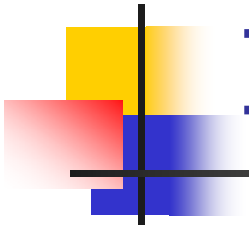
Representación de la Información

... en los Computadores



Información e Informática

- Un computador es una máquina que procesa información.
- La ejecución de un programa implica el tratamiento de los datos.
- Para que el computador ejecute un programa es necesario darles dos tipos de información:
  - las instrucciones que forman el programa y
  - los datos con los que debe operar ese programa.
- Los aspectos más importantes de la Informática relacionados con la información son:
  - cómo <representarla> y
  - cómo <materializarla> o <registrarla> físicamente.

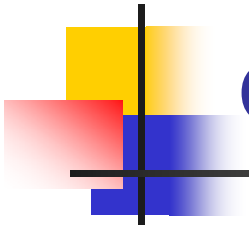
Cómo se da la información a un computador?

- Se la da en la forma usual escrita que utilizan los seres humanos;
  - con ayuda de un alfabeto o conjunto de símbolos, denominados caracteres.

- Categorías de los caracteres:
  - **Caracteres alfabéticos:** son los mayúsculas y minúsculas del abecedario inglés:

    A, B, C, D, E,…, X, Y, Z, a, b, c, d,…, x, y, z
  - **Caracteres numéricos:** están constituidos por las diez cifras decimales:

    Ø, 1, 2, 3, 4, 5, 6, 7, 8, 9
    - El cero suele marcarse con una raya inclinada (ø) para evitar posibles confusiones con la O mayúscula.



Cont…

- **Caracteres especiales:** son los símbolos no incluidos en los grupos anteriores, entre otros los siguientes:

) (, * / ; : Ñ ñ = ! ? . ≈ ‘’ & > # < { Ç } SP

  - Con SP representamos el carácter o espacio en blanco, tal como el que separa dos palabras.
- **Carácter de control:** representan órdenes de control, como el carácter indicador de fin de línea o el carácter indicador de sincronización de una transmisión de que se emita un pitido en un terminal, etc.
  - Muchos de estos son generados e insertados por el propio computador.
- **Caracteres Gráficos:** son símbolos o módulos con los que se pueden representar figuras (o iconos) elementales.

Cont...

- Toda comunicación con un computador convencional se realiza según los caracteres que admitan sus dispositivos de E / S.
- Toda instrucción o dato se representará por un conjunto de caracteres tomados del alfabeto definido en el sistema a utilizar.
- El diseño de un sistema informático resulta mas fácil, su realización menos compleja y su funcionamiento muy fiable, si se utilizan solo dos valores o estados posibles.
- Estos valores conceptualmente se representan por

cero (0) y	apagada y	0 voltios y
uno (1)	encendida	3.5 voltios

etc. **(BIT)**

101

101

0110

010

01

0

0101

0

0110

0110

01

Codificación y Decodificación

- Al tener que <traducir> toda la información suministrada al computador a ceros y unos, es necesario establecer una correspondencia entre el conjunto de todos los caracteres

  α **= { A, B, C, D, ..., Z, a, b,..., z, 0, 1, 2, 3, ..., 9, /, +, (,), ... }**

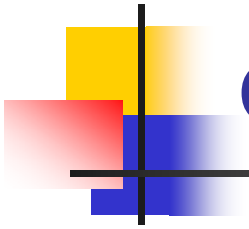
- y el conjunto binario

  β **= { 0, 1 } n**

- **Codificación** o representación de los elementos de un conjunto (α) mediante los de otro (β) de forma tal que a cada elemento de α le corresponda un elemento distinto de β (n bits).

- Estos códigos de trasformación se denominan códigos de Entrada / Salida (E/S) o códigos externos.

- Las operaciones aritméticas con datos numéricos se suelen realizar en una representación más adecuada para este objetivo que la obtenida con el código de E/S.



Cont...

DATO: Característica de una información expresada en forma adecuada para su tratamiento.

- Representación de los datos (valores):
  - Valores analógicos.
  - Valores discretos o digitales.
- Necesidad de convertir los valores analógicos a discretos.
  - **Sistema digita**l: Sistema de N estados estables
  - **Dígito**: Variable capaz de asumir un estado.
- Los dígitos se agrupan para representar más estados.

