



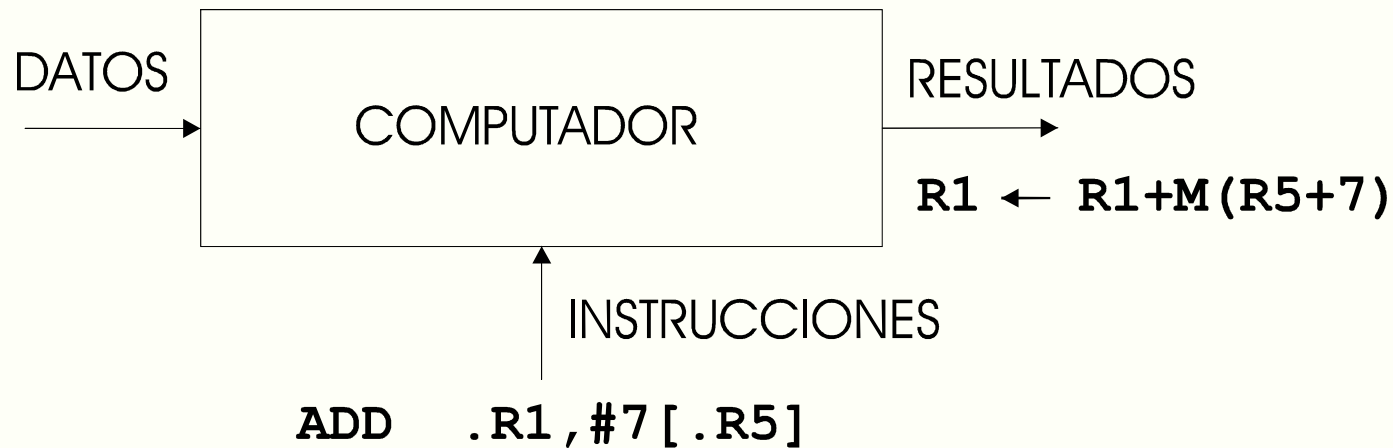
# Representación y Aritmética

1. Introducción
  - Representaciones alfanuméricas y numéricas
  - Operador y estructura de la ALU
2. Representación en coma fija
  - Binario sin signo
  - Complemento a 2
  - Complemento a 1
  - Signo-magnitud
  - Exceso a M
3. Representación en coma flotante
  - Definición, rango y resolución
  - Normalización y bit implícito
  - Suma y resta
  - Redondeo y bits de guarda
  - Estándar IEEE 754

## BIBLIOGRAFÍA

- Fundamentos de los computadores. Pedro de Miguel. Editorial Paraninfo, 9ª edición, 2004.
- Estructura y diseño de computadores. Patterson-Hennessy. Editorial Reverté, 2000
- Organización y arquitectura de computadores. Stallings. Prentice Hall, 7ª edición, 2006
- Computer Arithmetic Systems. Omondi. Prentice Hall International, 1994
- Estructura de computadores: Problemas resueltos. García Clemente y otros. RAMA, 2006

## INFORMACIÓN QUE LLEGA AL COMPUTADOR



- Datos e Instrucciones definidos por:
  - Símbolos (letras, números, caracteres ...)
  - Ideas (operaciones, movimientos, modificaciones ...)

## REPRESENTACIÓN DE LA INFORMACIÓN (2)

- **CONDICIONANTES DEL COMPUTADOR**
  - **CIRCUITOS INTEGRADOS DEL COMPUTADOR:**  
**UTILIZACIÓN DEL SISTEMA BINARIO**
  - **EL COMPUTADOR ES FINITO:**  
**LAS REPRESENTACIONES SON ACOTADAS**
  - **DISEÑO DE SUS UNIDADES FUNCIONALES:**  
**EXISTEN TAMAÑOS PRIVILEGIADOS** (byte, palabra, ..)
  
- **MODOS DE REPRESENTACIÓN**
  - **REPRESENTACIONES ALFANUMÉRICAS**
  - **REPRESENTACIONES NUMÉRICAS**
  - **REPRESENTACIONES REDUNDANTES**
  - **REPRESENTACIONES GRÁFICAS**
  - **REPRESENTACIONES ETIQUETADAS**

## REPRESENTACIONES ALFANUMÉRICAS (1)

- REPRESENTAN:
  - Las 26 letras del alfabeto (Mayúsculas y minúsculas)
  - Los 10 dígitos decimales
  - Un conjunto de caracteres especiales (+ , - = < ...)
  - Un conjunto de caracteres de control (no visibles)
  
- CARACTERÍSTICAS:
  - Facilidad para comprobar un carácter numérico
    - ASCII: desde H'30 hasta H'39
  - Fácil equivalencia Mayúsculas y minúsculas
    - ASCII: desde H'41 (A) hasta H'5A (Z)
    - ASCII: desde H'61 (a) hasta H'7A (z)
  - Fácil comprobación si es carácter de control
    - ASCII: desde H'00 (NUL) hasta H'1F (US)
    - ASCII: excepción H'7F (DEL)

## TABLA DE CÓDIGOS ASCII

Carácter más significativo

HEX	0	1	2	3	4	5	6	7
0	NUL	DLE	Space	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	Bell	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Carácter menos significativo

## REPRESENTACIONES ALFANUMÉRICAS (2)

- ASCII 8 bits:
  - Norma ISO que añade la representación de caracteres no presentes en inglés.
  - á es el 0xE1 é es el 0xE9. Á es el 0xC1 y É es el 0xC9.
  
- UTF-8:
  - Codificación de un carácter con varios bytes.
    - Codificación es de un byte: carácter es ASCII de 7 bits.
    - Codificación es de dos bytes: carácter pertenece a lenguas romances y otras.
    - 110000xx 10xxxxxx: xxxxxxxx es el carácter ISO de 8 bits. á es 0xC3 0xA1
    - Codificación es de tres bytes: carácter es de lenguas asiáticas.
    - Codificación es de cuatro bytes: otros.



