Parte III – Sistemas de representación de información

1. Introducción

Los ordenadores necesitan información con la que trabajar para poder resultar útiles. Si bien, dado que están compuestos de circuitos electrónicos, nos encontramos con el problema de que sólo trabajan con **dos estados**: el **paso de corriente** (que se representa con un **1**) y el **no paso de corriente** (que se representa con un **0**).

Esta característica de los equipos electrónicos genera la necesidad de transformar los datos, tanto numéricos como alfanuméricos, en una representación binaria para que el ordenador los pueda procesar. Esta información se divide en **órdenes** y **datos**, que pueden ser de diferentes tipos atendiendo al tratamiento que se les dé.

El tratamiento automático de la información establece **tres fases de procesamiento**: entrada, proceso y salida, que se clasifican de la siguiente forma:

- **De entrada**: son los que se introducen desde soportes de información (discos duros, pen-drive, DVDs) o a través de periféricos de entrada (como teclado, ratón o escáner). Se corresponden con la fase de entrada.

- **Intermedios**: son los que se obtienen y usan en la fase de proceso, que tiene lugar en el hardware del ordenador.

- **De salida**: también se denominan resultados y aparecen en la fase de salida. Los visualizamos a través de los periféricos de salida (como el monitor o la impresora).

Los datos también se pueden clasificar atendiendo a **si su valor cambia o no durante el proceso**:

- **Constantes**: permanecen inamovibles durante la ejecución del proceso o programa que los usa, como por ejemplo el número Pi en un programa que calcula la superficie de un círculo, pues su valor será el mismo para cualquier dato de entrada y no variará a lo largo de la ejecución del mismo.

- **Variables**: se modifican durante el proceso dependiendo del código del programa. Por ejemplo, los intereses que un usuario debe pagar al banco por su hipoteca calculados por un programa, pues dependiendo del interés y las condiciones, el resultado variará.

En el ordenador, tal y como vemos, el conjunto de instrucciones se ejecuta sobre un conjunto de datos. Esta información es suministrada al ordenador mediante símbolos o caracteres:

Caracteres	
Alfabéticos	a, b, c,... y, z, A, B, C... Y, Z
Numéricos	0, 1, 2, 3... 9
Especiales	*, +, -, [,], (,), {, }, ?...
De control	Fin de línea, pitido, avance de página...
Gráficos	♣, ♠, ®, ©, ™, €, £, ¥...

Los caracteres del primer y segundo grupo se denominan **caracteres alfanuméricos** y los pertenecientes a los tres primeros grupos, **caracteres de texto**.

2. Sistemas de numeración

Se define sistema de numeración como el conjunto de símbolos utilizados para la representación de cantidades, así como las reglas que rigen dicha representación.

Un sistema de numeración se distingue por su **base**, que es el **número de símbolos que utiliza**, y se caracteriza por ser el **coeficiente que determina cuál es el valor de cada símbolo** dependiendo de su **posición**. El sistema de numeración que utilizamos normalmente es el sistema decimal, de base 10. El **sistema decimal** utiliza diez dígitos o símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Dependiendo de la posición que ocupe un dígito dentro de una cifra, representará las unidades, decenas, centenas, millares, etc. Por esto, se dice que los sistemas de numeración son **posicionales**.

Por ejemplo, en este sistema el valor del número 6 839 se puede expresar como sumas de potencias de la base 10:

$$(6 \cdot 10^3) + (8 \cdot 10^2) + (3 \cdot 10^1) + (9 \cdot 10^0) = 6\,839$$

Que, de hecho, es como expresamos oralmente esta cifra:

«seis mil / ochocientos / treinta / y nueve»

Podemos definir también un sistema de numeración como un conjunto de dígitos y reglas que permiten representar datos numéricos. La principal regla es que un mismo dígito tiene distinto valor según la posición que ocupe.

