

# Guía de ejercicios # 7

## Flags y Saltos

Organización de Computadoras 2021 C3

UNQ

Los objetivos de esta práctica son:

- Comprender qué son y para qué se utilizan los Flags.
- Conocer qué operaciones modifican los Flags y cómo los modifican.
- Comprender los conceptos de saltos absolutos o condicionales.
- Introducir conceptos de iteración o bucles en programas.
- Notar la relación entre invocación a rutinas y saltos.

## Arquitectura Q4

### Características

- Tiene 8 registros de uso general de 16 bits: R0..R7.
- La memoria utiliza direcciones de 16 bits.
- Contador de programa (*Program counter*) de 16 bits.
- *Stack Pointer* de 16 bits. Comienza en la dirección 0xFFEF.
- Flags: Z, N, C, V (Zero, Negative, Carry, oVerflow). Instrucciones que alteran Z y N: ADD, SUB, CMP, DIV, MUL, AND, OR, NOT. Las 3 primeras además calculan C y V.

### Instrucciones de dos operandos

Formato de Instrucción

CodOp (4b)	Modo Destino (6b)	Modo Origen (6b)	Destino (16b)	Origen (16b)
---------------	----------------------	---------------------	------------------	-----------------

Tabla de instrucciones

Operación	Cod Op	Efecto
MUL	0000	Dest $\leftarrow$ Dest * Origen
MOV	0001	Dest $\leftarrow$ Origen
ADD	0010	Dest $\leftarrow$ Dest + Origen
SUB	0011	Dest $\leftarrow$ Dest - Origen
CMP	0110	Dest - Origen
DIV	0111	Dest $\leftarrow$ Dest % Origen

## Instrucciones de un operando origen

### Formato de Instrucción

CodOp (4b)	Relleno (000000)	Modo Origen (6b)	Operando Origen (16b)
---------------	---------------------	---------------------	--------------------------

### Tabla de instrucciones

Operación	Cod Op	Efecto
CALL	1011	$[SP] \leftarrow PC$ ; $SP \leftarrow SP - 1$ ; $PC \leftarrow \text{Origen}$
JMP	1010	$PC \leftarrow \text{Origen}$

## Instrucciones sin operandos

### Formato de Instrucción

CodOp (4b)	Relleno (00000000000000)
---------------	-----------------------------

### Tabla de instrucciones

Operación	CodOp	Efecto
RET	1100	$SP \leftarrow SP + 1$ ; $PC \leftarrow [SP]$

## Salto condicionales

### Formato de Instrucción

Cod_Op (8)	Desplazamiento(8)
------------	-------------------

Los primeros cuatro bits del campo Cod\_Op es la cadena  $1111_2$ . Si **al evaluar la condición de salto** el resultado es 1, se le suma al PC el valor del desplazamiento, representado en  $CA2(8)$ . En caso contrario la instrucción no hace nada.

Codop	Op.	Descripción	Condición de Salto
0001	JE	Igual / Cero	Z
1001	JNE	No igual	not Z
0010	JLE	Menor o igual	Z or ( N xor V )
1010	JG	Mayor	not ( Z or ( N xor V ) )
0011	JL	Menor	N xor V
1011	JGE	Mayor o igual	not ( N xor V )
0100	JLEU	Menor o igual sin signo	C or Z
1100	JGU	Mayor sin signo	not ( C or Z )
0101	JCS	Carry / Menor sin signo	C
0110	JNEG	Negativo	N
0111	JVS	Overflow	V

## Modos de direccionamiento

Modo	Codificación
Inmediato	000000
Directo	001000
Registro	100rrr

## 1 Flags y Saltos

### Ejemplo de cálculo de Flags:

Considerando la operación de suma  $1001 + 1110$  en  $BSS(4)$ , es posible ver:

$$\begin{array}{r} \text{+} \quad 1 \ 0 \ 0 \ 1 \\ \text{acarreo} \quad 1 \ 1 \ 1 \ 0 \\ \hline 0 \ 1 \ 1 \ 1 \end{array}$$

Los valores de los flags luego de la operación, son los siguientes:

- **Z (Zero)** = 0. Ya que el es distinto de 0000.
- **N (Negative)** = 0. El resultado comienza con 0.
- **C (Carry)** = 1. La operación tiene acarreo.
- **V (Overflow)** = 1. Ambos operandos representarían números negativos (a pesar de que estemos trabajando en  $BSS()$ ), por lo que la suma, debiera ser también un número negativo, y en ese caso no lo es.