## REPRESENTACIONES NUMÉRICAS

- LIMITACIONES DE UNA REPRESENTACIÓN
  - NÚMERO FINITO DE NÚMEROS REPRESENTABLE:  
**RANGO DE REPRESENTACIÓN** (Intervalo entre el mayor y el menor número representables)
  - NÚMERO FINITO DE BITS PARA LA REPRESENTACIÓN:  
**RESOLUCIÓN** (Diferencia entre los valores de un número representable y el inmediato siguiente)
  - OPERACIONES CON RESULTADOS NO REPRESENTABLES:  
**DESBORDAMIENTO** (Cuando un resultado está fuera del rango de representación)
  
- SISTEMAS POSICIONALES CON BASE

$b = \text{base} = n^0 \text{ natural} > 1$

$\text{Rep}(X) = (\dots x_2 x_1 x_0 x_{-1} x_{-2} \dots)$  con  $x_i \in \{b-1, b-2, \dots, 1, 0\}$

$$V(X) = \sum_{i=-\infty}^{i=\infty} x_i b^i = \sum_{i=0}^{i=\infty} x_i b^i + \sum_{i=1}^{i=\infty} x_{-i} b^{-i}$$

## CAMBIO DE BASE (1)

- Parte Entera =  $\dots x_2 b^2 + x_1 b^1 + x_0 b^0$   
Dividiendo la Parte Entera por  $b$  se obtiene:
  - Cociente =  $\dots x_2 b^1 + x_1 b^0$
  - Resto =  $x_0$
- Parte Fraccionaria =  $\dots ,x_{-1} b^{-1} + x_{-2} b^{-2} + x_{-3} b^{-3} + \dots$   
Multiplicando la Parte Fraccionaria por  $b$  se obtiene:
  - Parte Entera =  $x_{-1}$
  - Parte Fraccionaria =  $\dots , x_{-2} b^{-1} + x_{-3} b^{-2} + \dots$
- Relación  $b=2^K$  . Cada  $K$  bits de la representación binaria de un número constituyen un dígito en su representación en base  $b$ .
- Conversión de base  $b=2^K$  a decimal:  
 $010101,1010_{(2)} = 2^4 + 2^2 + 2^0 + 2^{-1} + 2^{-3} = 21,625_{(10)}$   
 $A27,8C_{(16)} = 10 \times 16^2 + 2 \times 16^1 + 7 \times 16^0 + 8 \times 16^{-1} + 12 \times 16^{-2} = 2599,546875_{(10)}$

## CAMBIO DE BASE (2)

- Ejemplo: Expresar  $N = 2202,735_{(10)}$  en base 16, 8 y 2.

$$2202 = 16 \times 137 + 10 \rightarrow x_0 = 10 \text{ (A)}$$

$$137 = 16 \times 8 + 9 \rightarrow x_1 = 9 \text{ y } x_2 = 8$$

$$0,735 \times 16 = 11,760 \rightarrow x_{-1} = 11 \text{ (B)}$$

$$0,760 \times 16 = 12,160 \rightarrow x_{-2} = 12 \text{ (C)}$$

Seguir hasta obtener el número de dígitos deseado

$$N = 89A,BC \dots_{(16)}$$

Expandiendo cada dígito hexadecimal en 4 bits:

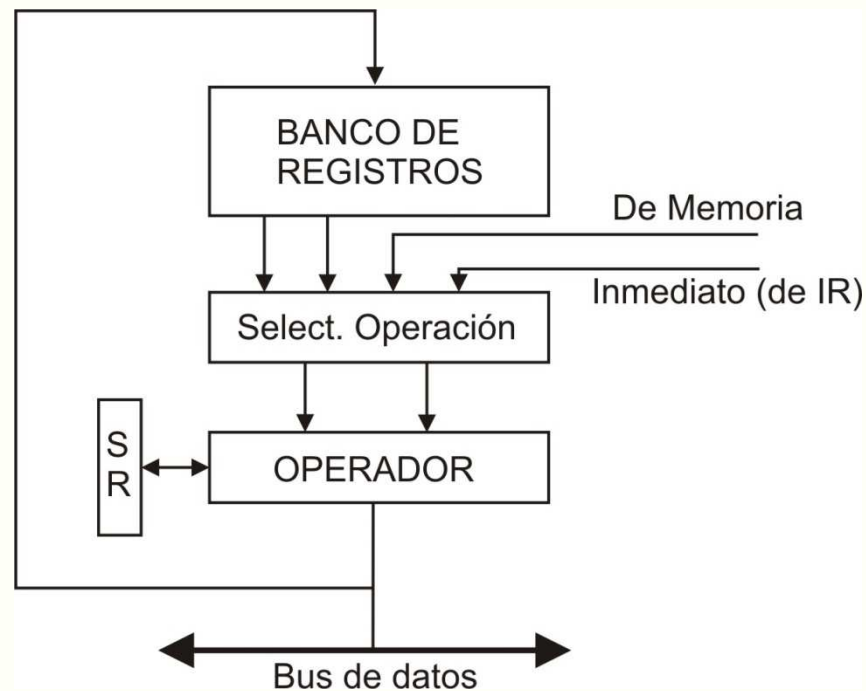
$$N = 1000 \ 1001 \ 1010, \ 1011 \ 1100 \ \dots_{(2)}$$

Agrupando cada 3 bits en un dígito octal:

$$N = 100 \ 010 \ 011 \ 010, \ 101 \ 111 \ 00?_{(2)} = 4232,57 \ \dots_{(8)}$$

## OPERADOR Y ESTRUCTURA DE LA ALU

- Operador: circuito que realiza una operación
- Registro de estado (SR). Los flags más usuales son: Acarreo (C), Cero (Z), Signo (S), Desbordamiento (V), Paridad (P), Resta (N), Acarreo BCD (H)
- Estructura de la ALU (modelo de ejecución Registro-Memoria)