El sistema binario

El sistema de numeración binario **utiliza solo dos dígitos (0 y 1)** para representar cantidades, por lo que su base es 2. Cada dígito de un número representado por este sistema se denomina **bit** (**bi**nary digi**t**). Deriva directamente del paso de la corriente eléctrica por los componentes electrónicos del equipo, por lo que el ordenador lo usa a nivel interno.

Los bits tienen distinto valor dependiendo de la posición que ocupan; por eso este sistema también es **posicional**. Estos valores vienen determinados por una potencia de base 2 a la que vamos a llamar **peso**. Así, por ejemplo, el número binario 1 011,01 expresado en decimal quedaría así:

$$(1 \cdot 2^3) + (0 \cdot 2^2) + (1 \cdot 2^1) + (1 \cdot 2^0) + (0 \cdot 2^{-1}) + (1 \cdot 2^{-2}) = 11{,}25$$

Expresar un número decimal en sistema binario

Vamos a pasar a binario el número decimal 54. Para ello:

1. Calculamos el número de dígitos necesarios para representar 54. El número 54 es **mayor** que 25 = 32, pero es menor que 26 = 64; entonces, con **seis dígitos binarios** podremos representar el número decimal.

2. Realizamos una tabla con los seis dígitos, y luego sumamos los pesos donde hay un 1.

Pesos asociados							
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
		1	1	0	1	1	0
32 + 16 + 0 + 4 + 2 + 0 = 54							

3. También podemos realizar el cálculo con divisiones sucesivas del número 54 por 2 hasta llegar a un cociente menor de 2.

54	2				
0	27	2			
Sexto ↑	**1**	13	2		
	Quinto ↑	**1**	6	2	
		Cuarto ↑	**0**	3	2
			Tercero ↑	**1**	**1**
				Segundo ↑	*Primero* ↑

Por tanto, el número decimal 54 se representa en código binario como 110110. Lo escribimos así:

$$54_{(10} = 110110_{(2}$$

Conversión de una fracción decimal a binario

La forma más sencilla para convertir una fracción decimal a binario consiste en **multiplicar sucesivamente la parte fraccionaria por 2 hasta que dé 0 como resultado**. La parte entera de cada multiplicación formará los bits del número binario. A veces, puede ocurrir que la parte fraccionaria no desaparezca; es decir, no dará nunca 0. En estos casos se realizan varias multiplicaciones hasta tener los suficientes dígitos que permitan no sobrepasar un determinado error. Por ejemplo, si se desea un error inferior a 2^{-10} (0,0000000002), calcularemos hasta 10 dígitos.

Vamos a pasar a binario el número decimal 12,125:

1. **Parte entera**: sumamos los pesos o realizamos las divisiones correspondientes: $12_{(10} = 1100_{(2}$

2. **Parte fraccionaria**:

   - 0,125 · 2 = **0**,250 → **0** (el primer dígito es 0; la nueva parte fraccionaria es 0,250).
   - 0,250 · 2 = **0**,500 → **0** (el segundo dígito es 0; la nueva parte fraccionaria es 0,500).
   - 0,500 · 2 = **1**,000 → **1** (el tercer dígito es 1; **como la parte fraccionaria es 0**, finalizamos).

3. **Resultado**: $12{,}125_{(10} = 1100{,}001_{(2}$

Conversión de una fracción binaria a decimal

El resultado es la suma de los productos de los resultados de multiplicar cada dígito por la base elevado a la posición que ocupa pero en negativo.

Vamos a pasar a decimal el número binario 110,0011:

1. **Parte entera**: sumamos los pesos y para ello nos fijamos en la tabla: $110_{(2} = 6_{(10}$

Pesos asociados							
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
					1	1	0
4 + 2 + 0 = 6							

2. **Parte fraccionaria**:
$(0 \cdot 2^{-1}) + (0 \cdot 2^{-2}) + (1 \cdot 2^{-3}) + (1 \cdot 2^{-4}) = (0 \cdot 0{,}5) + (0 \cdot 0{,}25) + (1 \cdot 0{,}125) + (1 \cdot 0{,}0625) = 0{,}1875$

3. **Resultado**: $110{,}0011_{(2} = 6{,}1875_{(10}$

Suma y resta en binario

Al igual que con el sistema decimal, en el sistema binario podemos realizar las **operaciones aritméticas**: **suma**, **resta**, **multiplicación** y **división**. La suma binaria es parecida a la suma en decimal, con la diferencia de que se manejan **solo dos dígitos, el 0 y el 1**. Si el resultado de la suma excede de 1, se agrega un acarreo a la suma parcial siguiente.

