalent POS expression using the complementary relationship as follows,

$$Y = \Pi M(2,4,8,9,10,13,14)$$

Implementation Using NAND Gates

- Using NAND gates
  - Get an equivalent expression

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

  - Using de Morgan's law

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

  - Can be generalized
    - Majority function

$$A\,B + B\,C + AC = \overline{\overline{A\,B} \cdot \overline{BC} \cdot \overline{AC}}$$

Idea: NAND Gates: Sum-of-Products, NOR Gates: Product-of-Sums

Signed Magnitude Representation

- Signed Magnitude (SM) is a method for encoding signed integers.
- The Most Significant Bit is used to represent the sign. ‘1’ is used for a ‘-’ (negative sign), a ‘0’ for a ‘+’ (positive sign).
- The format of a SM number in 8 bits is:

  Smmmmmmm

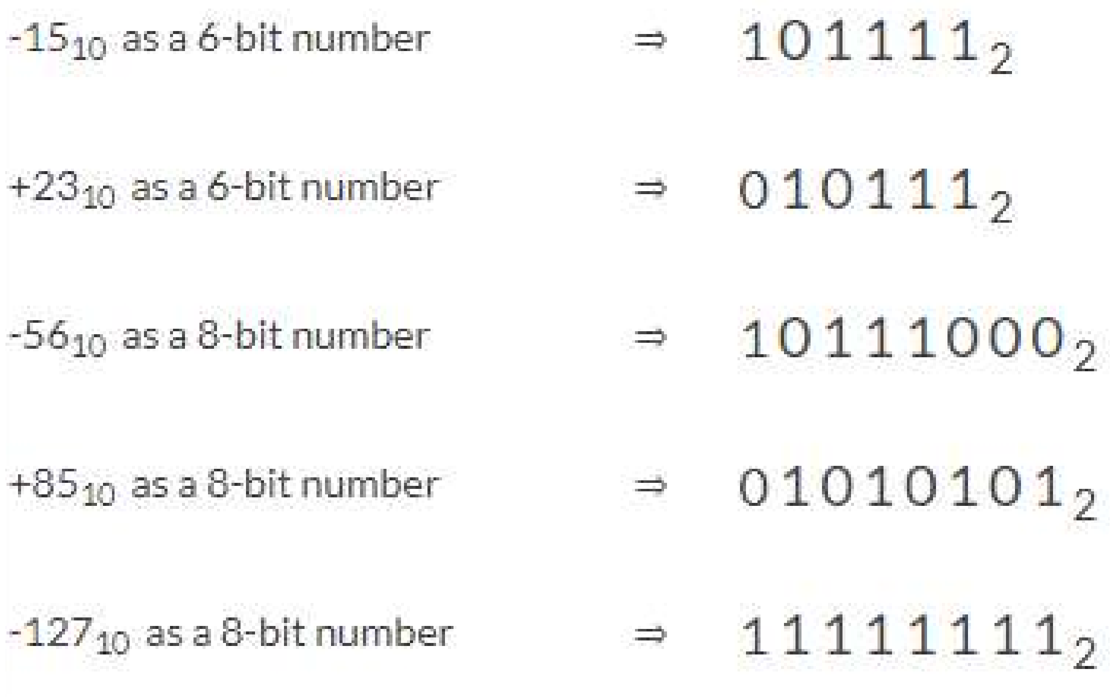
- where ‘s’ is the sign bit and the other 7 bits represent the magnitude.

Examples on sign-magnitude representation of signed numbers

+9 is represented as **0**1001

-9 is represented as **1**1001

…32 16 8 4 2 1



-15_{10} as a 6-bit number	⇒	101111_2
$+23_{10}$ as a 6-bit number	⇒	010111_2
-56_{10} as a 8-bit number	⇒	10111000_2
$+85_{10}$ as a 8-bit number	⇒	01010101_2
-127_{10} as a 8-bit number	⇒	11111111_2

Examples

- What are the decimal values of the following 8-bit sign-magnitude numbers?
  - 10000011 = -3
  - 00000101 = +5
  - 11111111 = ?
  - 01111111 = ?
- Represent the following in 8-bit sign-magnitude:
  - -15 = 10001111
  - +7 = 00000111
  - -1 = ?

1's Complement Representation

- Positive numbers are represented using normal binary equivalent while negative numbers are represented by the 1's complement (complement) of the normal binary representation of the magnitude.

  +9 is represented as 01001

  -9 is represented as 10110 (obtained by complementing the binary representation of 9).

