66.70 Estructura del Computador

Sistemas de representación numérica

✓¿Qué número representa “**312**” ?

Sistemas para la representación de números

Símbolos:

- Desarrollo histórico: marcas, nudos en una cuerda, simbología, …

Organización:

- Sistemas aditivos y Sistemas posicionales

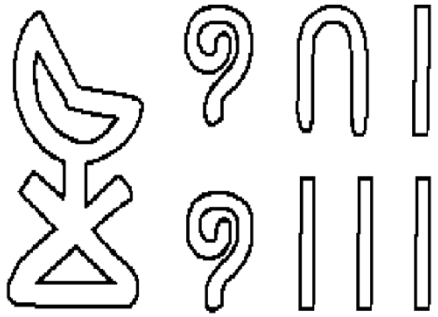
Sistemas de Numeración **Aditivos**

El sistema egipcio:

vara 1 | talón 10 | cuerda 100 | flor 1 000

dedo 10 000 | pez 100 000 | hombre asustado 1 000 000

El número 1214



Sistemas de numeración posicionales

El sistema árabe

- ✓ Símbolos que lo forman

٠ ١ ٢ ٣ ٤ ٥ ٦ ٧ ٨ ٩

0 1 2 3 4 5 6 7 8 9

- ✓ Con una cantidad **limitada de símbolos** repr. todo número
- ✓ Es decimal (base 10)

– *Desarrollado en India antes del siglo VII e introducido en Europa por los árabes.*
– *Babilonios, chinos y mayas en distintas épocas llegaron al mismo principio*
– *Entre el sistema actual y el de los Indios sólo hay diferencias en la forma que escribimos los dígitos*

El sistema decimal no es el único sistema numérico posicional

Un sistema numérico posicional *queda definido por:*

- ✓ Símbolos
- ✓ Cantidad de símbolos (base)
- ✓ Peso de cada posición (generalmente son potencias crecientes de la base)
- ✓ Cantidad de posiciones

Representar $42|_{10}$ en diferentes bases

- base 3, octal, base 2, hexadecimal. etc.

Conversión entre diferentes bases

Casos:

a. Conversión de cualquier base a base 10

b. Conversión de base 10 a otra base

c. Conversión entre dos bases diferentes de 10

d. Bases potencias de otras bases

Métodos de conversión entre bases

- Cualquier base a base 10 -> sumatoria
- De base 10 a otra base
  - Divisiones sucesivas
  - Estimación en base a los pesos
- Base que es potencia de otra base:
  - Agrupar y convertir cada grupo en un dígito
  - Desagrupar dígito a dígito

Cantidad de dígitos necesaria para un mismo rango de representación

Binary (base 2)	Octal (base 8)	Decimal (base 10)	Hexadecimal (base 16)
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F

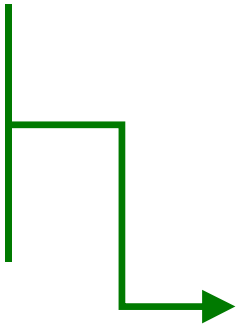
- Representar 262 en binario y en hexadecimal

Rango de representación

Símbolos (cuáles y cuántos)

Cantidad de dígitos

Peso de cada posición



Rango

- Cuántos valores distintos?
- Cuál es el valor máximo?
- Cuál es el valor mínimo?

Comparar procesadores de 8, 16 y 32 bits

Representación de
NUMEROS CON PARTE FRACCIONARIA

1) Sistemas de punto fijo

- Puede aplicarse a una base cualquiera, incluyendo la binaria
- Cómo convertir un número en base diez a otra base
- Cómo convertir un número a base 10
- Precisión de la conversión

