ARQUITECTURAS AVANZADAS DE PROCESADORES

UNIDAD IV. ARITMÉTICA PARA COMPUTADORES

ARITMÉTICA PARA COMPUTADORES

- Objetivo: Estudio de la instrucciones aritméticas de punto fijo y punto flotante del procesador MIPS. Diseño de la Unidad aritmético/lógica del procesador MIPS

- Hay muchas instrucciones aritméticas que no se tratará su estudio por haber sido estudiadas en la asignatura de Arquitectura de Computadores de 2º del Grado de Ingeniero en Informática

- Bibliografía:
  - Estructura y diseño de computadores – Patterson y Hennessy – Cuarta Edición
  - Estructura y diseño de computadores – Patterson y Hennessy – Tercera Edición, Volumen 1

ARITMÉTICA PARA COMPUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

- La ALU de 32 bit Se diseña a partir de la de 1 bit
- Elementos básicos:

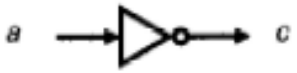
1. Puerta and ($c = a \cdot b$)

a	b	$c = a \cdot b$
0	0	0
0	1	0
1	0	0
1	1	1

2. Puerta or ($c = a + b$)

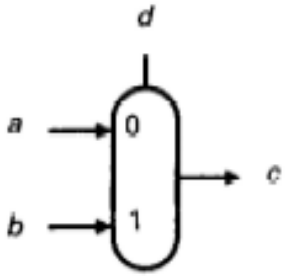
a	b	$c = a + b$
0	0	0
0	1	1
1	0	1
1	1	1

3. Inversor ($c = \bar{a}$)



a	$c = \bar{a}$
0	1
1	0

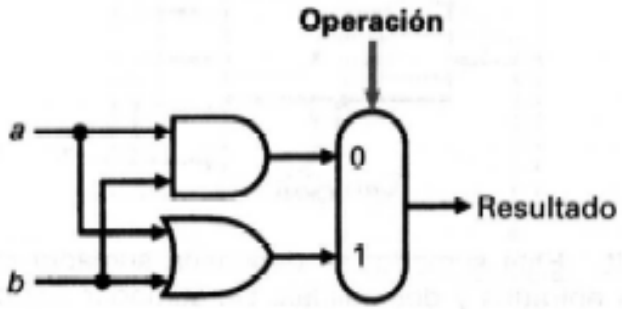
4. Multiplexor
(si $d = 0$ $c = a$;
sino $c = b$)



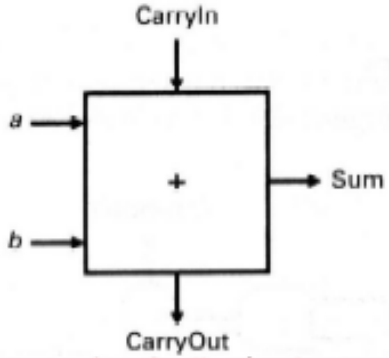
d	c
0	a
1	b

ARITMÉTICA PARA COMPUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

- Bloque lógico AND y OR



- Bloque aritmético (sumador completo)

