TEMA II
REPASO. SISTEMAS DE NUMERACIÓN USUALES EN INFORMÁTICA.

INTRODUCCIÓN.

Entendemos por sistema de numeración, la forma de representar cantidades mediante un sistema de valor posicional.

Los ordenadores efectúan las operaciones utilizando una representación para los datos basada en el sistema de numeración en base dos (binario natural).

REPRESENTACIÓN POSICIONAL DE LOS NÚMEROS.

Un sistema de numeración en base B, utiliza para representar los números, un alfabeto compuesto por B símbolos o cifras. Así todo número se puede representar por un conjunto de cifras teniendo cada una de ellas un valor dentro del número que depende de:

a) De la cifra en sí.
b) De la posición que ocupa dentro del número.

El sistema de numeración decimal (Base=10) utiliza un alfabeto de diez símbolos {0,1,2,3,4,5,6,7,8,9} y la base tiene valor 10 .Por ejemplo el número 5321.5 puede obtenerse como :

5321. 5 = 5000 + 300 + 20 + 1 + 0.5

Osea

3 2 1 0 -1 ← Posiciones de cada cifra dentro del número

5321. 5 = **5** x 10^3 + **3** x 10^2 + **2** x 10^1 + **1** x 10^0 + **5** x 10^{-1}

SISTEMA DE NUMERACIÓN EN BASE 2 O BINARIO.

El sistema de numeración en binario utiliza un alfabeto de dos símbolos { 0 , 1 } denominadas cifras binarias o bits y la Base=2. Ejemplo de número binario sería el 11011010 . 101

Para **transformar un número de binario a decimal** , se multiplica cada dígito binario por la base elevada al lugar que ocupa el dígito dentro de la cifra :

$$100111.11 = 1x2^5 + 0x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 1x2^0 + 1x2^{-1} + 1x2^{-2} =$$
$$32 + 4 + 2 + 1 + 0.5 + 0.25 = 39.75$$

Para transformar un número de decimal a binario :

a) La **parte entera** binaria se obtiene dividiendo (divisiones enteras) sucesivas veces la parte entera del número decimal y tomando el último cociente y los restos en orden inverso al obtenido :

39 | 2
19 **1** | 19 | 2
1 9 | 2
1 4 | 2
0 2 | 2
0 **1**

b) La **parte fraccionaria** del número binario se obtiene multiplicando por dos la parte fraccionaria del número decimal de partida y las partes fraccionarias que se van obteniendo, tomando como número binario las partes enteras obtenidas :

$$\begin{array}{r} 0.75 \\ \times\ 2 \\ \hline \mathbf{1}.5 \end{array} \qquad \begin{array}{r} 0.5 \\ \times\ 2 \\ \hline \mathbf{1}.0 \end{array}$$

OPERACIONES ARITMÉTICAS

SUMA				***RESTA***				***PRODUCTO***		
A	B	A + B	Acarreo	A	B	A - B	Debe	A	B	A x B
0	0	0		0	0	0		0	0	0
0	1	1		0	1	1	1	0	1	0
1	0	1		1	0	1		1	0	0
1	1	0	1	1	1	0		1	1	1

Ejemplo :

```
   11011011         10010001          1011011
 + 01111011       -  1110111            x 101
 ----------       ----------        ---------
  101010110        00011010           1011011
                                     0000000 .
                                    1011011....
                                    ---------
                                    111000111
```

OPERACIONES LÓGICAS (Tablas de verdad)

OR			AND			XOR			NOT	
A	B	A OR B	A	B	A AND B	A	B	A XOR B	A	NOT A
0	0	0	0	0	0	0	0	0	0	1
0	1	1	0	1	0	0	1	1	1	0
1	0	1	1	0	0	1	0	1		
1	1	1	1	1	1	1	1	0		

Ejemplo :

(0 Or 1) And Not 0 = 1

1 Xor (1 And (0 Or Not 1)) = 1

REPRESENTACIÓN EN COMPLEMENTOS

El **complemento a la base menos uno** de un número es el resultado de restar cada una de las cifras del número a la base menos uno del sistema de numeración utilizado.

Complemento a 9 del número decimal 56 sería = 99 - 56 = 43

Complemento a 1 del número binario 11011 sería = 11111 - 11011 = 00100

El **complemento a la base de un número** se obtiene sumando 1 al complemento menos uno :

Complemento a 10 del número decimal 56 sería = 43 + 1= 44

Complemento a 2 del número binario 11011 sería = 00100 + 1 = 00101

Conociendo la notación en complementos, se pueden **restar** dos números sumando al minuendo el complemento a la base del sustraendo y despreciando el acarreo:

$$\begin{array}{r} 1011011 \\ -\,1001110 \\ \hline \mathbf{0001101} \end{array} \quad \xrightarrow{\text{Complemento a 1}} \quad \begin{array}{r} 1011011 \\ +0110001 \\ 1 \\ \hline \mathbf{0001101} \end{array}$$

La representación en complementos permiten realizar las operaciones de resta mediante sumas, reduciendo así el número de circuitos necesarios en la ALU.

BASE INTERMEDIAS.

OCTAL : Se define como Base=8 y utiliza el alfabeto {0,1,2,3,4,5,6,7}.

$377)_8 = 3 \times 8^2 + 7 \times 8^1 + 7 \times 8^0 = 192 + 56 + 7 = 255$

HEXADECIMAL : Se define como Base = 16 y utiliza el alfabeto {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}.

$5FF)_{16} = 5 \times 16^2 + 15 \times 16^1 + 15 \times 16^0 = 1280 + 240 + 15 = 1535$

Tabla de equivalencias

Decimal	Binario	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	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

Cada dígito en octal equivale a un grupo de tres bits de forma que para pasar de octal a binario basta sustituir cada dígito octal por sus tres bits equivalentes.

$377)_8$ = 011 111 111

Análogamente cada dígito hexadecimal equivale a un grupo de cuatro bits :

$5FF)_{16}$ = 0101 1111 1111

Para pasar de binario a estas bases intermedias , se agrupan las cifras binarias desde el punto decimal hacia la izquierda y derecha y cada grupo de tres o cuatro bits se sustituye por su equivalente octal o hexadecimal respectivamente. Por ejemplo el número binario 1111110.01

001 111 110 . 010 = $1\ 7\ 6\ .\ 2\ )_8$

0111 1110 . 0100 = $7\ E\ .\ 4\ )_{16}$

EJERCICIOS

1. Transformar a binario, octal y hexadecimal los números en decimal 3245 y 543.65
2. Pasar a decimal el número binario 110101101001.1011
3. Pasar de hexadecimal a octal el número FFD8
4. Realiza las siguientes operaciones en binario:

   100111101 + 1101111

   100001011 - 11111

   101011010 x 1011
5. Resolver la tabla de verdad de las siguientes operaciones lógicas :

   (a OR b) XOR a (a AND b) AND NOT a
6. Utilizando el complemento a la base del sustraendo realiza las siguientes operaciones de resta en binario y comprueba el resultado tranformando los datos a decimal.

   10001101101 - 101110111

   11011011011 - 100110010
7. Realiza las siguientes operaciones en hexadecimal y comprueba los resultados transformando los términos a decimal .

FA5 + FBA F095 - EFFD