TEMA 1

REPRESENTACION Y COMUNICACIÓN DE LA INFORMACION

Introducción. Definición de información.

La información es todo aquello que puede ser manejado por un sistema, ya sea como entrada, como proceso, o bien como resultado. De esta forma, podemos clasificar a los sistemas informáticos como sistemas de flujo de información (si la información de entrada y salida es la misma) y sistemas de tratamiento de la información, en los que la información que entra y la que sale es distinta, ya que ha sufrido alguna manipulación.

La información, para que sea útil a nuestro ordenador debe estar representada por símbolos. Tales símbolos por si solos no constituyen la información, sino que la representan.

La información se puede clasificar como:

- Datos numéricos, generalmente números del 1 al 9
- Datos alfabéticos, compuestos solo por letras
- Datos alfanuméricos, combinación de los dos anteriores

Físicamente la información puede almacenarse en el interior de un computador mediante dos estados ,estos dos estados se representan conceptualmente como cero (0) y uno (1),
El bit (binary digit), es la unidad mínima de almacenamiento

Un Byte es un conjunto de 8 bits.
La capacidad de memoria de un computador se mide en Bytes. Como el Byte es una unidad relativamente pequeña se usan múltiplos:

- KiloByte KB = 2^{10} =1.024 Bytes
- MegaByte MB = 2^{20} = 1.048.576 Bytes
- GigaByte GB = 2^{30} = 1.073.741.824 Bytes
- TeraByte TB = 2^{40}
- PetaByte PB = 2^{50}

Sistemas de numeración.

Es un sistema de representación de números que asocia a cada uno una representación única, y permite realizar algoritmos simples, así como ejecutar operaciones aritméticas. El más usado es el sistema de numeración decimal, que surgió gracias a que se utilizaban los dedos de la mano para contar las cosas.

En los sistemas de numeración, cada cantidad se representa en forma de potencias sucesivas del sistema en que estamos, como puede ser base 2, base 10, base 16, etc.

Sistema decimal

En este sistema se representan los números en forma de potencias sucesivas de 10.

Vg.:

Expresar el número 7824 en base 10.

$$7000+800+20+4= 7*10^3 + 8*10^2 + 2*10^1 + 4*10^0$$

Ejercicios

Calcular los siguientes números en base 10

$53_{(10} = 5*10^1 + 3*10^0$

$7_{(10} = 7*10^0$

Sistema binario.

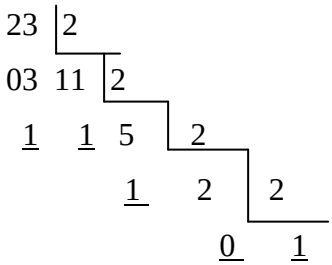
Comenzó a utilizarse casi a la vez que los ordenadores. Por construcción, los primeros ordenadores estaban formados por interruptores, que únicamente podían tener dos estados: 1- encendido y 0 – apagado. No obstante, en la actualidad aun siguen utilizándose, ya que los actuales ordenadores registran la información en forma de impulsos eléctricos. Así, las memorias y soportes de información en forma de 1 – paso ó 0 – no paso de corriente eléctrica.

Los números decimales, para poder ser almacenados en el ordenador deben ser representados en código binario, es decir, como sumas de potencias de 2.

Ejemplo:

Pasar **de decimal a binario** el número 23.

El método es dividir sucesivamente entre 2 hasta que el cociente sea 1



A continuación, se escribe el último cociente y los restos de derecha a izquierda

10111

Se concluye que

$23_{(10} = 10111_{(2}$

Para comprobar el resultado, se realiza el proceso inverso, es decir, se pasa **de binario a decimal**. Sería como sigue:

$10111_{(2} = 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 1*2^0 = 16 + 0 + 4 + 2 + 1 = 23_{(10}$

Ejercicios

1) *Pasar de decimal a binario los siguientes números:*

   $16_{(10} = 10000_{(2}$

   $93_{(10} = 1011101_{(2}$

2) *Pasar de binario a decimal lo siguientes números:*

   $1101110_{(2} = 110_{(10}$

   $1100_{(2} = 12_{(10}$

3) *Codificar en binario los 50 primeros números decimales, en código de 6 bits. Identificar la regla en la que se basa la escritura de números binarios consecutivos en su cambio a base decimal.*

