

### 1.3 Complement:

Complements are used in digital computers to **simplify the subtraction operation** and for logical manipulation. There are two types of complements for each base- $r$  system: the radix complement and the diminished radix complement. The first is referred to as the  $r$ 's complement and the second as the  $(r - 1)$ 's complement.

#### 1.3.1 Binary numbers Complement:

##### 1- One's (first) Complement:

$$1's \text{ complement} = r^n - N - 1$$

where  $n$  : number of bits

$N$ : binary number

$r$  : system base

Simply the 1's complement of binary number is the number we get by changing each bit (0 to 1) and (1 to 0).

**Example:** the first complement of  $(101100)_2$

**Solution:**

binary number	101100
1's complement	010011

##### 2- The Two's (second) Complement:

The equation is:

$$2's \text{ complement} = r^n - N$$

Simply the 2's complement is equal to 1's complement added by one.

**Example:** find the 2's complement of  $(101101)_2$

**Solution:**

binary number	101101
1's complement	010010
2's complement	$010010 + 1 = 010011$

### 1.4 Binary Arithmetic Operations

#### 1- Addition:-

$$\begin{array}{r}
 0 + 0 = 0 \\
 0 + 1 = 1 \\
 1 + 0 = 1 \\
 1 + 1 = 0 \quad \text{carry 1}
 \end{array}$$

**Example:** Add the two binary numbers (001) and (100)

$$\begin{array}{r} 001 \\ + 100 \\ \hline 101 \end{array}$$

**Example:** Add the two binary numbers (111) and (001)

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

## 2- Subtraction:-

$$\begin{array}{l} 0 - 0 = 0 \\ 0 - 1 = 1 \quad \text{borrow 1} \\ 1 - 0 = 1 \\ 1 - 1 = 0 \end{array}$$

**Example:** subtract the binary number (100) from (101)

**Solution:**

$$\begin{array}{r} 101 \\ - 100 \\ \hline 001 \end{array}$$

**Example:** subtract the binary number (1101) from (1110)

**Solution:**

$$\begin{array}{r} \phantom{1} \phantom{1} \phantom{1} \phantom{1} \phantom{1} \\ \phantom{1} \phantom{1} \phantom{1} \phantom{1} \phantom{1} \\ - \phantom{1} \phantom{1} \phantom{1} \phantom{1} \phantom{1} \\ \hline \phantom{1} \phantom{1} \phantom{1} \phantom{1} \phantom{1} \end{array}$$

## Subtraction Using 1's Complement:

Add M to 1's complement of N ( subtracted ) and check the carry: If an end carry occur, add 1 to the least significant bit. And if an end carry does not occur, take the 1's complement of the number obtained in step 1 and place a negative sign in front.

**Example:** Given the two binary numbers  $X = 1010100$  and  $Y = 1000011$ , perform the subtraction (a)  $X - Y$  and (b)  $Y - X$  by using 1's complements.

**Solution:**

$$(a) X - Y = 1010100 - 1000011$$

$$\begin{array}{r} X = \quad 1010100 \\ 1\text{'s complement of } Y = + \quad 0111100 \\ \text{Sum} = \quad 10010000 \\ \text{End-around carry} = + \quad \underline{\quad 1} \\ \text{Answer: } X - Y = \quad 0010001 \end{array}$$

$$(b) Y - X = 1000011 - 1010100$$

$$\begin{array}{r} Y = \quad 1000011 \\ 1\text{'s complement of } X = + \quad 0101011 \\ \text{Sum} = \quad 1101110 \end{array}$$

There is no end carry. Therefore, the answer is  $Y - X = -(1\text{'s complement of } 1101110) = -0010001$ .

### Subtraction Using 2's Complement:

Apply the 2's complement to the subtracted N and then add it to M, if an end carry occur, discard it. If an end carry does not occur, apply 2's complement on the number that obtained in step 1.

**Example:** Given the two binary numbers  $X = 1010100$  and  $Y = 1000011$ , perform the subtraction (a)  $X - Y$  and (b)  $Y - X$  by using 2's complements

**Solution:**

$$(a) \begin{array}{r} X = \quad 1010100 \\ 2\text{'s complement of } Y = + \quad 0111101 \\ \text{Sum} = \quad 10010001 \\ \text{Discard end carry } 2^7 = - \quad \underline{10000000} \\ \text{Answer: } X - Y = \quad 0010001 \end{array}$$

$$(b) \begin{array}{r} Y = \quad 1000011 \\ 2\text{'s complement of } X = + \quad 0101100 \\ \text{Sum} = \quad 1101111 \end{array}$$

There is no end carry. Therefore, the answer is  $Y - X = -(2\text{'s complement of } 1101111) = -0010001$ .

