Udaya K. Dayan

Government ICT Professional

(Sri Lanka Information Communication Technology [SLICT] - Service)

G.C.E (A/L) - Information Communication Technology (ICT)

Number Systems

0714146866/0779911074
http://www.dayansir.edu.lk
dayan@dayansir.edu.lk

Number Systems

While the concept of number system was present in the 'Abacus ' considered as the first calculating machine of the world, it has progressed up to the computer of today.

The number system used for the representation of data in the computer is as follows;

Number System	Base Value	Number and Alphabetic character used
1. Binary	2	0, 1
2. Octal	8	0, 1, 2, 3, 4, 5, 6, 7
3. Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
4. Hexa - decimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Decimal number system

- This number system has 10 digits of 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
- When the value of a particular number exceeds the largest number 9 in that number set, the multiples of 10 of the number of values are transferred to the next (left) place value. Every place value is multiplied by ten to get the next place value.

  E.g. :- 3456 $= 3 \times 10^3 + 4 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$

  $= 3000 + 400 + 50 + 6$

  $= 3456$

  The place values in decimal number are multiple values of 10. Therefore the base value of the decimal number system is 10.

Binary number system

- The binary number system has two digits which can be represent two states.
- These two states are, represented by digits "0" and "1".
- Therefore a number system with the two digits can be used here.
- There are multiplications of 2 in place values of the binary number system. They are as follows.

2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	
16	8	4	2	1	1/2	1/4	1/8	Place values

- Therefore, the base value of the binary number system is 2.
- As the computer works on electricity and is an electronic device, its functions are controlled by two states.
- These two states are, where the power is ON and OFF (As two different levels of voltage)
- The every place value is multiply by 0 or 1 (digits of binary number system) to get the value of binary number.

E.g. :- 11010_2 $= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 1\times16+1\times8+0\times4+1\times2+0\times1$

$= 26_{10}$

Therefore, $11010_2 = 26_{10}$

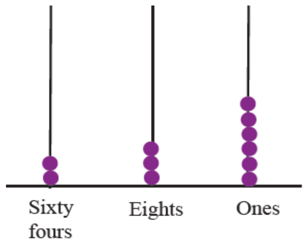
One location (one digits) is call a bit. There are 5 bits in the above number.

Octal number system

- The base value of the octal number system is 8.
- The digits are 0, 1, 2, 3, 4, 5, 6 and 7.

8^2	8^1	8^0	8^{-1}	8^{-2}
64	8	1	1/8	1/64

place values



E.g. :- $673_8 = 6 \times 8^2 + 7 \times 8^1 + 3 \times 8^0$

$= 6\text{x}64+7\text{x}8+3\text{x}1$

$= 443_{10}$

Therefore, $673_8 = 443_{10}$

Hexadecimal number system

- The base value of hexadecimal number system is16.
- There are 16 digits in the hexadecimal number system. Digits over value 9 need two digits. Therefore, A, B, C, D, E, F also used as remaining digits. All digits are as follows.

  Digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- The minimum value is 0 and maximum value is F ($=15_{10}$).
- The values represented by the digits are as follows

Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

E.g. :- $BC12_{16} = B(11) \times 16^3 + C(12) \times 16^2 + 1 \times 16^1 + 2 \times 16^0$

$= 11\times16^3 + 12\times16^2 + 1\times16^1 + 2\times16^0$

$= 11\text{x}4096 + 12\text{ x}256 + 1\text{x}16 + 2\text{x}1$

$= 45056 + 3072 + 16 + 2$

$= 48146$

Therefore, $BC12_{16} = 48146_{10}$

4096	256	16	1
16^3	16^2	16^1	16^0

Relationship among Decimal, Binary, Octal and Hexadecimal

Figure 3.8 - Relationship among Decimal, Binary, Octal and Hexadecimal

	Decimal	Binary	Octal	Hexadecimal	
	0	0	0	0	
2^0	1	1	1	1	8^0, 16^0
2^1	2	10	2	2	
	3	11	3	3	
	4	100	4	4	
	5	101	5	5	
	6	110	6	6	
	7	111	7	7	
2^3	8	1000	10	8	8^1
	9	1001	11	9	
	10	1010	12	A	
	11	1011	13	B	
	12	1100	14	C	
	13	1101	15	D	
	14	1110	16	E	
	15	1111	17	F	
2^4	16	10000	20	10	16^1
	17	10001	21	11	
	18	10010	22	12	
	19	10011	23	13	
	20	10100	24	14	
	21	10101	25	15	
	22	10110	26	16	
	23	10111	27	17	
	24	11000	30	18	

3.3.1 Most Significant Digit (MSD) and Least Significant Digit (LSD)

Given below in Table 3.9 are the most and least significant digits of a round figure or a decimal number.