Decimal	***Binario***	***Decimal***	***Binario***	***Decimal***	***Binario***	***Decimal***	***Binario***
0	000000	15	001111	30	011110	45	101101
1	000001	16	010000	31	011111	46	101110
2	000010	17	010001	32	100000	47	101111
3	000011	18	010010	33	100001	48	110000
4	000100	19	010011	34	100010	49	110001
5	000101	20	010100	35	100011	50	110010
6	000110	21	010101	36	100100		
7	000111	22	010110	37	100101		
8	001000	23	010111	38	100110		
9	001001	24	011000	39	100111		
10	001010	25	011001	40	101000		
11	001011	26	011010	41	101001		
12	001100	27	011011	42	101010		
13	001101	28	011100	43	101011		
14	001110	29	011101	44	101100		

Sistema octal

Es el sistema de numeración en base 8. Los números incluidos en este sistema son:

{0, 1, 2, 3, 4, 5, 6, 7}

Paso **de decimal a octal**

Se divide el número entre 8, tomándose los restos y el último cociente, de derecha a izquierda

Ejemplo:

$8361_{(10} = 20251_{(8}$

```
8361 | 8
036    1045 | 8
 41      24   130 | 8
  1       05   50  16 | 8
           5    2   0    2
```

Ejercicios:

Pasar a octal los siguientes números decimales:

$23_{(10} = 27_{(8}$

$54_{(10} = 66_{(8}$

Realizar una tabla en la que se represente el paso de decimal a octal de los 30 primeros números

Decimal	Octal	Decimal	Octal	Decimal	Octal
0	0	10	12	20	24
1	1	11	13	21	25
2	2	12	14	22	26
3	3	13	15	23	27
4	4	14	16	24	30
5	5	15	17	25	31
6	6	16	20	26	32
7	7	17	21	27	33
8	10	18	22	28	34
9	11	19	23	29	35

Paso **de octal a decimal**

Se multiplica cada cifra del número por la potencia de 8 equivalente a su posición.

Ejemplo:

$5721_{(8} = 5*8^3 + 7*8^2 + 2*8^1 + 1*8^0 = 5*512 + 7*64 + 2*8 + 1*1 = 3025_{(10}$

Ejercicios

Pasar de octal a decimal los siguientes números

$403_{(8} = 259_{(10}$

$63_{(8} = 51_{(10}$

$5_{(8} = 5_{(10}$

Paso **de octal a binario**

Opción 1: Se pasa de octal a decimal y de decimal a binario

Ejercicios

Pasar a binario los siguientes números en octal

$41_{(8} = 33_{(10} = 100001_{(2}$

$352_{(8} = 234_{(10} = 1101010_{(2}$

$76_{(8} = 65_{(10} = 111110_{(2}$

$1593_{(8}$ = *No es octal por contener el número 9*

Opción 2: Considerando que ocho es potencia de 2 ($8 = 2^3$), se realiza una tabla en la que estén contenidos todos los dígitos binarios con tres bits, y después se traduce cada dígito octal a su correspondiente binario.

Octal	Binario
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Ejemplo

Pasar de octal a binario usando la tabla

$41_{(8} = 100001_{(2}$

$352_{(8} = 011101010_{(2}$

$76_{(8} = 111110_{(2}$

Paso **de binario a octal**

Se toman grupos de tres dígitos de derecha a izquierda y se busca la correspondencia en la tabla. Si faltan dígitos a la izquierda se rellenan con ceros hasta conseguir los 3 bits.

Ejercicios

Pasar de binario a octal los siguientes números:

$110110_{(2} = 66_{(8}$

$100101_{(2} = 45_{(8}$

Sistema hexadecimal

Corresponde a un sistema de numeración en base 16. Los dígitos que faltan desde el 10 se suplen con letras del abecedario. Los dígitos hexadecimales son:

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

Paso **de hexadecimal a decimal**

Se multiplica cada cifra por la correspondiente potencia de 16, en función del lugar que ocupe en la propia cifra.