## OPERACIONES DE LA ALU (1)

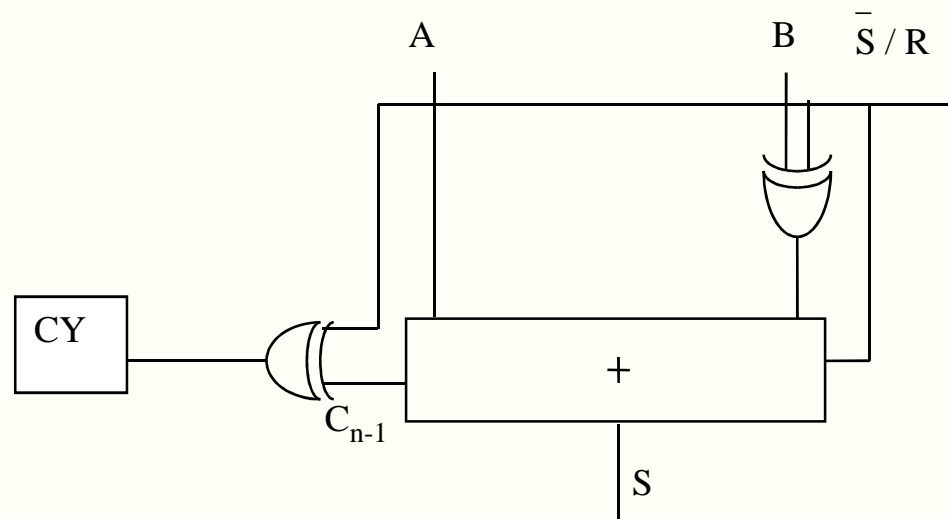
- OPERACIONES LÓGICAS (NOT, OR, AND, XOR, ...)
  - Actúan sobre los operandos bit a bit:
  - $(1001) \text{ XOR } (0101) = 1100$
  
- DESPLAZAMIENTOS
  - Lógicos: rellenan los huecos generados con ceros, ya sean a la derecha o a la izquierda
  - Aritméticos: realizan la multiplicación por 2 (a la izquierda) o división por 2 (a la derecha). Dependen de la representación:
    - Multiplicación en complemento a 2: Se rellena el hueco con 0 y hay desbordamiento si cambia de signo.
    - División en complemento a 2: Siendo  $A=(a_{n-1} \dots a_1 a_0)$  y  $a=|A|$ , si  $A < 0$  resulta  $A/2=2^{-1}(2^n-a)=(2^n-a/2)-2^{n-1}$ , por lo que hay que poner un 1 en el hueco generado
  - Concatenados: entre registros y con biestables (acarreo)
  - Circulares o rotaciones

## OPERACIONES DE LA ALU (2)

- EXTENSIÓN DE SIGNO
  - Representa un dato de  $n$  bits con  $m$  bits,  $m > n$
  - Depende de la representación. En complemento a 2 con  $a = |A|$  siendo  $A < 0$ ,  $2^m - a = (2^n - a) + (2^m - 2^n)$  por lo que hay que rellenar con 1 los  $(m - n)$  bits añadidos
  
- CAMBIO DE SIGNO
  - Dado un número  $a$ , se obtiene  $-a$ .
  - Depende del sistema de representación utilizado.
  
- SUMA/RESTA
  - Depende del sistema de representación utilizado.

## BINARIO SIN SIGNO

- $\text{Rep}(X) = (x_{n-1} \ x_{n-2} \ \dots \ x_1 \ x_0)$        $V(X) = \sum_{i=0}^{n-1} x_i 2^i$
- Rango =  $[0, 2^{n-1}]$     Resolución = 1
- $A-B = A+[(2^{n-1}-B)+1]-2^n = S + C_{n-1} \cdot 2^n - 2^n$
- Desbordamiento (OVF) con CY (biestable de acarreo)
  - SUMA:  $C_{n-1} = 1$  y  $S/R = 0$  (CY=1, carry =1)
  - RESTA:  $C_{n-1} = 0$  y  $S/R = 1$  (CY=1, borrow =1)



## ENTEROS EN COMPLEMENTO A 2 (1)

- $\text{Rep}(X) = (x_{n-1} \ x_{n-2} \ \dots \ x_1 \ x_0)$ 
  - $x_{n-1} = 0: X \geq 0$ , Igual que binario puro
  - $x_{n-1} = 1: X < 0$ ,  $\text{Rep}(X) = 2^n - |X|$
  - $\text{Rep}(X) + \text{Rep}(-X) = 2^n$

$$V(X) = -x_{n-1} 2^{n-1} + \sum_{i=0}^{n-2} x_i 2^i$$

Ejemplo:  $n = 6$ ,  $A = 7$ ,  $B = 101110$

$$A = 000111 \quad -A = 1000000 - 000111 = 111001 = 111000 + 1$$

(-A se representa invirtiendo los bits de A y sumando 1)

$$|B| = 1000000 - 101110 = 010010 = 18, \quad B = -18$$

$$\text{Valor máximo} = 011111 = 2^5 - 1 = 31$$

$$\text{Valor mínimo} = 100000 = -2^5 = -32$$

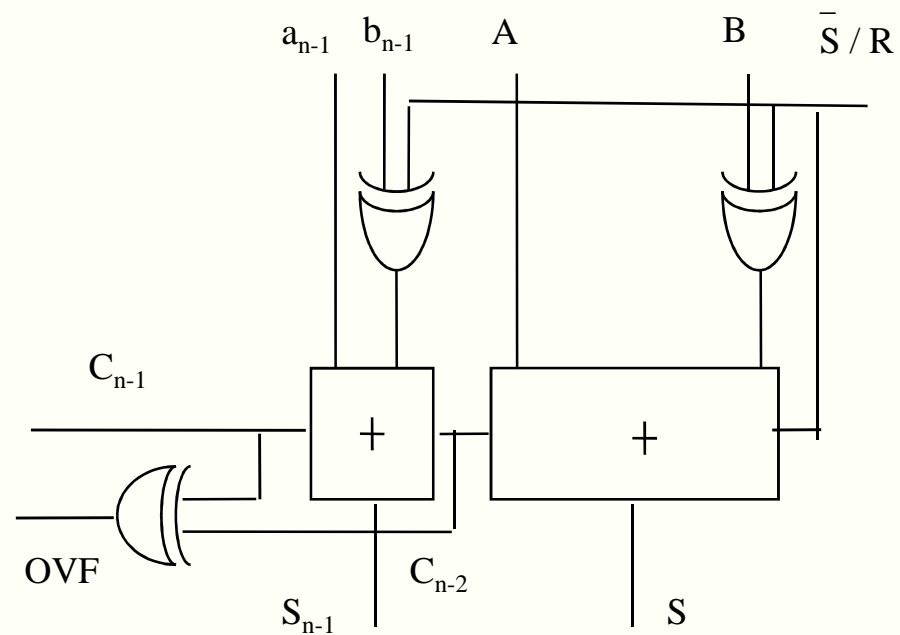
- Rango =  $[-2^{n-1}, -1] \cup [0, 2^{n-1}-1]$  Resolución = 1
  - Rango de representación asimétrico
  - Representación del cero única



