UD 01_10 – Sistema Informático. Hardware.
Representación de la información.

CONTENIDO

1 GUÍA

- Lo que nos interesa principalmente de este tema es entender cómo se expresan los números en diferentes bases, en concreto en decimal, binario, octal y hexadecimal y cómo se realizan cambios de base para números enteros mayores o iguales que cero.
- También veremos brevemente cómo se realiza la codificación de caracteres alfanuméricos.
- El sistema binario y el cambio de base entre binario y decimal nos servirán en unidades posteriores para entender cómo se aplican las máscaras en redes TCP/IP. El resto de sistemas de numeración también son utilizados en el mundo de la informática o puedes necesitarlos en tu trabajo como programador.
- Aunque en las explicaciones que hay a continuación se trabaja en ocasiones con números con parte fraccionaria, los ejercicios los realizaremos sólo para números enteros, por lo que, si no tienes tiempo, no insistas en la conversión de la parte fraccionaria, se considera materia de ampliación.
- Al realizar los cambios de base, puedes comprobar que no te has equivocado utilizando una calculadora, pero asegúrate de que entiendes cómo funciona la construcción de los números en las diferentes bases y cómo es posible cambiar entre los distintos sistemas.
- Al final del tema tienes enlaces a algunos materiales de ampliación relacionados con las operaciones en binario, el Álgebra de Boole y la representación interna de números reales y los números con signo.
- En el último apartado se recogen algunos ejercicios para practicar con los distintos sistemas de numeración y manejar las tablas de caracteres y la codificación Unicode.

2 LA INFORMACIÓN Y SU REPRESENTACIÓN INTERNA.

El ordenador es la máquina que se utiliza para **procesar** (recoger, tratar, almacenar y mostrar) **información**. La transmisión de información entre las personas y los ordenadores puede hacerse de muchas maneras, mediante letras y números (caracteres alfanuméricos), como los introducidos a través de un teclado o los que vemos en un texto en el ordenador, mediante sonidos, como los introducidos a través de un micrófono o los que escuchamos a través de altavoces, mediante vídeos o imágenes, a través de cámaras, etc. Pero la forma de entender la información que tenemos las personas no es la misma que la utilizada en el ordenador, por lo que es necesaria una codificación o traducción.

Los ordenadores, debido a su construcción, solamente pueden trabajar en **forma binaria**. Un ordenador está compuesto de circuitos electrónicos sobre los cuales sólo se puede evaluar si hay o no hay corriente, si está activado o desactivado, encendido o apagado, si hay tensión o no la hay; por lo tanto, sólo se reconocen **dos estados o valores**:

- "1" si hay tensión o corriente en un punto
- "0" si no hay tensión.

Sin embargo, el ordenador, para mostrarnos la información numérica, no utiliza el **<u>sistema binario</u>**, sino que utiliza otros sistemas de numeración como son el **octal, hexadecimal y decimal**. En este tema nos centraremos en los sistemas de numeración y en la representación alfanumérica.

3 SISTEMAS DE NUMERACIÓN

Un **sistema de numeración** es el conjunto de símbolos y reglas que se utilizan para representar cantidades o datos numéricos (números). Estas reglas son diferentes para cada sistema de numeración considerado, pero una regla común a todos es que para construir números válidos en un sistema de numeración determinado sólo se pueden utilizar los símbolos permitidos en ese sistema (para indicar el sistema de numeración utilizado se añade su base como subíndice al número).

Ejemplos:

- el número $125_{(10)}$ es un número válido en el sistema decimal, pero el número $12A_{(10)}$ no lo es, ya que utiliza un símbolo *A* no válido en el sistema decimal.
- el número $35_{(8)}$ es un número válido en el sistema octal, pero el número $39_{(8)}$ no lo es, ya que el símbolo *9* no es un símbolo válido en el sistema octal.
- el número $F1E4_{(16)}$ es un número válido en el sistema hexadecimal, pero el número $FKE4_{(16)}$ no lo es, ya que el símbolo *K* no es un símbolo válido en el sistema hexadecimal.

En los siguientes apartados veremos las reglas y funcionamiento que rigen los sistemas de numeración más utilizados en el ámbito de la Informática.

Los sistemas de numeración que veremos serán **sistemas posicionales,** es decir, utilizan un **conjunto de símbolos cuyo significado o valor depende de su posición relativa al punto decimal**. En el sistema decimal, que es el que solemos utilizar a diario, sabemos que no tiene el mismo valor un 1 si está situado en la posición de las unidades que si está situado en la posición de las centenas.

Ejemplo de un sistema de numeración NO posicional sería la numeración romana. Por ejemplo, en el número 33 en decimal, el 3 representa unidades o decenas según el lugar que ocupa: se asigna un peso según la posición. En cambio, en numeración romana 33 sería XXXIII, los tres últimos dígitos representan unidades, y los tres primeros son decenas. O el número 100 decimal se representa con un único dígito, C.