  Important Note: Negative numbers will always have a 1 in the MSB while positive numbers will have a 0 in the MSB.

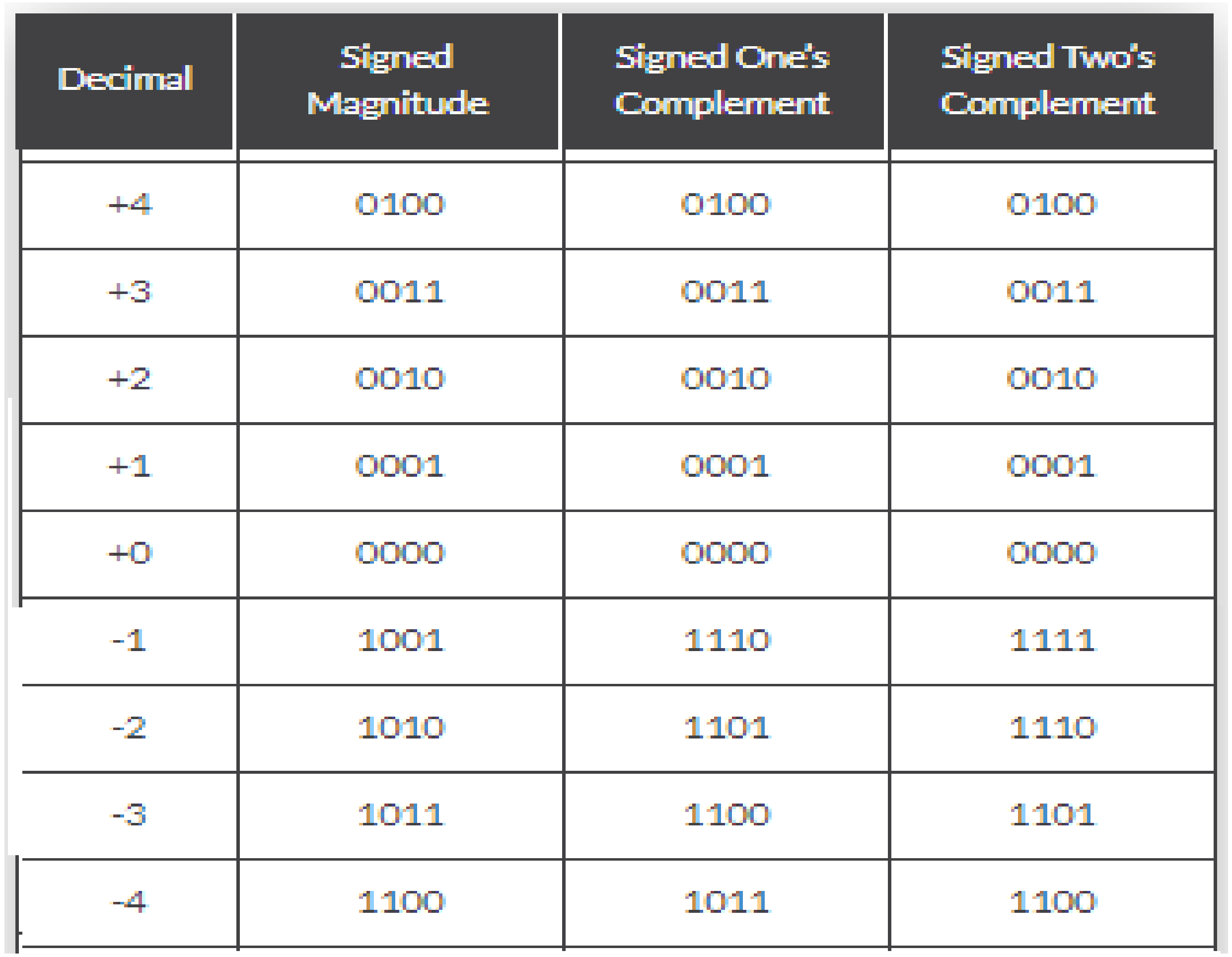
2's Complement Representation

- Positive numbers are represented using normal binary equivalent while negative numbers are represented by the 2's complement (complement) of the normal binary representation of the magnitude. The 2's complement of a binary number equals its 1's complement + 1.

  +9 is represented as 01001

  -9 is represented as 10111 (Now leave the LSB 1 unchanged and complement all the remaining bits).

Example:



Decimal	Signed Magnitude	Signed One's Complement	Signed Two's Complement
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100