



Ejercicios repaso 1° parcial

- 1) Utilizar el algoritmo de Booth para multiplicar 010011 (multiplicando) por 011011 (multiplicador).

El algoritmo de Booth permite hacer multiplicaciones en Complemento a 2.

Ej. $19 = 010011 \times 27 = 011011$

LEYES:

0 a 1	Reg. A + C2 (M) (multiplicando) y desplazo a la derecha
1 a 1	desplazo a la derecha
0 a 0	desplazo a la derecha
1 a 0	Reg. A + M (multiplicando) desplazo a la derecha
	si el resultado es negativo (1 adelante) antes de desplazar introduzco un 1 en el bit de mayor peso y luego desplazo (así mantengo el signo), si no, introduzco un 0

Acción	Reg A	Reg Q.	Q ₋₁ (bit de arrastre inicia en 0)	M	Contador
	000000	010011	0	011011 C2 = 100101	6
0 a 1 Reg. A + C2 (M)	<u>+100101</u> 100101	010011 010011			
→ Como el resultado es negativo mantengo el signo	110010	101001	1		5
1 a 1 → Como el resultado es negativo mantengo el signo	111001	010100	1		4
1 a 0 → Reg. A + C2 (M)	<u>+011011</u> 010100	010100	1		
→	001010	001010	0		3
0 a 0 →	000101	000101	0		2
0 a 1	<u>+100101</u>	000101			



Universidad Nacional de la Patagonia San Juan Bosco
Facultad de Ingeniería. Sede Puerto Madryn
ARQUITECTURA DE COMPUTADORAS

Reg. A + C2 (M)	101010	000101			
→ Como el resultado es negativo mantengo el signo	110101	000010	1		1
1 a 0 → Reg. A + M	$\frac{+100101}{010000}$	000010			
→	001000	000001			0

Resultado 001000000001 = 513

2) Realizar las siguientes divisiones (en magnitud y signo)

a) $100101101 \div 01010 = (-45) \div 10$

b) $001100100 \div 10111 = 100 \div (-7)$

Contador: Se carga con el número de bits del divisor y lleva la cuenta de los desplazamientos realizados.

Acumulador: Se carga con la parte más significativa del dividendo. Va almacenando los resultados.

Q: Se carga con la parte menos significativa del dividendo. Al finalizar el algoritmo queda almacenado el cociente.

D: Se carga con el divisor. (es el sustraendo de la resta).

Ej: 100 / 7

01100100 / 0111

ACCIÓN	REG. ACUMUL.	REG Q.	REG D.	CONTADOR
	<u>0</u> 110	0100	0111	4
← desplazo hasta que aparezca un 1 y se pueda hacer la resta	1100	1000		3
resto y	<u>0</u> 111 0101	1000 1001	Cuando se pueda hacer la resta entra un uno en Q ₀	
← desplazo	1011	0010		
resto y	<u>0</u> 111 0100	0010 0011	Cuando se pueda hacer la resta entra un uno en Q ₀	2
← desplazo	1000	0110		
resto y	<u>0</u> 111 0001	0110 0111		1
← desplazo	0010	1110		



Universidad Nacional de la Patagonia San Juan Bosco
Facultad de Ingeniería. Sede Puerto Madryn
ARQUITECTURA DE COMPUTADORAS

Cociente 1110: 14

Resto 0010: 2

- 3) Realizar las siguientes operaciones con números en punto flotante en el formato IEEE 754 (en las que las partes significativas se truncan a cuatro dígitos decimales). Indique los resultados en forma normalizada). Calcule el error cometido.

a) $C30C0000 + C1500000$

b) $3B370000 + 39F\ 68000$

Representación de números en punto flotante

Según esta normalización, los números pueden ser representados de acuerdo a los siguientes formatos según correspondan a simple o doble precisión.