### Ejercicios:

1. Realizar las siguientes operaciones en  $BSS(4)$  y calcular los flags a partir de los resultados de las mismas.
  - (a)  $1010 + 1001$
  - (b)  $1011 - 1011$
  - (c)  $0010 + 1101$
  - (d)  $0010 - 0111$
  - (e)  $1100 - 1000$
2. Dar los valores de R3 y R4 (y calcular los flags de la primer instrucción) que hagan que se ejecute la instrucción RET:
  - (a) 

```
rutina: CMP  R3, R4
        JLE fin
        CALL boom
fin: RET
```

```

(b)          rutina: CMP  R3, R4
                JL  fin
                CALL boom
                fin: RET

```

3. Indique verdadero o falso. Justifique.

- (a) La instrucción JMP no modifica el SP.
- (b) Los flags se modifican con todas las instrucciones de dos operandos.
- (c) La instrucción JE es un salto incondicional.

## 2 Ensamblado de saltos

### Ejercicios:

4. Considere el siguiente programa.

```

rutina: MOV R3, [0x0A0A]
        SUB R0, 0x0001
        JE fin
        MOV R3, 0xFFFF
fin: RET

```

- (a) Ensamblar a a partir de la celda **CAFE**.
- (b) ¿Que valor tiene el desplazamiento del salto **JE**?
- (c) ¿A que celda queda asociada la etiqueta **fin**?
- (d) Explique que hace el programa

5. Considere el siguiente programa.

```

rutina: CMP R0, 0x0000
        JL menoracero
        CMP R0, 0x000A
        JG mayoradiez
        MOV R3, 0x0001
        JMP fin
menoracero: MOV R3, 0x0000
        JMP fin
mayoradiez: MOV R3, 0x0002
        fin: RET

```

- (a) Ensamblar a a partir de la celda **F0CA**.
- (b) ¿Que valor tiene el desplazamiento del salto **JL**?
- (c) ¿Que valor tiene el desplazamiento del salto **JG**?
- (d) ¿A que celda queda asociada la etiqueta **menoracero**?

- (e) ¿A que celda queda asociada la etiqueta `mayoradiez`?
  - (f) ¿A que celda queda asociada la etiqueta `fin`?
  - (g) Explique que hace el programa
6. Dado el siguiente mapa de memoria, simule la ejecución de un programa que comienza en la celda A893, asumiendo que R0 = 0000 y R1 = F000

	...
A893	6821
A894	FC04
A895	1980
A896	FFFF
A897	A000
A898	A89B
A899	1980
A89A	AAAA
A89B	C000
	...

### 3 Estructura condicional

Ver ejemplo de rutinas con saltos absolutos y condicionales, al final de la sección

- 7. Escribir un programa que, si el valor en R0 es igual al valor en R1, ponga un 1 en R2, 0 en caso contrario.
- 8. Escribir un programa que, si el valor en R7 es negativo, le sume 1, o le reste 1 en caso contrario.
- 9. Escribir un programa que, si la suma entre R3 y R4 es menor a 512, guarde el resultado en la celda 1000, sino en la celda 2000.
- 10. Escribir un programa que, si el valor en R1 es 0 guarde el valor almacenado en la celda 0FFE en R7, en caso contrario que guarde la suma entre R2 y R3.
- 11. Implementar las siguientes rutinas según su documentación asumiendo que los valores que manejan están en C A2(16):

- (a)
 

```

; ----- min
; REQUIERE: Valores a comparar en R0 y R1.
; MODIFICA: ???
; RETORNA : En R5 el valor mínimo entre
;           R0 y R1.
      
```
- (b)
 

```

; ----- multiplo
; REQUIERE: Valores en R1 y R2.
; MODIFICA: ???
; RETORNA : Un 1 en R0 si el numero que esta
;           en R2 es múltiplo de R1, un 0 en
;           caso contrario.
      
```

12. Usando la rutina `multiplo`, hacer una rutina `esPar` que dado un número en R1, retorne en R0 un 1 si el número de R1 es par, un 0 en caso contrario. Documente la rutina.
13. Usando la rutina `min`, hacer un programa que le sume R3 el valor más chico entre lo que está guardado en las celdas `CAFE` y `1882`.

### Ciclo de ejecución y accesos

14. Suponer que la instrucción `CMP [AAAA],R1` está ensamblada a partir de la celda 0000.
  - (a) ¿Qué celdas se acceden durante la búsqueda de instrucción?
  - (b) ¿Qué celdas se acceden durante la búsqueda de operandos?
  - (c) ¿Qué celdas se acceden durante el almacenamiento de operandos?
15. Suponer que la instrucción `JE esIgual` está ensamblada a partir de la celda 0000.
  - (a) ¿Qué celdas se acceden durante la búsqueda de instrucción?
  - (b) ¿Qué celdas se acceden durante la búsqueda de operandos?
  - (c) ¿Qué celdas se acceden durante el almacenamiento de operandos?