

TEMA 1: INTRODUCCIÓN A LOS SISTEMAS DIGITALES.

1.1. Sistemas Analógicos y Digitales.

Magnitud **analógica** es aquella que puede tomar cualquier valor real dentro de un margen determinado de forma continua.

Ej: la temperatura en la habitación:

---x---x-----x----- T
20 21.5 30

Ej: Velocidad de un coche: 72,3 Km/h.

Ej: Estatura de una persona: 1,83 m.

Ej: Cantidad de lluvia caída: 13,2 l/m².

El mundo es analógico, esencialmente.

Magnitud **digital** es aquella que sólo puede tomar un valor dentro de un conjunto finito de valores preestablecidos.

Ej: El grupo al que pertenece un alumno de 1º: 1º A, 1º B, 1º C, 1º D.

Ej: El día de la semana (LUNES, MARTES, ...)

Ej: Los meses del año (ENERO, FEBRERO, ... DICIEMBRE).

Magnitud **digital binaria** es aquella que sólo puede tomar un valor dentro de un conjunto de 2 valores posibles.

Ej: cualquier pregunta cuya respuesta sea: SI,NO.

Todas ellas son fácilmente asimilables a SI/NO, Verdadero/Falso, Todo/Nada, 0/1.

Toda la informática se basa en magnitudes digitales binarias.

Se trabaja con los dos estados de una magnitud binaria, representados habitualmente como 0, 1, y físicamente representados por dos niveles de tensión distintos (pueden ser 0 V y 5 V).

¿Por qué 2 niveles y no más?

- Porque tecnológicamente es muy fácil fabricar dispositivos que presenten dos estados bien diferenciados, y no es tan fácil si el número de estados es mayor.
- Existe una herramienta matemática muy sencilla y adecuada para representar y procesar la información: la lógica binaria.

COMPARACION:

Ventajas Digitales / Inconvenientes Analógicos.

Mayor inmunidad al ruido.
Sencillez y economía al manejar pocos valores.
Seguridad y precisión.
Facilidad de almacenamiento.

Ventajas Analógicos / Inconvenientes Digitales.

El mundo real es ANALOGICO (CONVERSION A/D).

1.2. Sistema de Numeración Binario.

Base 10 (decimal). $432 = 4 \cdot 10^2 + 3 \cdot 10^1 + 2 \cdot 10^0$ [0,1,...9]

Especial interés.

Base 2 (binaria). $1001 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 9$ [0,1]

Base 8 (octal) [0,1,...7]

Base 16 (hexadecimal) [0,1,... 9, A, B, C, D, E, F]

Ej: $4DH = 4D_{(16)} = 4 \cdot 16^1 + 13 \cdot 16^0 = 77$

Si hay **decimales**, se usan las potencias negativas:

Ej: $54,3_8 = 5 \cdot 8^1 + 4 \cdot 8^0 + 3 \cdot 8^{-1} = 44,375$

El paso de cualquier base a decimal es el ya visto.

Vamos a ver el contrario. Poner 12 en base 8. Se puede hacer de dos formas.

Primera forma. Haciendo la tabla a mano:

Decimal	00	01	02	03	04	05	06	07	08	09	10	11	12	13
Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15

Segunda forma. Si dividimos 12 entre 8 sale un cociente y un resto. Teniendo en cuenta que en una división

$$\text{Dividendo} = \text{Divisor} * \text{Cociente} + \text{Resto}$$

queda

$$\begin{array}{r} 12 \quad | \quad 8 \\ \hline 4 \quad | \quad 1 \end{array} \quad \begin{array}{l} \text{Es decir } 12 = 8 * 1 + 4 \\ \text{o bien } 12 = 1 * 8^1 + 4 * 8^0 \end{array}$$

Los coeficientes que acompañan a las potencias de 8 son 1 y 4, precisamente **el último cociente menor que la base** (en este caso 8) **y los restos** (en este caso sólo uno) **en orden contrario a cómo aparecieron**.