Ejemplo

$A70D4_{(16} = 10*16^4 + 7*16^3 + 0*16^2 + 13*16^1 + 4*16^0 = 684244_{(10}$

Ejercicios

Pasar a decimal los siguientes números hexadecimales

$F5CCA_{(16} = 1006794_{(10}$

$BBACE_{(16} = 768718_{(10}$

$J14AB_{(16}$ = No es un número hexadecimal, ya que la j no forma parte de sus dígitos

Paso **de decimal a hexadecimal**

Opción 1: Sucesivas divisiones entre 16, quedándonos con los restos y el último cociente, escritos luego de derecha a izquierda.

Opción 2: Pasar a binario. Crear una tabla de correspondencia binaria a cada dígito hexadecimal de 4 bits ($16 = 2^4$), y de ahí a hexadecimal

Paso **de binario a hexadecimal**

Con la tabla de conversión, se toman de 4 en 4 dígitos de derecha a izquierda, supliendo con ceros las carencias de dígitos a la izquierda.

Hexadecimal	Binario	Hexadecimal	Binario
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Ejemplo

Pasar a hexadecimal los siguientes números en binario

$1001_{(2} = 9_{(16}$

$111111010_{(2} = 1FA_{(16}$

$10011_{(2} = 13_{(16}$

$101111101111010_{(2} = 5F7A_{(16}$

$1010100001_{(2} = 2A1_{(16}$

Paso de **octal a hexadecimal**

Se pasa de octal a binario (3 bits) usando la tabla, y de este a hexadecimal (4 bits) usando también la tabla.

Ejercicios

Pasar a hexadecimal los siguientes números

$70431_{(8} = 111000100011001_{(2} = 7119_{(16}$

$352_{(8} = 011101010_{(2} = 0EA_{(16}$

Paso de **hexadecimal a octal**

Se pasa de de hexadecimal a binario (4 bits) usando la tabla, y de este a octal (3 bits) usando también la tabla.

Ejemplos

Pasar a octal los siguientes números

$ABC07D_{(16} = 101010111100000001111101_{(2} = 52740175_{(8}$

$F549E10_{(16} = 11110101010010011111000010000_{(2} = 1725117020_{(8}$

$8963_{(16} = 1000100101100011_{(2} = 104543_{(8}$

Paso de una base a otra cuando hay decimales

Paso **de decimal a binario**

Se separa la parte entera de la fraccionaria. La parte entera se traduce como siempre, es decir, dividiendo sucesivamente entre 2. En cuanto a la parte fraccionaria, se multiplica sucesivamente por 2, quedándonos con la parte entera que vaya resultando de cada operación.

Ejemplo

Pasar a binario

$15,35_{(10}$

Se separan enteros de fraccionarios

$15_{(10} = 1111_{(2}$

$0,35_{(10} =$

0,35x2 = 0,70 se toma el 0

0,70x2 = 1,40 se toma el 1

0,40x2 = 0,80 se toma el 0

0,80x2 = 1,60 se toma el 1

quedando:

$1111,0101_{(2}$

Paso **de binario a decimal**

Se multiplica cada cifra por potencias sucesivas de 2, y los decimales por potencias negativas.

Ejemplo

$1111,0101_{(2} = 2^3 + 2^2 + 2^1 + 2^0 + 0 * 2^{-1} + 1 * 2^{-2} + 0 * 2^{-3} + 1 * 2^{-4} = 15 + 0,25 + 0,0625 = 15,3125_{(10}$

Ejercicios

Pasar a binario los siguientes números decimales.

$171{,}076_{(10}$ = $10101011{,}0001_{(2}$

$497{,}30214_{(10}$ = $111110001{,}0100_{(2}$

Paso **de decimal a octal con decimales**

La parte entera se divide por sucesivamente entre 8, y la parte fraccionaria se multiplica sucesivamente por 8, quedándonos con la parte entera.