Table 3.9 – The Most and Least Significant Positional Value of a number

329	3	9
1237.0	1	7
58.32	5	2
0.0975	9	5
0.4	4	4

Binary Number	MSB	LSB
1001	1 = (2^3)	1 = (2^0)
011.101	1 = (2^1)	1 = (2^{-3})

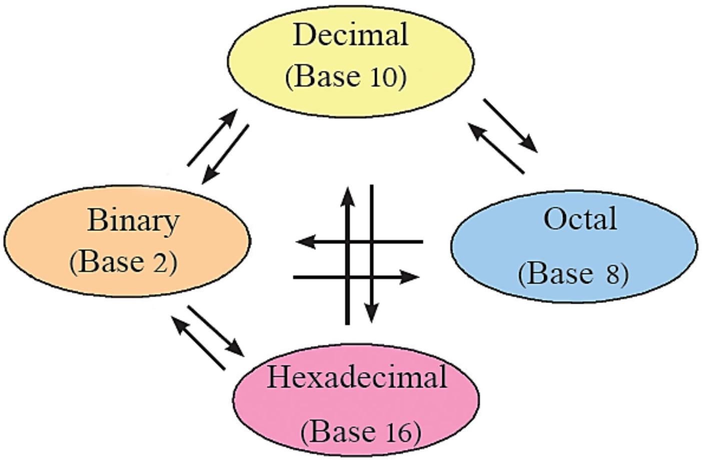
Find the most significant digit and the least significant digit of the following numbers.

(i) 56870_{10} (ii) 154.01_{10} (iii) 23.080_8 (iv) $AD\,239_{16}$
(v) 0.00110_2

Find the most significant bit and the least significant bit of the following numbers.

(i) 1000_2 (ii) 011101_2 (iii) 0.11001_2 (iv) 1.0010_2
(v) 0.00110_2

Conversion Between Number Systems



1.

Example

Converting number 1101_2 to a decimal number.

$$\begin{array}{cccc} 1 & 1 & 0 & 1 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

$$\begin{aligned} 1101_2 &= (1\times2^3) + (1\times2^2) + (0\times2^1) + (0\times2^0) \\ &= (1\times8) + (1\times4) + (0\times2) + (1\times1) \\ &= 8 + 4 + 0 + 1 \end{aligned}$$

$1101_2 = 13_{10}$

1101_2

$1 \times 2^0 = 1$
$0 \times 2^1 = 0$
$1 \times 2^2 = 4$
$1 \times 2^3 = 8$
13

$1101_2 = 13_{10}$

Activity

Convert the following binary numbers to decimal numbers.
(i) 101_2 (ii) 111010110_2 (iii) 1010010111_2

2. Converting Octal Numbers to Decimal Numbers

Example

Converting number 1275_8 to a decimal number.

1	2	7	5
↓	↓	↓	↓
8^3	8^2	8^1	8^0

$$
\begin{aligned}
1275_8 &= (1\times 8^3) + (2\times 8^2) + (7\times 8^1) + (5\times 8^0) \\
&= (1\times 512) + (2\times 64) + (7\times 8) + (5\times 1) \\
&= 512 + 128 + 56 + 5
\end{aligned}
$$

$$1275_8 = 701_{10}$$

1275_8

- $5\times 8^0 = 5$
- $7\times 8^1 = 56$
- $2\times 8^2 = 128$
- $1\times 8^3 = 512$

Total: 701

$$1275_8 = 701_{10}$$

(i) 230_8 (ii) 745_8 (iii) 2065_8

3. Converting Hexadecimal Numbers to Decimal Numbers

Example

Converting number $AB2_{16}$ to a decimal number.

