SISTEMAS DE NUMERACIÓN

Un sistema de numeración es un conjunto de símbolos y reglas que permiten representar datos numéricos. La norma principal en un sistema de numeración posicional es que ***un mismo símbolo tiene distinto valor según la posición que ocupe***.

Sistema de numeración decimal:

El sistema de numeración que utilizamos habitualmente es el **decimal,** que se compone de diez símbolos o dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9) a los que otorga un valor **dependiendo de la posición** que ocupen en la cifra: unidades, decenas, centenas, millares, etc.

El valor de cada dígito está asociado al de una potencia de base 10, número que coincide con la cantidad de símbolos o dígitos del sistema decimal, y un exponente igual a la posición que ocupa el dígito menos uno, contando desde la derecha.

En este sistema el número 528, por ejemplo, significa:

5 centenas + 2 decenas + 8 unidades, es decir:

500 + 20 + 8 o, lo que es lo mismo,

$$5\cdot10^{2}+2\cdot10^{1}+8\cdot10^{0}=528$$

En el caso de números con decimales, la situación es análoga aunque, en este caso, algunos exponentes de las potencias serán negativos, concretamente el de los dígitos colocados a la derecha del separador decimal. Por ejemplo, el número 8245,97 se calcularía como:

8 millares + 2 centenas + 4 decenas + 5 unidades + 9 décimos + 7 céntimos

8000 + 200 + 40 + 5 + 0,9 + 0,07 = 8245,97

$$8\cdot10^{3}+2\cdot10^{2}+4\cdot10^{1}+5\cdot10^{0}+9\cdot10^{-1}+7\cdot10^{-2}=8245{,}97$$

Sistema de numeración binario.

El sistema de numeración binario utiliza sólo dos dígitos, el cero (0) y el uno (1), que tienen distinto valor dependiendo de la posición que ocupen. El valor de cada posición es el de una potencia de base 2, elevada a un exponente igual a la posición del dígito menos uno. Se puede observar que, tal y como ocurría con el sistema decimal, la base de la potencia coincide con la cantidad de dígitos utilizados (2) para representar los números.

De acuerdo con estas reglas, el número binario 1011 tiene un valor que se calcula así:

$$1\cdot2^{3}+0\cdot2^{2}+1\cdot2^{1}+1\cdot2^{0}=8+0+2+1=11$$

y lo escribimos así: $1011_{2}=11_{10}$

Conversión entre números decimales y binarios

Convertir un número decimal al sistema binario es muy sencillo: basta con realizar divisiones sucesivas por 2 y colocar los restos obtenidos, en cada una de ellas. Para formar el número binario tomaremos los restos **en orden inverso** al que han sido obtenidos. Por ejemplo:

77 : 2 = 38 Resto: 1
38 : 2 = 19 Resto: 0
19 : 2 = 9 Resto: 1
9 : 2 = 4 Resto: 1
4 : 2 = 2 Resto: 0
2 : 2 = 1 Resto: 0
1 : 2 = 0 Resto: 1

$77_{10} = 1\ 0\ 0\ 1\ 1\ 0\ 1_2$

La cantidad de dígitos necesarios, para representar un número en el sistema binario, dependerá del valor de dicho número en el sistema decimal. En el caso anterior, para representar el número 77 han hecho falta siete dígitos. Para representar números superiores harán falta más dígitos. Por ejemplo, para representar números mayores de 255 se necesitarán más de ocho dígitos, porque $2^8=256$ y, por tanto, 255 es el número más grande que puede representarse con ocho dígitos.

Es importante distinguir entre los números que pueden representarse con n dígitos binarios, que es 2^n, y el mayor de esos números, que es una unidad menos, es decir, $2^n - 1$.

