Sistemas de computación 1

Eejrcitación Von Neumann

1) Carga el valor 00FF en el Registro R0 y el valor 000F en el Registro R1. Realiza las siguientes operaciones secuencialmente: Resta el contenido de R1 a R0, almacenando el resultado en R2.

1. Realiza un OR bit a bit entre R0 y R1, almacenando el resultado en R3.
2. Incrementa el valor de R2 en una unidad.
3. Realiza un NOT bit a bit sobre el contenido de R3, almacenando el resultado en R4.
4. Compara el valor final de R2 con el valor final de R4.

Posición memoria	**Código binario**	**Mnemotécnico**	**Código Hexadecimal**
0100	0010 1000 0000 0000	MOVH R0, 00	2800
0101	0010 0000 1111 1111	MOVL R0, FF	20FF
0102	0010 1001 0000 0000	MOVH R1, 00	2900
0103	0010 0001 0000 1111	MOVL R1, 0F	210F
0104	0100 1010 0000 0100	SUB R2, R0, R1	4A04
0105	0101 0011 0000 0100	OR R3, R0, R1	5304
0106	1000 1010 0000 0000	INC R2	8A00
0107	1000 0011 0000 0000	NOT R3	8300
0108	0000 1100 0110 0000	MOV R4, R3	0C60
0109	0110 1010 1000 0000	COMP R2, R4	6A80

2) Carga el valor 0055 en el Registro R0 y el valor 00AA en el Registro R1. Realiza las siguientes operaciones secuencialmente:

1. Decrementa el valor de R0 en una unidad.
2. Realiza una operación XOR entre R0 y R1, almacenando el resultado en R2.
3. Incrementa el valor de R1 en dos unidades (realiza dos INC).
4. Realiza la operación NOT sobre el contenido de R2, almacenando el resultado en R3.
5. Realiza una resta entre R3 y R2 y lo guarda en R4

Posición memoria	**Código binario**	**Mnemotécnico**	**Código Hexadecimal**
0100	0010 1000 0000 0000	MOVH R0, 00	2800
0101	0010 0000 0101 0101	MOVL R0, 55	2055
0102	0010 1001 0000 0000	MOVH R1, 00	2900
0103	0010 0001 1010 1010	MOVL R1, AA	21AA
0104	1001 0000 0000 0000	DEC R0	9000
0105	0110 0010 0000 0100	XOR R2, R0, R1	6204
0106	1000 1001 0000 0000	INC R1	8900
0107	1000 1001 0000 0000	INC R1	8900
0108	1000 0010 0000 0000	NOT R2	8200
0109	0000 1011 0100 0000	MOV R3, R2	0B40
010A	0100 1100 0110 1000	SUB R4, R3, R2	4C68

3) Teniendo en cuenta las siguientes instrucciones y cargando los espacios de memoria dados, interpretar que esta realizando el conjunto de instrucciones:

-Los datos en memoria son:

0200 → 000A
0201 → 000B
0202 → 000C
0203 → 000D
0204 → 000E

-Las instrucciones son:

100. MOVH R6, 02
101. MOVL R5, 05
102. MOV R4, [R6]
103. ADD R7, R7, R4
104. INC R6
105. DEC R5
106. COMP R5, R0
107. (Cond) BRNZ –6 Z=0

-RESOLUCION

100. MOVH R6, 02 (0010 1110 0000 0010 2E02)
101. MOVL R5, 05 (0010 0101 0000 0101 2505)
102. MOV R4, [R6] (0001 0100 1100 0000 14C0)
103. ADD R7, R7, R4 (0100 0111 1111 0000 47F0)
104. INC R6 (1000 1110 0000 0000 8E00)
105. DEC R5 (1001 0101 0000 0000 9500)
106. Comp R5, R0 (0110 1101 0000 0000 6D00) (Comparar si R5 =0, si no es igual, saltar a hacer de nuevo 4)
107. (Cond) BRNZ –6 (1111 0101 1111 1010 F5FA)

-INTERPRETACIÓN

100. Se colocan en los primeros 8 bits del registro R6, el número 02.
101. Se colocan en los últimos 8 bits del registro R5, el número 05.
102. Se copia al registro R4 el contenido de la posición de memoria cuya dirección esta en R6 (en este caso es 0200).
103. Se suman los registros R4 y R7 guardandose el resultado en R7.
104. Se incrementa en 1 el valor del registro R6.
105. Se decrementa en 1 el valor del registro R5.
106. Compara R5 con R0 efectuando una resta, dicho resultado sirve como condicional para la señal de bandera y salto en memoria en la siguiente instrucción.
107. En caso de que la señal de bandera no sea cero, efectúa el salto de memoria 6 posiciones atrás.

