SISTEMAS NUMÉRICOS POSICIONALES

Aplicaciones prácticas en la informática

Autor: Prof. Víctor Zamora Lorente

Índice de contenidos

1. SISTEMAS NUMÉRICOS POSICIONALES

Probablemente hayas olvidado que 123 es la suma de 100+20+3. En principio, tal vez consideres que es un aspecto sin la mayor importancia, pero a la hora de entender los sistemas numéricos usados en la informática, pronto te darás cuenta de que no es así.

Si analizamos con más detenimiento el número 123, éste tiene dos aspectos que a menudo obviamos: por un lado es la representación numérica de una cuantía, concretamente 123 unidades, pero al mismo tiempo, es la agrupación de 1 centena, 2 decenas y 3 unidades, es decir, 1*100 + 2*10 + 3*1 ¿Por qué pasa esto?.

Todo parte de una evidencia muy básica, tenemos 10 dedos en las manos. Este detalle sin importancia es vital para entender como empezamos a utilizar el sistema decimal y, en general, para comprender los distintos sistemas numéricos con otras bases que no sean el 10.

Dos conceptos: la base y la posición

1.1. ¿QUÉ ES LA BASE NUMÉRICA?

La base numérica representa dos conceptos bien definidos: la capacidad expresiva, en lo referente a el número de símbolos que tenemos para expresar una determinada cuantía, por otro lado, es el grado de agrupamiento.

a) Capacidad expresiva: símbolos por dígito

En el sistema decimal, tenemos la capacidad de representar hasta 10 magnitudes con sólo un símbolo numérico… ¿Cuáles son?

Nada es 0, la unidad es 1, un par es 2, un trío es 3… Hasta que llegamos al 9. ¿Qué pasa aquí?, ¿por qué no llegamos hasta el 10? Es de nuevo el momento de pensar en tus manos.

Cuando llegamos al10, lo que ocurre es que no podemos seguir contando con nuestros dedos; ¿qué podríamos hacer para seguir contando? La solución es hacer lo que hacemos con el sistema decimal, contabilizamos un grupo de diez (decena) y seguimos contando. Y así, es como el ser humano alumbró al 10.

Resumen:

- ✔ Cada dígito decimal tiene una capacidad expresiva de 10 símbolos: 0,1,2,3,4,5,6,7,8,9 (10 símbolos para representar desde la nada hasta 9)

b) Grado de agrupamiento

El grado de agrupamiento representa el otro significado de la base numérica. Éste representa cómo representamos las cuantía mediante un sistema de una determinada base. Sigamos con el decimal.

Siguiendo con el ejemplo inicial, ¿cómo puedo representar la magnitud 123 unidades?.

Cabría plantearse coger 12 decenas y 3 unidades, ¿verdad? 12*10+3*1 = 120+3 = 123, pero esto, como hemos visto anteriormente no es posible porque ¡nos faltan dedos!, no podemos contar más allá de 10 y cuando contamos 10 lo que hacemos es agruparlos. Así, cuando tenemos 10 unidades, hacemos una decena y la representamos exactamente así: 10 (1*10+0*1). Cuando tenemos 10 decenas, lo que hacemos es hacer una centena, y la representamos así: 100 (100*1+0*10+0*1). Y así sucesivamente.

Resumen:

- ✔ Cuando tenemos 10 grupos unidad, 10 grupos de decenas, 10 grupos de centenas, agrupamos estos en un grupo mayor que contiene exactamente 10 elementos del grupo anterior.

c) ¿Qué pasaría si sólo tuviésemos una mano?

Si sólo tuviésemos una mano, lo que ocurriría es que la base numérica pasaría a ser 5, entonces, dejarían de existir los símbolos 9,8,7,6 y ¡5!. ¿Cómo? ¿Por qué?. Veámoslo.

En el sistema pental (base 5), tenemos la capacidad de representar hasta 5 magnitudes con sólo un símbolo numérico… ¿Cuáles son?

Nada es 0, la unidad es 1, un par es 2, un trío es 3… Hasta que llegamos al 4. ¿Qué pasa aquí?, ¿por qué no llegamos hasta el 5? Es el momento de pensar que tan sólo tienes una mano.