Cont...

- **Código:** Ley de correspondencia entre valores de información y combinaciones de dígitos de un sistema digital utilizadas para representarlos.
- **Codificación:** Información -> Código

azul	----> 0		azul	----> 100
verde	----> 1	ó	verde	----> 101
rojo	----> 2		rojo	----> 111

- **Decodificación:** Código -> Información

azul	<---- 0		azul	<---- 100
verde	<---- 1	ó	verde	<---- 101
rojo	<---- 2		rojo	<---- 111

- **Código binario:** Cuando el sistema digital utilizado tiene sólo 2 estados (0,1).

Sistemas de numeración usuales en informática

- Los computadores suelen efectuar las operaciones aritméticas utilizando una representación para los datos numéricos basada en el sistema de numeración base dos (sistema **binario**).

- También se utilizan los sistemas de numeración, preferentemente el **octal** y **hexadecimal**, para obtener códigos intermedios.

- Un número expresado en uno de estos dos códigos puede transformarse directa y fácilmente a binario y viceversa.
  - Por lo que a veces se utilizan como paso intermedio en las transformaciones de decimal a binario y viceversa.

Representación posicional de los números

- Un sistema de numeración en base **b** utiliza para representar los números un alfabeto compuesto por **b** símbolos o cifras.
- Todo número se expresa por un conjunto de cifras, contribuyendo cada una e ellas con un valor que depende de:

  a) la cifra en sí, y
  b) la posición que ocupe dentro del número.

- En el sistema de numeración decimal (sistema en base 10):
  - b = 10 y el alfabeto está constituido por diez símbolos o cifras decimales:

{0,1,2,3,4,5,6,7,8,9}

Cont...

- por ejemplo, el número 3278.52 puede obtenerse como suma de:

$$
\begin{array}{r}
3000 \\
200 \\
70 \\
8 \\
0.5 \\
0.02 \\
\hline
3278.52
\end{array}
$$

- se verifica que:

$$\mathbf{3278.52} = 3*10^{3} + 2*10^{2} + 7*10^{1} + 8*10^{0} + 5*10^{-1} + 2*10^{-2}$$

Cont...

Representación de un número en una base b:

Forma abreviada:

$$N = \ldots n_4\ n_3\ n_2\ n_1\ n_0\ .\ n_{-1}\ n_{-2}\ n_{-3} \ldots$$

Valor:

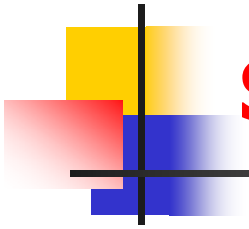
$$N = \ldots n_4 * b^4 + n_3 * b^3 + n_2 * b^2 + n_1 * b^1 + n_0 * b^0 + n_{-1} * b^{-1} \ldots$$

- **Para representar un número:**
  - Resulta más cómodo que los símbolos (cifras) del alfabeto o la base de numeración sean los menos posibles, pero ,
  - Cuanto menos es la base, mayor es el número de cifras que se necesitan para representar una cantidad dada.

Sistemas de Numeración

- Binario
- Octal
- Hexadecimal

Binario	Decimal	Octal	Hexadecimal
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	8	10	8
1001	9	11	9
1010	10	12	A
1011	11	13	B
1100	12	14	C
1101	13	15	D
1110	14	16	E
1111	15	17	F



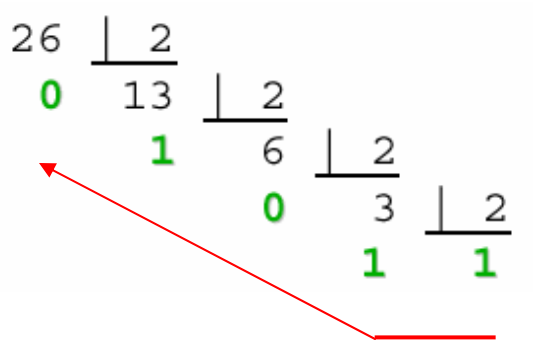
Sistema de numeración binario

- La base es 2 (b=2) sólo
- se necesitan dos símbolos :

