CHAPTER 2: NUMBER SYSTEM AND BINARY ARITHMETIC

Introduction to Number System

Definition: Number system is the way to represent a number in different forms.

Types of Number system:

1. **Binary Number System:** It is the number system with base value 2 means it has only two digits to represent the data. The digits are (0, 1). E.g. 00,01,10,11,100….
2. **Decimal Number System:** It is the number system with base value 10 means it has 10-digits to represent the data. The digits are(0-9). Eg. 0,1,2,3,4,5,6 ………
3. **Octal Number System:** It is the number system with base value 8 means it has 8 digits to represent the data. The digits are (0-7).
4. **Hexadecimal Number System :** It is the number system with base value 16 means it has 16 digits to represent the data. The digits are (0-15). Eg. 0,1,2,3…….,9,A,B,C,D,E,F

Bits & Bytes:

1 bit(binary digit) = 1 **digit**. For example: 1

1 byte = 8-**bits**

1 kilo byte= 2^{10} = 1024 **bytes**

1 mega byte = $2^{10} * 2^{10} = 2^{20}$ = 1024 **kilo bytes**

1 giga byte= 2^{30}= 1024 **mega bytes**

1 tera byte= 2^{40}= 1024 **giga bytes**

Binary Number System:

In binary number system is made up of 2 digits- 0 and 1. We use these two digits to represent data.

0	Start at 0
1	Then 1
???	Then no other symbol

So we count in the same way as using decimal number system. For example:

Decimal number system start at 0 and then 1,2,3,4,5,6,7,8,9… now what after nine repeat the no in combination such as start at 0 again the add 1 to the left of 0 resultant 10 , 11 ,12… so on. 100, 1000 etc.

The same method we follow in Binary number system:

Decimal Number	Binary Value	Decimal Number	Binary Value
0	0	6	110
1	1	7	111
2	10	8	1000
3	11	9	1001
4	100	10	1010
5	101	11	1011

So confused!
how we get this?

Lets do a little Mathematics:

	2^5	2^4	2^3	2^2	2^1	2^0	=
	32	16	8	4	2	1	✓
0=	0	0	0	0	0	**0**	
1=	0	0	0	0	0	**1**	
2=	0	0	0	0	**1**	**0**	
3=	0	0	0	0	**1**	**1**	
4=	0	0	0	**1**	**0**	**0**	
5=	0	0	0	**1**	**0**	**1**	
6=	0	0	0	**1**	**1**	**0**	
7=	0	0	0	**1**	**1**	**1**	
8=	0	0	**1**	**0**	**0**	**0**	
9=	0	0	**1**	**0**	**0**	**1**	
10=	0	0	**1**	**0**	**1**	**0**	

Confuse again? Have a look on the bold letter 2, 1 is representing that 2

STOP? Are you able to create the binary counting

If no, Just look at the power values : $2^0, 2^1, 2^2, 2^3, 2^4 ....2^n$

$2^0 = 1$

$2^1 = 2$

$2^3 = 8$

2^4=16 so on…..

Lets calculate 2, we have only two digits **0** and **1.**

For 2 , I can write in front of 2^1 as its equal to 2.

Hmmm…..again what about 3? If I will add 2+1 =3 so there for 1 is assigned in front of the power of 2^0 and 2^1.
Still not get : Ok read this,

Decimal

Well how do we count in Decimal?	0	Start at 0
	...	Count 1,2,3,4,5,6,7,8, and then...
	9	This is the **last digit** in Decimal
	10	So we start back at 0 again, but add 1 on the left

The same thing is done in binary ...

	Binary	
	0	Start at 0
•	1	Then 1
••	10	Now start back at 0 again, but **add** 1 **on the left**
•••	11	1 more
••••	???	But NOW what ... ?

Other Number System

The octal system

In this system the base is eight. The allowed digits are 0 – 7 where as 8 is not allowed.

Typical number

N= $(4526.23)_8$

In polynomial term may be represented as follow

$(4\times8^3)+(5\times8^2)+(2\times8^1)+(6\times8^0)+(2\times8^{-1})+(3\times8^{-2})$ - POLYNOMIAL TERM

= $(2390.296875)_{10}$ = $(2390\ 19/64)_{10}$ - DECIMAL TERM

The hexadecimal

The base here is 16 and the allowed digits are 0 – 9 and A – F

A typical number

$N = (A1F.1C)_{16}$

In Polynomial may be represented as

$(A\times16^{2}) + (1\times16^{1}) + (F\times16^{0}) + (1\times16^{-1}) + (C\times16^{-2}) =$

$(10\times16^{2}) + (1\times16^{1}) + (15\times16^{0}) + (1\times16^{-1}) + (12\times16^{-2}) =$

$(2591\ 28/256)_{10} = 2591.109375_{10}$

Conversion between Number System

To convert a decimal number into binary number system. Follow the following Steps:-

Step-1: Divide the Number by 2 (as 2 is the base of the binary number system).