Ejemplo

$171{,}076_{(10}$ =

Parte entera

$171_{(10}$ = $253_{(8}$

Parte fraccionaria

$0{,}076_{(8}$

0,076*8 = 0,608 se toma el 0

0,608*8 = 4,864 se toma el 4

0,864*8 = 6,912 se toma el 6

0,912*8 = 7,296 se toma el 7

queda el número $253{,}0467_{(8}$

Ejercicio

Pasar a octal el siguiente número decimal

$9325{,}8654_{(10}$ = $22155{,}6730_{(8}$

Paso **de decimal a hexadecimal con decimales**

La parte entera se divide por sucesivamente entre 16, y la parte fraccionaria se multiplica sucesivamente por 16, quedándonos con la parte entera.

Ejemplo

Pasar a hexadecimal el siguiente número en decimal

$171{,}076_{(10}$

Parte entera

$171_{(10}$ = AB

Parte fraccionaria

0,076*16 = 1,216 se toma el 1

0,216*16 = 3,456 se toma el 3

0,456*16 = 7,296 se toma el 7

0,296*16 = 4,736 se toma el 4

El número resultante es $AB,1374_{(16}$

Ejercicio

Pasar a hexadecimal el siguiente número

$9325,8654_{(10} = 246D, DD8A_{(16}$

Paso **de octal a binario con decimales**

Se utiliza la tabla y se traducen de dígito en dígito

Ejemplos

$7540,321_{(8} = 111101100000, 011010001_{(2}$

$53,7654_{(8} = 101011,111110101100_{(2}$

Paso **de hexadecimal a binario con decimales**

Se utiliza la tabla y se traducen de dígito en dígito

Ejemplos

$AB10A,97C_{(16} = 10101011000100001101,011110011100_{(2}$

$5E,40DA_{(16} = 01011110,0100000011011010_{(2}$

Paso **de octal a hexadecimal con decimales**

Se pasa a binario usando la tabla, y de ahí a hexadecimal. Se completan los bits necesarios con ceros a ambos lados.

Ejemplos

Pasar a hexadecimal los siguientes números octales

$7540, 321_{(8} = 111101100000, 011010001_{(2} = F60,688_{(16}$

$53,7564_{(8} = 101011,111110101100_{(2} = 28,FAC_{(16}$

Paso **de hexadecimal a octal con decimales**

Se pasa a binario usando la tabla y de ahí a octal.

Ejemplos

Pasar a octal los siguientes números hexadecimales

$AB10D,79C_{(16} = 01010101100010000 1101, 011110011100_{(2} = 2530415,3634_{(8}$

$5E,40DA_{(16} = 001011110,0100000011011010_{(2} = 136,2015_{(8}$

Representación de numeros negativos

Modulo y signo

En el sistema Signo – magnitud los números positivos y negativos tienen la misma notación para los bits de magnitud pero se diferencian en el bit del signo. El bit del signo es el bit situado más a la izquierda en el número binario:

- En números positivos se emplea el bit "0".
- En números negativos se emplea el bit "1".

- El número no debe estar complementado.

Ejemplo

El número decimal 21 se expresa en binario de 6 bits 010101, donde el primer bit "0" denota el bit de una magnitud positiva. El número decimal –21 se expresa en binario 110101, donde el primer bit "1" denota el bit de una magnitud negativa.

Ejemplo : Representar el número decimal -39 en formato SM de 8 bits.
En primer lugar se pasa a binario el módulo del número -39, estos es, se expresa en binario natural el número 39 utilizando 7 bits. El resultado es 00100111.
Tras ello, se añade el bit de signo, y como en este caso se trata de un número negativo, éste será 1, resultando finalmente que el número binario pedido es 100100111.

Ejemplo : Sea la secuencia de bits 10010101. A qué valor corresponde dicha secuencia si representa a:
a) Un número codificado en binario natural
b) Un número de codificado en SM de 8 bits

La respuesta del primer apartado es inmediata si se realiza la conversión de binario a decimal del modo usual, resultando:

$$10010101_2 = 1 + 4 + 16 + 128 = 149_{10}$$

Para el segundo apartado, de inmediato sabemos que se trata de un número negativo, ya que está en SM y su MSB es 1. Para el cálculo del módulo realizamos la conversión de binario a decimal pero con los 7 bits de la izquierda. Estos es:

$$0010101_2 = 1 + 4 + 16 = 21_{10}$$

De modo que el resultado final es que $10010101_{SM} = -21_{10}$.