Otro ejemplo: expresar 92 en base 8

$$\begin{array}{r} 92 \quad | \quad 8 \\ \hline 4 \quad | \quad 11 \quad | \quad 8 \\ \hline \quad \quad | \quad 3 \quad | \quad 1 \end{array} \quad \begin{array}{l} \text{Es decir } 92 = 8 * 11 + 4, \text{ pero a su vez } 11 \text{ se puede poner} \\ \text{como } 11 = 8 * 1 + 3, \text{ con lo cual } 92 \text{ se puede expresar} \\ \text{como } 92 = 8 * [8 * 1 + 3] + 4 = 1 * 8^2 + 3 * 8^1 + 4 * 8^0 \end{array}$$

Es decir 92 en octal se expresaría como 134_8 , que coincide con la lectura del último cociente y los restos de las operaciones en sentido contrario al de aparición.

Esta es la forma general de cambiar de base, en particular para base 2. Ejemplo: $120 = 1111000_2$.

Se debe observar que en base 2 cada 1 contribuye con importancia **doble** que se vecino de la derecha. Cada uno contribuye con una importancia que es una potencia de 2. Por tanto, otra forma de expresar un número en binario consiste en descomponerlo como suma de potencias de 2 y poner 1 en los lugares correspondientes. El $120 = 64+32+16+8$. Para ello $120-64=56$, $56-32=24$, $24-16=8$.

Los cambios entre las bases 2, 8 y 16 son inmediatos, sin necesidad de pasar por base 10.

Base 2 a 8 y viceversa. Agrupaciones de 3 <=====> carácter octal.

Base 2 a 16 y viceversa. Agrupaciones de 4 <=====> carácter hexadecimal.

Ejemplos: $564_{(8)} = 101\ 110\ 100_{(2)}$ $100\ 101\ 001_{(2)} = 451_{(8)}$
 $926_{(16)} = 1001\ 0010\ 0110_{(2)}$ $0001\ 0100\ 1011_{(2)} = 14B_{(16)}$

Una forma especialmente rápida de pasar un número a binario es mediante el paso intermedio por hexadecimal. Ej: $179 = B3_{(16)} = 10110011_{(2)}$.

Con decimales: Entre bases 2, 8 y 16 es directo. Ej: $110,101_{(2)} = 6,5_{(8)}$

De base 10 a cualquier base se hace así:

Ejemplo: Pasar 13,2 a octal.

$13,2 = 13+0,2$. $0,2 \times 8 = 1,6$; $0,6 \times 8 = 4,8$; $0,8 \times 8 = 6,6$ etc...

$13,2 = 15,146...$ en octal

Otro ejemplo: Pasar 25,4 a binario: $25 = 11001_{(2)}$

$0,4 \times 2 = 0,8$; $0,8 \times 2 = 1,6$ $0,6 \times 2 = 1,2$ $0,2 \times 2 = 0,4$ $0,4 \times 2 = 0,8$ y se repite.

$25,4 = 11001,0110\ 0110\ 0110\ 0110$.

La multiplicación y división rápidas por 2^n en binario.

Para multiplicar un número binario por 2, se desplaza una posición a la izquierda, rellenando con ceros la derecha.

Para multiplicar un número binario por 2^n , se desplaza "n" posiciones a la izquierda, rellenando con ceros la derecha.

Para dividir un número binario por 2, se desplaza una posición a la derecha, pasando la unidad a la derecha de la coma.

Para dividir un número binario por 2^n , se desplaza "n" posiciones a la derecha, pasando "n" bits a la derecha de la coma.

Suma en binario (tabla de sumar):

C ₀	a	b	a+b	C ₁
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Ejemplo:

$7 + 5 = 12$

7: $0\ 1\ 1\ 1$
5: $0\ 1\ 0\ 1\ +$

12: $1\ 1\ 0\ 0$

Las operaciones dan el mismo resultado independientemente de la base.

A cada uno de los 0, 1 se les llama dígito binario (**BINARY DIGIT**). **BIT**.

Con n bits se pueden representar 2^n números distintos. Ejemplo n = 3.

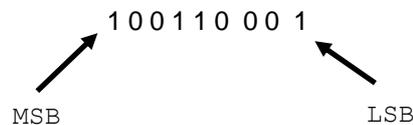
000, 001, 010, 011, 100, 101, 110, 111 que representan de 0 a 7. Desde 0 hasta 2^{n-1} .