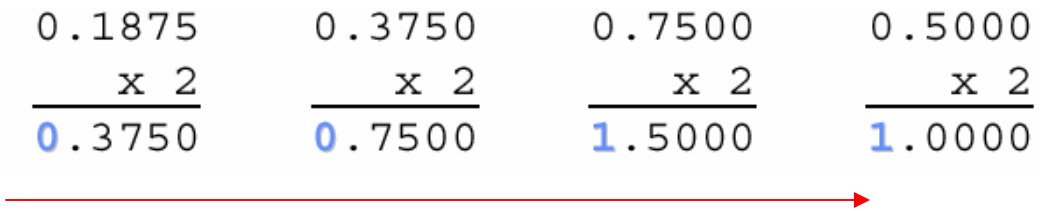
{ 0, 1 }

Conversión de Decimal a Binario

- Se aplica el método de las **"divisiones y multiplicaciones** " sucesivas con la base como divisor y multiplicador (b = 2).
- **Ejemplo:** 26.1875 $_{)10}$ = **11010.0011** $_{)2}$
- Para la parte entera:



- Para la parte fraccionaria:



Conversión de Binario a Decimal

- Se desarrolla la representación binaria (con b=2) y se opera el polinomio en decimal.
- **Ejemplos:**

$$110100_{)2} = 1\cdot 2^5 + 1\cdot 2^4 + 0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 52_{)10}$$

$$10100.001_{)2} = 1\cdot 2^4 + 0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 + 0\cdot 2^{-1} + 0\cdot 2^{-2} + 1\cdot 2^{-3} = 20.125_{)10}$$

- Realmente basta con sumar los pesos (2_i) de las posiciones (i) en las que hay un 1.

Operaciones aritméticas con variables binarias

- Las operaciones aritméticas básicas son la suma, resta, multiplicación y división.

Suma aritmética con varias variables

a	b	a + b
0	0	0
0	1	1
1	0	1
1	1	0 y me llevo 1

Resta aritmética con varias variables

a	b	a - b
0	0	0
0	1	1 y me adeudo 1
1	0	1
1	1	0

Multiplicación aritmética con varias variables

a	b	a * b
0	0	0
0	1	0
1	0	0
1	1	1

División aritmética con varias variables

a	b	a / b
0	0	indeterminado
0	1	0
1	0	∞
1	1	1

Ejemplos:

Efectuar las siguientes operaciones aritméticas binarias:

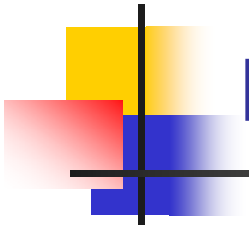
```
  1110101
+ 1110110
----------
 11101011
```

```
  1101010
- 1010111
----------
  0010011
```

```
  1101010
x      11
----------
  1101010
+1101010
----------
 100111110
```

```
  1010011
x      10
----------
  0000000
+1010011
----------
 10100110
```

```
1101.010 | 101
-101     |----
 00110   | 10.101
  -101   |
   00110 |
   - 101 |
     001 |
```



Representación en complementos

- Para representar un número negativo se puede utilizar
  - Complemento a la base
  - Complemento a la base – 1

- Las sumas y restas quedan reducidas a sumas.
- Este sistema de representación de sumo interés ya que reduce la complejidad de la unidad aritmético lógica (no son necesarios circuitos específicos para restar).

Complemento a la base menos 1

El complemento a la base menos uno de un número, N, es el número que resulta de restar cada una de las cifras de N a la base menos uno del sistema de numeración que este utilizando.

Podemos restar dos números sumando al minuendo el complemento a la base menos uno del sustraendo. La cifra que se arrastra del resultado se descarta y se suma al resultado así obtenido.

Complemento a la base menos 1

En base 10 (Complemento a 9)

- Complemento a la base menos uno (a nueve) de 63 es 36;

$$\begin{array}{r} 99 \\ -\ 63 \\ \hline 36 \end{array}$$

- Si queremos resta 63 a 77

De manera normal	Con Complemento a la base menos 1 (a 9)
$\begin{array}{r} 77 \\ -63 \\ \hline 14 \end{array}$	$\begin{array}{r} 77 \\ +36 \\ \hline (1)13 \\ +\ \ 1 \\ \hline 14 \end{array}$

Cont…

- Complemento a nueve de 16 es 83;

```
  99
- 16
----
  83
```

- Queremos hacer 1100-0016:

De manera normal

```
 1100
-0016
----
 1084
```

Con Complemento a la base menos 1 (a 9)

```
 9999
-0016
----
 9983
```

```
  1100
 +9983   Complemento a 9
------
(1)1083
+ 0001
------
  1084
```

En base 2 (Complemento a 1)

- Complemento a la base menos uno (a uno) del número 10010 es:

$$\begin{array}{r} 11111 \\ -10010 \\ \hline 01101 \end{array}$$

- Complemento a uno de 101010 es:

$$\begin{array}{r} 111111 \\ -010101 \\ \hline 101010 \end{array}$$

Cont…

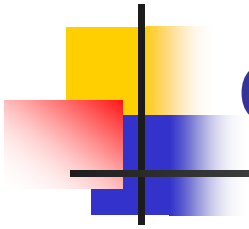
- Queremos Restar 1000111 – 10010:

De manera normal

```
  1000111
- 0010010
---------
  0110101
```

- Con complemento a 1 (de 0010010):

```
  1000111
+ 1101101   →  Complemento a 1 de 0010010
---------
 (1)0110100
+  0000001
---------
  0110101
```



Cont…

Fácilmente se observa que para transformar un número binario, N, a complemento a 1 basta con **cambiar en N los unos por los ceros y los ceros por los unos**.

Complemento a la base

El complemento a la base de un número, N, es el número que resulta de restar cada una de las cifras del número N a la base menos uno del sistema que se esté utilizando y, posteriormente, sumar uno a la diferencia obtenida.

Se pueden restar dos números sumando al minuendo el complemento a la base del sustraendo y despreciando, en su caso, el acarreo del resultado.

Complemento a la base

En base 10 (Complemento a 10)

- Complemento a la base (a diez) de 63 es 37;

$$\begin{array}{r} 99 \\ -\ 63 \\ \hline 36 \\ +\ 1 \\ \hline 37 \end{array}$$

- Si queremos resta 63 a 77

De manera normal	Con Complemento a la base (a 10)
$\begin{array}{r} 77 \\ -63 \\ \hline 14 \end{array}$	$\begin{array}{r} 77 \\ +37 \\ \hline (1)14 \end{array}$

En base 2 (Complemento a 2)

- Complemento a la base (a dos) del número 10010 es:

$$
\begin{array}{r}
11111 \\
-10010 \\
\hline
01101 \\
+\ 1 \\
\hline
01110
\end{array}
$$

- Complemento a dos de 101010 es:

$$
\begin{array}{r}
111111 \\
-010101 \\
\hline
101010 \\
+\ 1 \\
\hline
101011
\end{array}
$$

Cont…

- Queremos Restar 1000111 – 10010:

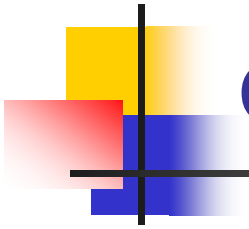
De manera normal

$$\begin{array}{r} 1000111 \\ -\ 0010010 \\ \hline 0110101 \end{array}$$

- Con complemento a 2 (de 0010010):

$$\begin{array}{r} 1000111 \\ +\ 1101110 \\ \hline (1)0110101 \end{array}$$

→ **Complemento a 2 de 0010010**



Cont…

Observamos que para transformar un numero binario, N, a complemento a 2 **basta con cambiar los 0 por 1 y los 1 por 0 de N y sumar 1 al resultado**.

Esto puede también ser visto como:

Recorrer el número desde el bit menos significativo hasta el mas significativo y dejar los bits iguales hasta el primer uno y luego cambiar los ceros por unos y los unos por ceros

Sistema de numeración octal

- La base es 8
- El conjunto de símbolos es:

{ 0, 1, 2, 3, 4, 5, 6, 7 }

Conversión de octal a decimal

- Se desarrolla el polinomio con b=8 y se opera en decimal.

Conversión de decimal a octal

- Aplicar el método de "divisiones y productos" con divisor y multiplicador 8.

Conversión "rápida" de binario a octal

- Agrupar cifras binarias de 3 en 3 y transformar con la tabla 1.

Conversión "rápida" de octal a binario

- Convertir cada cifra octal mediante la tabla

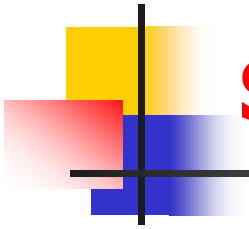
Cont...