### 3- Multiplication:

$$\begin{array}{l} 0 \times 0 = 0 \\ 0 \times 1 = 0 \\ 1 \times 0 = 0 \\ 1 \times 1 = 1 \end{array}$$

**Example:** Multiply the two binary numbers  $(111)_2$  and  $(101)_2$ .

$$\begin{array}{r} \phantom{\times} 111 \\ \times \phantom{00} 101 \\ \hline \phantom{+} 111 \\ + \phantom{00} 0000 \\ \hline \phantom{+} 11100 \\ \phantom{+} 100011 \end{array}$$

### 4- Division

Binary division is again similar to its decimal counterpart:

**Example:** divide the number  $(11011)$  on  $(101)$

$$\begin{array}{r} \phantom{101} 101 \\ \phantom{101} \overline{) 11001} \\ \underline{\phantom{101} 101} \phantom{00} \\ \phantom{101} 00101 \\ \phantom{101} \underline{\phantom{101} 101} \phantom{00} \\ \phantom{101} \phantom{00} 000 \end{array}$$

## 1.5 Binary Codes

The electronic digital systems like computers, microprocessors etc., are required to process data which may include numbers, alphabets or special characters. The binary system of representation is the most extensively used one in digital systems i.e, digital data is represented, stored and processed as group of binary digits (bits). Hence the numerals, alphabets, special characters and control functions are to be converted into binary format. The process of conversion into binary format is known as binary coding. Several binary codes have developed over the years. Some of them are discussed in this section.

1. Binary coded decimal (BCD).
2. Gray code.
3. ASCII code

### 1- Binary Coded Decimal (BCD)

Internally, digital computers operate on binary numbers. When interfacing to humans, digital processors, e.g. pocket calculators, communication is decimal-based.

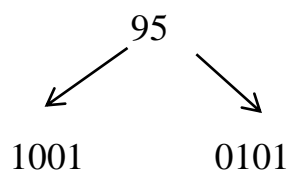
Input is done in decimal then converted to binary for internal processing. For output, the result has to be converted from its internal binary representation to a decimal form.

One commonly used code is the *Binary Coded Decimal* (BCD) code which corresponds to the first 10 binary representations of the decimal digits 0-9. The BCD code requires 4 bits to represent the 10 decimal digits. Since 4 bits may have up to 16 different binary combinations, a total of 6 combinations will be unused

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

**Example:** Convert  $(95)_{10}$  into BCD code .

**Solution:**



### 2- Gray Code

The Gray code consists of 16 4-bit code words to represent the decimal Numbers 0 to 15. For Gray code, successive code words differ by only one bit from one to the next as shown in the table and further illustrated in the Figure.

#### Binary Number to Gray Code Conversion:

The procedures of conversion from binary to gray code are:

- 1- put down the MSB
- 2- start from the MSB, adding without carry each two adjacent bits

**Example:** convert the  $(10110)_2$  into gray code.

**Solution:**

1	-	+	→	0	-	+	→	1	-	+	→	1	-	+	→	0	Binary
↓				↓				↓				↓					
1				1				0				1					Gray

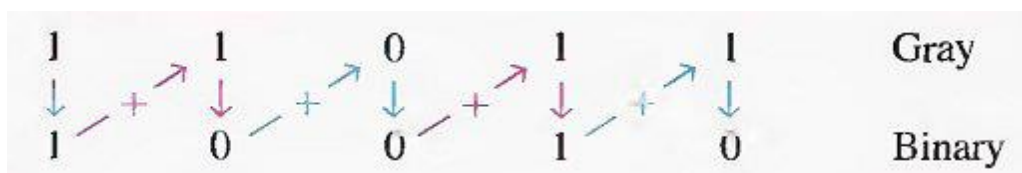
**Gray Code to Binary Number Conversion:**

The procedure of conversion from gray code to binary are:

- 1- put down the MSB
- 2- start from the MSB adding without carry each result binary bit with the lower gray code bit

**Example:** convert the  $(11011)_{\text{gray}}$  into binary.

**Solution:**

**3- ASCII Code**

American Standard Codes for Information Interchanging (ASCII) is the most widely used alphanumeric code. It is pronounced as 'ASKEE'. This is basically a 7-bit code and so, it has  $2^7 = 128$  possible code groups. The ASCII code can be used to encode both the lowercase and uppercase characters of the alphabet (52 symbols) and some special symbols as well, in addition to the 10 decimal digits. This code is used to exchange the information between input/output device and computers, and stored into the memory.

*American Standard Code for Information Interchange (ASCII)*

$b_4b_3b_2b_1$	$b_7b_6b_5$							
	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	-	o	DEL