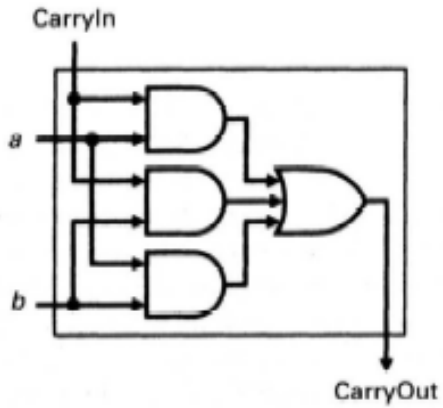


- Tabla del sumador

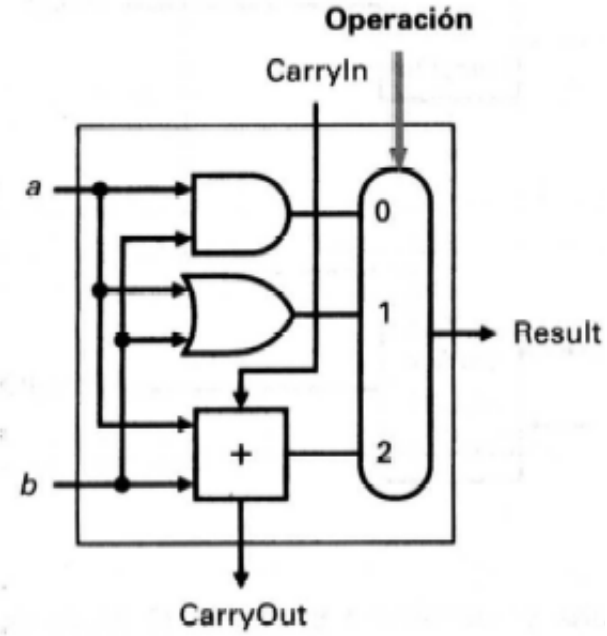
Entradas			Salidas		Comentarios
a	*b*	CarryIn	CarryOut	Sum	
0	0	0	0	0	$0 + 0 + 0 = 00_{dos}$
0	0	1	0	1	$0 + 0 + 1 = 01_{dos}$
0	1	0	0	1	$0 + 1 + 0 = 01_{dos}$
0	1	1	1	0	$0 + 1 + 1 = 10_{dos}$
1	0	0	0	1	$1 + 0 + 0 = 01_{dos}$
1	0	1	1	0	$1 + 0 + 1 = 10_{dos}$
1	1	0	1	0	$1 + 1 + 0 = 10_{dos}$
1	1	1	1	1	$1 + 1 + 1 = 11_{dos}$

ARITMÉTICA PARA COMPUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

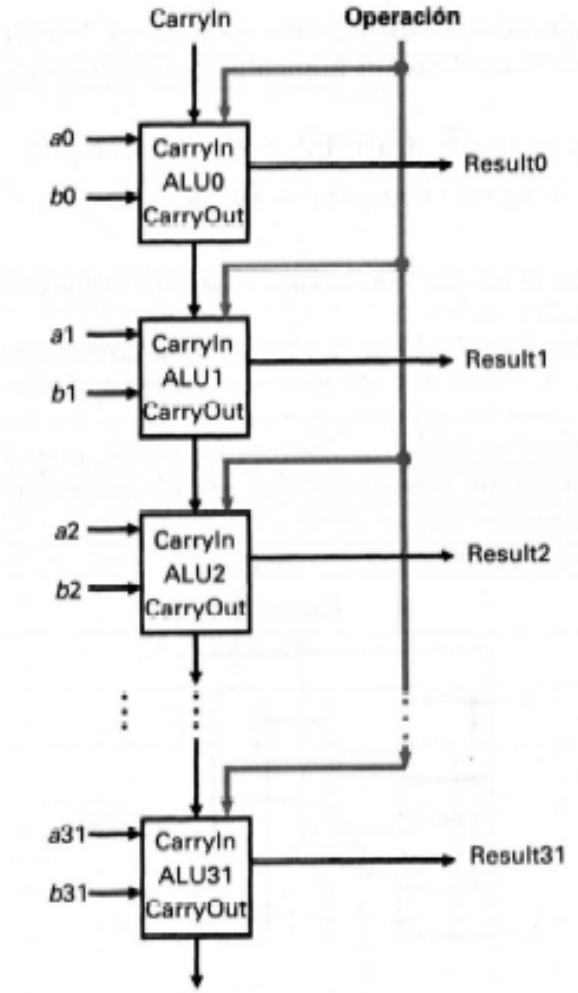
- Bloque sumador



- ALU completa de 1 bit (operaciones and, or y suma)