## ENTEROS EN COMPLEMENTO A 2 (2)

- Suma y Resta:  $A - B = A + (-B) = A + [2^n - 1 - \text{Rep}(B)] + 1$
- Análisis de OVF:

A	B	A+B	$C_{n-1}$	OVF
a	b	a+b	0	$S_{n-1}=1 \ C_{n-2}=1$
$2^n - a$	$2^n - b$	$2^n + 2^n - (a+b)$	1	$S_{n-1}=0 \ C_{n-2}=0$
a(>b)	$2^n - b$	$2^n + (a-b)$	1	NO $C_{n-2}=1$
a(<b)	$2^n - b$	$2^n - (b-a)$	0	NO $C_{n-2}=0$



## ENTEROS EN COMPLEMENTO A 1

- $\text{Rep}(X) = (x_{n-1} \ x_{n-2} \ \dots \ x_1 \ x_0)$ 
  - $x_{n-1} = 0: X \geq 0$ , Igual que binario puro
  - $x_{n-1} = 1: X \leq 0$ ,  $\text{Rep}(X) = 2^n - 1 - |X|$
  - $\text{Rep}(X) + \text{Rep}(-X) = 2^n - 1$

$$V(X) = x_{n-1}(1 - 2^{n-1}) + \sum_{i=0}^{n-2} x_i 2^i$$

Ejemplo:  $n = 6$ ,  $A = 7$ ,  $B = 101110$

$$A = 000111 \quad -A = 111111 - 000111 = 111000$$

(-A se representa invirtiendo los bits de A)

$$|B| = 111111 - 101110 = 010001 = 17, \quad B = -17$$

$$\text{Valor máximo} = 011111 = 2^5 - 1 = 31$$

$$\text{Valor mínimo} = 100000 = -011111 = -(2^5 - 1) = -32$$

- Rango =  $[-(2^{n-1}-1), -1] \cup [0, 2^{n-1}-1]$  Resolución = 1
  - Rango de representación simétrico
  - Doble representación del cero: 000...000 y 111...111

## ENTEROS EN SIGNO-MAGNITUD

- $\text{Rep}(X) = (x_{n-1} \ x_{n-2} \ \dots \ x_1 \ x_0)$ 
  - $x_{n-1}$  = bit de signo
  - $x_{n-1} = 0: X \geq 0$  y  $x_{n-1} = 1: X \leq 0$

$$V(X) = (1 - 2 \cdot x_{n-1}) \cdot \sum_{i=0}^{n-2} x_i 2^i$$

Ejemplo:  $n = 6$ ,  $A = 7$ ,  $B = 101110$

$A = 000111$      $-A = 100111$      $B = -14$      $-B = 001110$

- Rango y resolución igual que en complemento a 1
- Suma  $A+B=(-1)^S \times M$ , siendo  $A=(-1)^{S_A} \times M_A$  y  $B=(-1)^{S_B} \times M_B$  y utilizando un sumador en binario sin signo:
  1. Si  $S_A=S_B$  ir a 5
  2. Si  $M_A < M_B$  ir a 4
  3.  $S=S_A$ ,  $M=M_A-M_B$ , FIN
  4.  $S=S_B$ ,  $M=M_B-M_A$ , FIN
  5.  $S=S_A=S_B$ ,  $M=M_A+M_B$ , si  $CY=1$  hay OVF, FIN

## ENTEROS EN EXCESO A "M"

- $\text{Rep}(X) = (x_{n-1} x_{n-2} \dots x_1 x_0)$ 
  - $\text{Rep}(X) = V(X) + M$
  - Normalmente  $M=2^{n-1}$  ó  $M=2^{n-1}-1$  (el usado en el estándar IEEE)

Ejemplo:  $n = 6$ ,  $M=32$ ,  $A = 7$ ,  $B = 001110$

$$A = 7+32 = 39 = 100111 \quad B = 001110 - 32 = 14-32 = -18$$

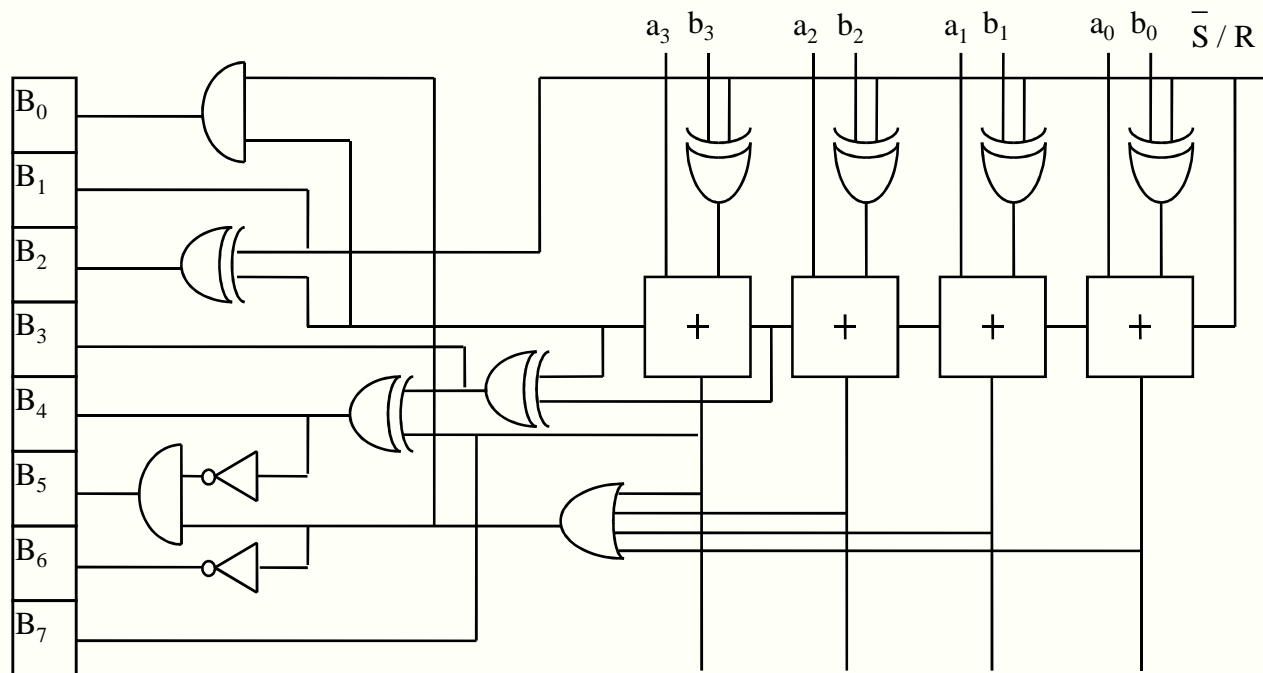
$$\text{Valor máximo} = 111111 = 63 - 32 = 31$$

