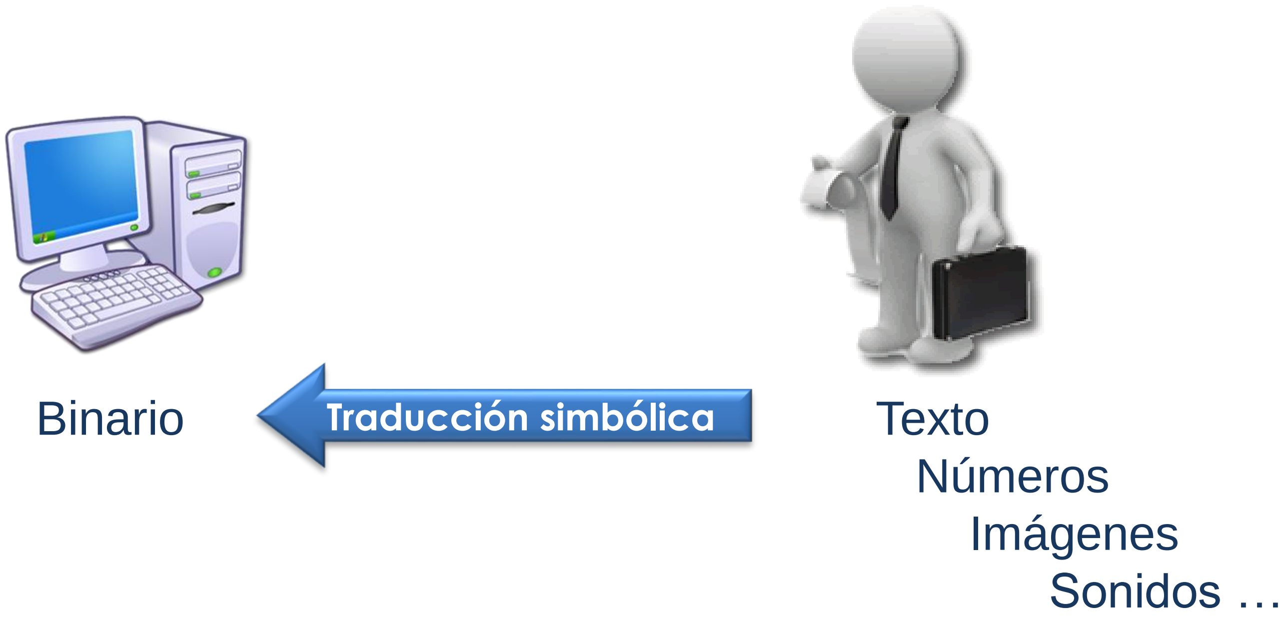
UCA
Universidad
de Cádiz

1. Aritmética binaria, formatos de representación numérica y arquitectura de Von Neumann

Contenidos

1.- Tipos de información: instrucciones y datos

- Los seres humanos utilizamos diferentes tipos de información en función de nuestros sentidos.
- Los ordenadores solo pueden procesar, directamente, información binaria.
- Para que el ordenador pueda utilizar un determinado tipo de información es necesario transformarla a binario: traducción simbólica



1.- Tipos de información: instrucciones y datos

- Todo ordenador maneja un conjunto finito de valores:
  - Binario (dos estados)
  - Ancho de los buses, tamaño de los registros, …

Ej.: En un registro/bus de 8 bits caben 256 valores distintos

¿Cómo representar un estado de ánimo?

- Algunos tipos de información son infinitos y continuos, mientras que la información se almacena de forma finita y discontinua.

Conjunto Números Naturales → Infinito

1, 2, 3, … ∞

Capacidad del ordenador → Finita: ej. registro de 16 bits

1, 2, 3, …, 65536, <u>65537, … ∞</u>

no se pueden representar

Características de la representación

• Toda representación elegida tiene limitaciones. Es preciso describir estas limitaciones y para ello se utiliza principalmente dos atributos:

- **Rango de representación:**

  Intervalo entre el menor y mayor elemento representable.

- **Resolución de la representación:**

  Diferencia entre un elemento representable y el siguiente.

  - Nos ofrece una idea del máximo error cometido en la representación de una cantidad.
  - La resolución puede ser constante o no.