SIMPLE PRECISIÓN

S	Exponente	Mantisa
1 bit	8 bits	23 bits

DOBLE PRECISIÓN

S	Exponente	Mantisa
1 bit	11 bits	52 bits

Donde:

S es el bit de signo del número y es **0** para **+** y es **1** para **-**

EXP es el exponente de 2 que está representado en exceso 127 para simple precisión y exceso 1023 para doble precisión.

MANT es la mantisa y cumple con la condición de ser la parte del número representado que queda a la derecha de la coma cuando a la izquierda de ella hay un sólo 1.

Por lo tanto el número representado se obtiene así:

$$(-1)^S \cdot 2^{(EXP-127)} \cdot 1, MANT \quad \text{simple precisión}$$

$$(-1)^S \cdot 2^{(EXP-1023)} \cdot 1, MANT \quad \text{doble precisión}$$

Para representar los números cercanos a cero se utiliza un formato desnormalizado, haciendo $EXP=0$ y $MANT \neq 0$.

El número que se representa cuando $EXP=0$ es:

$$(-1)^S \cdot 2^{-126} \cdot 0, MANT \quad \text{simple precisión}$$

$$(-1)^S \cdot 2^{-1022} \cdot 0, MANT \quad \text{doble precisión}$$



Universidad Nacional de la Patagonia San Juan Bosco
Facultad de Ingeniería. Sede Puerto Madryn
ARQUITECTURA DE COMPUTADORAS

En esta normalización el cero se representa con EXP=0 y MANT=0.

a) C30C0000 + C1500000

$$C30C0000 \rightarrow 1 \mid 100\ 0011\ 0 \mid 000110000000000000000000 = -1.00011 \times 2^7$$

$$C1500000 \rightarrow 1 \mid 100\ 0001\ 0 \mid 101000000000000000000000 = -1.101 \times 2^3$$

Igualamos los exponentes al mayor de los dos y sumamos. El resultado obtenido se vuelve a normalizar. No olvidar el bit implícito que no está en la mantisa.

$$-1.000\ 1100 \times 2^7$$

$$\underline{-0.000\ 1101 \times 2^7}$$

$$-1.001\ 1001 \times 2^7$$

$$-1.001\ 1001 \times 2^7 \rightarrow C3190000$$

b) 3B370000 + 39F68000

$$3B370000 \rightarrow 0 \mid 01110110 \mid 011011100000000000000000 = 1.0110111 \times 2^{-9}$$

$$\text{Exponente} = 118 - 127 = -9$$

$$39F68000 \rightarrow 0 \mid 01110011 \mid 111011010000000000000000 = 1.11101101 \times 2^{-12}$$

$$\text{Exponente} = 115 - 127 = -12$$

Igualamos los exponentes al mayor de los dos y sumamos. El resultado obtenido se vuelve a normalizar. No olvidar el bit implícito que no está en la mantisa.

$$1.11101101 \times 2^{-12} \rightarrow 0.00111101101 \times 2^{-9}$$

$$1.0110111000 \times 2^{-9}$$

$$\underline{0.00111101101 \times 2^{-9}}$$

$$1.10101011101 \times 2^{-9}$$

$$0 \mid 01110110 \mid 101010111010000000000000$$

Resultado: 3B55D000

- 4) **Escribir** un programa que defina **4 variables**: *a, b, c* y *d* inicializadas a 0. Debe cargar en los registros *x5, x6, x7* y *x8* las **direcciones** de cada una de ellas usando 4 pseudoinstrucciones **la**. Ensamblar el programa. Abre la tabla de símbolos y comprueba que en los registros *x5-x8* se han cargado las direcciones de las variables *a-d* respectivamente. ¿Cuántos bytes ocupa el programa?