Suma binaria
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0, acarreo 1

Resta binaria
0 - 0 = 0
0 - 1 = 1, acarreo 1
1 - 0 = 1
1 - 1 = 0

El acarreo de la resta se suma al siguiente dígito del sustraendo.

Suma SIN acarreo

		1	0	0	0	0	→	16
+	1	0	1	0	0	1	→	41
	1	1	1	0	0	1	→	57

Suma CON acarreo

				1	1	1			Acarreo
				↓	↓	↓			
	1	0	1	0	1	1	1	→	87
+		1	0	0	0	0	1	→	33
	1	1	1	1	0	0	0	→	120

Resta SIN acarreo

	1	1	1	0	1	0	1	→	117
-		1	0	0	0	0	1	→	33
	1	0	1	0	1	0	0	→	84

Resta CON acarreo

- Cuando nos encontramos con el primer 0 - 1, el resultado es 1 y nos llevamos 1, que sumaremos al siguiente sustraendo.
- Si al sumar nos volvemos a llevar 1 (caso de sumar 1 de acarreo + 1 en sustraendo), ese 1 pasa al siguiente sustraendo, y así sucesivamente hasta que dé 0.

	1	1	0	0	1	0	1	→	101
		↓ 1	↓ 1		↓ 1				Acarreo
-			1	1	0	1	1	→	27
	1	0	0	1	0	1	0	→	74

Resta CON acarreo y decimales

	1	0	0	0	1	,	0	1	→	17,25
	↓ 1	↓ 1	↓ 1	↓ 1	↓ 1					Acarreo
-		1	0	1	1	,	1	1	→	11,75
	0	0	1	0	1	,	1	0	→	5,5

Multiplicación binaria

Se realiza como en la multiplicación decimal, con la diferencia de que luego se hacen las sumas en binario.

Multiplicación binaria
$0 \cdot 0 = 0$
$0 \cdot 1 = 0$
$1 \cdot 0 = 0$
$1 \cdot 1 = 1$

Si en la suma de una multiplicación nos juntamos con cuatro 1 en una columna, primero sumamos 1 + 1 = 0, y me llevo 1 para la siguiente suma de la siguiente columna; continuamos sumando 1 + 1 = 0, y me vuelvo a llevar 1 para sumar a la siguiente columna, con lo que el resultado será 0 y me llevo dos 1, que se sumarán con los elementos de la columna siguiente.

Multiplicación de 23 (10111) por 14 (1110).

					1	0	1	1	1	→	23
				x		1	1	1	0	→	14
					0	0	0	0	0		
				1	0	1	1	1			
			1	0	1	1	1				
+		1	0	1	1	1					
	1	0	1	0	0	0	0	1	0	→	322

División binaria

Se efectúa como en la división decimal, pero las multiplicaciones y las restas internas se hacen en binario.

Dividir 282 (100011010) entre 10 (1010)

1	0	0	0	1	1	0	1	0	1	0	1	0	
0	0	1	1	1	1				1	1	1	0	0
		0	1	0	1	0							
			0	0	0	0	1	0			Cociente	→	28
											Resto	→	2

El sistema octal

Los primeros sistemas informáticos utilizaban solo el sistema binario para interpretar y transformar los datos, con lo que las labores de programación eran bastante tediosas; se recurrió entonces al uso de sistemas intermedios que permitían una fácil traducción hacia y desde el sistema binario. Estos sistemas son el **octal** y el **hexadecimal**.