El proceso para convertir un número del sistema binario al decimal es aún más sencillo; basta con desarrollar el número, teniendo en cuenta que el valor de cada dígito está asociado a una potencia de 2, cuyo exponente es 0 en el bit situado más a la derecha, y se incrementa en una unidad según vamos avanzando posiciones hacia la izquierda, tal y como se muestra en el siguiente ejemplo:

$$1010011=1\cdot2^6+0\cdot2^5+1\cdot2^4+0\cdot2^3+0\cdot2^2+1\cdot2^1+1\cdot2^0=83$$

$$1010011_2 = 83_{10}$$

SISTEMAS DE NUMERACIÓN OCTAL Y HEXADECIMAL

El inconveniente de la codificación binaria es que la representación de algunos números resulta muy larga. Por este motivo se utilizan otros sistemas de numeración que resulten más cómodos de escribir: el sistema octal y el sistema hexadecimal. Afortunadamente, resulta muy fácil convertir un número binario a octal o a hexadecimal.

Sistema de numeración octal

En el sistema octal, los números se representan mediante **ocho** dígitos diferentes: 0, 1, 2, 3, 4, 5, 6 y 7. Cada dígito tiene, naturalmente, un valor distinto dependiendo del lugar que ocupen. El valor de cada una de las posiciones viene determinado por las potencias de base 8. La conversión de un número decimal a octal, y viceversa, se realiza del mismo modo que la de los números binarios, aunque, lógicamente, se emplea como base el número 8 en vez del 2.

La conversión de un número decimal a octal se hace del mismo modo: mediante divisiones sucesivas por 8 y colocando los restos obtenidos en orden inverso. Por ejemplo:

122 : 8 = 15 Resto: 2
15 : 8 = 1 Resto: 7
1 : 8 = 0 Resto: 1

$122_{10} = 172_8$

La conversión de un número octal a decimal es igualmente sencilla. Por ejemplo:

$$237_8 = 2 \cdot 8^2 + 3 \cdot 8^1 + 7 \cdot 8^0 = 128 + 24 + 7 = 159_{10}$$

$$237_8 = 159_{10}$$

SISTEMA DE NUMERACIÓN HEXADECIMAL

En este sistema, los números se representan con dieciséis símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F. Se utilizan los caracteres A, B, C, D, E y F representando las cantidades decimales 10, 11, 12, 13, 14 y 15 respectivamente, porque no hay dígitos mayores que 9 en el sistema decimal. El valor de cada uno de estos símbolos depende, como es lógico, de su posición, que se calcula mediante potencias de base 16.

Ensayemos la conversión decimal a hexadecimal del número 1735:

1735 : 16 = 108	Resto: 7	
108 : 16 = 6	Resto: C (12_{10})	$1735_{10} = 6C7_{16}$
6 : 16 = 0	Resto: 6	

Ensayemos también la conversión inversa, de hexadecimal a decimal del número 1A3F:

$$1\,A3F_{16} = 1\cdot 16^3 + A\cdot 16^2 + 3\cdot 16^1 + F\cdot 16^0 = 6719_{10}$$

$$1A3F_{16} = 6719_{10}$$

Conversión de números binarios a octales y hexadecimales

Cada dígito de un número octal equivale a tres dígitos en el sistema binario. Por tanto, el modo de convertir un número entre estos sistemas de numeración equivale a "expandir" cada dígito octal a tres dígitos binarios, o en "contraer" grupos de tres caracteres binarios a su correspondiente dígito octal. Por ejemplo:

$$101001011_2 = 513_8$$

$$750_8 = 111101000_2$$

Análogamente, la conversión entre números hexadecimales y binarios se realiza "expandiendo" o "contrayendo" cada dígito hexadecimal a cuatro dígitos binarios. Por ejemplo:

$$101001110011_2 = A73_{16}$$

$$1F6_{16} = 000111110110_2$$

En caso de que los dígitos binarios no formen grupos completos (de tres o cuatro dígitos, según corresponda), se deben añadir ceros a la izquierda hasta completar el último grupo. Por ejemplo:

$$101110_2 = 00101110_2 = 2E_{16}$$

ARITMÉTICA BINARIA

La Unidad Aritmético Lógica, en la CPU del procesador, es capaz de realizar operaciones aritméticas, con datos numéricos expresados en el sistema binario. Naturalmente, esas operaciones incluyen la adición, la sustracción, el producto y la división. Las operaciones se hacen del mismo modo que en el sistema decimal, pero debido a la sencillez del sistema de numeración, pueden hacerse algunas simplificaciones que facilitan mucho la realización de las operaciones.