- Ejemplo:
  - Haciendo uso de la tabla convertir
    $10001101100.11010_{(2} = N_{(8}$

$$10|001|101|100.110|10\ )_2 = 2154.64\ )_8$$

- Ejemplo:
  - Haciendo uso de la tabla convertir $537.24\ )_8 = N\ )_2$

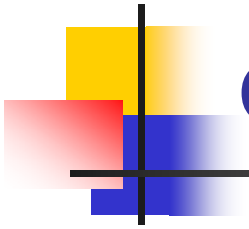
$$537.24\ )_8 = 101|011|111.010|100\ )_2$$



Sistema de numeración hexadecimal

- La base es 16
- El conjunto de símbolos es:

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}



Cont...

Conversión de Hexadecimal a decimal

- Se desarrolla el polinomio con b=16 y se opera en decimal.

Conversión de Decimal a hexadecimal

- Aplicar el método de "divisiones y productos" con divisor y multiplicador 16.

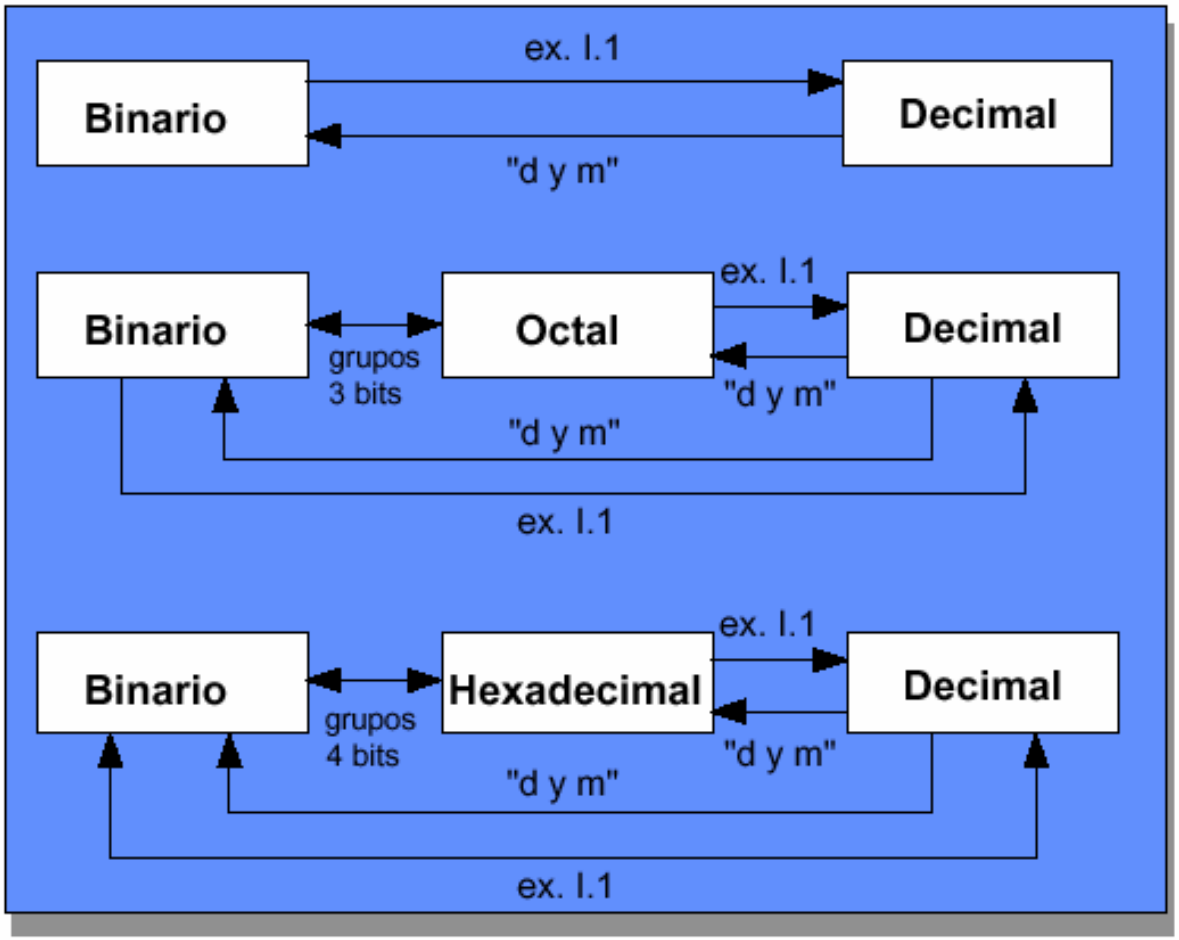
Conversión "rápida" de binario a hexadecimal