$$\text{Valor mínimo} = 000000 = 0 - 32 = -32$$

- Rango =  $[-M, -1] \cup [0, 2^n-1-M]$  Resolución = 1

## BIESTABLES DE ESTADO (1)

- El sumador-restador para operandos de 4 bits incorpora una lógica para generar 8 señales que se almacenan en los biestables de estado, que podrán utilizarse como condiciones en las instrucciones de salto. Deducir el significado de cada biestable, al realizar la operación  $A-B$ , considerando que:
  - Los operandos están en aritmética sin signo
  - Los operandos están en aritmética de complemento a dos



## BIESTABLES DE ESTADO (2)

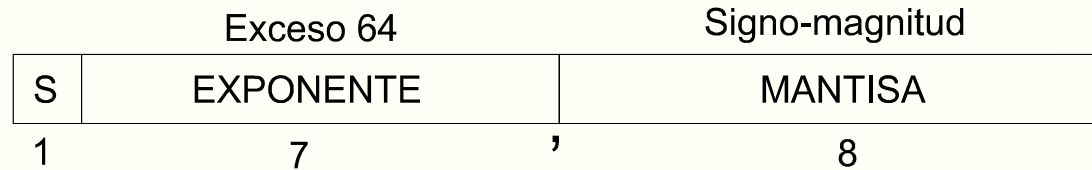
- BINARIO SIN SIGNO
  - $B0=1 \rightarrow \text{Carry}=1$  y  $S=1 \rightarrow$  No hay OVF y positivo  $\rightarrow A>B$
  - $B1=1 \rightarrow$  La operación es Resta
  - $B2=1 \rightarrow$  Resta y  $C=1 \rightarrow \text{Borrow}=1 \rightarrow A<B$
  - $B3, B4, B5$  y  $B7$  no tienen sentido
  - $B6=1 \rightarrow Z=1 \rightarrow A=B$
  
- COMPLEMENTO A 2
  - $B0$  no indica nada
  - $B1 = N =$  flag de resta,  $B2 = C =$  flag de acarreo,  $B3 = V =$  flag de OVF
  - $B4=1 \rightarrow \text{OVF} \oplus \text{Signo} = 1 \rightarrow A<B$ 
    - Si no hay OVF el signo es negativo  $\rightarrow$  Resultado real negativo
    - Si hay OVF el signo es positivo  $\rightarrow$  Resultado real negativo
  - $B6 = Z =$  flag de cero,  $B7 = S =$  flag de signo

## REPRESENTACIÓN EN COMA FLOTANTE (1)

- $V(X) = M \cdot r^E$  (notación científica)
  - $M$  = mantisa o fracción ( $p$  bits)
  - $r$  = base o radix
  - $E$  = exponente o característica ( $q$  bits)
  
- $\text{Rep}(X) = (e_{q-1} e_{q-2} \dots e_1 e_0 m_{p-1} m_{p-2} \dots m_1 m_0)$
  
- CARACTERÍSTICAS:
  - Normalmente  $r = 2^k$  ( $r = 2, 8, 16$ )
  - Mantisa: coma fija con signo y base  $r$
  - Exponente: Entero y base 2

REPRESENTACIÓN EN COMA FLOTANTE (2)

- Rango Exponente = [-64, 63]



- Rango Mantisa:

,00000000 = 0

,00000001 =  $2^{-8}$

.....

,11111111 =  $1-2^{-8}$

- Rango =  $\pm [2^{-8} \cdot 2^{-64}, (1-2^{-8}) \cdot 2^{63}] \cup 0$

- Resolución =  $2^{-8} \cdot 2^E$

- $A = H'C63C = 1 \ 1000110 \ ,00111100$

$A = - ,00111100 \cdot 2^6 = -15_{(10)}$

$A = - ,01111000 \cdot 2^5 = -15_{(10)} \rightarrow A = H' \ C578$

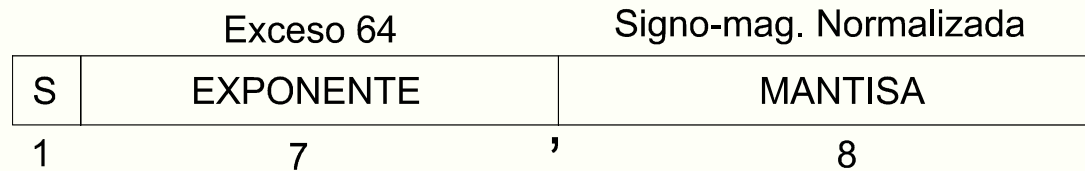
$A = - ,11110000 \cdot 2^4 = -15_{(10)} \rightarrow A = H' \ C4F0$



## REPRESENTACIÓN EN COMA FLOTANTE (3)

- NORMALIZACIÓN

Un número en coma flotante está con su mantisa normalizada si al desplazar la mantisa un dígito a la izquierda y decrementar el exponente en 1 cambia el valor del número



- Rango Mantisa:

,10000000 =  $2^{-1}$

.....