Si queremos signo hace falta un bit más o sacrificar un bit en detrimento del margen de representación.

Se llama bit más significativo, también conocido por las iniciales inglesas **MSB** (Most Significant Bit) al bit de mayor peso, el colocado a la izquierda, de un número binario.

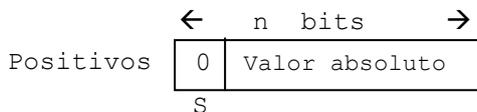
Se llama bit menos significativo, también conocido por las iniciales inglesas **LSB** (Least Significant Bit) al bit de menor peso, el colocado a la derecha, de un número binario.

Ejemplo:



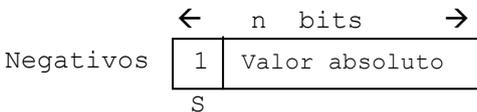
1.3. Representación de números enteros.

a) Signo magnitud:



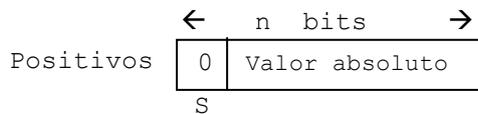
Margen de representación

Desde $-(2^{n-1}-1)$ hasta $+(2^{n-1}-1)$



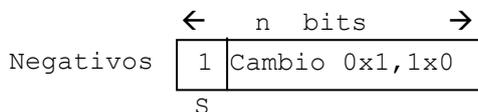
El 0 tiene doble representación
Ej: n=4, desde -7 hasta +7, pasando doble 0.

b) Complemento a 1:



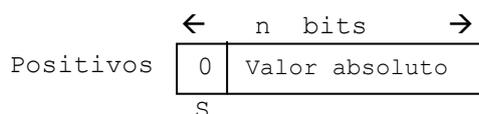
Margen de representación

Desde $-(2^{n-1}-1)$ hasta $+(2^{n-1}-1)$



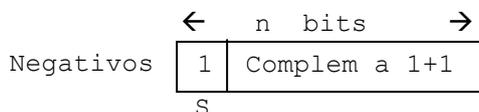
El 0 tiene doble representación
Ej: n=4, desde -7 hasta +7, pasando doble 0

c) Complemento a 2:



Margen de representación

Desde $-(2^{n-1})$ hasta $+(2^{n-1}-1)$



El 0 tiene representación simple.
Ej: n=4, desde -8 hasta +7, pasando simple 0.

Ejemplo: poner -5 en complemento a 2.

+5: 0101 $\xrightarrow{\text{ca1}}$ 1010 $\xrightarrow{+1}$ 1011. Como empieza por 1 es NEGATIVO.

La operación de complemento a 2 es reversible, es decir el ca2 de 1011 es: 0100 + 1 = 0101

Otra forma de calcular el complemento a 2.

Se lee el número de derecha a izquierda y se transcribe igual que está hasta que se encuentra el primer 1. A partir de ese momento, manteniendo el 1 intacto, los restantes dígitos que haya a su izquierda se cambian.

Ejemplo 1:

Número: 00010010 (se sobreentiende que lo que hay a la izquierda son todos 0)

Su complemento a 2: 11101110 (se sobreentiende que lo que hay a la izquierda son todos 1)

Ejemplo 2:

Número: (0000000) 101001110000

Su complemento a 2: (1111111) 010110010000

Ejemplo 3:

Número: 0101001

Su complemento a 2: 1010111

Finalmente veamos una tabla de representación de números binarios con 4 bits:

Decimal	Signo-Magnitud	Complemento a 1	Complemento a 2
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
0	0000	0000	0000
0	1000	1111	NO PROCEDE
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	NO SE PUEDE	NO SE PUEDE	1000

Aritmética en complemento a 2

Para restar, se hace la suma del minuendo con el ca2 del sustraendo.