- Agrupar cifras binarias de 4 en 4 y transformar con la tabla
  - Ejemplo: 0010|0101|1101|1111 . 1011|1010 $_{(2}$ = 25DF.BA $_{(16}$

Conversión "rápida" de hexadecimal a binario

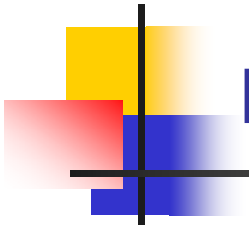
- Convertir cada cifra hexadecimal mediante la tabla
  - Ejemplo: 1ABC.C4 $_{(16}$ = 0001|1010|1011|1100 . 1100|0100 $_{(2}$

Resumen de cambios de base



Ejercicios en clases…

- Hacer las operaciones en binario:
  - $101011101_{)2} + 101001010_{)2} = N_{)8}$
  - $1100101011_{)2} + 100101101_{)2} = N_{)10}$
  - $101011101_{)2} - 10001010_{)2} = N_{)16}$
  - $110001011_{)2} – 10101101_{)2} = N_{)16}$
  - $10101.0101_{)2} * 2_{)10} = N_{)2}$
  - $1101.1010_{)2} * 25_{)10} = N_{)10}$
  - $1010100_{)2} / 2_{)10} = N_{)8}$
  - $10101.101_{)2} / 101_{)2} = N_{)2}$

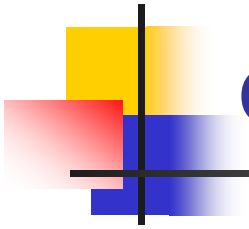


Representación Numérica

- Para la representación de los datos numéricos se debe tener en cuenta que las operaciones de la ALU están sujetas a las siguientes restricciones:
  - Los registros son de tamaño fijo.
  - Puede existir desbordamiento.
  - Presentan problemas con los números negativos.

- Es necesario, por ello, introducir nuevas formas de numeración basadas, por supuesto, en la representación binaria.
- Al conjunto de estas representaciones y su funcionamiento se le denomina aritmética binaria.

- En aritmética binaria debemos distinguir:
  - Representación para números enteros
  - Representación de números reales.



Cont...

- Números de precision finita
  - En la mayoría de las computadoras, la cantidad de memoria disponible para guardar números se fija en el momento de su diseño.
  - Con un poco de esfuerzo, el programador puede llegar a representar números 2 o 3 veces más grandes que este tamaño prefijado
  - Al hacerlo no termina de cambiar la naturaleza del problema: la cantidad de dígitos disponibles para representar un número siempre será fija.
  - Llamamos a estos números de precisión finita.

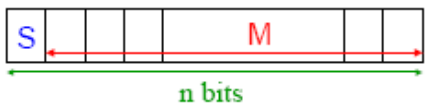
Representación de números enteros

- **Enteros sin signo**
  - No hace falta codificación.

- **Enteros con signo**
  - Los mas usuales son integer y long
  - Complemento a 1, Complemento a 2, representación signo-magnitud y exceso 2n-1
  - Todas se basan en tener 1 bit para el signo y el resto de la cifra (n-1 bits) para codificar el número entero a representar.
  - Se distingue entre números:
    - **Positivos:** Se almacenan con el bit de signo puesto a 0, y el valor absoluto
    - **Negativos:** Se almacenan con el bit de signo puesto a 1, y el complemento a 1 ó 2 del valor absoluto.