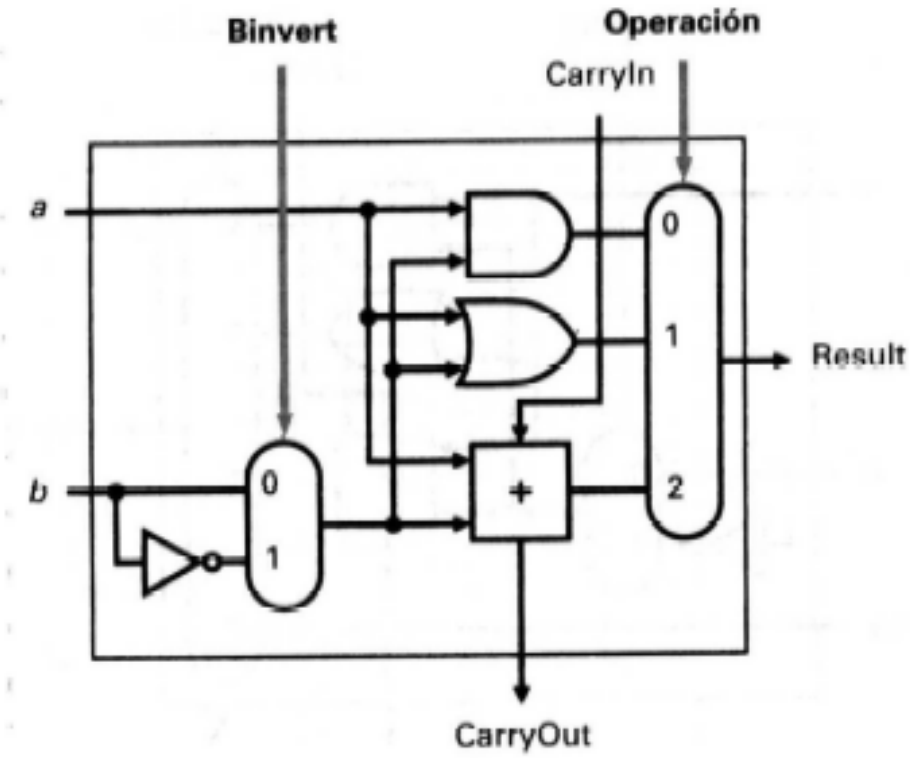
- ALU de 32 bits



ARITMÉTICA PARA COMPUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

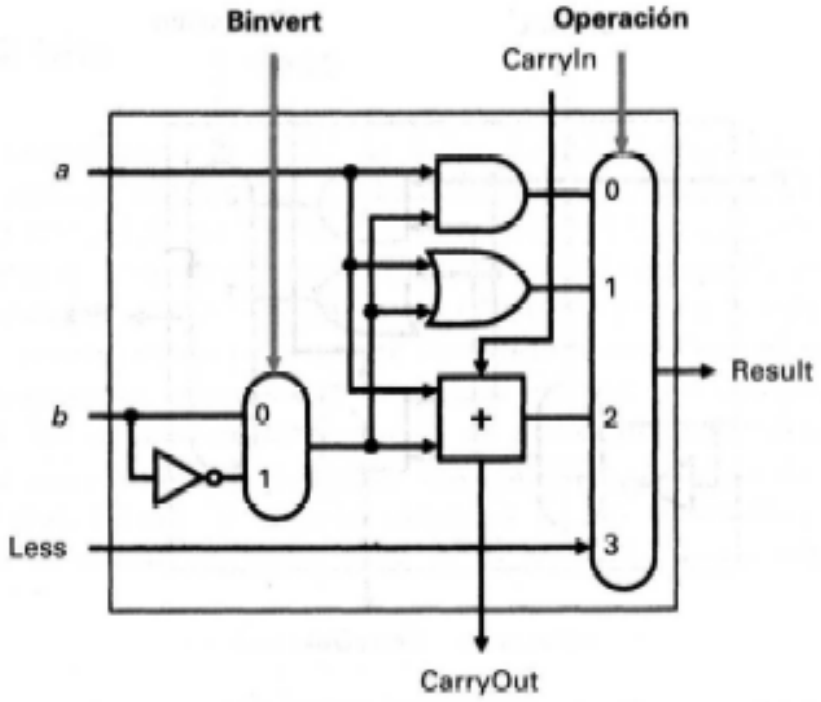
- ALU de 1 bit con las operaciones:

  A AND b ; a OR b ; a + b ; a + b´



ARITMÉTICA PARA COMPUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

- Operación inicializar sobre menor que (set-on-less-than)
- Hay que expander el multiplexor e introducir una entrada denominada Less