Casos posibles. Ejemplos

1º caso: resultado positivo correcto.

$$7-5=+7+(-5)$$

0111	+7
1011 +	-5 +
0010	+2

2º caso: resultado negativo correcto.

$$\begin{array}{r}
 5-7=+5+(-7) \\
 \begin{array}{r}
 0101 \\
 1001 + \\
 \hline
 1110
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 +5 \\
 -7 + \\
 \hline
 -2
 \end{array}$$

3º caso: resultado positivo con desbordamiento (OVERFLOW).

$$\begin{array}{r}
 7+5=12 \\
 \begin{array}{r}
 0111 \\
 0101 + \\
 \hline
 \pm 100
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 +7 \\
 +5 \\
 \hline
 \text{FALSO, bit de signo -}
 \end{array}$$

4º Caso: resultado negativo con desbordamiento (OVERFLOW).

$$\begin{array}{r}
 -7-5 = -7+(-5) \\
 \begin{array}{r}
 1001 \\
 1011 + \\
 \hline
 \oplus 100
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 -7 \\
 -5 \\
 \hline
 \text{FALSO, bit de signo +}
 \end{array}$$

Veamos cómo en los dos últimos casos con 1 bit más no se produce desbordamiento

3º caso: resultado positivo sin desbordamiento.

$$\begin{array}{r}
 7+5=12 \\
 \begin{array}{r}
 00111 \\
 00101 + \\
 \hline
 01100
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 +7 \\
 +5 + \\
 \hline
 +12 \text{ CORRECTO}
 \end{array}$$

4º caso: resultado negativo sin desbordamiento.

$$\begin{array}{r}
 -7-5 = -7+(-5) \\
 \begin{array}{r}
 11001 \\
 11011 + \\
 \hline
 \pm 10100
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 -7 \\
 -5 + \\
 \hline
 -12 \text{ CORRECTO}
 \end{array}$$

¿Cómo saber si se ha producido desbordamiento en una operación en complemento a 2?

Viendo si los dos últimos acarrees son iguales. Si son iguales, no se ha producido desbordamiento. Si son distintos, sí se ha producido desbordamiento.

Ejemplo 1:

$$\begin{array}{r}
 7-5=+7+(-5) \\
 \begin{array}{r}
 \mathbf{11}110 \text{ ACARREOS} \\
 0111 \\
 1011 + \\
 \hline
 \pm 0010
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 +7 \\
 -5 + \\
 \hline
 +2 \text{ CORRECTO}
 \end{array}$$

Ejemplo 2:

$$\begin{array}{r}
 7+5=12 \\
 \begin{array}{r}
 \mathbf{01}110 \text{ ACARREOS} \\
 0111 \\
 0101 + \\
 \hline
 \oplus 1100
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 +7 \\
 +5 + \\
 \hline
 -4 \text{ INCORRECTO}
 \end{array}$$

Aritmética en complemento a 1.

La operación en complemento a 1 es igual que en complemento a 2 con la única diferencia de que además de sumar los dos operandos, se suma el acarreo que resulte de la operación como si fuera el LSB de un tercer operando.

Ejemplo 1:

```

00000 ACARREOS
 0110   +6
 0001 +   +1
-----
00111
  0 +
-----
00111   +7
    
```

Ejemplo 2:

```

11000 ACARREOS
 1101   -2
 1110 +   -1
-----
11011
  1 +
-----
11100   -3
    
```

1.4. Códigos más usuales.

BINARIO: Simplemente pasar a base 2.

BCD natural o simplemente **BCD: Binary Coded Decimal**. Decimal codificado en binario. Igual que en binario pero sólo con los 10 dígitos decimales (desde 0 hasta 9, ambos incluidos). Relación de pesos 8 4 2 1 (ponderado).

BCD exceso 3: un número en BCD exceso 3 se representa igual que dicho número más 3 en BCD (no es ponderado). Autocomplementario. El código de N es el ca1 del código de 9-N. Ejemplo, el código de 3 es el complemento a 1 del código de $9 - 3 = 6$.

BCD Aiken: La relación de pesos no es 8 4 2 1 como en BCD natural, sino 2 4 2 1 (ponderado). Autocomplementario. Se pueden definir códigos BCD con otra relación de pesos distinta, incluso de pesos negativos.

GRAY: Entre dos códigos consecutivos sólo hay un bit de diferencia, considerando el primero (0000) y el último (1000) como consecutivos también. Código reflejado. La 2ª mitad es la imagen especular de la primera, cambiando el primer bit. Se genera a partir del binario poniendo el MSB igual y los restantes bits con el siguiente criterio: si no hay cambio de bit se pone un 0 y si hay cambio de bit se pone un 1. Ejemplo, 4 == 0100 en binario y 4 == 0110 en GRAY.