Cuando llegamos al 5, lo que ocurre es que agruparíamos esas 5 unidades en un grupo de 5 y pasaríamos a representarlo de la siguiente forma: $10_{5.}$ Tranquilidad, vamos a verlo detenidamente.

¿Acaso 1*5 + 0*1 no es 5+0 que es igual a 5? Pues es eso exactamente lo que representa 10_5, representa un grupo de 5 más 0 unidades.

… Creo que lo estamos entendiendo, ¿cómo se representa en base 5 la magnitud 26?.

Si somos observadores, con dos dígitos en base 5 podemos representar desde el 00 hasta 44, que corresponderían a las magnitudes 0 y 4*5+4*1 = 20 + 4 = 24 respectivamente. Vaya, parece ser que nos quedamos cortos, necesitamos coger un grupo mayor (otro dígito), en este caso de 25 (5 grupos de 5) para poder representar la magnitud 26.

Calculemos el número: 1*25 + 0*5 + 1*1 = 25 + 0 + 1 = 26 *et voilà, c'est très facile*

d) Corolario

Tal vez andes algo confuso con esto de necesitar otro dígito para representar el 26 en el sistema pental (base 5); pero verás que es muy sencillo. ¿Entiendes que con dos dígitos decimales sólo puedes representar desde el 00 hasta el 99?

Pues la forma de proceder es la misma, sólo que como se ha explicado anteriormente, en el sistema pental cada dígito sólo dispone de 5 símbolos por dígito que representan las magnitudes 0, 1, 2, 3 y 4... 5 magnitudes, dependiendo de la posición que ocupen, porque si ocupan la posición 0, son unidades, pero si ocupan la posición 1 estamos hablando de grupos de 5 o quintetos, si ocupan la posición 2, son grupos de 25 (5 grupos de 5)... Y así sucesivamente.

Veamos algunos ejemplos:

123_{10} → 1*100 + 2*10 + 3*1 = 123

123_5 → 1*25 + 2*5 + 3*1 = 25 +10+3 = 38

402_{10} → 4*100 + 0*10 + 2*1 = 402

402_5 → 4*25+ 0*5+ 2*1 = 100 + 2 = 102

1.2. LA POSICIÓN

Una vez explicado el concepto de base, es el momento de analizar cómo la posición de cada uno de los dígitos representa a un grupo. Analicemos cómo influye esta con distintas bases:

	7	6	5	4	3	2	1	0
Base: 10 [0,9]	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0
	10000000	1000000	100000	10000	1000	100	10	1
Base: 5 [0,4]	5^7	5^6	5^5	5^4	5^3	5^2	5^1	5^0
	78125	15625	3125	625	125	25	5	1
Base: 2 [0,1]	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1

Como podemos observar en la tabla, sea cual sea la base, **el primer dígito (posición 0) siempre representa a las unidades**. El segundo dígito (posición 1), representa a un **grupo del tamaño de la base** de la posición anterior, es decir:

- ✔ Base 10 → grupo de 10 unidades (decena) → $\mathbf{10^1}$ → 10
- ✔ Base 5 → grupo de 5 unidades (quinteto) → $\mathbf{5^1}$ → 5
- ✔ Base 2 → grupo de 2 unidades (par) → $\mathbf{2^1}$ → 2

Siguiendo con este razonamiento, el tercer dígito (posición 2) representa a un **grupo del tamaño de la base** de la posición anterior, es decir:

- ✔ Base 10 → grupo de 10 decenas (centena) → $\mathbf{10^2}$ → 100
- ✔ Base 5 → grupo de 5 quintetos (veinticinco) → $\mathbf{5^2}$ → 25
- ✔ Base 2 → grupo de 2 pares (cuarteto) → $\mathbf{2^2}$ → 4

Y así sucesivamente con cada una de las posiciones. Por lo tanto, lo que nos indica la posición es el **tamaño en unidades que tiene ese grupo** atendiendo a la **base** que usemos. Veamos un ejemplo más:

1011_{10} → 1000 + 0 + 10 + 1 = 1011
1011_5 → 125 + 0 + 5 + 1 = 131
1011_2 → 8 + 0 + 2 + 1 = 11

a) Corolario

Una vez identificado qué representa la posición en un sistema numérico, podemos concluir que **la posición del dígito es el exponente de la base** (recuerda que la posición primera es 0). Asimismo, hemos observado que con **n dígitos**, se pueden representar **B^n magnitudes** que irían desde el **0 hasta B^n-1** (p.e. con 2 dígitos decimales, podemos representar 100 magnitudes que van desde el 00 al 99).