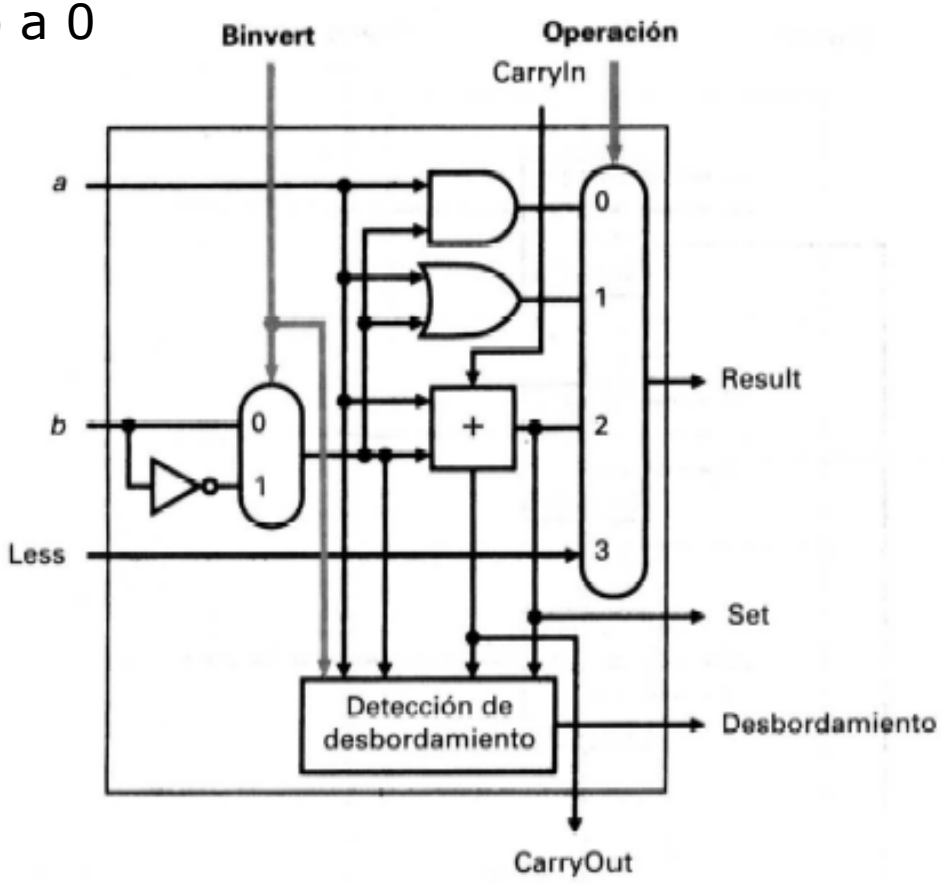


ARITMÉTICA PARA COM PUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

- La operación que se debe realizar es:

Si (Rs – Rt) < 0 => ponemos el lsb a 1

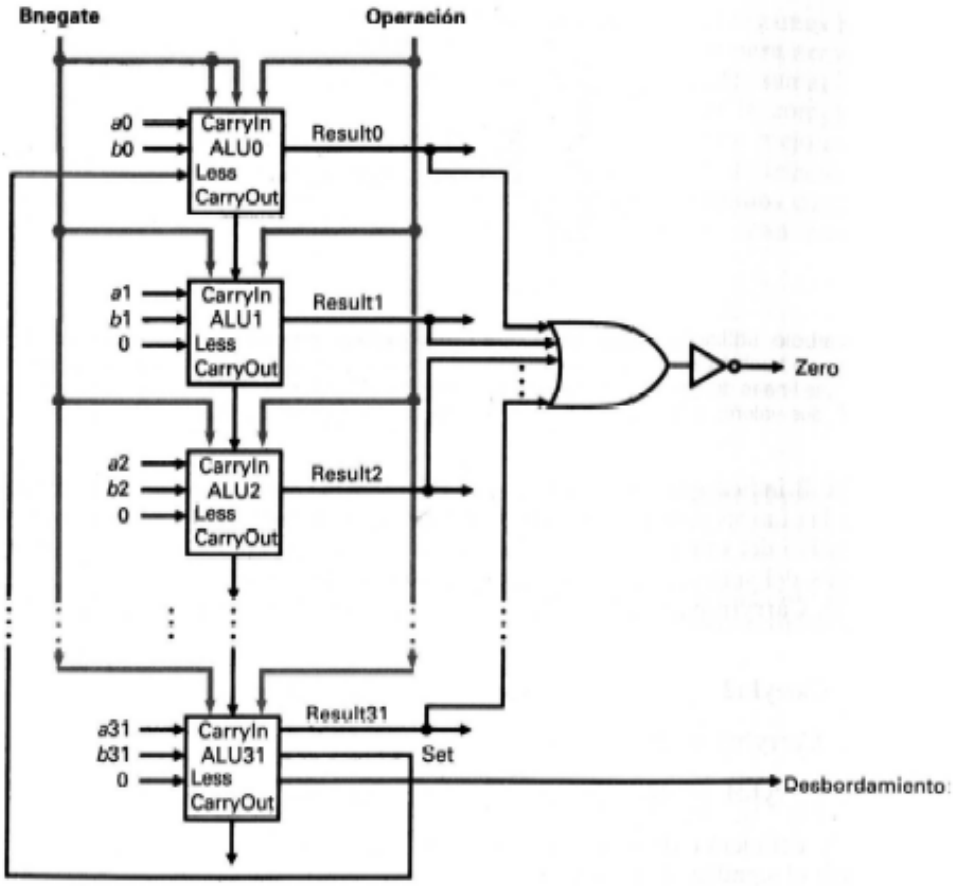
Si (Rs – Rt) ≥ 0 => ponemos el lsb a 0



ARITMÉTICA PARA COM PUTADORES

CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

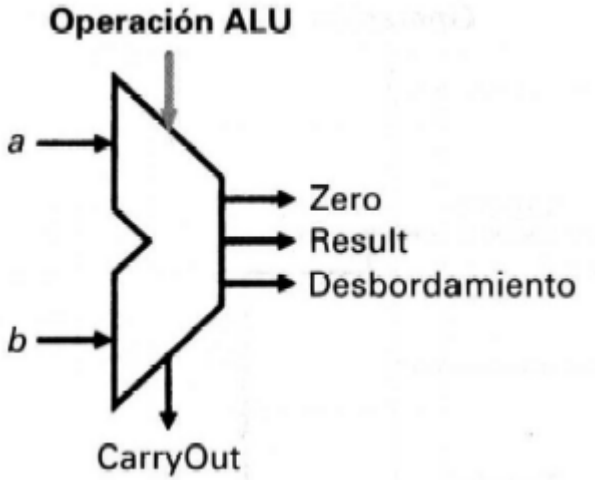
- Conectamos el msb (bit de signo) de la salida del sumador al lsb
- Aquí se incluye el detector de desbordamiento para operaciones suma y resta
- Se incluye la función Z (bit de status Z) para los saltos condicionados
- La instrucción set-on-less-than y el desbordamiento está implementado para números con signo
- ALU completa de 32 bits



ARITMÉTICA PARA COMPUTADORES
CONSTRUCCIÓN DE UNA UNIDAD ARITMÉTICO LÓGICA

- Bloque ALU y tabla de operaciones:

Líneas de control de la ALU	Función
000	And
001	Or
010	Sumar
110	Restar
111	Inicializar sobre menor que



- Problema en la propagación del acarreo cuando el número de bits aumenta (caso del MIPS 32 bits). Lentitud a la hora de realizar la suma, pues la suma no es válida hasta que no se realice la propagación del último acarreo.

- **TAREA: ¿Cómo se puede resolver este problema?**

ARITMÉTICA PARA COMPUTADORES

OPERACIONES LÓGICAS Y ARITMÉTICAS

- Resumen de las instrucciones aritméticas lógicas que se han tratado

Categoría	Instrucción	Ejemplo	Significado	Comentarios
Aritmética	sumar	add $1,$2,$3	$1 = $2 + $3	3 operandos; excepción posible
	restar	sub $1,$2,$3	$1 = $2 - $3	3 operandos; excepción posible
	sumar inmediato	addi $1,$2,100	$1 = $2 + 100	+ constante; excepción posible
	sumar sin signo	addu $1,$2,$3	$1 = $2 + $3	3 operandos; no excepciones
	restar sin signo	subu $1,$2,$3	$1 = $2 - $3	3 operandos; no excepciones
	sumar inmediato sin signo	addiu $1,$2,100	$1 = $2 + 100	+ constante; no excepciones
	transferir desde reg. coprocesador	mfc0 $1,$epc	$1 = $epc	Usado para conseguir excepción PC
Lógicas	and	and $1,$2,$3	$1 = $2 & $3	3 reg. operandos; AND Logica
	or	or $1,$2,$3	$1 = $2|$3	3 reg. operandos; OR Logica
	and inmediato	and $1,$2,100	$1 = $2&100	reg. Logica AND, constante
	or inmediato	or $1,$2,$100	$1 = $2|100	reg. Logica OR, constante
	desplazamiento lógico a la derecha	sll $1,$2,$10	$1 = $2 << 10	Desplaza izquierda por constante
	desplazamiento lógico a la izquierda	srl $1,$2,$10	$1 = $2 >> 10	Desplaza derecha por constante

ARITMÉTICA PARA COMPUTADORES
OPERACIONES LÓGICAS Y ARITMÉTICAS

- Resumen de las instrucciones aritméticas lógicas que se han tratado

Salto condicional	inicializa sobre menor que	slt $1,$2,$3	si ($2 < $3)$1 = 1; si no $1 = 0	Compara menor que; comp. a 2
	inicializa menor que inmediato	slti $1,$2,100	si ($2 < 100)$1 = 1; si no $1 = 0	Compara constante menor que; comp. a 2
	inicializa menor que sin signo	**sltu $1,$2,$3**	**si ($2 < $3) $1 = 1, si no $1 = 0**	**Compara menor que; número natural**
	inicializa menor que inmediato sin signo	**sltiu $1,$2,100**	**si ($2 < 100)$1 = 1; si no $1 = 0**	**Compara < constante, número natural**

ARITMÉTICA PARA COMPUTADORES

EXCEPCIONES EN MIPS

- Cuando se produce una excepción, se resuelve mediante una interrupción.
- El MIPS incluye un contador de excepción EPC para almacenar la dirección de la instrucción que ha causado la excepción.
- Hay una instrucción *mfc0* que sirve para mover el registro EPC a un registro de propósito general para que el MIPS tenga la opción de volver a la instrucción que produjo la excepción con una instrucción de salto a través de registro
- Se deja al compilador y al sistema operativo la forma de tratar las excepciones de tipo aritmético.