2) Sistemas de punto flotante

Representación de ENTEROS CON SIGNO

- *Magnitud y signo*
- *Complemento a la base menos 1*
- *Complemento a la base*

Representación de ENTEROS CON SIGNO

Decimal	Two's Complement	Ones' Complement	Signed Magnitude
–8	1000	—	—
–7	1001	1000	1111
–6	1010	1001	1110
–5	1011	1010	1101
–4	1100	1011	1100
–3	1101	1100	1011
–2	1110	1101	1010
–1	1111	1110	1001
0	0000	1111 or 0000	1000 or 0000
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	0100	0100
5	0101	0101	0101
6	0110	0110	0110
7	0111	0111	0111

Representación en magnitud y signo

- **Forma de representación**

+ N ⟶ Idem binario puro

- N ⟶ Primer bit a izq es 1

- **Rango representable**

$$(-2^{n-1} + 1)_{10} \leq x \leq (2^{n-1} - 1)_{10}$$

Con 4 bits

1111	−7
1110	−6
1101	−5
1100	−4
1011	−3
1010	−2
1001	−1
1000 or 0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Representación en complemento a la base menos 1

En binario (base 2) => "COMPLEMENTO A 1"

- **Forma de representación**

  + N ⟶ Idem binario puro

  - N ⟶ $C_{b-1}(N) = b^n - 1 - N$

  b: *base* **n**: *cant. de dígitos*

  En binario:

  - N ⟶ $C_1(N) = 2^n - 1 - N$

  Se puede obtener haciendo la resta o ***invirtiendo bit a bit***

- **Rango representable**

$$(-2^{n-1} + 1)_{10} \leq x \leq (2^{n-1} - 1)_{10}$$

¿ Cuál es el rango representable con 8 bits, 16 bits y 32 bits ?

Con 4 bits

1000	–7
1001	–6
1010	–5
1011	–4
1100	–3
1101	–2
1110	–1
1111 or 0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Representación en complemento a la base

En binario (base 2) => "COMPLEMENTO A 2"

- **Forma de representación**

  + N ⟶ Idem binario puro

  - N ⟶ $C_b(N) = b^n - N$

  b: *base* **n**: *cant. de dígitos*

  En binario:

  - N ⟶ $C2(N) = 2^n - N$

  Se puede obtener haciendo la resta
  *o **invirtiendo bit a bit y sumando 1*** ⟵ Basado en complemento a 1

- **Rango representable**

$$(-2^{n-1})_{10} \leq x \leq (2^{n-1} - 1)_{10}$$

¿ Cuál es el rango representable con 8 bits, 16 bits y 32 bits ?

Con 4 bits

1000	−8
1001	−7
1010	−6
1011	−5
1100	−4
1101	−3
1110	−2
1111	−1
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Representación de ENTEROS CON SIGNO

Decimal	Two's Complement	Ones' Complement	Signed Magnitude
–8	1000	—	—
–7	1001	1000	1111
–6	1010	1001	1110
–5	1011	1010	1101
–4	1100	1011	1100
–3	1101	1100	1011
–2	1110	1101	1010
–1	1111	1110	1001
0	0000	1111 or 0000	1000 or 0000
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	0100	0100
5	0101	0101	0101
6	0110	0110	0110
7	0111	0111	0111

Convertir a base 10
números en compl. a 1 y compl. a 2

- **Si el bit más significativo es 0**
  - ✓ El número es positivo
  - ✓ Se convierte como si estuviera en binario puro (sumatoria de pesos)

- **Si el bit más significativo es 1**
  - ✓ El número es negativo
  - ✓ Fue obtenido complementando su módulo

¿ Justificación ?

Invierto ese proceso
calculando su complemento

Tengo su módulo
Convierto por sumatoria de pesos

¿ Cómo puedo distinguir si se trata de un entero con signo o sin signo ?
¿ Cómo puedo distinguir si se trata de complemento a 1 o complemento a 2 ?