Los sistemas de numeración se clasifican por su **base,** que es el número de representaciones distintas posibles con un sólo dígito. Nosotros estudiaremos los siguientes:

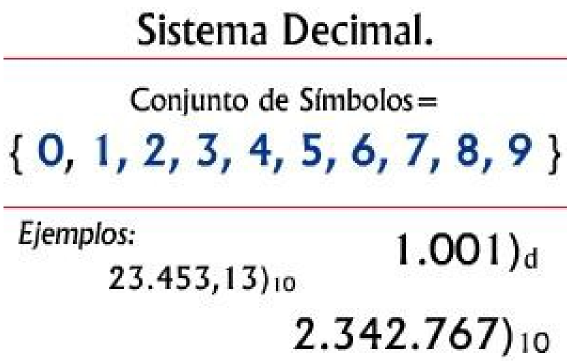
- Sistema Binario(base 2)
- Sistema Octal(base 8)
- Sistema Decimal(base 10)
- Sistema Hexadecimal(base 16)

3.1. SISTEMA DECIMAL

El sistema decimal es el más conocido por nosotros, pues es el que utilizamos todos los días.

- Los **símbolos o cifras** que utiliza el sistema decimal son los siguientes: **0 1 2 3 4 5 6 7 8 9**.

- La **base** de este sistema de numeración es **10**, que es también la cantidad de cifras o símbolos distintos que utiliza el sistema para la composición de los números.



En este sistema, un número se expresa como una cadena de estas cifras, donde cada cifra aporta un valor al número, valor que depende tanto del valor intrínseco de la propia cifra, como de la posición que ocupa en la cadena, al ser un sistema de numeración posicional.

Para relacionar una cantidad expresada en cualquier sistema de numeración con la misma cantidad expresada en sistema decimal, utilizamos el teorema fundamental de la numeración.

$$\boxed{N = \sum_{i=-d}^{n} (\text{dígito})_i * (\text{base})^i}$$

donde:

N = número en sistema decimal
base = base del sistema de numeración
i = posición que ocupa un dígito respecto a la coma
d = número de dígitos a la derecha de la coma
n = número de dígitos a la izquierda de la coma menos 1
dígito = cada uno de los que componen el número

Esta fórmula corresponde a la representación:

$$\ldots X_{-3}X_{-2}X_{-1}X_0 \,.\, X_1X_2X_3\ldots_{(b} = \ldots + X_3 * b^3 + X_2 * b^2 + X_1 * b^1 + X_0 * b^0 + X_{-1} * b^{-1} + X_{-2} * b^{-2} + X_{-3} * b^{-3} + \ldots$$

Supongamos una cantidad expresada en un sistema cuya base es **B** y representamos por Xi cada uno de los dígitos que contiene dicha cantidad, donde el subíndice indica la posición del dígito respecto a la coma o punto decimal, posición que hacia la izquierda de la coma se numera desde 0 en adelante y de 1 en 1, y hacia la derecha se numera desde -1 y con incremento -1.

Por ejemplo, la interpretación de las representaciones de las cantidades 1994 y 3.1416 del sistema decimal (base=10) será:

$$1994_{(10} = 1 * 10^3 + 9 * 10^2 + 9 * 10^1 + 4 * 10^0$$

$$3.1416_{(10} = 3 * 10^0 + 1 * 10^{-1} + 4 * 10^{-2} + 1 * 10^{-3} + 6 * 10^{-4}$$

El teorema fundamental de la numeración **relaciona una cantidad expresada en cualquier sistema de numeración con la misma cantidad expresada en el sistema decimal.**

Algunos ejemplos de uso del teorema fundamental de la numeración:

Ejemplo1: Supongamos la cantidad 201.1 expresada en el sistema de numeración de base 3 que utiliza los dígitos 0, 1 y 2 para la representación de cantidades. ¿Cuál será la representación de la misma cantidad en el sistema decimal?

$$201.1_{(3} = 2 * 3^2 + 0 * 3^1 + 1 * 3^0 + 1 * 3^{-1} = 18 + 0 + 1 + 0.333 = 19.333_{(10}$$

Ejemplo 2: Supongamos la cantidad 516 expresada en el sistema de numeración de base 7 que utiliza los dígitos 0, 1, 2, 3, 4, 5 y 6 para la representación de cantidades. ¿Cuál será la representación de la misma cantidad en el sistema decimal?

$$516_{(7} = 5 * 7^2 + 1 * 7^1 + 6 * 7^0 = 254 + 7 + 6 = 258_{(10}$$

Ejemplo 3: Supongamos la cantidad 0.111 expresada en el sistema de numeración de base 2 que utiliza los dígitos 0 y 1 para la representación de cantidades. ¿Cuál será la representación de la misma cantidad en el sistema decimal?

$$0.111_{(2} = 1 * 2^{-1} + 1 * 2^{-2} + 1 * 2^{-3} = 0.5 + 0.25 + 0.125 = 0.875_{(10}$$

(Recordad que $X^{-1}= 1/X^1$; es decir que $2^{-7}= 1/2^7= 0,0078125$)

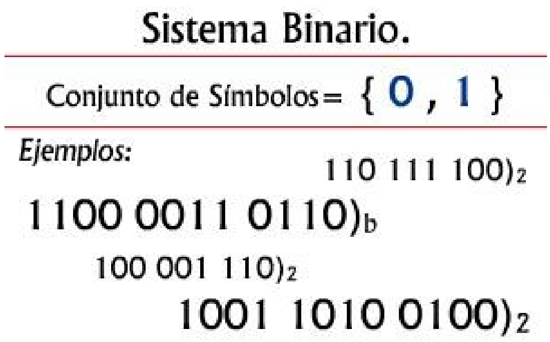
El teorema aplicado a la inversa nos sirve para obtener la representación de una cantidad decimal en cualquier otro sistema de numeración, por medio de divisiones sucesivas por la base, como veremos más adelante.