ARITMÉTICA PARA COMPUTADORES

ARITMÉTICA DE MULTIMEDIA

- Un problema que surge en la aritmética en los procesadores de propósito general es la manipulación de datos de diferentes tamaños, como es el caso a la hora de trabajar los vectores de diferentes tamaños, lo que denominamos procesamiento vectorial tipo SIMD (single instruction, multiple data).
- Este problema se intentó resolver mediante las unidades MMX (SSE) que se incorporaron en la microarquitectura de los procesadores.
- Otro problema que surge es la **saturación**: es una solución para los desbordamientos producidos por operaciones aritméticas que consiste en dar el resultado al número mayor positivo o menor negativo en lugar del resultado correcto.

ARITMÉTICA PARA COMPUTADORES

MULTIPLICACIÓN EN MIPS

- Para que en la multiplicación no se produzca desbordamiento, se debe presentar el resultado de la multiplicación en dos registros
- MIPS proporciona dos registros para Hi y Lo para almacenar el producto de 64 bits
- Dos instrucciones para multiplicar: *mult* (multiplicación con signo) y *multu* (multiplicación sin signo)
- Se utiliza dos instrucciones *mflo* y *mfhi* para mover el resultado a los registros
- En general, se utiliza el algoritmo de suma y desplazamiento para la multiplicación de enteros, y de desplazamiento para multiplicaciones por potencias de 2.
- También hay versiones de MIPS que utilizan el algoritmo de booth para resolver el problema de la multiplicación con signo en número en representación en complemento a dos.

ARITMÉTICA PARA COMPUTADORES
DIVISIÓN EN MIPS

- La división con signo se realiza con el mismo hardware que el de la multiplicación cuando se trata de dividir números negativos en la representación en SM.
- El problema es cuando trabajamos en representación complemento a dos. No hay algoritmo que resuelva este problema: verificar signos y pasar a positivo, y después proceder a la división.
- En MIPS se utiliza el algoritmo de división en SM, pero hay versiones actuales que incorporan unidades de punto flotante que realizan la división directamente en punto flotante.
- Utiliza los mismo registros que en la multiplicación Hi (contiene el resto) y Lo (contiene el cociente) para almacenar el resultado, y los mismas instrucciones para mover el resultado de la división a registros (*mflo* y *mfhi*).