A	B	2
↓	↓	↓
16^2	16^1	16^0

$$
\begin{aligned}
AB2_{16} &= (A\times 16^2) + (B\times 16^1) + (2\times 16^0) \\
&= (10\times 256) + (11\times 16) + (2\times 1) \\
&= 2560 + 176 + 2
\end{aligned}
$$

$$AB2_{16} = 2738_{10}$$

$AB2_{16}$

- $2\times 16^0 = 2$
- $11\times 16^1 = 176$
- $10\times 16^2 = 2560$

Total: 2738

$$AB2_{16} = 2738_{10}$$

Activity

Convert the following hexadecimal numbers to decimal numbers

(i) $1A_{16}$ (ii) $7EF_{16}$ (iii) $A49_{16}$

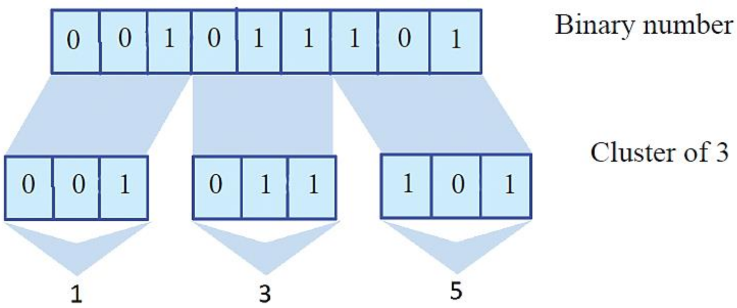
4. Converting Binary Numbers to Octal Numbers

Write the equivalent three binary digits groups for each octal digit. Remove the zeros from left which has no values. Put all together to get the binary equivalent number.

$137_8 = 001,011,111$

$= 001011111_2$

$= 1011111_2$



Activity

Convert the following binary numbers to octal numbers.

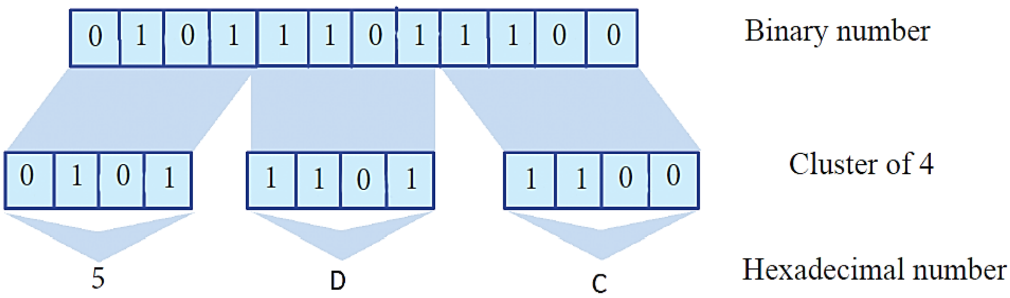
(i) 10011001_2 (ii) 111100111_2 (iii) 10101010110_2

5. Converting Binary Numbers to Hexadecimal Numbers

- First, divide the number into four-bit clusters from the right corner to the left corner.
- Write hexadecimal numbers separately for each cluster.
- Write these numbers in order from the left corner to the right corner and write down the base.

Example

Converting number 10111011100_2 to a hexadecimal number.



$10111011100_2 = 5DC_{16}$

Activity

Convert the following binary numbers to hexadecimal numbers.

(i) 11011010_2 (ii) 11111001101_2 (iii) 10011100011_2

6. Converting Octal numbers to Binary Numbers

Example

Converting number 457_8 to a binary number.

- Firstly, write each digit in octal number in three bits.
- Secondly, write down all the bits together to get the binary number for the octal number.

4	5	7
100	101	111

$457_8 = 100101111_2$

Activity

Convert the following octal numbers to binary numbers.

(i) 10_8 (ii) 245_8 (iii) 706_8

7. Converting Octal numbers to Hexadecimal Numbers

We have learned above that an octal number can be indicated in three digits when it is converted to a binary number.

Thus, each digit in octal numbers should be written in three digits when it is converted to base two.

Example

Converting number 1057_8 to a hexadecimal number.

- First, write each digit in octal number in three bits.
- Divide the binary number you get into four-bit clusters from the right corner to the left corner.
- Write the related hexadecimal number for each cluster.

1	0	5	7
001	000	101	111

00 1 0	0 0 1 0	1111
2	2	15
2	2	F

$1057_8 = 22F_{16}$

Activity

Convert the following octal numbers to hexadecimal numbers

(i) 320_8 (ii) 475_8 (iii) 1673_8

8. Converting Hexadecimal Numbers to Binary Numbers

When a hexadecimal number is converted to a binary number, each digit in that number should be indicated in a four-bit binary number.

Example

Converting number $2AE_{16}$ to a binary number.