- 5) **Escribir** un programa que recorra una TABLA de diez números enteros y determine cuántos elementos son mayores que X. El resultado debe almacenarse en una dirección etiquetada CANT. El programa debe generar además otro arreglo de palabras llamado RES cuyos elementos sean ceros y unos. Un '1' indicará que el entero correspondiente en el arreglo TABLA es mayor que X, mientras que un '0' indicará que es menor o igual.
- 6) Hacer un diagrama de tiempos usando adelantamiento de datos, analizar los riesgos (hazards) en la ejecución de esta porción de código y buscar alguna alternativa para optimizar el tiempo

```
lw t3, 0(t5)
add t8, t6, t3
addi t2, t5, 0x4
sw t5, 0(t2)
```

Diagrama de tiempos sin adelantamientos

lw t3, 0(t5)	IF	ID	EX	MEM	WB							
add t8, t6, t3		IF	ID			EX	MEM	WB				
addi t2, t5, 0x4			IF			ID			EX	MEM	WB	
sw t5, 0(t2)							IF		ID	EX	MEM	WB

Diagrama de tiempos con adelantamientos

lw t3, 0(t5)	IF	ID	EX	MEM	WB							
add t8, t6, t3		IF	ID			EX	MEM	WB				
addi t2, t5, 0x4			IF			ID	EX	MEM	WB			
sw t5, 0(t2)							IF	ID	EX	MEM	WB	

Tarda 5 ciclos sin contar el llenado de la tubería

Con reordenamiento y adelantamiento

lw t3, 0(t5)	IF	ID	EX	MEM	WB							
addi t2, t5, 0x4		IF	ID	EX	MEM	WB						
add t8, t6, t3			IF	ID	EX	MEM	WB					
sw t5, 0(t2)				IF	ID	EX	MEM	WB				

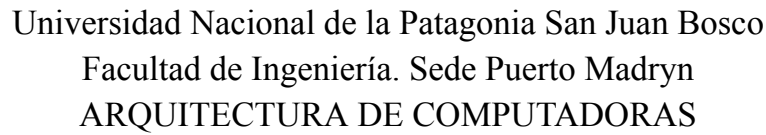


Tarda 4 ciclos sin contar el llenado de la tubería

- 7) Analiza las dependencias de datos de la siguiente secuencia de instrucciones en una máquina segmentada Mips usando un diagrama de tiempos

```
lw t3 , 0 ( t5 )  
add t8 , t6 , t3  
add t9 , t8 , t5  
add t7 , t2 , t1  
sw t5 , 0 ( t2 )
```

- Analiza la mejora a través de un diagrama de tiempo usando adelantamiento de datos.
- Analiza la mejora a través de un diagrama de tiempo usando reordenamiento.
- Analiza la mejora a través de un diagrama de tiempo usando reordenamiento y adelantamiento de datos.


$$\frac{T_1}{T_2} = \frac{5}{5} = 1$$



Universidad Nacional de la Patagonia San Juan Bosco
Facultad de Ingeniería. Sede Puerto Madryn
ARQUITECTURA DE COMPUTADORAS

- 8) Explica a nivel transferencia entre registros el proceso de búsqueda, decodificación y ejecución de la instrucción: `addi t2, t8, 0x5a`

BÚSQUEDA

$MAR \leftarrow PC$

READ ($MDR \leftarrow MP[MAR]$)

$IR \leftarrow MDR$

DECODIFICACIÓN

$\mu MAR \leftarrow OPC$ $PC \leftarrow PC + 1$

EJECUCIÓN

$BusA \leftarrow R[rs]$; $BusB \leftarrow R[rt]$

$BusS \leftarrow BusA + BusB$

$R[rd] \leftarrow BusS$

- 9) Describe a nivel transferencias entre registros el proceso de búsqueda, decodificación y ejecución de la instrucción: `beq t3, t4, salto`

BÚSQUEDA

$MAR \leftarrow PC$

READ ($MDR \leftarrow MP[MAR]$)

$IR \leftarrow MDR$

DECODIFICACIÓN

$\mu MAR \leftarrow OPC$ $PC \leftarrow PC + 1$

EJECUCIÓN

$BUS_A \leftarrow R[t3]$; $BUS_B \leftarrow R[t4]$

$DST \leftarrow PC + \text{Desplazamiento}$

$Z \leftarrow BUS_A - BUS_B$

SI Z ENTONCES $PC \leftarrow DST$