Características de la representación

La representaciones usadas en un computador son causa y efecto de las características del mismo (tanto de la estructura como de la arquitectura)

Representación seleccionada

⬇

Tipos de datos

Operaciones sobre datos

⬇

Características de la máquina

Tamaño de información característico

- Octeto o byte: secuencia de 8 bits
- Palabra: Información manipulada en paralelo en el interior del computador (32 bits típico, 16, 8, 4 y 36, 48, 64)

↑palabra ⇒ ↑nº representables ⇒ ↑potencia de cálculo

- Media palabra:

  Para cuando no hace falta gran precisión. Ahorra “sitio”

- Doble palabra:

  Para mejorar precisión de cálculo. Necesita de más “sitio”

- Resolución de acceso a memoria:

  Tamaño mínimo de información que es capaz de leer directamente de memoria.

Sistemas de numeración posicionales-Repaso Definición

Binario: Base 2, {0,1}

Posición	S	4	3	2	1	0	-1	-2	-3	-4
Peso	-/+	16	8	4	2	1	½	¼	1/8	1/16

Hexadecimal Base 16, {0,1,2, 3,..., 8, 9, A, B, ..., F }

Posición	S	4	3	2	1	0	-1	-2	-3	-4
Peso	-/+	16^4	16^3	16^2	16^1	16^0	16^{-1}	16^{-2}	16^{-3}	16^{-4}

Sistemas de numeración posicionales - Repaso Conversión

- Decimal a binario: *Ej: $318{,}6_{(10}$*

► <u>Parte entera</u>: Potencias implícitas

$$318 = 256+32+16+8+4+2 = 2^{8} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2^{1}$$

► *posiciones 8, 5, 4, 3, 2, 1*

► <u>Parte fraccionaria</u>: parte entera del producto por 2

0,6 x 2 = 1,2 ► 1 en posición -1 ; 0,2 x 2 = 0,4 ► 0 en posición -2 ;

0,4 x 2 = 0,8 ► 0 en posición -3 ; 0,8 x 2 = 1,6 ► 1 en posición -4

$318{,}6_{(10} = 100111110{,}1001_{(2}$

- Binario a hexadecimal: convertir grupos de 4 cifras binarias a derecha e izquierda de la coma

Bin.	1	0	0	1	1	1	1	1	0	1	0	0	1
Hex.	1	3				E				9			
13E,9													

Códigos de Representación

- <u>Signo magnitud</u>: Signo = 0 (si es positivo), Signo = 1 (si es negativo)
  $-2^{n-1} + 1 \leq x \leq 2^{n-1} - 1$
- <u>Complemento a 1</u>: binario (si es positivo), complementado (si es negativo)
  $-2^{n-1} + 1 \leq x \leq 2^{n-1} - 1$
- <u>Complemento a 2</u>: binario (si es positivo), C1+1 (si es negativo)
  $-2^{n-1} \leq x \leq 2^{n-1} - 1$
- <u>Exceso</u>: Se suma el exceso 2^{n-1}-1 al número que se quiere representar
  $-2^{n-1} + 1 \leq x \leq 2^{n-1}$

$$R = A + 2^{n-1}-1$$

Número	S-M	C-1	C-2	Exc 4bits $-7 \leq x \leq +8$	Exc 5 bits $-15 \leq x \leq +16$	Exc 6 bits $-31 \leq x \leq +32$
3	0 11	011	011	1010	10010	
-12	1 1100	10011	10100	Imposible	00011	010011
23	0 10111	010111	010111	Imposible		
-72						

Códigos de Representación - Ejemplo -

Disponemos de un registro de **6 bits** y queremos representar el número **-23**

Signo magnitud	C - 1	C - 2	Exceso
$-2^{n-1} + 1 \leq x \leq 2^{n-1} - 1$ $-31 \leq x \leq +31$	$-2^{n-1} + 1 \leq x \leq 2^{n-1} - 1$ $-31 \leq x \leq +31$	$-2^{n-1} \leq x \leq 2^{n-1} - 1$ $-32 \leq x \leq +31$	$-2^{n-1} + 1 \leq x \leq 2^{n-1}$ $-31 \leq x \leq +32$
como el número es - el bit de signo es 1	como el número es - se complementan todos sus dígitos	como el número es - se complementan todos sus dígitos y después se suma 1	exceso = $2^{n-1}-1 = 31$ se suma el exceso $-23 + 31 = 8$
$\boxed{1}$ 1 0 1 1 1	0 1 0 1 1 1 → 1 0 1 0 0 0	0 1 0 1 1 1 → 1 0 1 0 0 0 + 1 1 0 1 0 0 1	0 0 1 0 0 0
Resultado:	**Resultado:**	**Resultado:**	**Resultado:**
$\boxed{1}$ 1 0 1 1 1	1 0 1 0 0 0	1 0 1 0 0 1	0 0 1 0 0 0

Formatos numéricos

Objetivo:

- Cuando tenemos un número de cifras binarias considerables, aparece el problema de cómo representar este Nº en un espacio limitado.