2	A	E
0010	1 010	1110

$2AE_{16} = 1010101110_2$

> **Activity**
>
> Convert the following hexadecimal numbers to binary numbers
>
> (i) 78_{16} (ii) $B2C_{16}$ (iii) $4DEF_{16}$

9. Converting Hexadecimal Numbers to Octal Numbers

First, the hexadecimal number should be converted to a binary number and then it should be converted to an octal number.

Example

Converting number $23A_{16}$ to an octal number.

2	3	A
0010	0011	1010

001	0 0 0	1 1 1	010
1	0	7	2

$23A_{16} = 1072_8$

> **Activity**
>
> Convert the following hexadecimal numbers to octal numbers.
>
> (i) 320_{16} (ii) $A7B_{16}$ (iii) $10ED_{16}$

10. Conversion of Decimal Numbers to Binary Numbers

Example

Converting number 12_{10} to a binary number.

➢ First, divide this number by 2 writing the remainders.

```
2 | 12
   2 | 6    — 0   ↑
      2 | 3  — 0   | Remainder
        2 | 1 — 1  |
            0 — 1  |
            ↗
      Quotient
```

E.g.:- convert 0.3125_{10} to binary

	0.3125	x2
0	.625	x2
1	.25	x2
0	.50	x2
1	.00	

$0.3125_{10} = 0.0101_2$

: following decimal numbers to binary numbers.

(ii) 472_{10} (iii)1163_{10}

11. Converting Decimal Numbers to Octal Numbers

Example

Converting 158_{10} to an octal number.

➢ Firstly, divide this number by 8 and write down the remainder.

```
8 | 158
   8 | 19   — 6   ↑
      8 | 2  — 3   | Remainders
          0  — 2   |
         ↗
   Quotient
```

Activity

Convert the following decimal numbers to octal numbers.

(i) 155_{10} (ii) 472_{10} (iii) 1163_{10}

12. Converting Decimal Numbers to Hexadecimal Numbers

Example

Converting number 38_{10} to a hexadecimal number.

➢ Firstly, divide this number by 16 and write down the remainders.

```
16 | 38
   16 | 2  — 6   ↑
         0 — 2   | remainders
        ↗
   Quotient
```

Activity

Convert the following decimal numbers to hexadecimal numbers.

(i) 256_{10} (ii) 478_{10} (iii) 1963_{10}

➢ Secondly, write down all the remainders from bottom to top.

$38_{10} = 26_{16}$

Sign-Magnitude.

The sign and magnitude method is commonly an 8 bit system that uses the most significant bit (MSB) to indicate a positive or a negative value. By convention, a '0' in this position indicates that the number given by the remaining 7 bits is positive, and a most significant bit of '1' indicates that the number is negative. This interpretation makes it possible to create a value of negative zero.

E.g. :- $+45_{10}$ in signed binary is 00101101_2

-45_{10} in signed binary is 10101101_2

One's Complement

In one's complement, positive numbers are represented as usual in regular binary. However, negative numbers are represented differently. To negate a number, replace all zeros with ones, and ones with zeros - flip the bits. Thus, 12 would be 00001100, and -12 would be 11110011. As in signed magnitude, the leftmost bit (most significant bit-MSB) indicates the sign (1 is negative, 0 is positive). To compute the value of a negative number, flip the bits and translate as before.

When representing positive and negative numbers in 8-bit ones complement binary form, the positive numbers are the same as in signed binary notation.

E.g.: -120 is represented in one's complement form as 10000111_2 and

-60 is represented in one's complement form as 11000011_2

The ones complement system still has two ways of writing 0_{10} ($00000000_2 = +0_{10}$ and $11111111_2 = -0_{10}$);

Two's Complement.

A single set of bits is used. To form a negative number, start with a positive number, complement each bit and add one. This interpretation includes one more negative value than positive values (to accommodate zero).

E.g. :- -5 is represented in two's complement form as 11111011_2 and $+5_{10}$ as 00000101_2.

$+5 = 00000101_2$, 1^s complement of 00000101_2 is 11111010_2

2^s complement of $00000101_2 = 11111010_2 + 1 = 11111011_2$

-5_{10} is represented in two's complement as 11111011_2 (In hexadecimal FB_{16})

	Usage
Sign Magnitude	Used only when we do not add or subtract the data. They are used in analog to digital conversions. They have limited use as they require complicated arithmetic circuits.
One's Complement	Simpler design in hardware due to simpler concept.
Two's Complement	Makes it possible to build low-cost, high-speed hardware to perform arithmetic operations.