Trata de contestar a estas preguntas:

- ¿Cuántos números distintos se pueden representar con 1 cifra decimal?
  - Evidentemente, se pueden representar 10 números, los que van del 0 al 9.
- ¿Cuántos números distintos se pueden representar con 2 cifras decimales?
  - Con dos cifras se pueden representar 100 números, los que van del 0 (00) al 99.
- ¿Cuántos números distintos se pueden representar con 3 cifras decimales?
  - Con tres cifras decimales se pueden representar 1000 números, los que van del 0 (000) al 999.

Generalizando, ¿cuántos números distintos se pueden representar con n cifras decimales? **Con n cifras decimales se pueden representar 10^n números, los que van del 0 al 10^n – 1**.

3.2. SISTEMA BINARIO



- La **base** de este sistema de numeración es **2.**

- Los símbolos o dígitos que se utilizan para la representación de los números son exclusivamente los siguientes: **0 1**.

- Cada cifra o dígito de un número representado en este sistema se denomina **bit**, que es la menor unidad de información posible en un ordenador.

Así, por ejemplo:

1 bit = se refiere a un número de 1 dígito binario
2 bits = se refiere a un número de 2 dígitos binarios
. . .
n bits = se refiere a un número de n dígitos binarios

Al igual que el sistema decimal, el sistema binario es un sistema de numeración posicional, que recordemos que quiere decir que **el valor de cada cifra viene dado tanto por su valor intrínseco como por su posición dentro de la cadena de cifras que forman el número binario**.

Ejemplo:
Veamos a qué número decimal corresponde el número binario 1011 aplicando el teorema fundamental de la numeración:

$$1011_{(2} = (1*2^3) + (0*2^2) + (1*2^1) + (1*2^0) = 8 + 0 + 2 + 1 = 11_{(10}$$

En el apartado anterior hemos formulado una serie de preguntas cuyas respuestas habrás encontrado sencillas al tratarse del sistema decimal. Trata de responder a las mismas preguntas pero formuladas sobre el sistema de numeración binario:

- ¿Cuántos números distintos se pueden representar con 1 cifra binaria; es decir, con 1 bit?
  - Con un bit se pueden representar 2^1 números; es decir, 2 números, los que van del 0 al 1.
- ¿Cuántos números distintos se pueden representar con 2 cifras binarias?
  - Con dos bits se pueden representar 2^2 números; es decir, 4 números:

$00 = 0_{(10}$
$01 = 1_{(10}$
$10 = 2_{(10}$
$11 = 3_{(10}$

Es decir, los que van del 00 al 11, o lo que es lo mismo, los que van del 0 al 3 en decimal.

- ¿Cuántos números distintos se pueden representar con 3 cifras binarias?
  - Con tres bits se pueden representar 2^3 números; es decir, 8 números:

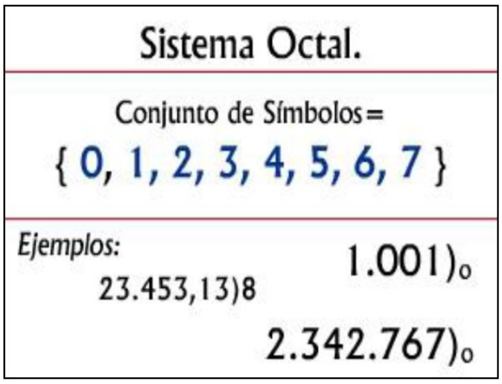
$000 = 0_{(10}$
$001 = 1_{(10}$
$010 = 2_{(10}$
$011 = 3_{(10}$
$100 = 4_{(10}$
$101 = 5_{(10}$
$110 = 6_{(10}$
$111 = 7_{(10}$

Es decir, los que van del 000 al 111, o lo que es lo mismo, los que van del 0 al 7 en decimal.

Generalizando, ¿cuántos números distintos se pueden representar con n cifras binarias? **Con n cifras binarias se pueden representar 2^n números, los que van del 0 al 2^n – 1**.

3.3. SISTEMA OCTAL

El sistema octal, al igual que el sistema decimal y el sistema binario, es un sistema de numeración de los llamados posicionales cuya **base** es **8** y que, por tanto, utiliza **ocho símbolos** o **cifras** distintas para componer sus números: **0 1 2 3 4 5 6 7**.



Ejemplo:
Utilizando el teorema fundamental de la numeración, vamos a averiguar a qué número decimal corresponde el número 54 octal.

Número octal $54_{(8} = (5*8^1) + (4*8^0) = 40 + 4 = 44_{(10}$

3.3.1. CONVERSIÓN ENTRE OCTAL Y BINARIO

Al ser 8 potencia de 2, la conversión entre las bases 8 y 2 es muy sencilla. Para pasar de octal a binario, pasamos cada dígito a binario utilizando tres bits. Para pasar de binario a octal, agrupamos en grupos de 3 bits empezando por la derecha y traducimos esos grupos a octal.