,11111111 =  $1-2^{-8}$

- Rango =  $\pm [2^{-1} \cdot 2^{-64}, (1-2^{-8}) \cdot 2^{63}]$

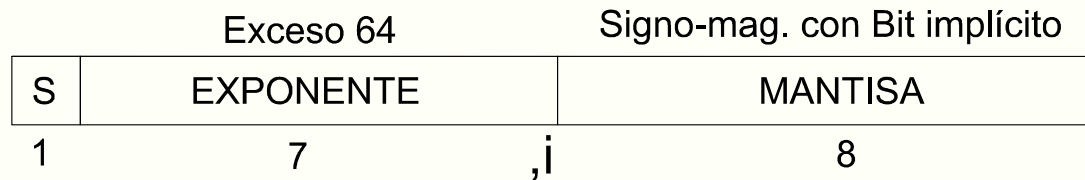
- Problemas de la normalización:

- Resultados de operaciones no normalizados
- El cero no es representable

REPRESENTACIÓN EN COMA FLOTANTE (4)

- BIT IMPLÍCITO

A un número en coma flotante con  $r=2$  y su mantisa en signo magnitud y normalizada, puede dejarse el bit más significativo como implícito ya que tiene que ser un 1



- Rango Mantisa:

,1 00000000 =  $2^{-1}$

.....

.....

,1 11111111 =  $1-2^{-9}$

- Rango =  $\pm [2^{-1} \cdot 2^{-64}, (1-2^{-9}) \cdot 2^{63}]$

## SUMA Y RESTA EN COMA FLOTANTE (1)

### SOLUCIÓN ANALÍTICA:

$$A = MA \cdot r^{EA} \quad B = MB \cdot r^{EB}$$

$r = 2^k$ ; Las mantisas MA y MB normalizadas

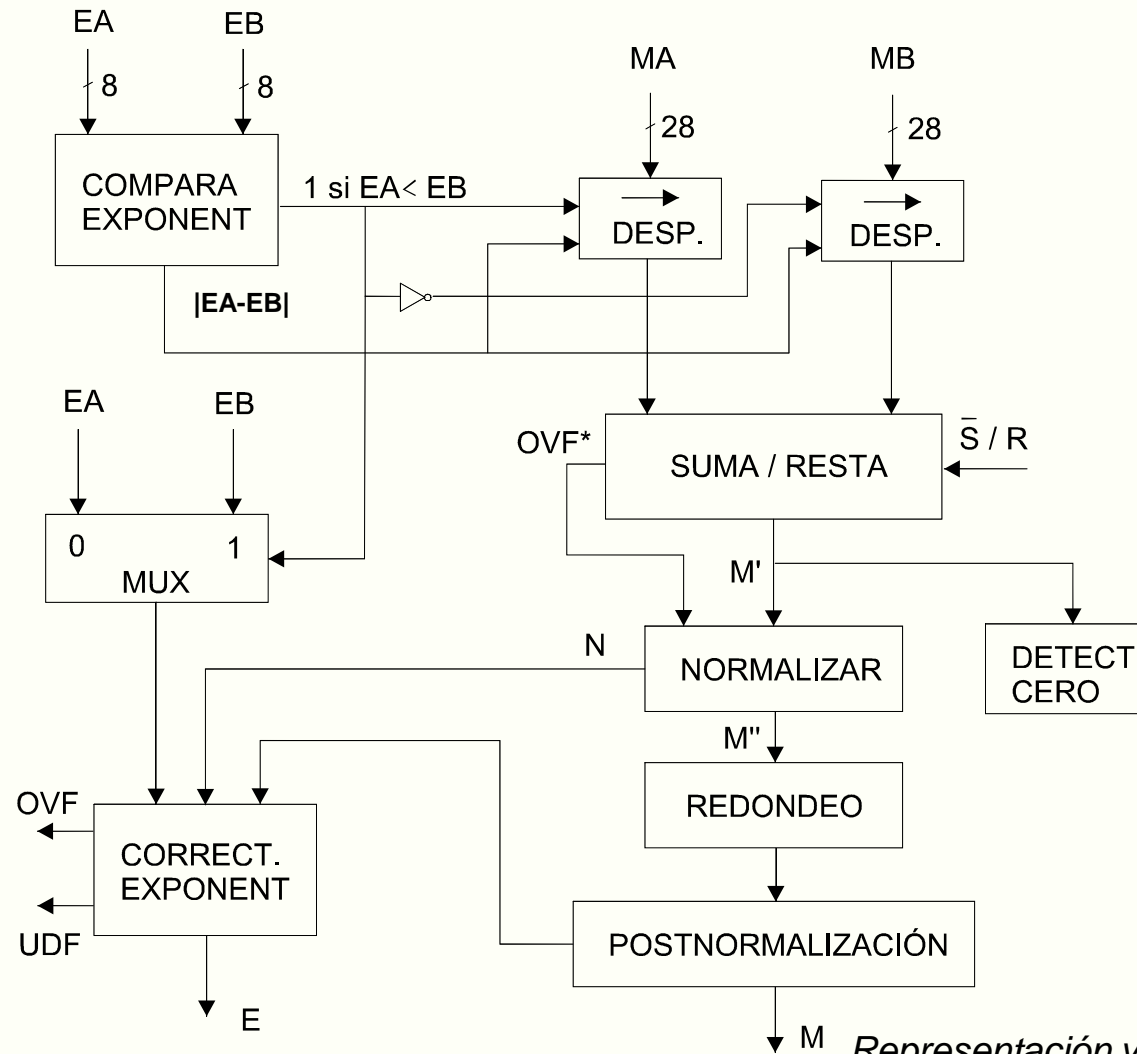
- $EA > EB$  siendo  $d = EA - EB$   
 $A \pm B = (MA \pm MB \cdot r^{-d}) r^{EA}$
- $EA < EB$  siendo  $d = EB - EA$   
 $A \pm B = (MA \cdot r^{-d} \pm MB) r^{EB}$