Representación en complemento a 1 (C1)

El complemento a 1 en binario se obtiene cambiando los unos por ceros y los ceros por unos. La representación de números positivos en complemento a 1 sigue las mismas reglas del sistema signo-

magnitud y la representación de los números negativos en complemento 1 es el complemento a 1 del número positivo.
Ejemplo
El número decimal 21 se expresa en complemento a 1 a 6 bits como 010101, donde el primer bit "0" denota el bit de una magnitud positiva.
El complemento 1 a 6 bits del decimal –21, se obtiene por medio del complemento a 1 del número positivo 010101 el cual es 101010.
Ejemplo
Un forma de obtener el complemento 1 de un número binario es utilizar un circuito digital compuesto por inversores (compuertas NOT).

Algunas características de este tipo de representación son:

- Existe la misma cantidad de números positivos y negativos (2^{n-1} -1 de cada tipo)
- Existen dos representaciones del 0: { +0, -0 }
- El total de números distintos posibles de representar con n bits es de 2^{n}-1, desde el menor número negativo -(2^{n-1} - 1) al positivo de mayor valor +(2^{n-1} - 1).

Representación en complemento a 2 (C2)

Los computadores utilizan la representación binaria en complemento a 2 para representar números negativos. La representación de números positivos en complemento a 2 sigue las mismas reglas del sistema signo-magnitud y la representación de los números negativos en complemento a 2 se obtiene de la siguiente forma:

1. Se representa el número decimal dado en magnitud positiva.
2. El número de magnitud positiva se representa en forma binaria positiva.
3. Se obtiene el complemento 1 del número binario obtenido en el paso anterior mediante el cambio de los unos por ceros y viceversa.
4. Al complemento 1 se le suma uno y el resultado es la representación en el complemento 2.

Estas son algunas de las características de la representación en C2:

- Existe un número positivo menos que de números negativos
- Existe una única representación del 0: { +0 }
- El rango de números posibles de representar con n bits es de 2^{n}, desde el -2^{n-1} al número $+(2^{n-1}-1)$.

Ejemplo

Representar el número -5_{10} en binario, utilizando el complemento a 2 con 5 bits.

1. Escribimos el número $+5_{10}$ en binario de 5 bits

   0101

2. Obtenemos el complemento a 1 de 0101

   1010

3. Al complemento de número anterior se la suma 1. El resultado es 1011.
4. Obtenemos el número 1011 en complemento a 2.

Ejemplo

Obtener el complemento a 2 del número positivo de 8 bits 000001012 ($+5_{10}$).

El equivalente en complemento a 1 es 11111010. El complemento a 2 del número es 11111011. Comprobando los pesos en decimal se puede demostrar la obtención del negativo del número inicial utilizando el método del complemento a 2:

$111110112 = (-128 + 64 + 32 +16 + 8 + 0 + 2 + 1)10 = - 5_{10}$

En la representación en complemento 2 el primer bit del lado más significativo puede interpretarse como el signo, siendo cero para números positivos y 1 para números negativos. Se puede comprobar que si a una cantidad negativa expresada en complemento 2 se le saca su complemento 2, se obtiene la magnitud positiva correspondiente.

El complemento a1 es mas facil de implementar utilizando componentes digitales, pero presenta la desventaja de tener dos representaciones del +0 y -0
Que complica los circuitos electrónicos capaces de detectar el cero. La mayoria de las m

Aritmética Binaria

Al igual que se hacen operaciones aritméticas entre números expresados en decimal con una serie de reglas, y se obtienen resultados también en decimal, esas mismas operaciones se pueden realizar entre números binarios obteniéndose los resultados en binario.