Tendremos en cuenta esta tabla de conversión:

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

Vamos a pasar el **número binario** 1011011100 **a octal**:

- Agrupamos de tres en tres empezando por la derecha:

  1 011 011 100

- Como el dígito de más a la izquierda queda sólo, podemos rellenar con ceros a la izquierda si así nos aclaramos mejor al principio:

  001 011 011 100

- Ahora buscamos la equivalencia de cada grupo de tres en la tabla anterior:

  001 bin = 1 oct

  011 bin = 3 oct

  011 bin = 3 oct

  100 bin = 4 oct

- Simplemente tenemos que agrupar los números en orden y tendremos el número en octal: 1334.

Para pasar de **octal a binario**, convertimos cada uno de los dígitos octales a grupos de tres dígitos binarios. Veamos un ejemplo:

- Convertimos el número 713 a binario. Para ello, utilizamos la tabla anterior:

  7 oct = 111 bin

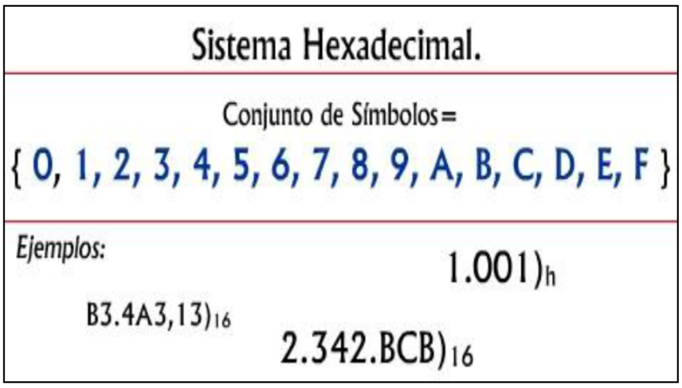
  1 oct = 001 bin

  3 oct = 011 bin

- Ahora agrupamos y ya tenemos el número en binario: 111001011 bin = 713 oct

3.4. SISTEMA HEXADECIMAL

- El sistema hexadecimal, al igual que los sistemas anteriores, es un sistema de numeración posicional, cuya **base** es **16**.
- El sistema hexadecimal permite expresar la información binaria de una forma compacta y sencilla, ya que con un dígito hexadecimal se puede representar un número de cuatro dígitos binarios, teniendo en cuenta que con cuatro dígitos binarios podemos representar 16 números diferentes.



Para la base 10, tenemos 10 dígitos diferentes: del 0 al 9; para la base 2, nos servimos de dos de esos dígitos que ya teníamos para la base 10: el 0 y el 1. Pero en la base 16, donde tenemos 16 dígitos diferentes, no podemos valernos sólo de los dígitos de la base decimal, ya que sólo hay 10 diferentes, y necesitamos 16.

La solución es utilizar letras para representar los 6 dígitos que nos faltan. Tenemos entonces que los dígitos hexadecimales son: **0 1 2 3 4 5 6 7 8 9 A B C D E F**. . A equivale a 10 en base 10. B equivale a 11 en base 10. C equivale a 12 en base 10. D equivale a 13 en base 10. E equivale a 14 en base 10. F equivale a 15 en base 10.

Ejemplos:
Número hexadecimal 54 = $(5*16^1) + (4*16^0) = 80 + 4 = 84_{(10}$
Número hexadecimal BC3 = $(3*16^0) + (C*16^1) + (B*16^2) = (3*16^0) + (12*16^1) + (11*16^2) = 3 + 192 + 2816 = 3011_{(10}$

Al igual que en octal, la base del sistema hexadecimal es una potencia de base 2, por lo que la conversión entre las bases 2 y 16 es inmediata, como en octal, sólo que ahora los grupos serán de cuatro dígitos binarios.

3.4.1. CONVERSIÓN ENTRE HEXADECIMAL Y BINARIO

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

Vamos a **pasar el número binario** 1011011100 a **hexadecimal**:

- Agrupamos de cuatro en cuatro empezando por la derecha:

  10 1101 1100

- Como a la izquierda quedan sólo dos dígitos, podemos rellenar con ceros a la izquierda si así nos aclaramos mejor al principio:

  0010 1101 1100

- Ahora buscamos la equivalencia de cada grupo de cuatro en la tabla anterior:

  0010 bin = 2 hex

  1101 bin = D hex

1100 bin = C hex

- Simplemente tenemos que agrupar los números en orden y tendremos el número en hexadecimal: 2DC.

Para **pasar de hexadecimal a binario**, convertimos cada uno de los dígitos hexadecimales a grupos de cuatro dígitos binarios. Veamos un ejemplo:

- Convertimos el número hexadecimal 713 a binario. Para ello, utilizamos la tabla anterior:

  7 oct = 0111 bin

  1 oct = 0001 bin

  3 oct = 0011 bin

- Ahora agrupamos y ya tenemos el número en binario: 011100010011 bin = 713 hex.