4) Considerando que la CPU Elemental no tiene una instrucción de multiplicación directa, diseñar un procedimiento que realice una multiplicación de dos números (asumiendo que son positivos) utilizando sumas repetidas. Los números por multiplicar deben pasarse en R1 y R2, y el resultado debe devolverse en R0.

Posición memoria	**Código binario**	**Mnemotécnico**	**Código Hexadecimal**
0100	0010 0001 0000 0010	MOVL R1, 02	2102
0101	0010 0010 0000 0011	MOVL R2, 03	2203
0102	0100 0000 0000 0100	ADD R0, R0, R1	4004
0103	1001 0010 0000 0000	DEC R2	9200
0104	0110 1010 0110 0000	COMP R2, R3	6A60
0105	1111 0101 1111 1100	BRNZ -4	F5FC

5) Cargar dos números desde la memoria, sumarlos y guardar el resultado en otra posición de memoria. Asumir que la posición de memoria [R5] contiene el primer número y [R6] el segundo. El resultado debe almacenarse en [R7].
NOTA: R5 y R6 así como las direcciones de memoria a las que apunta hay que cargarlas previamente.

-Los datos en memoria son:

0200 → 0004
0201 → 0007
0202 → (resultado)

Posición memoria	**Código binario**	**Mnemotécnico**	**Código Hexadecimal**
0100	0010 1101 0000 0010	MOVH R5, 02	2D02
0101	0010 1110 0000 0010	MOVH R6, 02	2E02
0102	0010 0110 0000 0001	MOVL R6, 01	2601
0103	0010 1111 0000 0010	MOVH R7, 02	2F02
0104	0010 0111 0000 0010	MOVL R7, 02	2702
0105	0001 0000 1010 0000	MOV R0, [R5]	10A0
0106	0001 0001 1100 0000	MOV R1, [R6]	11C0
0107	0100 0010 0000 0100	ADD R2, R0, R1	4204
0108	0001 1111 0100 0000	MOV [R7], R2	1F40

6) Implementar una llamada a una subrutina que realice una operación lógica y retorne. La subrutina (SUB_OR) debe realizar una operación OR entre R1 y R2 y guardar el resultado en R3

```
MOVL R1, #5 -> 0x2105
MOVL R2, #3 -> 0x2203
CALL SUB_OR_OFFSET -> 0xD006 (Asumiendo un offset de 0x06)
NOP -> 0x0000
OR R3, R1, R2 -> 0x5328
RET -> 0xE000 (0109)
```

Posición memoria	Código binario	Mnemotécnico	Código Hexadecimal
0100	0010 0001 0000 0101	MOVL R1, 05	2105
0101	0010 0010 0000 0011	MOVL R2, 03	2203
0102	1101 0000 0000 0110	CALL +6	D006
0103	0000 0000 0000 0000	NOP	0000
0104	-	-	-
0105	-	-	-
0106	-	-	-
0107	-	-	-
0108	-	-	-
0109	0101 0011 0010 1000	OR, R3, R1, R2	5328
010A	1110 0000 0000 0000	RET	E000

7) Negar un número en un registro y luego saltar si el resultado es cero. El registro R4 contiene un valor. Si después de negarlo es cero, se debe saltar a una etiqueta FIN.

Posición memoria	Código binario	Mnemotécnico	Código Hexadecimal
0100	0010 0100 1111 1111	MOVL R4, FF	24FF
0101	0010 1100 1111 1111	MOVH R4, FF	2CFF
0102	1000 0100 0000 0000	NOT R4	8400
0103	0110 1100 0000 0000	COMP R4, R0	6C00
0104	1111 0100 0000 0011	BRZ +3	F403
0105	-	-	-
0106	-	-	-
0107	-	-	-
0108	FIN	FIN	FIN