Fixed point numbers

- In calculations involving fixed point numbers that have a fixed number of digits after the decimal point.

E.g. :- 763.2135 (The decimal point is located in the same position in each number)

179.4821

942.6956

Floating point

- The floating point number is used to represent
  - – Numbers with fractions, e.g., 3.1416
  - – Very small numbers, e.g., 0.000000001
  - – Very large numbers, e.g., 3.15576×10^9

e.g.:- -10.625 is represented in single precision floating point as below.

First convert it to binary 1010.101_2 (the whole and the fractional part separately

Put it in standard form 1.010101×2^3

Bias the exponent by using excess- k method $(2^{k-1}-1)$

K for exponent 8 = $2^{8-1}-1=128-1=127$

Biased value for exponent = $3+127 = 130_{10}$

exponent part in binary is 10000010_2

1	10000010	01010100000000000000000

– Sign, exponent, Mantissa: $(-1)^{sign} \times \text{Mantissa} \times 2^{exponent}$

– More bits for Mantissa give more accuracy

– More bits for exponent increases range

IEEE 754 floating point standard:

– Single precision: 8 bit exponent, 23 bit Mantissa

– Double precision: 11 bit exponent, 52 bit Mantissa

	Advantage	Disadvantage
Fixed Point Representation	Performance good. No need to rely on additional hardware or software logic.	Limited range of values can represent.
Floating point representation	Greater range of numbers is represented. Varying degrees of precision.	More storage space needed. Slower processing times. Lack of precision.

- Binary arithmetic operations - (integers only)
  - Addition, subtraction
- Logical operations
  - Bitwise logical operations

Concepts and terms to be highlighted:

- Binary number addition with/ without carry overs
- Binary number subtractions
- Bitwise logical operations using NOT, AND, OR, XOR operations.

Guidance for lesson plans:

- Add two binary numbers by writing one below another according to the place values.
- Subtract binary numbers by writing the small number below the big number.
- For the given binary number do the bitwise logical operations

Addition of binary numbers

$101100_2 + 1100_2 =$

$1\ 0\ 1\ 1\ 0\ 0_2$

$\underline{\ \ \ \ \ \ 1\ 1\ 0\ 0_2}$

$1\ 1\ 1\ 0\ 0\ 0_2$

$101100_2 + 1100_2 = 111000_2$

Subtraction of binary numbers

$0010\ 110_2 - 001100_2$

$1\ 0\ 1\ 1\ 0\ 0_2$

$\underline{\ \ \ \ \ \ 1\ 1\ 0\ 1_2}$

$0\ 1\ 1\ 1\ 1\ 1_2$

$101100_2 - 1101_2 = 011111_2$

$2\ A\ B_{16}$

$+\ C\ C\ 4_{16}$

$F\ 6\ F_{16}$

$6\ 7\ 5_8$

$+2\ 3\ 5_8$

1132_8

Bitwise operations

1. NOT operation

   For unsigned integers, the bitwise complement of a number is the "mirror reflection" of the number across the half-way point of the unsigned integers range. One use is to invert a grayscale image where each pixel is stored as an unsigned integer. It uses the below bit operations.

A	NOT A
0	1
1	0

   E.g. :- **NOT** 0111_2 (7_{10}) = 1000_2 (8_{10})

2. Bitwise AND operation

   This is often called bit masking. (By analogy, the use of masking tape covers, or masks, portions that should not be altered or portions that is not of interest. In this case, the 0 values mask the bits that are not of interest.)

   It uses the below bit operation

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

   E.g. :- 0101_2 (5_{10}) **AND** 0011_2 (3_{10})

   $0\ 1\ 0\ 1_2$

   $\underline{0\ 0\ 1\ 1_2}$

   $0\ 0\ 0\ 1_2\ (1_{10})$

   Therefore 0101_2 **AND** 0011_2 is 0001_2

3. Bitwise OR operation

A bitwise OR takes two bit patterns of equal length and performs the logical inclusive OR operation on each pair of corresponding bits. The result in each position is 0 if both bits are 0, while otherwise the result is 1. It uses the below bit operation

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

E.g. :- 0101_2 (5_{10}) **OR** 0011_2 (3_{10})

$0\ 1\ 0\ 1_2$

$\underline{0\ 0\ 1\ 1_2}$

$0\ 1\ 1\ 1_2$ (7_{10})