Capacidad expresiva: B^n (B → Base, n → número de dígitos)

Rango: [0, B^n- 1]

Ejemplo:

Supongamos que disponemos de un *byte*, es decir, 8 bits (*binary digits*). ¿Cuántos números puedo representar? ¿Cuáles?

Con 8 bits puedo representar 256 (2^8) números que van desde el 0 hasta el 255.

¿De qué tamaño es el grupo del último bit?

Como un *byte* son 8 bits, el último bit ocuparía la posición 7, por tanto sería de 128 unidades (2^7)

1.3. BASE 16: HEXADECIMAL

El sistema numérico hexadecimal, tiene la particularidad de que excede las 9 magnitudes por símbolo, concretamente del 0 al 15. Pero no podemos expresar una magnitud con más de un símbolo, dado que crearíamos confusión. Para ello, lo que hacemos es usar las letras del abecedario en mayúsculas. Así, el sistema hexadecimal es el siguiente:

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Por otro lado, el sistema hexadecimal tiene la característica de que su base es un múltiplo de 2, concretamente 16 es 2^4. Esta coincidencia que puede en principio parecer sin importancia, se traduce en que cada dígito hexadecimal puede ser representado exactamente con 4 bits o a la inversa, cada grupo de 4 bits puede ser representado por su equivalente en hexadecimal. Veamos algunos ejemplos.

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
1111	1110	1101	1100	1011	1010	1001	1000	0111	0101	0101	0100	0011	0010	0001	0000
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Este aspecto es especial útil ala hora de representar bytes y es por ello que las direcciones MAC se presentan en hexadecimal.

Ejemplos

Binario								Hexadecimal		Decimal
0	0	0	1	1	1	0	0	1	C	28
1	0	0	1	0	0	0	1	9	1	145
1	0	0	0	0	0	0	0	8	0	128
1	1	0	0	0	0	0	0	C	0	192
1	1	1	0	0	0	0	0	E	0	224
1	1	1	1	0	0	0	0	F	0	240
1	1	1	1	1	0	0	0	F	8	248
1	1	1	1	1	1	0	0	F	C	252
1	1	1	1	1	1	1	0	F	E	254
1	1	1	1	1	1	1	1	F	F	255

1.4. DIRECCIONES IP (DECIMAL/BINARIO)

Las direcciones IP son un número identificador de los nodos de la red compuesto por 32 bits que se agrupan en 4 bytes. Por lo tanto, se hace especialmente útil saber representar un byte a decimal y viceversa.

a) Decimal a Byte

Supongamos la dirección 192.168.1.152 ¿Cuál sería su representación en binario?

Observa con detenimiento 192 ¿Cómo podrías sumar fácilmente 192 con los grupos binarios?

	7	6	5	4	3	2	1	0
Base: 2 [0,1]	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
192	1	1	0	0	0	0	0	0

Efectivamente, 192 se obtiene fácilmente con 128+64, es decir, 11000000. Vayamos con 168.

	7	6	5	4	3	2	1	0
Base: 2 [0,1]	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
192	1	0	1	0	1	0	0	0

Aquí no ha sido tan fácil, pero sí que es sencillo. De primeras sumamos 128. Acto seguido observamos que 192 + 64 excedía 168, por tanto, hacemos 0 ese grupo binario. Posteriormente, intentamos agregar el grupo de 32 lo cual nos da la cercana magnitud de 160 (128+32). Ahora, sólo necesitamos 8, con lo cual hacemos 0 el grupo de 16, 1 el grupo de 8 y el resto es cero. Resultado:

10101000 = 1*27 + 0*26 + 1*25 + 0*24 + 1*23 + 0*22 + 0*21 + 0*20 = 128+0+32+0+8+0+0+0 = 168