### PASOS A SEGUIR:

1. Comparar exponentes
2. Desplazar mantisa de exponente menor
3. Sumar / Restar mantisas
4. Detectar resultado cero
5. Normalizar (si redondeo postnormalizar)
6. Corregir exponente
7. Detectar desbordamiento

## SUMA Y RESTA EN COMA FLOTANTE (2)

Esquema del sumador/restador en coma flotante:



## SUMA Y RESTA EN COMA FLOTANTE (3)

1. Comparar exponentes
  - Identificar mantisa a desplazar
  - Determinar el número de desplazamientos =  $|EA-EB|$
  - Utiliza un restador (puede haber OVF en esta resta)
2. Desplazar mantisa de exponente menor
  - Desplaza  $|EA-EB|$  dígitos
  - Desplazamientos aritméticos
3. Sumar / Restar mantisas
  - Depende de la representación de las mantisas
  - Depende del operador que se utilice
  - Puede haber OVF\*  $\rightarrow$  hay que normalizar
  - La resta no es conmutativa
4. Detectar resultado cero
  - Se detecta con el flag,  $Z=1$
  - Se devuelve la representación definida para el cero

## SUMA Y RESTA EN COMA FLOTANTE (4)

5. Normalización (mantisa signo y p bits de magnitud)
  - $OVF^*=1$ : desplaza dcha.  $M'$  y  $E \leftarrow E+1$
  - $OVF^*=0$ : desplaza izda.  $M'$  y  $E \leftarrow E+x$  ( $x=0,1,\dots,p-1$ )
  - $N=1,0,-1,\dots,-(p-1)$ =cantidad a sumar al exponente mayor
  - Si redondeo y postnormalización : desplaza dcha.  $M'$  y  $E \leftarrow E+1$
6. Corregir exponente
  - Seleccionar el exponente mayor
  - Sumar  $N$  (de la fase de normalización)
  - Sumar 1 si hay postnormalización tras el redondeo
7. Detectar desbordamiento
  - Si  $E >$  Exponente mayor, hay overflow (OVF)
  - Si  $E <$  Exponente menor, hay underflow (UDF)

## REDONDEO

- $A = \pm M \times 2^E$  donde  $M$  está representada por 6 bits. Se ha obtenido un resultado de 10 bits  $M = ,100100\ 1011$  que ha de ajustarse a 6 bits mediante técnicas de redondeo:
  - **Truncamiento:** Suprimir los bits sobrantes.  $M = ,100100$ . Error absoluto  $\varepsilon_a < 2^{-6}$  siempre por defecto
  - **Forzado a 1:** Truncamiento dejando siempre a 1 el bit menos significativo.  $M = 100101$ . Mismo error absoluto que en truncamiento, pero por defecto y por exceso
  - **Redondeo al más próximo:** Ajustar al extremo  $M_{i-1} = ,100100$  ó al  $M_i = ,100101$  más próximo sumando la mitad del intervalo,  $\frac{1}{2}(M_i - M_{i-1}) = 000000\ 1000$ .  $M = ,100101$ . Error absoluto  $\varepsilon_a \leq 2^{-7}$  por defecto y por exceso
  - **Redondeos a cero, a  $+\infty$  y a  $-\infty$ :** Ajustar al extremo  $M_i$  ó  $M_{i-1}$  que corresponda en la dirección ( $M$  a 0), ( $M$  a  $+\infty$ ) y ( $M$  a  $-\infty$ ), respectivamente

## DÍGITOS DE GUARDA Y BIT RETENEDOR (1)

- **Dígitos de guarda:** dígitos añadidos a la mantisa para obtener la precisión máxima. En el caso de mantisa en signo-magnitud se necesitarían dos bits de guarda, uno para normalizar el resultado y otro para redondeo
- **Bit retenedor:** bit que se añade para propagar el borrow en la resta. Al realizar los desplazamientos en la mantisa de menor exponente en la operación suma/resta, el bit retenedor se pone a 1 en el momento que pase un 1 y permanece ese valor independientemente de los bits que pasen después
- **Ejemplo:** mantisa normalizada en signo magnitud (1 bit de signo y 6 de magnitud) y exponente de 5 bits en exceso a 16.  
$$A = ,100001 \times 2^7 \quad B = ,100101 \times 2^3,$$
realizar A-B usando todos los bits necesarios para absorber todos los desplazamientos, con dos bits de guarda y con los dos bits de guarda más bit retenedor



## DÍGITOS DE GUARDA Y BIT RETENEDOR (2)

- Resultado con 4 bits adicionales:

$$\begin{aligned}
 A &= ,100001\ 0000 \times 2^7 \\
 B &= ,000010\ 0101 \times 2^7 \\
 A-B &= ,011110\ 1011 \times 2^7 \\
 \text{Nor.} &= ,111101\ 011 \times 2^6 \\
 \text{Red.} &+ \quad \quad \quad 1 \\
 A-B &= ,111101 \quad \quad \times 2^6 = D'61
 \end{aligned}$$

- Resultado con 2 bits de guarda:

$$\begin{aligned}
 A &= ,100001\ 00 \times 2^7 \\
 B &= ,000010\ 01 \times 2^7 \\
 A-B &= ,011110\ 11 \times 2^7 \\
 \text{Nor.} &= ,111101\ 1 \times 2^6 \\
 \text{Red.} &+ \quad \quad \quad 1 \\
 A-B &= ,111110 \quad \quad \times 2^6 = D'62
 \end{aligned}$$

## DÍGITOS DE GUARDA Y BIT RETENEDOR (3)

- Resultado con 2 bits de guarda y bit retenedor:

$$A = ,100001 \ 00 \ 0 \times 2^7$$

$$B = ,000010 \ 01 \ 1 \times 2^7$$

$$A-B = ,011110 \ 10 \ 1 \times 2^7$$

$$\text{Nor.} \quad ,111101 \ 01 \quad \times 2^6$$

$$\text{Red.} \quad + \quad \quad \quad 1$$

$$A-B = ,111101 \quad \quad \times 2^6 = D'61$$

- Resultado exacto y errores:

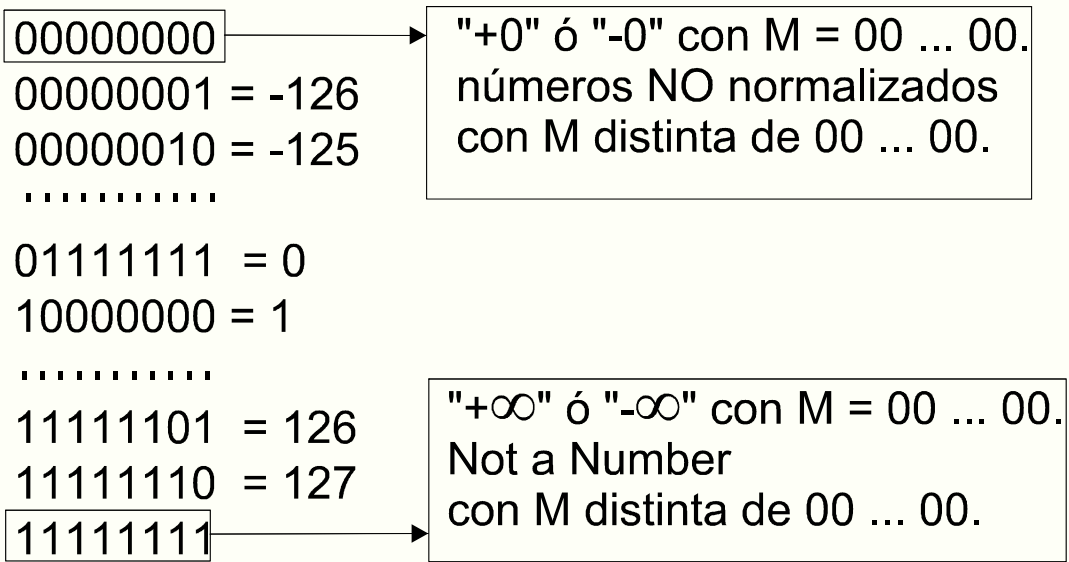
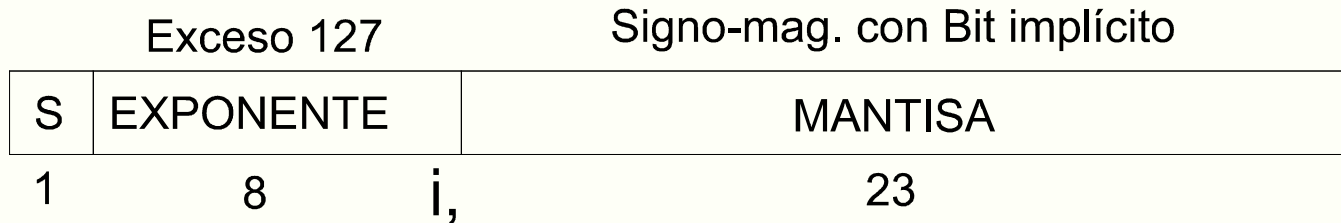
$$A = ,100001 \times 2^7 = D'66 \quad B = ,100101 \times 2^6 = D'4,625$$

$$A-B = D'61,375$$

- Error con 2 bits de guarda =  $|61,375-62|=0,625$
- Error con bit retenedor =  $|61,375-61|=0,375$

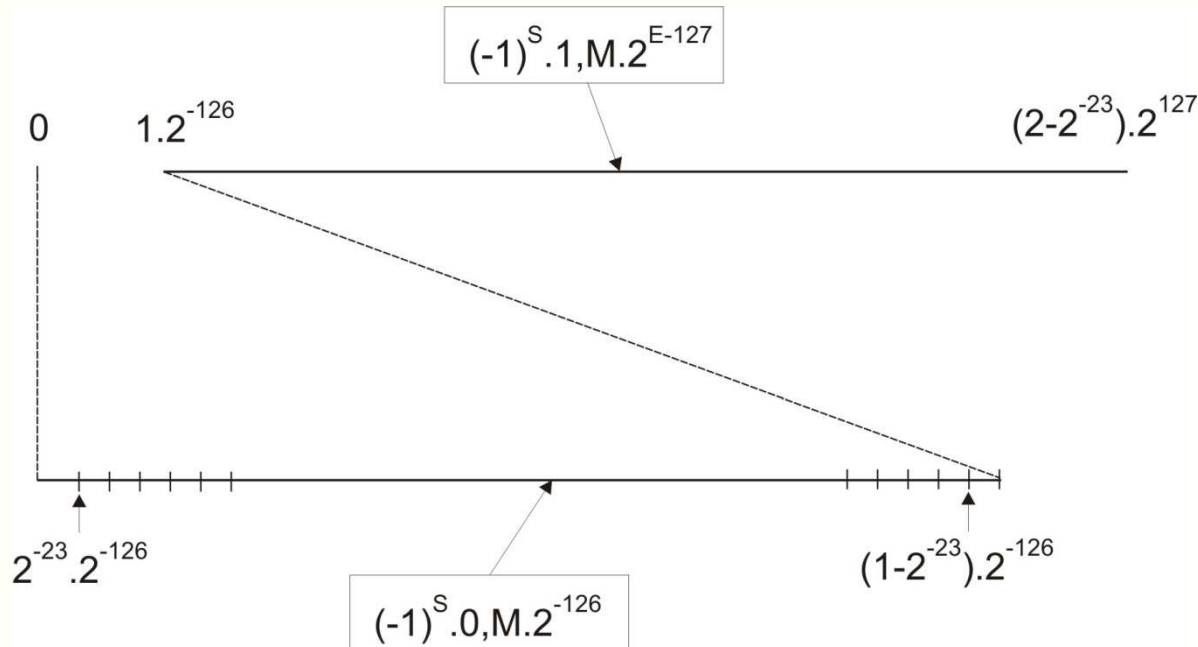
ESTÁNDAR IEEE 754 DE COMA FLOTANTE (1)

- SIMPLE PRECISIÓN



## ESTÁNDAR IEEE 754 DE COMA FLOTANTE (2)

- RANGO DE REPRESENTACIÓN:



- PRECISIÓN:

- 3 Bits adicionales (2 de guarda y 1 retenedor)
- Redondeos al más próximo, a  $+\infty$ , a  $-\infty$  y truncamiento