Therefore 0101_2 OR 0011_2 is 0111_2

4. Bitwise XOR operation

The bitwise XOR may be used to invert selected bits in a register (also called toggle or flip). Any bit may be toggled by XOR it with 1

It uses the below bit operations

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

E.g. :- 0010_2 (2_{10}) **XOR** 1010_2 (10_{10})

$1\ 0\ 1\ 0_2$

$\underline{0\ 0\ 1\ 0_2}$

$= 1\ 0\ 0\ 0_2$ (8_{10})

Therefore 0010_2 **XOR** 1010_2 is 1000_2

Fill the blanks

Binary	Octal	Hexadecimal	Decimal
10101010111	______________	______________	______________
______________	62542	______________	______________
______________	______________	3A0F	______________
______________	______________	______________	2052

Data Representation on Computers

Characters are represented on computers using several standard methods such as ASCII, BCD,
EBCDIC and UNICODE

BCD – (Binary Coded Decimal) CODE – This is a 4 bit code used for coding numeric values (0-9) only. 2^4=16 the remaining 6 (i.e. 1010, 1011, 1100, 1101, 1110, 1111) are invalid combinations.

E.g.:- 1000111_2 = $0100\ 0111_{BCD}$ = 47_{10}

Decimal	0	1	2	3	4	5	6	7	8	9

ASCII – ASCII (American Standard Codes for Information Interchange) normally uses 8 bits (1 byte) to store each character. However, the 8th bit is used as a check digit, meaning that only 7 bits are available to store each character. This gives ASCII the ability to store a total of 2^7 = 128 different values . The 7 bit ASCII code was originally proposed by the American National Standard Institute (ANSI). (IBM personal computers use ASCII).

EBCDIC (Extended Binary Coded Decimal Interchange Code) – The 8 bit EBCDIC is used primarily by large IBM mainframe computers and compatible equipment. It uses 256 different characters

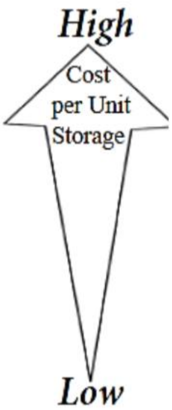
UNICODE – The 16 bit code provides unique code points to characters in many of the world's languages including Sinhala and Tamil. One of the promising proposals is named Unicode.

	Advantage	Disadvantage
BCD	• Easy to encode and decode decimals into BCD and vice versa. • Simple to implement a hardware algorithm for the BCD converter. • It is very useful in digital systems whenever decimal information is given either as inputs or displayed as outputs. • Digital voltmeters, frequency converters and digital clocks all use BCD as they display output information in decimal.	• Not space efficient. • Difficult to represent the BCD form in high speed digital computers in arithmetic operations, especially when the size and capacity of their internal registers are restricted or limited. •Require a complex design of Arithmetic and logic Unit (ALU) than the straight Binary number system. •The speed of the arithmetic operations slow due to the complete hardware circuitry involved.
ASCII	• Uses a linear ordering of letters. • Different versions are mostly compatible. • compatible with modern encodings	• Not Standardized. • Not represent world languages.
EBCDIC	• uses 8 bits while ASCII uses 7 before it was extended.	• Does not use a linear ordering of letters.
	• Contained more characters than ASCII.	• Different versions are mostly not compatible. • Not compatible with modern encodings
UNICODE	• Standardized. • Represents most written languages in the world • ASCII has its equivalent within Unicode.	• Need twice memory to store ASCII characters.

Following are the relationships between units which measure data storage capacity.

8 bits	=	1 byte
4 bits	=	1 nibble
1024 bytes	=	1 kilobyte (KB)
1024 kilobytes	=	1 Megabyte (MB)
1024 Megabytes	=	1 Gigabyte (GB)
1024 Gigabytes	=	1 Terabyte (TB)
1024 Terabytes	=	1 Petabyte (PB)

Register Memory
Cache Memory
Random Access Memory
Read Only Memory
Magnetic Tape
Flash Memory
Hard Disk
Digital Versatile Disc – DVD



Register Memory
1 KB
Cache memory
3 MB – 32 MB
Compact Disk (CD)
650 – 900 MB
Digital Versatile Disc
4.7 – 9 GB
Random Access Memory
01 – 64 GB
Read Only Memory (ROM)
Flash Memory
1 – 64 GB
Hard Disk
100 GB – 6 TB
Magnetic Tape
1 TB – 185 TB

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	