SUMA EN BINARIO

La tabla de sumar, en binario, es mucho más sencilla que en decimal. Sólo hay que recordar cuatro combinaciones posibles. Recuerda que en el sistema decimal había que memorizar unas 100 combinaciones.

SUMA	0	1
0	0	1
1	1	0 + a

Las sumas 0+0, 0+1 y 1+0 son evidentes:

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

Pero la suma de 1+1, que sabemos que es 2, debe escribirse en binario con dos cifras (10) y, por tanto 1+1 es 0 y se arrastra una unidad, que se suma a la posición siguiente a la izquierda.

Veamos algunos ejemplos:

$$\begin{array}{r} 010 \\ 101 \\ \hline 111 \end{array} \qquad \begin{array}{r} 2_{10} \\ 5_{10} \\ \hline 7_{10} \end{array} \qquad \begin{array}{r} 001101 \\ 100101 \\ \hline 110010 \end{array} \qquad \begin{array}{r} 13_{10} \\ 37_{10} \\ \hline 50_{10} \end{array}$$

$$\begin{array}{r} 1011011 \\ 1011010 \\ \hline 10110101 \end{array} \qquad \begin{array}{r} 91_{10} \\ 90_{10} \\ \hline 181_{10} \end{array} \qquad \begin{array}{r} 110111011 \\ 100111011 \\ \hline 1011110110 \end{array} \qquad \begin{array}{r} 443_{10} \\ 315_{10} \\ \hline 758_{10} \end{array}$$

SUSTRACCIÓN EN BINARIO

RESTA	0	1
0	0	1
1	1 + a	0

Restar en binario es, nuevamente, igual que la misma operación en el sistema decimal. Pero conviene repasar la operación de restar en decimal para comprender la operación binaria, que es más sencilla. Los términos que intervienen en la resta se llaman minuendo, sustraendo y diferencia.

Las sumas 0-0, 1-0 y 1-1 son evidentes:

$0 - 0 = 0$

$1 - 0 = 1$

$1 - 1 = 0$

La resta 0 - 1 se resuelve, igual que en el sistema decimal, tomando una unidad prestada de la posición siguiente: 10 -1, es decir, $2_{10} - 1_{10} = 1$

Esa unidad prestada debe devolverse, sumándola, a la posición siguiente. Veamos algunos ejemplos:

$$\begin{array}{r} 111 \\ 101 \\ \hline 010 \end{array} \qquad \begin{array}{r} 7_{10} \\ 5_{10} \\ \hline 2_{10} \end{array} \qquad\qquad \begin{array}{r} 10001 \\ 01010 \\ \hline 00111 \end{array} \qquad \begin{array}{r} 17_{10} \\ 10_{10} \\ \hline 7_{10} \end{array}$$

$$\begin{array}{r} 11011001 \\ 10101011 \\ \hline 00101110 \end{array} \qquad \begin{array}{r} 217_{10} \\ 171_{10} \\ \hline 46_{10} \end{array} \qquad\qquad \begin{array}{r} 111101001 \\ 101101101 \\ \hline 001111100 \end{array} \qquad \begin{array}{r} 489_{10} \\ 365_{10} \\ \hline 124_{10} \end{array}$$

A pesar de lo sencillo que es el procedimiento de restar, es facil confundirse. Tenemos interiorizado el sistema decimal y hemos aprendido a restar mecánicamente, sin detenernos a pensar en el significado del arrastre. Para simplificar las restas y reducir la posibilidad de cometer errores hay varias soluciones:

- Dividir los números largos en grupos. En el siguiente ejemplo, vemos cómo se divide una resta larga en tres restas cortas:

$$\begin{array}{r} 100110011101 \\ 010101110010 \\ \hline 010000101011 \end{array} = \begin{array}{r} 1001 \\ 0101 \\ \hline 0100 \end{array} \quad \begin{array}{r} 1001 \\ 0111 \\ \hline 0010 \end{array} \quad \begin{array}{r} 1101 \\ 0010 \\ \hline 1011 \end{array}$$

- Utilizando el **Complemento a dos**

Complemento a dos

El ***complemento a dos*** de un número ***N***, con ***n*** cifras, se define como $C_2^N = 2^n - N$.

Veamos un ejemplo: tomemos el número $N = 101101_2$ que tiene 6 cifras, y calculemos el complemento a dos de ese número:

$N = 45_{10}$ $n = 6$ $2^6 = 64$ y, por tanto: $C_2^N = 64 - 45 = 19 = 010011_2$

Complemento a uno

El ***complemento a uno*** de un número ***N***, con ***n*** cifras es, por definición, una unidad menor que el complemento a dos, es decir:

$$C_1^N = C_2^N - 1 \quad \text{y, por la misma razón,} \quad C_2^N = C_1^N + 1$$

Calculemos el ***complemento a uno*** del mismo número del ejemplo anterior:

$C_1^N = C_2^N - 1$

$$\begin{array}{r} 010011 \\ 000001 \\ \hline 010010 \end{array}$$

$C_1^N = 010010$

Da la sensación de que no va a ser más sencillo restar utilizando el complemento a dos, porque el procedimiento para calcular el complemento a dos es más difícil y laborioso que la propia resta. Pero es mucho más sencillo de lo que parece.

En realidad, el ***complemento a uno*** de un número binario es el número resultante de invertir UNOS y CEROS.

Si $N = 101101$

su ***complemento a uno*** es: $C_1^N = 010010$

y su ***complemento a dos*** es: $C_2^N = C_1^N + 1 = 010011$

Veamos otro ejemplo de cálculo de complementos:

Si $N = 0110110101$

El complemento a uno es: $C_1^N = 1001001010$ $C_1^N = 1001001010$

y el complemento a dos es: $C_2^N = 1001001011$

Restar en binario usando el complemento a dos

Y, por fin, vamos a ver cómo facilita la resta el complemento. La resta binaria de dos números puede obtenerse ***sumando al minuendo el complemento a dos del sustraendo***. Veamos algunos ejemplos:

a) Hagamos la siguiente resta, 91 – 46 = 45, en binario:

$$\begin{array}{r} 1011011 \\ 0101110 \\ \hline 0101101 \end{array} \qquad \begin{array}{r} 91_{10} \\ 46_{10} \\ \hline 45_{10} \end{array}$$

Tiene alguna dificultad, cuando se acumulan los arrastres a la resta siguiente. Pero esta misma resta puede hacerse como una suma, utilizando el complemento a dos del sustraendo:

$$\begin{array}{r} 1011011 \\ 1010010 \\ \hline 10101101 \end{array}$$

En el resultado nos sobra un bit, que se desborda por la izquierda. Como el número resultante no puede ser más largo que el minuendo, el bit sobrante se desprecia.

b) Hagamos esta otra resta, 219 – 23 = 196, utilizando el complemento a dos:

$219_{10} = 11011011_2$
$23_{10} = 00010111_2$

$C_2^{23} = 11101001$

$$\begin{array}{r} 11011011 \\ 11101001 \\ \hline 111000100 \end{array}$$

Y, despreciando el bit que se desborda por la izquierda, llegamos al resultado correcto:

$11000100_2 = 196_{10}$

¡Qué fácil!