3.4.1.1 CÓMO REALIZAR LAS TABLAS DE EQUIVALENCIAS ANTERIORES

Para realizar las tablas de equivalencias anteriores, basta con tener en cuenta cuál es el peso de cada uno de los bits. Para cuatro bits, los pesos serán los siguientes:

$2^3 = 8$	$2^2 =4$	$2^1 =2$	$2^0= 1$

Cualquier número del 0 al 15 en decimal (0 a F hexadecimal) se podrá hacer como suma de las potencias anteriores multiplicadas por cero o por uno (lo que estamos aplicando aquí es el teorema fundamental de la numeración).

Por ejemplo, el número 9 será la suma de 8 + 1, así que las potencias correspondientes tendrán que estar multiplicadas por uno y el resto por cero. El 9 será en binario: 1 0 0 1

$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0= 1$
1	0	0	1

Podemos ir rellenando la tabla empezando por 0, viendo qué suma de potencias corresponde a cada número y poniendo 1 o 0 en el lugar de cada potencia según le corresponda:

- Para el 0 pondremos todas las potencias a 0.
- Para el 1, sólo estará a 1 la potencia $2^0 = 1$.
- Para el 2, sólo estará a 1 la potencia $2^1 = 2$.
- Para el 3: 3 = 2 + 1, así que estarán a 1 las potencias $2^1 =2$ y $2^0= 1$.
- Para el 4, sólo estará a 1 la potencia $2^2 = 4$.
- Para el 5: 5= 4 + 1, estarán a 1 las potencias $2^2 = 4$ y $2^0= 1$.
- etc.

3.5. CAMBIOS DE BASE

Nota: aunque en algunos casos se incluye también el cambio de la parte fraccionaria, recordad que lo vamos a considerar materia de ampliación, así que ese cambio podéis obviarlo y centraros sólo en la parte entera.

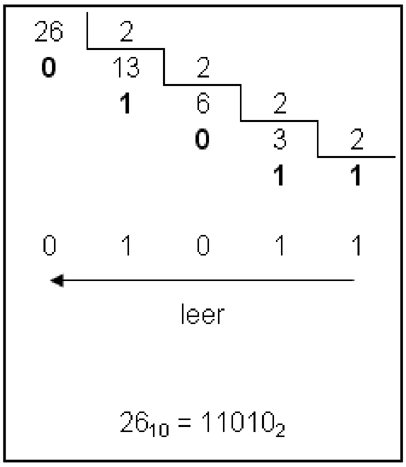
$0_{hex} = 0_{dec} = 0_{oct}$	0	0	0	0
$1_{hex} = 1_{dec} = 1_{oct}$	0	0	0	1
$2_{hex} = 2_{dec} = 2_{oct}$	0	0	1	0
$3_{hex} = 3_{dec} = 3_{oct}$	0	0	1	1
$4_{hex} = 4_{dec} = 4_{oct}$	0	1	0	0
$5_{hex} = 5_{dec} = 5_{oct}$	0	1	0	1
$6_{hex} = 6_{dec} = 6_{oct}$	0	1	1	0
$7_{hex} = 7_{dec} = 7_{oct}$	0	1	1	1
$8_{hex} = 8_{dec} = 10_{oct}$	1	0	0	0
$9_{hex} = 9_{dec} = 11_{oct}$	1	0	0	1
$A_{hex} = 10_{dec} = 12_{oct}$	1	0	1	0
$B_{hex} = 11_{dec} = 13_{oct}$	1	0	1	1
$C_{hex} = 12_{dec} = 14_{oct}$	1	1	0	0
$D_{hex} = 13_{dec} = 15_{oct}$	1	1	0	1
$E_{hex} = 14_{dec} = 16_{oct}$	1	1	1	0
$F_{hex} = 15_{dec} = 17_{oct}$	1	1	1	1

Tabla de equivalencia entre los números decimales del 0 al 15 y sus equivalentes en binario, octal y hexadecimal.
Fuente: https://es.wikipedia.org/wiki/Sistema_hexadecimal

3.5.1. CONVERSIÓN DE DECIMAL A BINARIO

Veamos cómo convertir un número representado en decimal a binario (este método también es útil para pasar de decimal a cualquier base, no sólo la binaria). Se irá dividiendo sucesivamente el número decimal y los cocientes que se van obteniendo entre 2 (la base) hasta que el cociente sea menor que 2. El número en binario será la secuencia del último cociente y todos los restos obtenidos en orden inverso.

Ejemplo:
Vamos a pasar el número 26 en decimal a sistema binario:



(No es necesario estudiar el cambio de la parte fraccionaria)
Si el número tiene parte fraccionaria, para pasarla a binario, multiplicamos sucesivamente la parte fraccionaria por 2 hasta que obtengamos 0. La parte entera de cada multiplicación formará los bits del número binario.

Ejemplo:
Supongamos que nuestro número es 26,625 en decimal. La parte entera ya la hemos calculado antes. A continuación calcularemos la parte fraccionaria:

0,625	* 2 = 1,250	1 (MSB, bit más significativo)
0,25	* 2 = 0,50	0
0,5	* 2 = 1,0	1 (LSB, bit menos significativo)

Finalmente concatenamos las dos secuencias obtenidas:

$26{,}625_{10} = 11010{,}101_2$

Otro método es el de distribución. Consiste en distribuir los unos necesarios entre las potencias sucesivas de 2 de modo que su suma resulte ser el número decimal a convertir.