JOHNSON 5 bits: va aumentando el número de unos desde la derecha, y posteriormente disminuye por la izquierda (no es ponderado).

Decimal	BINARIO	BCD natural	BCD exceso 3	BCD Aiken	GRAY	JOHNSON 5 bits
0	0000	0000	0011	0000	0000	00000
1	0001	0001	0100	0001	0001	00001
2	0010	0010	0101	0010	0011	00011
3	0011	0011	0110	0011	0010	00111
4	0100	0100	0111	0100	0110	01111
5	0101	0101	1000	1011	0111	11111
6	0110	0110	1001	1100	0101	11110
7	0111	0111	1010	1101	0100	11100
8	1000	1000	1011	1110	1100	11000
9	1001	1001	1100	1111	1101	10000
10	1010	NO	NO	NO	1111	NO
11	1011	NO	NO	NO	1110	NO
12	1100	NO	NO	NO	1010	NO
13	1101	NO	NO	NO	1011	NO
14	1110	NO	NO	NO	1001	NO
15	1111	NO	NO	NO	1000	NO
	PONDER	PONDER	AUTOCOM	AUTOCOM	REFLEJ	

EJERCICIOS RESUELTOS

1. Pasar a base decimal los siguientes números en distintas bases:

- $101001.001_{(2)} = 41.125$
- $342.3_{(5)} = 97.6$
- $675.03_{(8)} = 445.046875$
- $3DE4.4_{(16)} = 15844.25$

2. Expresar el número decimal 2223.39 en las bases binaria, base 6, octal y hexadecimal.

$100010101111.0110001111010111_{(2)}$
 $14143.2201235_{(6)}$
 $4257.30753_{(8)}$
 $8AF.63D7_{(16)}$

3. Realizar la operación $126 + 589$ en binario, base 7, octal Y hexadecimal, comprobando que coinciden los resultados.

126	0001111110	240	0176	$7E$
$589 +$	$1001001101 +$	$1501 +$	$1115 +$	$24D +$
-----	-----	-----	-----	-----
715	$1011001011_{(2)}$	$2041_{(7)}$	$1313_{(8)}$	$2CB_{(16)}$

4. Realizar la suma $159A_{(12)} + 859B_{(12)}$ en base 12. Usar para ello los 10 dígitos decimales más A (peso 10) y B (peso 11).

Base 12	Decimal
$159A_{(12)}$	$2566,93055$
$859B_{(12)} +$	$14663,53472 +$
-----	-----
$9B7A_{(12)}$	$17230,46527$

5. El número $543_{(x)} = 674_{(8)}$. ¿De qué base se trata?

Por los dígitos usados, la base debe ser mayor o igual que 6.

$$543_{(x)} = 5 \cdot x^2 + 4 \cdot x + 3 \quad \text{y por otro lado} \quad 674_{(8)} = 6 \cdot 64 + 7 \cdot 8 + 4 = 444$$

$5 \cdot x^2 + 4 \cdot x + 3 = 444$, o bien $5 \cdot x^2 + 4 \cdot x - 441 = 0$. Resolviendo la ecuación de 2º grado, sale $x = +9$, y $x = -9,8$ (ésta última se descarta). Por tanto 543 estaba en base 9.

6. Comprobar a priori si se pueden realizar sin desbordamiento las siguientes operaciones con el número de bits que se indican, trabajando en complemento a 2. En los casos en que sí se pueda, realizar la operación y comprobar el resultado:

		RESPUESTA	
•	$157 + 222$ con 7 bits	NO	
•	$-245 - 112$ con 10 bits	SÍ	
•	$344 + 134$ con 10 bits	SÍ	
•	$344 - 220$ con 8 bits	NO	
•	$344 - 569$ con 6 bits	NO	
•	$350 - 533$ con 8 bits	NO	

-245	1100001011	$+344$	0101011000
$-112+$	$1110010000+$	$+134+$	$0010000110+$
-----	-----	-----	-----
-357	1010011011	$+478$	0111011110
	S		S