MULTIPLICACIÓN BINARIA

POR	0	1
0	0	0
1	0	1

La multiplicación en binario es más fácil que en cualquier otro sistema de numeración.

Como los factores de la multiplicación sólo pueden ser CEROS o UNOS, el producto sólo puede ser CERO o UNO. En otras palabras, la tabla de multiplicar es muy fácil de aprender

En un ordenador, sin embargo, la operación de multiplicar se realiza mediante sumas repetidas. Eso crea algunos problemas en la programación porque cada suma de dos UNOS origina un arrastre, que se resuelven contando el número de UNOS y de arrastres en cada columna. Si el número de UNOS es par, la suma es un CERO y si es impar, un UNO. Luego, para determinar los arrastres a la posición superior, se cuentan las parejas de UNOS.

DIVISIÓN BINARIA

Igual que en el producto, la división es muy fácil de realizar, porque no son posibles en el cociente otras cifras que UNOS y CEROS.

Consideremos el siguiente ejemplo, 42 : 6 = 7, en binario:

```
(Dividendo)    1 0 1 0 1 0  | 110  (Divisor)
               - 1 1 0      | 111  (Cociente)
                 1 0 0 1
                 - 1 1 0
                   0 1 1 0
                     1 1 0
                     0 0 0
```

Se intenta dividir el dividendo por el divisor, empezando por tomar en ambos el mismo número de cifras (100 entre 110, en el ejemplo). Si no puede dividirse, se intenta la división tomando un dígito más (1001 entre 100).

Si la división es posible, entonces, el divisor sólo podrá estar contenido **una vez** en el dividendo, es decir, la primera cifra del cociente es un UNO. En ese caso, el resultado de multiplicar el divisor por 1 es el propio divisor. Restamos las cifras del dividendo del divisor y bajamos la cifra siguiente.

El procedimiento de división continúa del mismo modo que en el sistema decimal.

EJERCICIOS

1. Expresa, en código binario, los números decimales siguientes:
   c) 47
   d) 191
   e) 25
   f) 67
   g) 99
   h) 135
   i) 276.

2. Expresa, en el sistema decimal, los siguientes números binarios:
   a) 110111
   b) 111000
   c) 010101
   d) 101010
   e) 1111110

3. Dados dos números binarios: 01001000 y 01000100 ¿Cuál de ellos es el mayor? ¿Podrías compararlos sin necesidad de convertirlos al sistema decimal?

4. ¿Cuántos números diferentes se pueden escribir, utilizando el sistema binario de numeración, con sólo 3 dígitos? ¿Y con 16 dígitos?

5. Convierte los siguientes números octales en decimales:
   a) 45_8
   b) 125_8
   c) 625_8

6. Convierte los siguientes números decimales en octales:
   a) 63
   b) 513
   c) 119

7. Convierte los siguientes números binarios en octales:
   a) 1101101
   b) 101110
   c) 11011011
   d) 101101011

8. Convierte los siguientes números octales en binarios:
   a) 25_8
   b) 372_8
   c) 2753_8

9. Realiza las siguientes sumas de números binarios:
   a) 111011 + 110
   b) 111110111 + 111001
   c) 10111 + 11011 + 10111

10. Realiza las siguientes sumas de números octales:
    a) 365 + 23
    b) 2732 + 1265
    c) 65 + 1773

11. Suma los siguientes números hexadecimales:
    a) 17A + 3C
    b) 20F5 + 31B
    c) 2E70C + 1AA7F

12. Realiza las siguientes restas de números binarios:
    a) 111011 - 110
    b) 111110111 - 111001
    c) 1010111 - 11011 – 10011

13. Resta los siguientes números octales:
    a) 365 - 23
    b) 2732 - 1265
    c) 1773 – 65

14. Realiza las siguientes restas de números hexadecimales:
    a) 17A - 3C
    b) 20F5 - 31B
    c) 2E70C - 1AA7F