Sea por ejemplo el número 151, para el que se necesitarán las 8 primeras potencias de 2, ya que la siguiente, 2^8=256, es superior al número a convertir.

Se comienza poniendo un 1 en 128, por lo que aún faltarán 23, 151-128 = 23, para llegar al 151. Este valor se conseguirá distribuyendo unos entre las potencias cuya suma dé el resultado buscado y poniendo ceros en el resto. En el ejemplo resultan ser las potencias 4, 2, 1 y 0, esto es, 16, 4, 2 y 1, respectivamente.

Ejemplo:

2^0= 1|1
2^1= 2|1
2^2= 4|1
2^3= 8|0
2^4= 16|1
2^5= 32|0
2^6= 64|0
2^7= 128|1

$128 + 16 + 4 + 2 + 1 = 151_{(10} = 10010111_{(2}$

3.5.2. CONVERSIÓN DE OCTAL A DECIMAL

Empleamos el teorema fundamental de la numeración

$$746{,}12_8 = 7 * \mathbf{8}^2 + 4 * \mathbf{8}^1 + 6 * \mathbf{8}^0 + 1 * \mathbf{8}^{-1} + 2 * \mathbf{8}^{-2} =$$
$$= 448 + 32 + 6 + 0{,}125 + 0{,}03125 =$$
$$= 486{,}15625_{10}$$

3.5.3. CONVERSIÓN DE HEXADECIMAL A DECIMAL

Empleamos el teorema fundamental de la numeración

$$F9{,}E3B_{16} = F * \mathbf{16}^1 + 9 * \mathbf{16}^0 + E * \mathbf{16}^{-1} + 3 * \mathbf{16}^{-2} + B * \mathbf{16}^{-3} =$$
$$= 15 * \mathbf{16}^1 + 9 * \mathbf{16}^0 + 14 * \mathbf{16}^{-1} + 3 * \mathbf{16}^{-2} + 11 * \mathbf{16}^{-3} =$$
$$= 249{,}8894042969_{10}$$

3.5.4. CONVERSIÓN DE DECIMAL A HEXADECIMAL

Aplicamos divisiones sucesivas. Hay que tener en cuenta que los restos de 10 a 15 habrá que pasarlos a hexadecimal utilizando la letra correspondiente.

Por ejemplo, vamos a pasar el número 3490 decimal a hexadecimal.

3490 | 16

2 → Resto 218 | 16

10 → Resto 13 → último cociente

↓ ↓

A D

El número será DA2 hex = 3490 dec

3.5.5. CONVERSIÓN ENTRE SISTEMAS BINARIO, OCTAL Y HEXADECIMAL

Para pasar de binario a octal, tomamos los bits de tres en tres empezando a contar de derecha a izquierda y cada grupo de tres lo pasamos a binario. Si nos faltan bits a la izquierda, rellenaremos con ceros.

Por ejemplo, si tenemos el siguiente número 1101011_2 la conversión sería:

001 101 011_2 = 153_8
0110 1011_2 = $6B_{16}$

La conversión a la inversa sería semejante utilizando las tablas vistas en apartados anteriores

217_8 = 010 001 111_2
$D40_{16}$ = 1101 0100 0000_2

Ejemplos:

- de Octal y Hexadecimal a Binario
  $15{,}36_8$ = 001 101 , 011 110 $_2$
  $F9{,}E3B_{16}$ = 1111 1001 , 1110 0011 1011 $_2$

- de Binario a Octal y Hexadecimal
  111000011011.10000001_2 = 111 000 011 011 , 100 000 010 $_2$ =
  = 7 0 3 3 , 4 0 2 $_8$
  = 1110 0001 1011 , 1000 0001 $_2$ =
  = E 1 B , 8 1 $_{16}$

4 REPRESENTACIÓN ALFANUMÉRICA

A cada símbolo del conjunto {0, 1,... ,9, A, B, C,... X, Y, Z,+,-,*,...} se le puede **asociar** una combinación arbitraria de **señales binarias** que usualmente suele constar de palabras del mismo tamaño.

Las características de esta representación son las siguientes:

- Longitud del código que se utiliza para representar cada carácter (de ella depende el número de caracteres distintos representable).
- Codificación de cada carácter.

Debe destacarse que no existe un criterio determinado que indique qué código debe utilizarse para representar un carácter alfanumérico concreto (no existe ninguna regla que indique qué código es el más adecuado para representar la letra “a”, por ejemplo). Por este motivo se utilizan convenios, algunos de los cuales se han convertido en estándares con el paso del tiempo.

4.1. EJEMPLOS DE CÓDIGOS

4.1.1. A.S.C.I.I. (AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE)

- Su longitud es fija, igual para todos los códigos.
- El ASCII original tenía una longitud de código de 7 bits.
- Con el ASCII extendido se realizó una ampliación para caracteres internacionales, con una longitud de código de 8 bits.

En la siguiente tabla podemos ver el código, expresado en decimal y en hexadecimal, que corresponde a cada símbolo (números, caracteres, etc.).