SUMA

a	b	a+b
0	0	0
0	1	1
1	0	1
1	1	0 Me llevo 1

```
  1 1 1
  10110
+ 11100
--------
 110010
```

RESTA

a	b	a-b
0	0	0
0	1	1 Me llevo 1
1	0	1
1	1	0

Decimal

```
  54583055
- 19947053
----------
  34636002
```

Binario

```
  11010011
- 01101010
----------
  01101001
```

MULTIPLICACIÓN

a	b	a×b
0	0	0
0	1	0
1	0	0
1	1	1

```
    1101
  x 0101
  ------
    1101
   0000
  1101
  -------
  1000001
```

DIVISIÓN

a	b	a/b
0	0	Indeterm.
0	1	0
1	0	∞
1	1	1

```
1011011 | 111
------- | ----
  1000  | 1101
   00111
      000
```

NÚMEROS REALES (coma flotante).

Números como 10, 45 o -13 son valores enteros que se manipulan con las operaciones que ya conocemos. Otra cosa son los valores como 25,47 que no son enteros pero al igual que los anteriores forman parte del conjunto de los números reales. En binario también tienen representación los números con decimales: Cada cifra que haya después de la coma tiene igualmente un peso que depende de su posición, comienza por la izquierda con valor igual a 1/2 y decrece hacia la derecha, siempre multiplicando por 1/2 para obtener el siguiente. Veamos un ejemplo:

$11001{,}0112 = 1\cdot 16 + 1\cdot 8 + 0\cdot 4 + 0\cdot 2 + 1\cdot 1 + 0\cdot (1/2) + 1\cdot (1/4) + 1\cdot (1/8) = 25{,}37510$

Para pasar de decimal a binario se pasa normalmente la parte entera y la parte decimal se va multiplicando por 2 hasta que se anulan los decimales y los decimales binarios se obtienen con la parte entera que se obtiene en cada paso. Por ejemplo:

$25{,}37510 = 25 + 0{,}375 = 110012 +$ decimales

$0{,}375\cdot 2 = 0{,}750$ (primer decimal el 0); $0{,}75\cdot 2 = 1{,}50$ (segundo decimal el 1); $0{,}50\cdot 2 = 1{,}0$ (tercer decimal el 1).

Queda finalmente: $25{,}37510 = 11001{,}0112$

La forma anterior de proceder es complicada de implementar con electrónica digital o al menos existe otra manera mucho más sencilla que es la coma flotante. Sin coma flotante, es necesario seguir la pista a la posición de la coma cada vez que se hace una operación y al operar con los valores de dos registros no puede relacionarse directamente bit con bit de iguales posiciones, ya que al variar las posiciones de las comas hacen que los pesos de cada bit también se desplacen**.**

En coma flotante cada número se expresa con un valor entero y un exponente de la base del sistema, por ejemplo, el valor -43,425 será igual a $-43425\cdot 10^{-3}$ (notación científica), es decir, queda definido por el valor entero -43425 y el exponente -3. En los registros, se destina un número fijo de cifras para el valor entero y el resto de cifras para el exponente. Al multiplicar los valores de dos registros, simplemente se multiplican los enteros y se suman los exponentes. Al dividir, se dividen los enteros y se restan los exponentes. Para sumar o restar dos valores, se suman o restan los enteros y se mantienen los exponentes, siempre y cuando tengan igual exponente. Por ejemplo, no se puede sumar directamente $103\cdot 10^{-2}$ con $235\cdot 10^{-1}$, pero sí se puede sumar $103\cdot 10^{-2}$ con $2350\cdot 10^{-2}$ porque ya tienen igual exponente y el resultado será $(103 + 2350)\cdot 10^{-2.}$

Codigos binarios

Código BCD

Dado que nosotros trabajamos con sistema decimal, y la máquinas en sistema binario es habitual este código.
Codifica un número decimal asignando a cada dígito de éste, su equivalente binario,
Como el valor máximo de un dígito decimal es 9 necesitaremos 4 bits para codificar cada dígito
Es un código ponderado 1, 2, 4, 8,, lo que hace que coincida el código de cada cifra decimal por separado con su conversión a binario puro.