Suma de números binarios

Sistema numérico:

- **8 bits**
- **Enteros sin signo**

```
  01110101
+ 11010110
----------
  ????????
```

Sistema numérico:

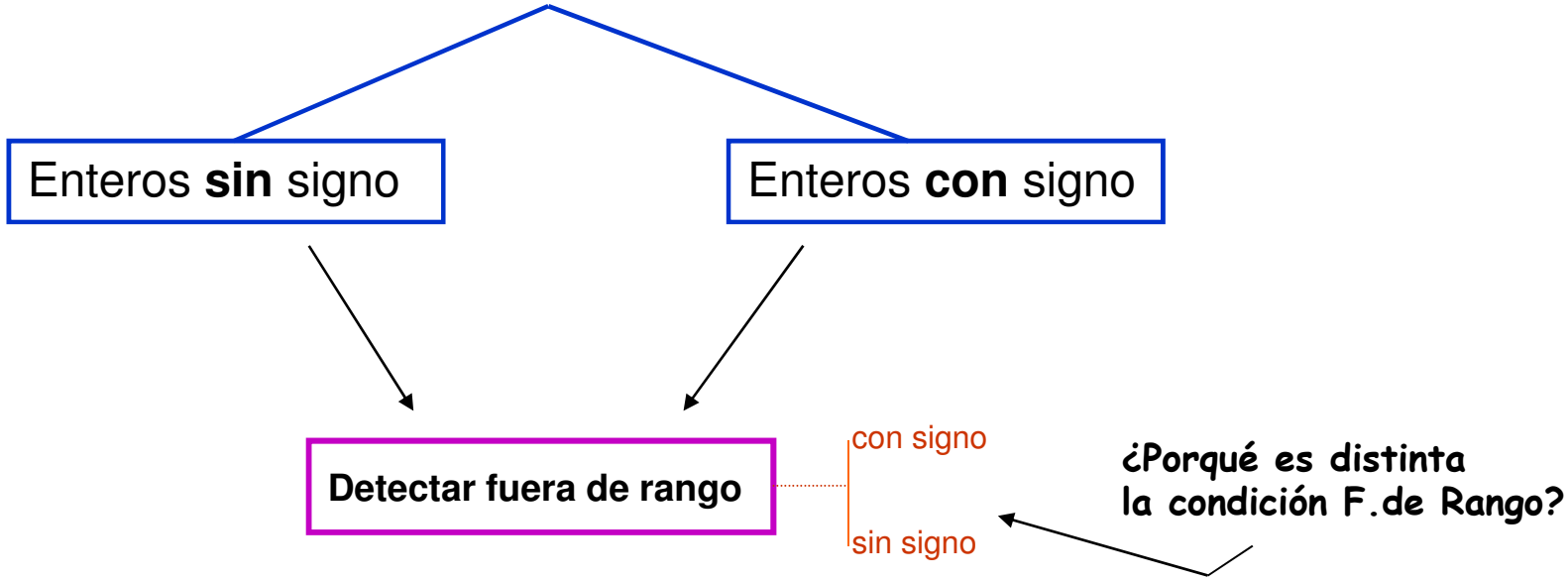
- **8 bits**
- **Enteros con signo**
- **Repres. en compl. a 2**

```
  01010110
+ 11010010
----------
  ????????
```

Resultado:

- **Suma**
- **Se fue de rango?**

Suma *de números binarios*



Indicadores (*flags*)

- C Carry
- V Overflow
- Z Cero
- N Signo
- P Paridad

*Operación **resta** en binario*

- *Forma directa*
- *Como suma del complemento*

Álgebra de números VS. microprocesador
Ley asociativa

El álgebra dice que:

$$a + (b + c) = (a + b) + c$$

Qué dice la computadora si el proceso se hace con 8 bits?

$$\begin{aligned} a &= 60 \\ b &= 105 \\ c &= -50 \end{aligned}$$

Cómo verificarlo en mi programa?

Suma de dos o más números definidos con ***distinta cantidad de bits***

- **Enteros sin signo**

Enteros con signo

“EXTENSIÓN” DEL SIGNO

Suma de
Números con parte fraccionaria

(Punto fijo)

Ejemplo:

	3.25	0011,0100
+		
	10.75	1010,1100
		????????