El sistema octal tiene como base de numeración 8, es decir, utiliza ocho símbolos para representar las cantidades. Estos símbolos son 0, 1, 2, 3, 4, 5, 6, 7. Para convertir de decimal a octal, y viceversa, procederemos como en el sistema binario.

Conversión de un número decimal a octal

Lo más sencillo son las divisiones sucesivas.

925	8		
920	115	8	
45	35	14	8
5	**3**	**6**	**1**
Cuarto ↑	*Tercero* ↑	*Segundo* ↑	*Primero* ↑

Para convertir un número octal a decimal, emplearemos el teorema fundamental de la numeración. Nos podremos guiar por los pesos asociados a cada dígito dependiendo de su posición. Para pasar a decimal, multiplicamos el dígito por la base elevada a su posición:

Pesos asociados				
8^4	8^3	8^2	8^1	8^0
4096	512	64	8	1
0	1	6	3	5

$$(1 \cdot 8^3) + (6 \cdot 8^2) + (3 \cdot 8^1) + (5 \cdot 8^0) \rightarrow (1 \cdot 512) + (6 \cdot 64) + (3 \cdot 8) + (5 \cdot 1) \rightarrow 925$$
$$1635_{(8} \rightarrow 925_{(10}$$

Conversión a octal de una fracción decimal

Se procede como en el sistema binario, con el método de multiplicaciones sucesivas, lo único que cambia es la base.

Vamos a convertir a octal el número $12{,}0625_{(10}$.

1. **Parte entera**: sumamos los pesos o realizamos las divisiones correspondientes: $12_{(10} = 14_{(8}$

2. **Parte fraccionaria**: realizamos multiplicaciones sucesivas por 8, quedándonos con la parte entera y multiplicando por la fraccionaria, hasta que dé 0. Si las fracciones no llegan a 0, se realizan varias multiplicaciones hasta tener los suficientes dígitos que permitan no sobrepasar un determinado error:

   - 0,0625 · 8 = **0**,5000 → **0** (el primer dígito es 0; la nueva parte fraccionaria es 0,5000).
   - 0,5000 · 8 = **4**,0000 → **4** (el segundo dígito es 4; **como la parte fraccionaria es 0**, finalizamos).

3. **Resultado**: $12{,}0625_{(10} = 14{,}04_{(8}$

Conversión a decimal de una fracción octal

Se procede a realizar esta conversión aplicando el teorema fundamental de la numeración: cada dígito tiene un peso según la posición que ocupe. El primer dígito de la parte fraccionaria se multiplica por la base elevada a –1; el segundo por la base elevada a –2, y así sucesivamente.

Vamos a convertir a decimal el número $11{,}3016_{(8}$.

1. **Parte entera**: en primer lugar realizamos los cálculos oportunos: $11_{(8} = 9_{(10}$

2. **Parte fraccionaria**:
   $(3 \cdot 8^{-1}) + (0 \cdot 8^{-2}) + (1 \cdot 8^{-3}) + (6 \cdot 8^{-4}) = 3/8 + 0 + 1/512 + 6/4096 = 0{,}375 + 0{,}001953125 + 0{,}00146484375 = 0{,}37841796875$

3. **Resultado**: $11{,}0316_{(8} = 9{,}37841796875_{(10}$

El sistema hexadecimal

El sistema hexadecimal tiene como base de numeración 16, es decir, utiliza dieciséis símbolos para representar las cantidades. Estos símbolos son 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

A los símbolos A, B, C, D, E y F se les asignan los valores que se muestran en la tabla a continuación. Para convertir de hexadecimal a decimal, y viceversa, procederemos como en los casos anteriores.

Símbolo	Valor decimal
A	10
B	11
C	12
D	13
E	14
F	15

Conversión de un número decimal a hexadecimal

Se realizan divisiones sucesivas, y para los restos entre 10 y 15 utilizamos las letras correspondientes.

Conversión del número 41 565 a hexadecimal.

41565	16		
095	2597	16	
156	099	162	16
125	037	2	10
13	5		
D	**5**	**2**	**A**
Cuarto ↑	*Tercero* ↑	*Segundo* ↑	*Primero* ↑

Conversión de un número hexadecimal a decimal

Para convertir un número hexadecimal a decimal, utilizaremos el teorema fundamental de la numeración.