Step-2 : Collect the remainder.

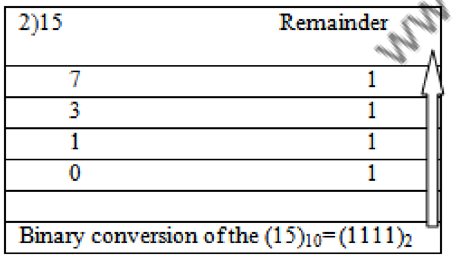
Step-3: Divide the quotient again with 2.

Step-4: Repeat the step 2 & 3 until the quotient is 0.

Step-5: Start from bottom, read the sequence of remainders upwards to the top.

Example-1

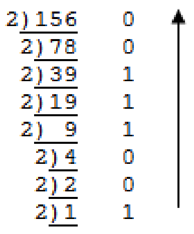
Convert $(15)_{10} = (?)_2$



2)15	Remainder
7	1
3	1
1	1
0	1

Binary conversion of the $(15)_{10} = (1111)_2$

Example-2: Convert $(156)_{10} = (?)_2$



2) 156	0
2) 78	0
2) 39	1
2) 19	1
2) 9	1
2) 4	0
2) 2	0
2) 1	1

Binary conversion of the $(156)_{10} = (0011100)_2$

Hexadecimal Number system:

This number system has a base value 16. We can use (0-15) decimal numbers to represent hexadecimal numbers. It uses 16 distinct numbers to represent the values.

Decimal numbers	Binary Number	Hexadecimal Number
0	00	0
1	01	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

ASCII (American Standard Code Interchange Information) Code: **ASCII** is a character set that is used to interchange information to binary and from binary to decimal. ASCII is a 8-bit character containing 256 characters.

0-31	Control code
32-127	Standard, alphabet A-Z and a-z
128-255	Special Symbols, non standard characters

Unicode: Unicode is a character set that is used to interchange information to binary language and from binary to decimal language. The latest version of Unicode is **Unicode 6.0**. It is a computing industry standard encoding scheme to represent a text.

To convert number from a non-decimal to decimal, we simply expand a given number as a polynomial and evaluate the polynomial using arithmetic as in the examples above.

When a decimal number is converted to any other system, the integer and the fraction potions of the number are handled differently. The radix divide technique is used to convert the integer portion and the radix multiply technique is used for the fraction portion.

Example

1. $(245)_{10}$ to binary

2	245		
2	122	1	LSB
2	61	0	
2	30	1	
2	15	0	
2	7	1	
2	3	1	
2	1	1	
	0	1	MSB

N = 11110101_2

2. 0.625_{10} to binary

0.625×2=	1.250	0.100	MSB
0.250×2=	0.500	0.000	
0.500×2=	1.000	0.001	LSB

N = 0.101_2

3. 245_{10} to hexadecimal

16	245		
16	5	15	MSB
2	0	5	LSB

$N = 5F_{16}$

4. $2AF_{16}$ to decimal
   $(2\times16^2)+(A\times16^1)+(F\times16^0) = (2\times16^2)+(10\times16^1)+(15\times16^0)$

   $= (512+160+15)_{10} = 687_{10}$

To convert hexadecimal to binary umber, simply replace each hexadecimal bit with its 4 bit equivalent binary bit

i.e. – 37_{16} = 00110111

- $C4_{16}$ = 11000100

To convert a binary number to its hexadecimal equivalent, simply group the binary bits at groups of 4. if necessary, may have to add 0's to complete the groups

Note

The leading zero that is added to complete the MSB assist us in making 4 bit binary group.

The grouping of a 4 bit binary number is referred as ***binary coded hexadecimal***

Introduction to Binary Arithmetic

Arithmetic circuits form point of the CPU. Mathematical operations include

Subtraction, multiplication, division and addition

Addition

a) Binary addition

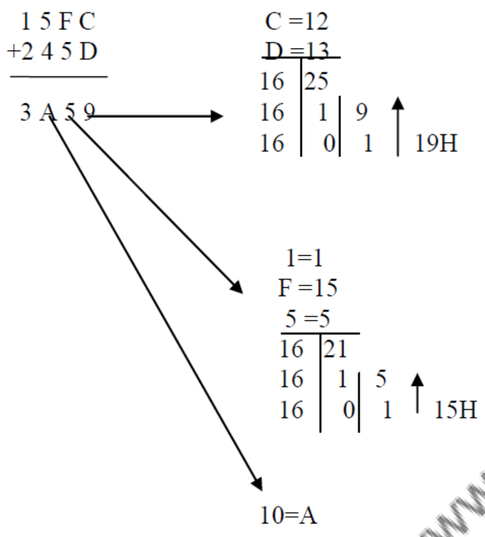
Binary addition takes in consideration of the following conditions