  - Permiten almacenar números desde
    - $-2^{(n-1)}$, hasta $+ (2^{(n-1)}) - 1$
    - Bytes: -128 a +127, words: -32768 a 32767

Signo y magnitud



rango: $[-2^{n-1}+1, +2^{n-1}-1]$

Representación de números reales (IEEE 754)

- ***Coma fija:*** La posición está fijada de antemano y es invariante.
  - Cada número se representa por n bits para la parte entera y m bits para la parte fraccionaria .
  - Nos ahorramos el punto
    - Dependerá de n y de m
  - Se puede producir un error de truncamiento.
  - Un mismo número en punto fijo puede representar a muchos números reales.
    - 1.25 (m=2), 1.256 (m=2), 1.2589 (m=2), 1.2596 (m=2), etc
  - El MSB es el signo
  - No todos los números reales pueden representarse con este formato

Cont…

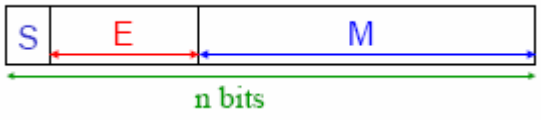
- ***Coma flotante:*** La posición de la coma es variable dependiendo del valor del exponente. Es de la forma:

  m 10exp (En decimal) m 2exp(En binario)

  - En decimal en la notación científica podemos escribir:

    1.9 x 109 o en forma corta 1.9E9

  - Tiene dos campos uno contiene el valor de la mantisa y el otro de valor del exponente.
  - El bit más significativo de la mantisa contiene el signo.
  - Existen tres formatos:
    - $Signo_N$ Mantisa Exponente → Directo
    - $Signo_M$ Exponente Mantisa → Comparación rápida
    - $Signo_E$ Exponente $Signo_N$ Mantisa → Precisión ampliada



Cont...

- Como un valor puede tener más de una representación, se normaliza la representación haciendo que el primer bit significativo de la mantisa ocupe la posición inmediatamente a continuación del signo.

mantisa normalizada

Sólo una cifra *a la izquierda* del punto.

$295.02*10^{-4} = 2.9502*10^{-2}$ (sumar 2 al exponente; 2 lugares a la izquierda)

En binario:

$110.01*2^{-4} = 1.1001*2^{-2}$

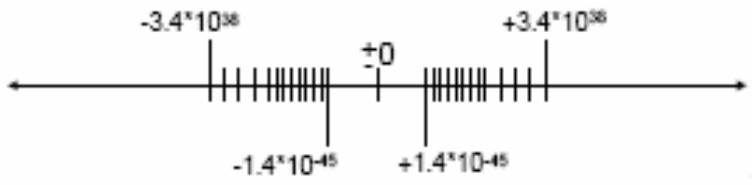
$0.0011*2^{-5} = 1.1000*2^{-8}$

- Trabajando mantisas normalizadas siempre el primer bit de la mantisa es el complemento del bit de signo, por lo que no es necesario incluirlo en la codificación.
- El bit que no se incluye recibe el nombre de bit implícito.
- Las características de los sistemas de representación en coma flotante son:
  - El exponente se representa en exceso a 2^{n-1}, siendo n el número de bits del exponente.
  - La mantisa es un número real normalizado, sin parte entera.
    - Su representación puede ser en cualquier sistema: módulo y signo, Complemento a 1 o Complemento a 2.
  - La base de exponenciación es una potencia de dos.

Cont...

- ***Representación en simple precisión:*** *Palabra de 32 bits.*
  - Signo Exponente Mantisa
  - 31 30 23 22 0
  - 1 bit 8 bits 23 bits
  - Un ejemplo en C es el float

1 bit de signo
8 bits de exponente: [-126, +127]
(*exceso* 127)
23 bits de mantisa *normalizada*,
con el primer bit *implícito*

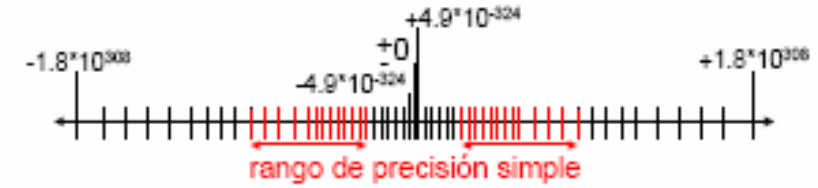


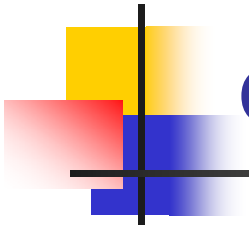
Cont...