- Representar números reales en un espacio de bits limitado

- Dos formatos:
  - Coma fija: más sencillo pero con menor rango de representación
  - Coma flotante: más complejo pero con una amplia capacidad de representación

Formatos numéricos: coma fija

Delimita tres zonas de representación Signo, Parte entera, Parte fraccionaria

- ❑ Hay que definir previamente el Nº de bits asignados
- ❑ La parte entera se introduce de derecha a izquierda
- ❑ La parte fraccionaria de introduce de izquierda a derecha
- ❑ El signo siempre es el bit más significativo
- ❑ Si el nº de bits a introducir > capacidad del formato aparecen truncamientos → Errores de representación

Definición: P.E. 4 bits, P.F. 5 bits (el signo se sobreentiende que es 1 bit)									
S	P.E.				P.F				
1	1	0	0	1	1	1	0	1	1
	←				→				

Formatos numéricos: coma flotante

Coma flotante: parte de la notación exponencial de un número

$$-12{,}456 = -1{,}2456 \times 10^{1}$$

$$0{,}0000000213 = 2{,}13 \times 10^{-8}$$

$$3916538910213 = 3{,}916538910213 \times 10^{12}$$

$$-11011{,}001 = -1{,}1011001 \times 2^{100}$$

- Partiendo de una representación en base (B)
- Cualquier número se representa mediante:
  - Signo (S),
  - Mantisa (M),
  - Exponente (E)

$$\mathbf{S\ M \times B^{E}}$$

Formatos numéricos: coma flotante

Coma flotante: Características

$$S\ M \times B^{E}$$

- La mantisa se normaliza: se ajusta a la forma fraccionaria 1,xxxx
- Puede utilizar bit implícito: no se representa el bit más significativo de la mantisa, porque se sobreentiende que es 1.
- La mantisa es fraccionaria: se introduce de izquierda a derecha.
- El exponente es un entero representado en exceso $2^{n-1}-1$. Siendo n el nº de bits del exponente

Formatos numéricos: Estándar IEEE754

- **Estándar** para el almacenamiento en coma flotante utilizado por la mayoría de los ordenadores.

S	Exponente	Mantisa

- Define dos formatos:
  - **Precisión simple**: 32 bits (S: 1, E: 8, M: 23). Variables de tipo float
  - **Precisión doble**: 64 bits (S: 1, E: 11, M: 52). Variables de tipo double

- Características:
  - ❖ Exponente: exceso 2^{n-1}-1
  - ❖ Mantisa: signo-magnitud, normalizada con bit implícito

Formatos numéricos: Estándar IEEE754

Precisión simple: 32 bits (signo: 1, exponente: 8, mantisa: 23)

S	Exponente	Mantisa

- El valor se calcula con la siguiente expresión:

$$N = (-1)S \times 1.M \times 2^{E-127}$$

- Donde:

    S = 0 indica número positivo, S =1 indica número negativo

    $0 < E < 255$ ($E=0$ y $E=255$ indican excepciones)

    00000000000000000000000 ≤ M ≤ 11111111111111111111111

- Bit implícito: Una vez normalizado, el bit más significativo es 1, **no** se almacena en M para dejar espacio para un bit más (aumenta la precisión)

Formatos numéricos: Estándar IEEE754

Ejemplo: representar en precisión simple este número

$+101110,01010111010000111110000011111000010011_2$

1º Normalizar:

$1,0111001010111010000111110000011111000010011_2 * 2^5$

2º Representar la mantisa utilizando bit implícito:

solo se cogen los 23 primeros bits el resto se trunca

<u>1</u>,**01110010101110100001111**<u>10000011111000010011</u>

BIT IMPLÍCITO **TRUNCADO**

3º Representar el exponente en exceso a $\mathbf{2^{n-1}-1}$:

5 + 127 = 132 = **10000100**

0	1	0	0	0	0	1	0	0	0	1	1	1	0	0	1	0	1	0	1	1	1	0	1	0	0	0	0	1	1	1	1

32 bits