Caracteres de control ASCII

DEC	HEX	Simbolo ASCII	
00	00h	NULL	(carácter nulo)
01	01h	SOH	(inicio encabezado)
02	02h	STX	(inicio texto)
03	03h	ETX	(fin de texto)
04	04h	EOT	(fin transmisión)
05	05h	ENQ	(enquiry)
06	06h	ACK	(acknowledgement)
07	07h	BEL	(timbre)
08	08h	BS	(retroceso)
09	09h	HT	(tab horizontal)
10	0Ah	LF	(salto de linea)
11	0Bh	VT	(tab vertical)
12	0Ch	FF	(form feed)
13	0Dh	CR	(retorno de carro)
14	0Eh	SO	(shift Out)
15	0Fh	SI	(shift In)
16	10h	DLE	(data link escape)
17	11h	DC1	(device control 1)
18	12h	DC2	(device control 2)
19	13h	DC3	(device control 3)
20	14h	DC4	(device control 4)
21	15h	NAK	(negative acknowle.)
22	16h	SYN	(synchronous idle)
23	17h	ETB	(end of trans. block)
24	18h	CAN	(cancel)
25	19h	EM	(end of medium)
26	1Ah	SUB	(substitute)
27	1Bh	ESC	(escape)
28	1Ch	FS	(file separator)
29	1Dh	GS	(group separator)
30	1Eh	RS	(record separator)
31	1Fh	US	(unit separator)
127	20h	DEL	(delete)

Caracteres ASCII imprimibles

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_	elCodigoASCII.com.ar		

ASCII extendido

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
128	80h	Ç	160	A0h	á	192	C0h	└	224	E0h	Ó
129	81h	ü	161	A1h	í	193	C1h	┴	225	E1h	ß
130	82h	é	162	A2h	ó	194	C2h	┬	226	E2h	Ô
131	83h	â	163	A3h	ú	195	C3h	├	227	E3h	Ò
132	84h	ä	164	A4h	ñ	196	C4h	─	228	E4h	õ
133	85h	à	165	A5h	Ñ	197	C5h	┼	229	E5h	Õ
134	86h	å	166	A6h	ª	198	C6h	ã	230	E6h	µ
135	87h	ç	167	A7h	º	199	C7h	Ã	231	E7h	þ
136	88h	ê	168	A8h	¿	200	C8h	╚	232	E8h	Þ
137	89h	ë	169	A9h	®	201	C9h	╔	233	E9h	Ú
138	8Ah	è	170	AAh	¬	202	CAh	╩	234	EAh	Û
139	8Bh	ï	171	ABh	½	203	CBh	╦	235	EBh	Ù
140	8Ch	î	172	ACh	¼	204	CCh	╠	236	ECh	ý
141	8Dh	ì	173	ADh	¡	205	CDh	═	237	EDh	Ý
142	8Eh	Ä	174	AEh	«	206	CEh	╬	238	EEh	¯
143	8Fh	Å	175	AFh	»	207	CFh	¤	239	EFh	´
144	90h	É	176	B0h	░	208	D0h	ð	240	F0h	
145	91h	æ	177	B1h	▒	209	D1h	Ð	241	F1h	±
146	92h	Æ	178	B2h	▓	210	D2h	Ê	242	F2h	‗
147	93h	ô	179	B3h	│	211	D3h	Ë	243	F3h	¾
148	94h	ö	180	B4h	┤	212	D4h	È	244	F4h	¶
149	95h	ò	181	B5h	Á	213	D5h	ı	245	F5h	§
150	96h	û	182	B6h	Â	214	D6h	Í	246	F6h	÷
151	97h	ù	183	B7h	À	215	D7h	Î	247	F7h	¸
152	98h	ÿ	184	B8h	©	216	D8h	Ï	248	F8h	°
153	99h	Ö	185	B9h	╣	217	D9h	┘	249	F9h	¨
154	9Ah	Ü	186	BAh	║	218	DAh	┌	250	FAh	·
155	9Bh	ø	187	BBh	╗	219	DBh	█	251	FBh	¹
156	9Ch	£	188	BCh	╝	220	DCh	▄	252	FCh	³
157	9Dh	Ø	189	BDh	¢	221	DDh	¦	253	FDh	²
158	9Eh	×	190	BEh	¥	222	DEh	Ì	254	FEh	■
159	9Fh	ƒ	191	BFh	┐	223	DFh	▀	255	FFh	

Fuente: elcodigoascii.com.ar

4.1.2. E.B.C.D.I.C. (EXTENDED BINARY CODED DECIMAL INTERCHANGE CODE)

- Surgido en 1964 con el sistema IBM S360.
- Longitud fija de 8 bits.
- Sólo se usa en algunos sistemas mainframe[1]. Tiende a desaparecer.
- Más información: https://en.wikipedia.org/wiki/EBCDIC

4.1.3. UNICODE

- Unicode incluye todos los caracteres de uso común en la actualidad.

- La versión de 2017, la 10.0, contiene unos 136 000 caracteres provenientes de distintos alfabetos, sistemas ideográficos y colecciones de símbolos (matemáticos, técnicos, musicales, iconos, emojis...).