0+0=0, 0+1=1, 1+0=1,1+1=0

When adding larger numbers, the resulting ones are carried to other higher columns e.g.

$$\begin{array}{r} 101 \\ +010 \\ \hline 111 \end{array} \quad \text{and} \quad \begin{array}{r} 1010 \\ +0011 \\ \hline 1101 \end{array}$$

b) hexadecimal addition
Let's add



Alternatively ADD the binary equivalence of the hexadecimal numbers

1	5	F	C
0001	0101	1111	1100
2	4	5	D
0010	0100	0101	1101
3	A	5	9
0011	1010	0101	1001

Subtraction

a) Binary subtraction

In arithmetic subtraction, the initial numeric quotients that are combined by subtraction are the minuend and the subtrahend.. the result of the subtraction is called the difference

$$\begin{tikzcd} A \arrow[r] & \text{Minu-end} \\ -B \arrow[r] & \text{Subtra-end} \\ C \arrow[r] & \text{Difference} \end{tikzcd}$$

To subtract from a larger binary number, subtract column by column borrowing from adjacent columns when necessary.

Remember when borrowing from adjacent column, there are two digits

<u>Example</u>

1. subtract 1001 from 1101

$$\begin{array}{r} 1001 \\ 1101 \\ \hline 0100 \end{array}$$

2. subtract 0111 from 1011

$$\begin{array}{r} 0111 \\ 1011 \\ \hline 0100 \end{array}$$

When subtracting, 0 becomes 2

Binary numbers can also be -ve

The procedure for this calculation is identical to that of decimal numbers because the smaller value is subtracted from the larger value and the negative sign placed in front of the results

$$\begin{array}{rr} 111 & 4 \\ 100 & 7 \\ \hline 100 & -3 \end{array}$$

There are two other methods available for doing subtraction and representation of a negative number.

i. ones' compliment

The procedure for subtracting numbers using ones' compliment is as follows

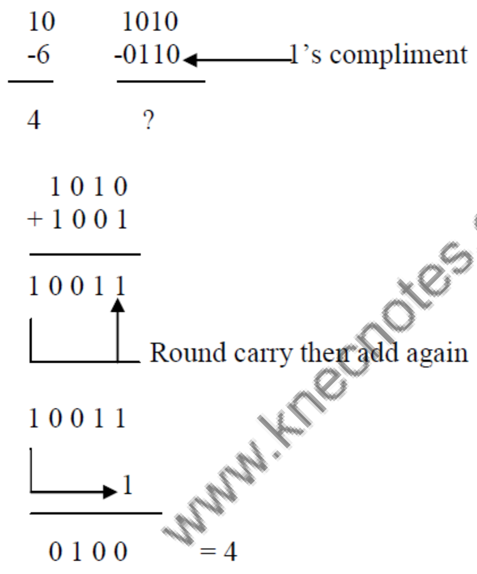
Step1-change the 0's of subtra-end to 1's and the 1's of subtra-end to 0's

step2-add the two numbers

Step3-remove the last carry and add it to the number

i.e. – end round carry

<u>Example</u>



When there is a carry in the end of the result, then we know the result is positive.

When there is no carry then we know the result is negative and we can now place a minus sign in front of the answer.

Example

01101 $\longrightarrow$ menuend

$\longrightarrow$

11011 $\longrightarrow$ subtraend

00100 1’s compliment of subtraend

___

01101 $\longrightarrow$

$\longrightarrow$

+ 00100

ii. two’s compliment

The general rule is that the two’s complement subtra-end is added to menu-end.

If the sign bit of the result is 0, then the result is the true difference and is assigned a positive sign.

If the sign bit is 1, the result is a two’s complement of the difference.

Any over flow produced by the calculation is lost.

Example

Subtract 11011 from 01101

Store 11011

Reverse bits 00100 1’s compliment

Add one + 1

00101 2’s compliment =

01101

+00101

___

10010

- 1

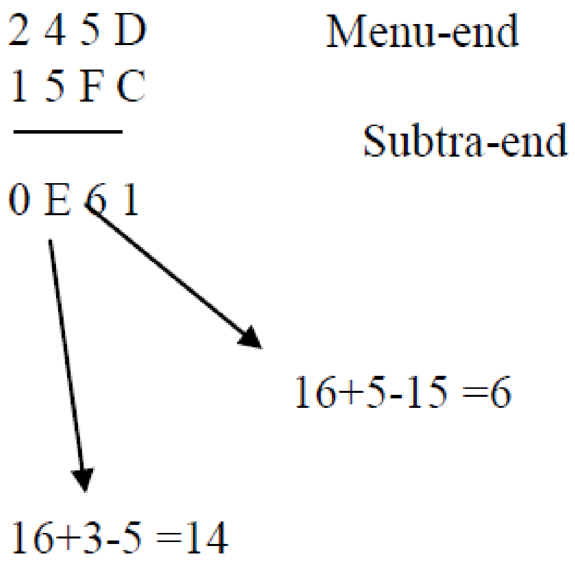
___

10001 $\longrightarrow$ reverse $\longrightarrow$ 01110 (14)