Suma de
Números con parte fraccionaria

(Punto fijo)

Ejemplo:

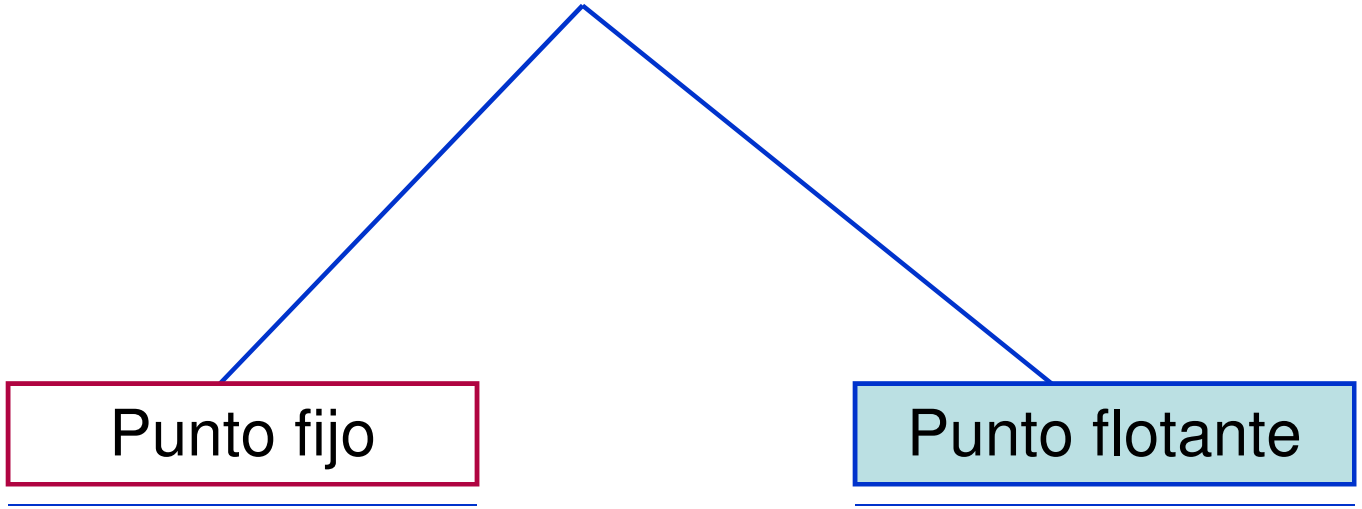
	3.25	0011,0100
+		
	10.75	1010,1100
		????????

- Opera como en números enteros
- Se implementa operando con números enteros y definiendo un factor de escala fijo

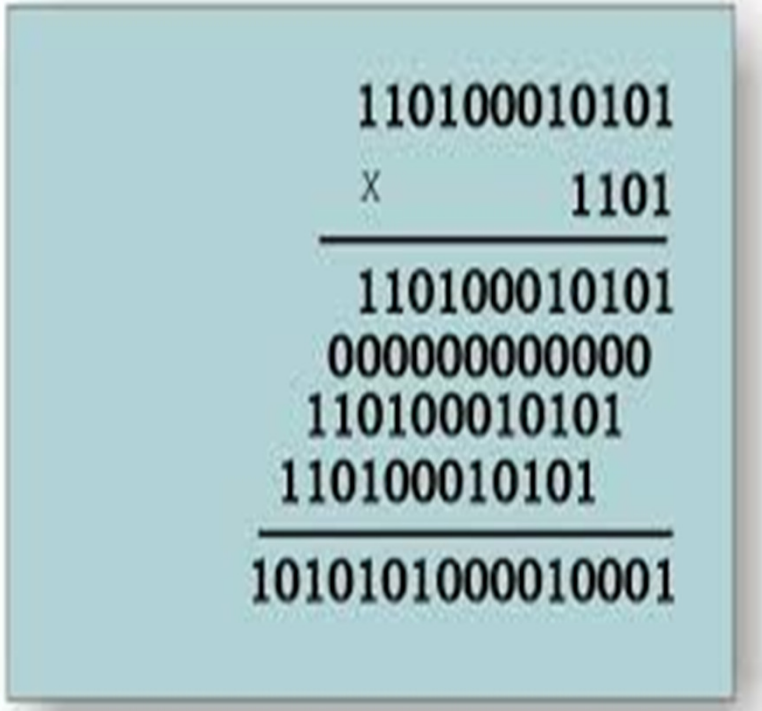
Cuánto vale en el ejemplo?

