

## Práctica 3 - Arquitectura ORGA1

### Descripción General

- ▷ Palabras de *16 bits*.
- ▷ Direccionamiento a palabra.
- ▷ Espacio direccionable de *65536* palabras.
- ▷ Espacio de direcciones dedicado a entrada/salida en las direcciones 0xFFF0 - 0xFFFF.
- ▷ Ocho registros de propósito general de *16 bits*: R0..R7.
- ▷ *Program counter* (PC) de *16 bits*.
- ▷ *Stack pointer* (SP) de *16 bits* inicializado en la dirección 0xFFEF.
- ▷ Los valores de los *flags* se calculan interpretando los operandos en complemento a 2.  
*Flags*: Z (*zero*), N (*negative*), C (*carry*), V (*overflow*).
- ▷ Todas las instrucciones alteran los *flags*, excepto MOV, CALL, RET, JMP y Jxx.
- ▷ De las que alteran los *flags*, todas dejan C y V en cero, excepto ADD, ADDC, SUB, CMP y NEG.

### Formato de instrucción

#### Tipo 1: Instrucciones de dos operandos

<i>4 bits</i>	<i>6 bits</i>	<i>6 bits</i>	<i>16 bits</i>	<i>16 bits</i>
cod. op.	destino	fuente	constante destino (opcional)	constante fuente (opcional)

operación	cod. op.	efecto
MOV $d, f$	0001	$d \leftarrow f$
ADD $d, f$	0010	$d \leftarrow d + f$ (suma binaria)
SUB $d, f$	0011	$d \leftarrow d - f$ (resta binaria)
AND $d, f$	0100	$d \leftarrow d \text{ and } f$
OR $d, f$	0101	$d \leftarrow d \text{ or } f$
CMP $d, f$	0110	Modifica los <i>flags</i> según el resultado de $d - f$ (resta binaria)
ADDC $d, f$	1101	$d \leftarrow d + f + \text{carry}$ (suma binaria)

Formato de operandos destino y fuente.

Modo	Codificación	Resultado
Inmediato	000000	c16
Directo	001000	[c16]
Indirecto	011000	[[c16]]
Registro	100rrr	Rrrr
Indirecto registro	110rrr	[Rrrr]
Indexado	111rrr	[Rrrr + c16]

c16 es una constante de *16 bits*.

Rrrr es el registro indicado por los últimos tres *bits* del código de operando.

Las instrucciones que tienen como destino un operando de tipo *inmediato* son consideradas como inválidas por el procesador, excepto el CMP.

#### Tipo 2: Instrucciones de un operando

Tipo 2a: Instrucciones de un operando destino.

<i>4 bits</i>	<i>6 bits</i>	<i>6 bits</i>	<i>16 bits</i>
cod. op.	destino	000000	constante destino (opcional)

operación	cod. op.	efecto
NEG $d$	1000	$d \leftarrow 0 - d$ (resta binaria)
NOT $d$	1001	$d \leftarrow \text{not } d$ (bit a bit)

El formato del operando *destino* responde a la tabla de formatos de operando mostrada más arriba.

Tipo 2b: Instrucciones de un operando fuente.

<i>4 bits</i>	<i>6 bits</i>	<i>6 bits</i>	<i>16 bits</i>
cod. op.	000000	fuente	constante fuente (opcional)

operación	cod. op.	efecto
JMP $f$	1010	$PC \leftarrow f$
CALL $f$	1011	$[SP] \leftarrow PC, SP \leftarrow SP - 1, PC \leftarrow f$

El formato del operando *f* responde a la tabla de formatos de operando mostrada más arriba.

### Tipo 3: Instrucciones sin operandos

	4 bits	6 bits	6 bits
cod. op.	000000	000000	

operación	cod. op.	efecto
RET	1100	$PC \leftarrow [SP+1], SP \leftarrow SP + 1$

### Tipo 4: Saltos condicionales

Las instrucciones en este formato son de la forma  $Jxx$  (salto relativo condicional). Si al evaluar la condición de salto en los *flags* el resultado es 1, el efecto es incrementar el PC con el valor de los 8 bits de desplazamiento, representado en *complemento a 2* de 8 bits. En caso contrario, la instrucción no produce efectos.

8 bits	8 bits
cod. op.	desplazamiento

Codop	Operación	Descripción	Condición de Salto
1111 0001	JE	Igual / Cero	Z
1111 1001	JNE	Distinto	not Z
1111 0010	JLE	Menor o igual	Z or ( N xor V )
1111 1010	JG	Mayor	not ( Z or ( N xor V ) )
1111 0011	JL	Menor	N xor V
1111 1011	JGE	Mayor o igual	not ( N xor V )
1111 0100	JLEU	Menor o igual sin signo	C or Z
1111 1100	JGU	Mayor sin signo	not ( C or Z )
1111 0101	JCS	Carry / Menor sin signo	C
1111 0110	JNEG	Negativo	N
1111 0111	JVS	Overflow	V

### ¿Cómo altera los flags?

Sea  $r$  el resultado de una instrucción que modifica los flags, el nuevo valor es el que sigue:

- $Z=1 \leftrightarrow r = 0x0000$ .
- $N=1 \leftrightarrow$  el bit más significativo de  $r$  es igual a 1.
- $C=1 \leftrightarrow$  se produjo *carry* durante una suma binaria o *borrow* durante una resta binaria.
- $V=1 \leftrightarrow$  la suma de dos números con signo produce un número *sin* signo ( $S + S = \bar{S}$ ) ó la suma de dos números *sin* signo produce un número con signo ( $\bar{S} + \bar{S} = S$ ) ó alguna analogía con la resta ( $S - \bar{S} = \bar{S}$  ó  $\bar{S} - S = S$ )

## El ensamblador

### Directivas

El ensamblador de código tiene una única directiva.

Directiva	Efecto
DW c16	Asigna en la posición correspondiente la constante c16