can now add a minus -01110 (-14)

b) Hexadecimal subtraction
Example

Subtract 15FCH from 245DH



Alternatively convert the decimal numbers to binary then subtract using the rules of binary subtraction.

Multiplication

c) Binary multiplication
Binary numbers are multiplied in the same manner as decimal numbers.

Rules

- 0×0=0, 0×1=0, 1×0=0, 1×1=1

- To multiply number with more than one digit, you form partial products and add them together.

$$\begin{array}{r} 5 \\ \times 6 \\ \hline 30 \end{array} \qquad \begin{array}{r} 101 \\ \times 110 \\ \hline 000 \\ 101 \\ 101 \\ \hline 11110 \end{array}$$

Computers cannot store partial products. The multiplication method used by the computer is **repeated additions**.

To determine 7×55, the computer can add 7, 55 times

A faster method of micro-processor system is the **add and shift method**

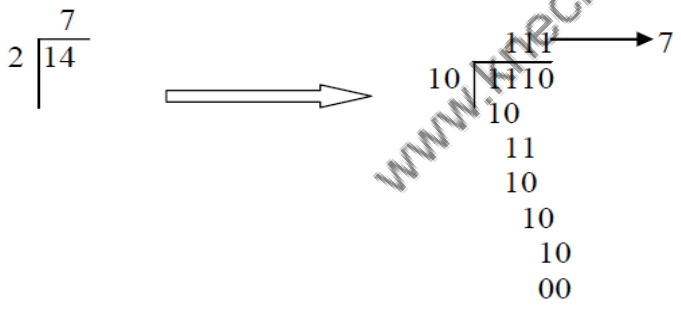
Division

d) Binary division

There are several methods of performing binary division. In the partial method also known as the restoring method, division is similar to decimal method

Example

Divide 14 by 2



Successive subtraction

Divisor is subtracted from the divided and from each successive remainder until a barrow is realized. The desired quotient is one less the number of subtraction needed to produce a borrow. This method is simple but slow for large numbers.

Representation of negative numbers

The examples shown so far have been using positive numbers. In practice, a digital system must represent all positive and negative numbers. To accommodate the sign of numbers, an additional digit known as the sign digit is included in the representation along with the magnitude digit. Thus to represent an n-digit number, we world need n+1 digit.

Typically, the sign of digit is the MSB

There are two ways to represent sign numbers

e) sign-magnitude system

in this representation, n+1 digit are used to represent a number where the MS digit is the sign digit and the remaining n-digit are the magnitude digit. The value of the sign digit is 0 for a positive number and r-1 for a negative number, where r is the radix of the number system.

Sign	Magnitude			Decimal
0	1	1	1	+7
0	1	1	0	+6
0	1	0	1	+5
0	1	0	0	+4
0	0	1	1	+3
0	0	1	0	+2
0	0	0	1	+1
0	0	0	0	0
1	0	0	1	-1
1	0	1	0	-2
1	0	1	1	-3
1	1	0	0	-4
1	1	0	1	-5
1	1	1	0	-6
1	1	1	1	-7

The sign and magnitude portions are handled separately in arithmetic using sign magnitude number

f) Compliment system

To compliment a binary number, change all 0's to 1's and all 1's to 0's. this is known as ones' compliment form of a binary number.

i.e. - 0110 = 1001 in compliment

The most common way to express a negative binary number is to show it as a two's compliment. A two's compliment is a binary number that shows when one is added to the first compliment

Signed Decimal	Ones' compliment	Two's compliment
+7	0111	0111
+6	0110	0110
+5	0101	0101
+4	0100	0100
+3	0011	0011
+2	0010	0010
+1	0001	0001
0	0000	0000
1	1110	1111
2	1101	1110
3	1100	1101
4	1011	1100
5	1010	1101
6	1001	1010
7	1000	1001

Using the two's compliment makes its easier for digital system to perform digital operation. The correct sign bit is generated by performing the two's compliment

Computer data formats/coding

Successful programming requires a precise data format.

Data appears in different forms as below

a) Alphanumeric codes

Codes have been developed to represent data as well as numbers and special symbols as '=', '&', etc

The codes are called alphanumeric codes