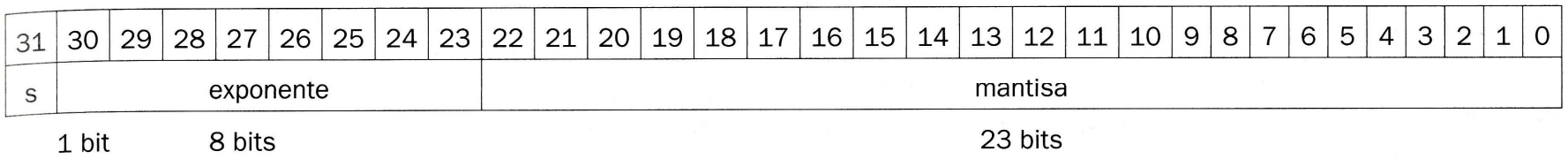
ARITMÉTICA PARA COMPUTADORES

ARITMÉTICA DE PUNTO FLOTANTE

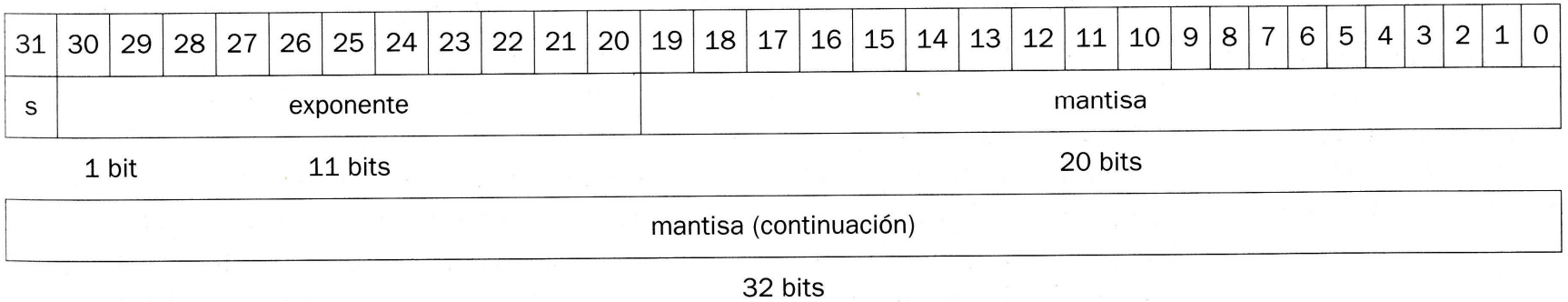
- Representación en punto flotante:

  (s) x M x 2^e donde ***s*** es el bit de signo, ***M*** es la mantisa normalizada y ***e*** es el exponente sesgado ($sesgo_{8\ bit}$ = 127; $sesgo_{11\ bit}$= 1023

  - Simple precisión



  - Doble precisión



ARITMÉTICA PARA COMPUTADORES

ARITMÉTICA DE PUNTO FLOTANTE

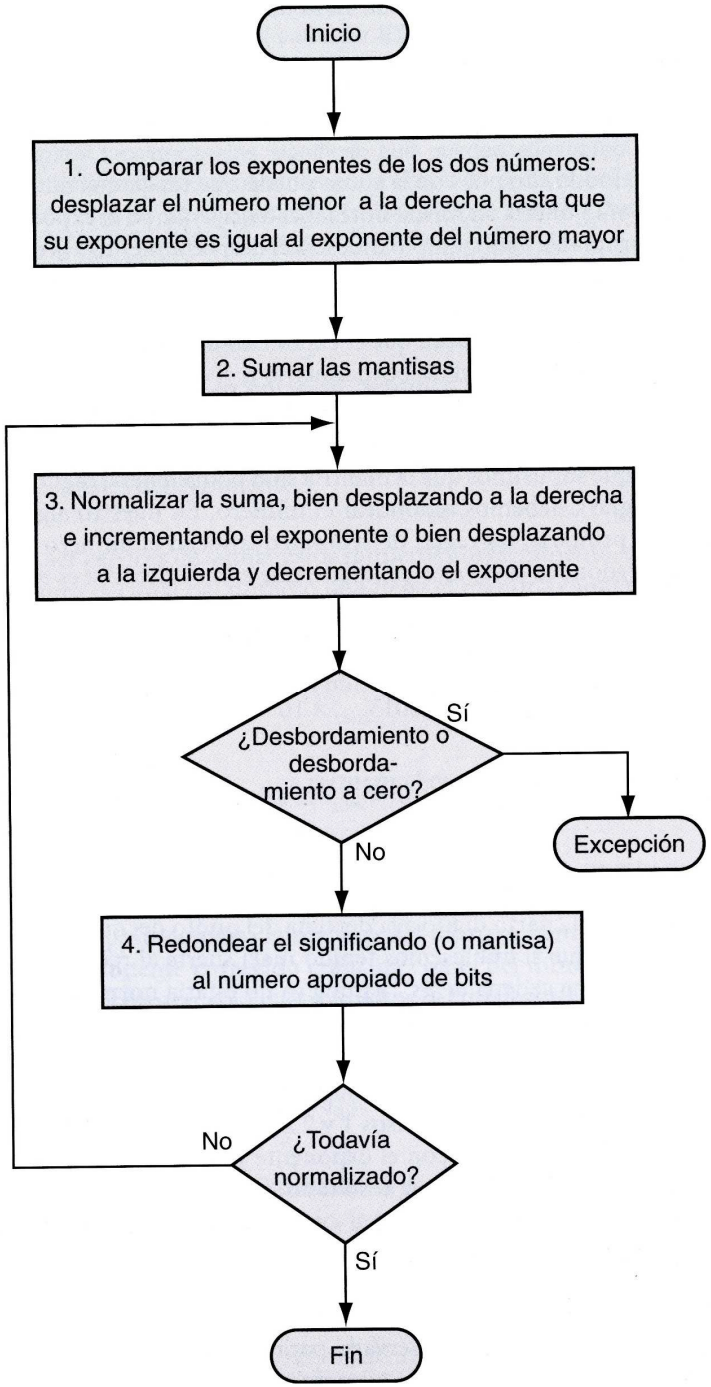
- Representación de números singulares:

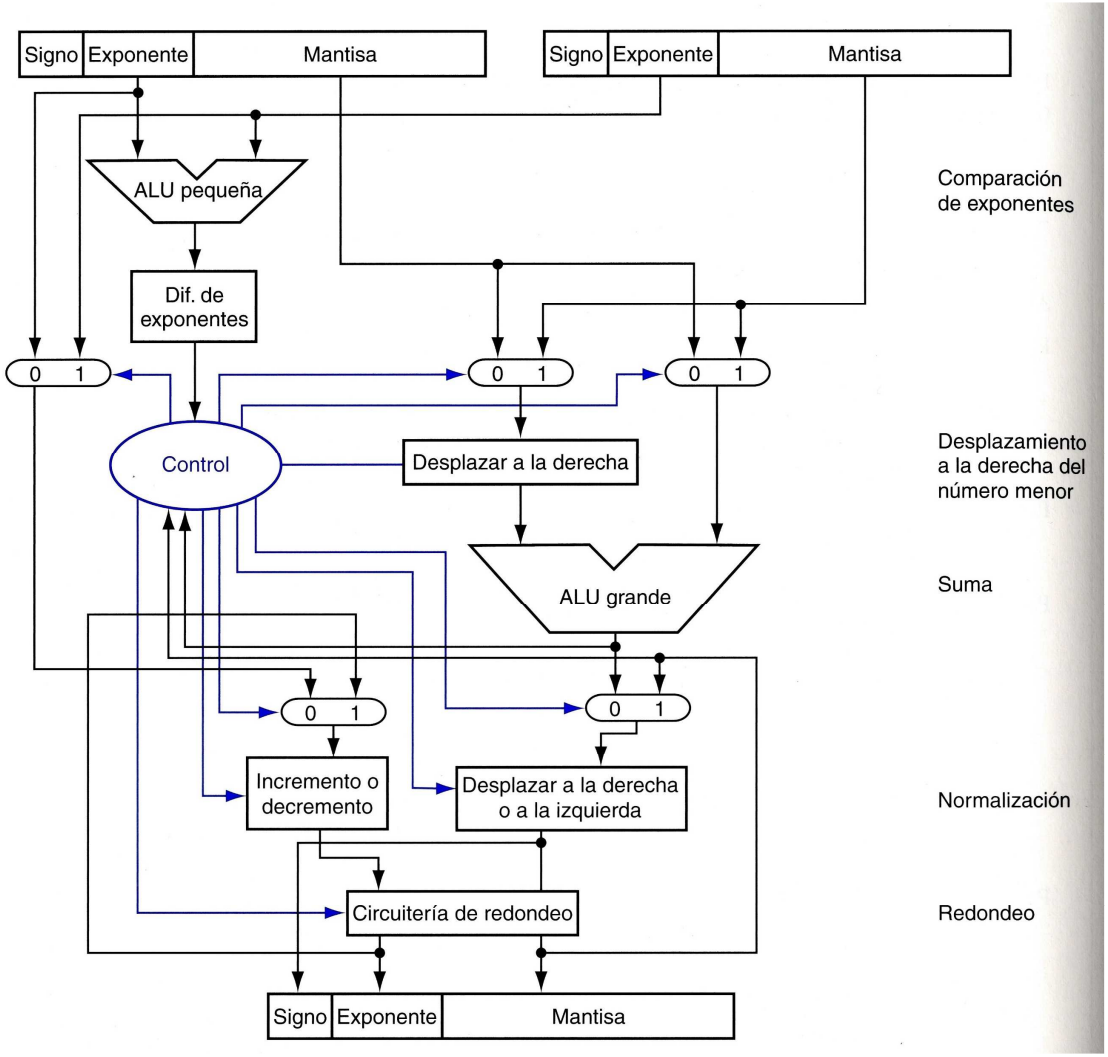
Precisión simple		Precisión doble		Objeto representado
Exponente	Mantisa	Exponente	Mantisa	
0	0	0	0	0
0	Distinto de cero	0	Distinto de cero	± número no normalizado
1–254	cualquiera	1–2046	cualquiera	± número en punto flotante
255	0	2047	0	± infinito
255	Distinto de cero	2047	Distinto de cero	NaN (*Not a Number*)

- Operaciones
  - Suma y resta
  - Multiplicación
  - División

ARITMÉTICA PARA COMPUTADORES

ARITMÉTICA DE PUNTO FLOTANTE





ARITMÉTICA PARA COMPUTADORES

ARITMÉTICA DE PUNTO FLOTANTE