Para pasar la cantidad $A1D_{(16}$ a decimal, multiplicamos el dígito por la base elevada a su posición:

$$(A \cdot 256) + (1 \cdot 16) + (D \cdot 1) = (10 \cdot 256) + 16 + 13 = 2560 + 16 + 13 = 2589$$
$$A1D_{(16} = 2589_{(10}$$

Conversión de una fracción decimal a hexadecimal

Se procede como en los casos anteriores.

Pesos asociados			
16^3	16^2	16^1	16^0
4096	256	16	1
	A	1	D

Conversión a hexadecimal de una fracción decimal

1. **Parte entera**: en primer lugar realizamos los cálculos oportunos: $28_{(10} = 1C_{(16}$

2. **Parte decimal**, realizamos multiplicaciones sucesivas por 16:

   - 0,1975 · 16 = **3**,16 → **3**
   - 0,16 · 16 = **2**,56 → **2**
   - 0,56 · 16 = **8**,96 → **8**
   - 0,96 · 16 = **15**,36 → **F**
   - 0,36 · 16 = **5**,76 → **5**
   - 0,76 · 16 = **12**,16 → **C**
   - 0,16 · 16 = **2**,56, se repite de nuevo.

3. **Resultado**: $12{,}1975_{(10} = 1C{,}328F5C28F5C2\ldots_{(16}$

Conversiones entre sistemas

De la misma manera que convertimos del sistema decimal al **binario**, **octal** y **hexadecimal**, y viceversa. También podemos convertir del binario al **octal** y **hexadecimal** y del **hexadecimal** al **octal**, etc.

Conversión hexadecimal-binario

Se sustituye cada dígito hexadecimal (0, 1, 2, …, D, E, F) por su representación binaria utilizando cuatro dígitos; así, el 0 se representa por 0000, el 1 por 0001, el 2 por 0010, etc. Se utilizan cuatro dígitos porque el valor más alto de este código, el 15, que se representa con la F, necesita cuatro dígitos: 1111.

Pasar a binario $73B{,}F1_{(16}$:

7	3	B	,	F	1
↓	↓	↓		↓	↓
0111	**0011**	**1011**	,	**1111**	**0001**

$73B{,}F1_{(16} = 11100111011{,}11110001_{(2}$

Conversión binario-hexadecimal

Se agrupan los dígitos binarios de cuatro en cuatro a partir del punto decimal hacia la izquierda y hacia la derecha, y se sustituye cada grupo de cuatro por su valor correspondiente en hexadecimal

Pasar a hexadecimal $101011011_{(2}$:

0001	0101	1011
↓	↓	↓
1	**5**	**B**

$101011011_{(2} = 15B_{(16}$

Conversión octal-binario

Procedemos como en la conversión hexadecimal-binario; se sustituye cada dígito octal por su representación binaria utilizando tres dígitos binarios. Se utilizan tres porque el valor más alto, el 7, necesita tres dígitos binarios: 111.

Pasar a binario 712,46$_{(8}$:

7	1	2	,	4	6
↓	↓	↓		↓	↓
111	**001**	**010**	,	**100**	**110**

712,46$_{(8}$ = 111001010,100110$_{(2}$

Conversión binario-octal

Se agrupan los dígitos de tres en tres a partir del punto decimal hacia la izquierda y hacia la derecha, sustituyendo cada grupo de tres por su equivalente en octal.

Pasar a octal 1110110,1100111$_{(2}$:

001	110	110	,	110	011	100
↓	↓	↓		↓	↓	↓
1	**6**	**6**	,	**6**	**3**	**4**

1110110,1100111$_{(2}$ = 166,634$_{(8}$

Conversión hexadecimal-octal y octal-hexadecimal

En esta conversión se realiza un paso intermedio; primero se pasa de hexadecimal a binario y luego de binario a octal o viceversa.