- Lista de caracteres Unicode:
  - http://www.unicode.org/charts/
  - https://en.wikipedia.org/wiki/List_of_Unicode_characters

- Más información:
  - http://www.unicode.org/standard/WhatIsUnicode.html
  - http://es.wikipedia.org/wiki/Unicode

- En la siguiente tabla se muestran los caracteres Unicode para alfabeto latino, y estos coinciden con el código ASCII. El código viene dado en hexadecimal y además de poder leerse debajo de cada carácter, se puede formar a partir de la numeración que encontramos en la parte de arriba de la tabla junto con la de la izquierda (primero se pondría el código de arriba y a continuación el de la izquierda).

1 https://es.wikipedia.org/wiki/Computadora_central

	000	001	002	003	004	005	006	007
0	NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	` 0060	p 0070
1	SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
2	STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
3	ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
4	EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
5	ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
6	ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
7	BEL 0007	ETB 0017	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
8	BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
9	HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
A	LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
B	VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
C	FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
D	CR 000D	GS 001D	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
E	SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
F	SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

C0 Controls and Basic Latin. Fuente: http://www.unicode.org/charts/PDF/U0000.pdf

4.1.4. UCS (UNIVERSAL CODED CHARACTER SET) O ISO/IEC 10646

Sus estándares y los de Unicode Consortium prácticamente van a la par. Por ejemplo, el estándar ISO/IEC 10646:2017 es como la versión 10 de Unicode, excluyendo algunos caracteres y símbolos.

Más información: https://en.wikipedia.org/wiki/Universal_Coded_Character_Set

5 AMPLIACIÓN

5.1. OPERACIONES EN BINARIO

En el siguiente enlace se explica cómo realizar sumas, restas, multiplicaciones y divisiones en binario: https://es.wikipedia.org/wiki/Sistema_binario#Operaciones_con_n.C3.BAmeros_binarios

5.2. ÁLGEBRA DE BOOLE

En este documento se explican las funciones lógicas del Álgebra de Boole y la simplificación de expresiones lógicas a través de los mapas de Karnaugh: http://arantxa.ii.uam.es/~ig/teoria/temas/IG_tema-5-2008-2009.pdf

5.3. REPRESENTACIÓN INTERNA DE LOS DATOS

5.3.1. REPRESENTACIÓN DE NÚMEROS CON SIGNO

https://es.wikipedia.org/wiki/Representaci%C3%B3n_de_n%C3%BAmeros_con_signo

5.3.2. REPRESENTACIÓN DE NÚMEROS EN COMA FLOTANTE

https://es.wikipedia.org/wiki/Coma_flotante

6 BIBLIOGRAFÍA

- A. Ramos, M.J. Ramos, S. Viñas. *Montaje y mantenimiento de equipos*. Ed. McGrawHill
- Unicode Consortium: http://www.unicode.org/
- https://es.wikipedia.org/wiki/Sistema_binario
- https://es.wikipedia.org/wiki/Sistema_hexadecimal
- https://es.wikipedia.org/wiki/Sistema_octal
- Libros sobre electrónica digital:
  - http://books.google.es/books?id=9HWEodKxTCYC&printsec=frontcover#v=onepage&q=&f=false (Roger L. Tokheim. Electrónica digital.)
  - http://books.google.es/books?id=bmLuH0CsIh0C&printsec=frontcover&dq=tocci&ei=A9rFSsqdKpv-yAS4mJyJBA#v=onepage&q=&f=false (Tocci. Sistemas digitales. Principios y aplicaciones)

7 EJERCICIOS

1. ¿Cuáles de los siguientes números son válidos en el sistema binario?

   1111111, 0000000, 101001, 0, 10, 2, A

2. ¿Cuáles de los siguientes números son válidos en octal?

   34, 81, B21, 7777

3. ¿Cuáles de los siguientes números son válidos en hexadecimal?

   B21, 7777, 101001, 0, 10, 2, A, F8A1B, M12

4. Comprueba que sabes elaborar las tablas de equivalencia entre sistema binario-octal y binario-hexadecimal.
5. Pasa los números binarios 100010011 y 1001110 a octal.
6. ¿Cómo representarías el número en octal 453 en binario?
7. ¿Cómo representarías el número decimal 598 en hexadecimal? ¿Y el 332?
8. ¿Cómo representarías el número binario 100111101100 en hexadecimal? ¿Y el número binario 1001110010011?
9. ¿Cómo representarías el número hexadecimal ABC en binario?
10. ¿Cómo representarías el número octal 231 en hexadecimal? ¿Y el número 3A hexadecimal en octal?
11. Convertir a hexadecimal y a binario las siguientes cantidades en decimal:

    757 , 123, 356

12. ¿Qué código Unicode corresponde al carácter d?¿Coincide con el código ASCII de d? ¿Cómo se escribiría ese código en sistema decimal?
13. ¿Qué carácter corresponde al código Unicode $01011010_{(2}$?
14. La letra ñ no la encontrarás en la tabla Unicode anterior, tienes que buscar en otra. Puedes encontrarla entre los distintos alfabetos que encuentras aquí: http://www.unicode.org/charts/ - European Scripts – Latin-1 Supplement. ¿Qué código Unicode le corresponde?