Cifra decimal	Código binario
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
Valor	8 4 2 1

Código exceso 3

A cada número binario se le suma 3. La ventaja de este sistema es que es auto complementado esto es, si complementamos todos los bits, se obtiene el complemento A9 del número. , Otra de las ventajas de este código es que todos los números tienen al menos un BIT a 1, esto facilita distinguir entre cero y ausencia de información.

Cifra decimal	Código Exceso 3
0	0 0 1 1
1	0 1 0 0
2	0 1 0 1
3	0 1 1 0
4	0 1 1 1
5	1 0 0 0
6	1 0 0 1
7	1 0 1 0
8	1 0 1 1
9	1 1 0 0
Valor	8 4 2 1

Códigos alfanumericos

Código BCDIC
– Standard Binary Coded Decimal Interchange Code
– Usualmente este código utiliza 6 bits (hasta 64 caracteres)
– A veces se añade un bit adicional para verificar posibles errores de transmisión o grabación
– EBCDIC: Extended BCDIC. Utiliza 8 bits para representar cada carácter (hasta 256 símbolos distintos)

• **Código ASCII**
– American Standar Code for Information Interchange
– Versión reducida de 7 bits, usualmente se incluye un octavo bit para detección de errores de transmisión o grabación
– Versión extendida de 8 bits, con caracteres gráficos
– Hoy en día es de los más utilizados

Ejercicios sobre sistemas de representación.

1 Convertir a binario los siguientes números:

a) 145_{10} _____

b) 500_{10} _____

c) $323{,}375_{10}$ _____

d) $FA0{,}C7_{16}$ _____

2 Convertir a decimal los siguientes números binarios:

a) 1010101010.1 _____

b) 11100.011 _____

3 $27{,}1875_{10} =$ _____ $_2$

4 Convertir a hexadecimal los siguientes números binarios:

a) 1110111001 _____

b) 10010.011011 _____

5 $111_{16} =$ _____ $_{10} =$ _____ $_2$

6 El nº de 8 bits 01111111 en complemento a 2, representa el número _____ $)_{10}$

7 El nº de 8 bits 11111111 en complemento a 2, representa el número _____ $)_{10}$

8 El número decimal con signo –35 es igual a _____en complemento a 2 con 8 bits.

9 El número decimal con signo +35 es igual a _____en complemento a 2 con 8 bits.

10 00010111 = _____ BCD

11 Pasar a hexadecimal el número binario 1011100010011.00001111011101

_____

12 El mayor número entero sin signo representable con 5 bits es _____

Sistemas de numeración y codificación de la información

1.- Convertir a decimal los siguientes números binarios:

a) 1110 b) 11100 c) 10111

d) 110011,11 e) 101010,01 f) 1011100,10101

g) 1111111,11111

2.- Convertir a binario los siguientes números decimales:

a) 61 b) 93 c) 186

d) 0,0981 e) 789,0423 f) 90754

g) 21,02

3.- Realizar las siguientes operaciones aritméticas en binario:

a) 11 + 01 b) 1001 + 101 c) 110011 + 1001001

d) 1100 – 1001 e) 11010 – 10111 f) 0,100101 – 10100,01101

g) 1001 * 110 h) 111 * 101 i) 100 / 10

j) 1001 / 1101 k) 1110 / 1101

4.- Expresar en formato binario de 8 bits signo-modulo los siguientes números decimales:

a) +29 b) -85 c) +100 d) -123

5.- Expresar cada número decimal como un número de 8 bits en sistema de complemento a 1:

a) -34 b) +57 c) -99 d) +115

6.- Expresar cada número decimal como un número de 8 bits en sistema de complemento a 2:

a) +12 b) -68 c) +101 d) -125

7.- Determinar el valor decimal correspondiente a cada número binario con signo suponiendo que están

expresados en: 1) Signo-módulo 2) Complemento a 1 3) Complemento a 2

a) 10011001 b) 01110100 c) 10111111

8.- Convertir a decimal los siguientes números hexadecimales:

a) 23H b) 1AH c) 5C2H d) 700H e) 8DH

9.- Convertir a hexadecimal los siguientes números decimales:

a) 8 b) 33 c) 2890 d) 4019 e) 6500