- ***Representación en doble precisión:*** *Palabra de 64 bits.*
  - Signo Exponente Mantisa
  - 63 62 52 51 0
  - 1 bit 11 bits 52 bits

  - Un ejemplo en C es el Double

1 bit de signo
11 bits de exponente: [-1022, +1023]
(*exceso* 1023)
52 bits de mantisa *normalizada*,
con el primer bit *implícito*





Cont...

- Ejemplo 1:
  - -9.25_{10}

    Sean m =16, $n_E = 8$ ($\Rightarrow n_M = 7$) ,

    Pasamos a binario $\Rightarrow 9.25_{10} = 1001.01_2$

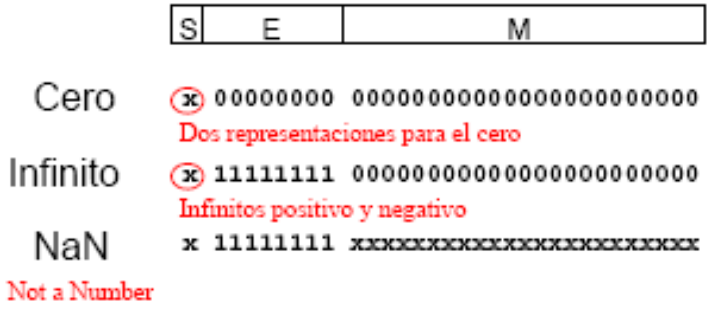
    Normalizamos $\Rightarrow 1.00101\ 2^3$

    Exponente (exceso a 2^7-1) $3_{10} = (127 + 3)_2 = 10000010$

1	1000 0010	0010 100
S_M	E	M

Cont…

Valores especiales



Números en Matlab

Todos los números son reales con precisión doble (64 bits)

eps = 2.2204e-016 = 2^-52

Se pueden *definir* enteros de 8, 16 y 32 bits

Pero no se pueden hacer operaciones!

Se han de convertir en reales para operar

Códigos de Entrada/Salida

- Asocian a cada símbolo una determinada combinación de bits.

  a = {0,1,2,...,8,9,A,B,...,Y,Z,a,b,...,y,z,*,",/,...}
  b = {0,1}n

- Con n bits podemos codificar $m=2^n$ símbolos distintos
- Para codificar m símbolos distintos se necesitan n bits,

$$n \geq \log_2 m = 3.32 \log (m)$$

Ejemplo:

- Para codificar las cifras decimales {0,1,2,3,4,5,6,7,8,9} se necesitarán :

  $n \geq 3.3221 \log(m) = 3.322$ bits

- es decir, 4 bits (para que se cumpla la relación)

- Por lo menos se necesitan 4 bits, pero pueden hacerse codificaciones con más bits de los necesarios. Tabla 2

- Con 4 bits no se usan $2^4 - 10 = 6$ combinaciones, y con 5 bits $2^5 - 10 = 22$ combinaciones.

Cont... Tabla 2

Alfabeto	Código I	Código II
0	0000	00000
1	1000	10001
2	0100	01001
3	1100	11000
4	0010	00101
5	1010	10100
6	0110	01100
7	1110	11101
8	0001	00011
9	1001	10010

Ejemplos de Códigos de E/S

- **Código ASCII**
  - El código ASCII se utiliza para representar caracteres.
  - Formado por 8 bits (cada carácter se expresa por un número entre 0 y 255)
  - Es un código estándar, independiente del lenguaje y del ordenador
  - Podemos distinguir dos grupos:
    - Los 128 primeros caracteres se denominan código ASCII estándar
      - Representan los caracteres que aparecen en una maquina de escribir convencional
    - Los 128 restantes se denominan código ASCII ampliado
      - Este código asocia un numero a caracteres que no aparecen en la maquina de escribir y que son muy utilizados en el ordenador tales como caracteres gráficos u operadores matemáticos.

- **Código EBCDIC**
  - Extended Binary Coded Decimal Interchange Code
    - Código Ampliado de Caracteres Decimales Codificados en Binario para Intercambio de Información
  - Es un sistema de codificación de caracteres alfanuméricos.
  - Cada carácter queda representado por un grupo de 8 bits.

- **Código Unicode**
  - Es de 16 bits, por lo que puede representar 65536 caracteres.
  - Es una extensión del ASCII para poder expresar distintos juegos de caracteres (latino, griego, árabe, kanji, cirílico, etc).