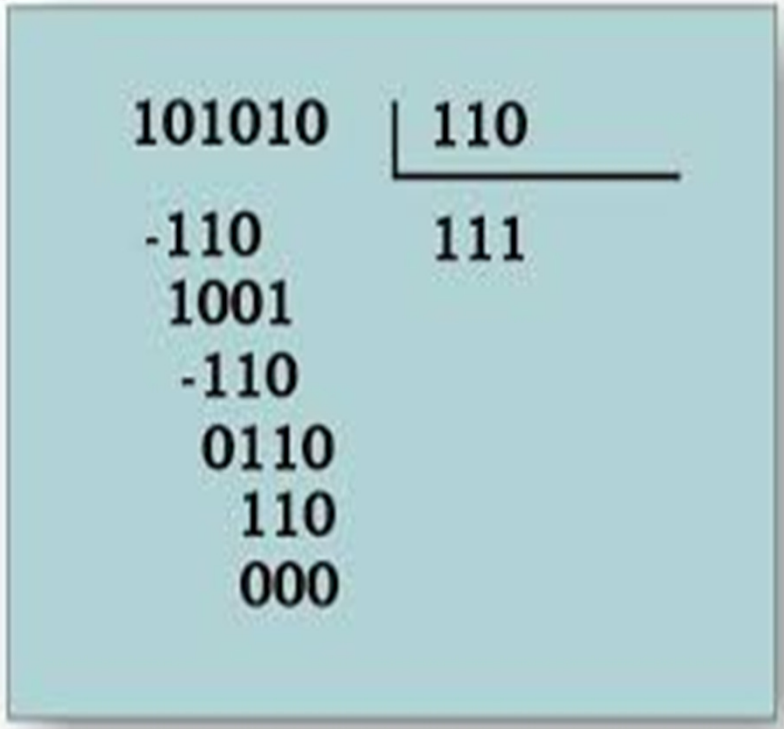
Representación de

Números con parte fraccionaria



Multiplicación y división de números enteros





Multiplicación y división de números enteros

- **Método general**

- **Por desplazamientos a derecha e izquierda**
  - ✓ Implementación sencilla
  - ✓ Alta velocidad de proceso

Errores numéricos famosos

25 de febrero de 1991 «Guerra del Golfo»

El sistema antimisiles falló …..

Los **misiles Scud** tienen una velocidad de 1,7 km/s, Mach 5

El **sistema antimisiles Patriot** sabiendo la velocidad predecía su posible futura ubicación y verificaba la trayectoria

La trayectoria del misil se calculaba en función de dos variables (números reales)
- Velocidad (dada por conocida)
- El último momento de tiempo en que fue detectado en el radar (mediante un **reloj interno**)

Cálculo del tiempo:
- Tiempo: variable entera medida en «décimas de segundo» => **convertir a segundos**

  => multiplico por 1/10 => en binario es: 0,0001100110011001100110011001100… (repite hasta el infinito)

Es aproximado a 0.00011001100110011001100 => error de conversión "apenas" 0,000000095 segundos no afecta el cálculo

Porqué falló?

Bibliografía

- GINZBURG M "Introducción a las Técnicas Digitales con Circuitos Integrados“. 8va Ed Bibl. Téc.Sup.1998

- HILL F, PETERSON G. "Teoría de Conmutación y Diseño Lógico”, Limusa.1992

- MURDOCCA M.J., HEURING V. P."Principios de Arquitectura de Computadoras", Prentice Hall, 2002

- Morris Mano – “Arquitectura de Computadores”