La Máquina **ORGA1**: Arquitectura y Seguimiento

Organización del Computador I

Verónica Coy

Departamento de Computación - FCEyN UBA

2^{do} Cuatrimestre 2017

Outline

Arquitectura

¿Qué es la Arquitectura de una Computadora?

Es el conjunto de reglas e instrucciones que describen su funcionalidad, así como su organización e implementación. Comprende tanto al *Set de Instrucciones* de la máquina como al diseño de su *Microarquitectura* y sus *circuitos lógicos*.

¿Qué es la Arquitectura de una Computadora?

Es el conjunto de reglas e instrucciones que describen su funcionalidad, así como su organización e implementación. Comprende tanto al *Set de Instrucciones* de la máquina como al diseño de su *Microarquitectura* y sus *circuitos lógicos*.

A nivel general, podemos afirmar que la máquina **ORGA1** tiene arquitectura de *von Neumann*. Esto implica que:

¿Qué es la Arquitectura de una Computadora?

Es el conjunto de reglas e instrucciones que describen su funcionalidad, así como su organización e implementación. Comprende tanto al *Set de Instrucciones* de la máquina como al diseño de su *Microarquitectura* y sus *circuitos lógicos*.

A nivel general, podemos afirmar que la máquina **ORGA1** tiene arquitectura de *von Neumann*. Esto implica que:

► Tiene un **CPU** que cuenta con una ALU y registros internos.

¿Qué es la Arquitectura de una Computadora?

Es el conjunto de reglas e instrucciones que describen su funcionalidad, así como su organización e implementación. Comprende tanto al *Set de Instrucciones* de la máquina como al diseño de su *Microarquitectura* y sus *circuitos lógicos*.

A nivel general, podemos afirmar que la máquina **ORGA1** tiene arquitectura de *von Neumann*. Esto implica que:

- ► Tiene un **CPU** que cuenta con una ALU y registros internos.
- ► Tiene una **Memoria** que guarda tanto datos como instrucciones.

¿Qué es la Arquitectura de una Computadora?

Es el conjunto de reglas e instrucciones que describen su funcionalidad, así como su organización e implementación. Comprende tanto al *Set de Instrucciones* de la máquina como al diseño de su *Microarquitectura* y sus *circuitos lógicos*.

A nivel general, podemos afirmar que la máquina **ORGA1** tiene arquitectura de *von Neumann*. Esto implica que:

- ► Tiene un **CPU** que cuenta con una ALU y registros internos.
- ► Tiene una **Memoria** que guarda tanto datos como instrucciones.
- ► Tiene dispositivos de **Entrada/Salida** para interactuar con el 'afuera' de la máquina.



Registros

Registros de Propósito General

- 8 registros de propósito general de 16 bits.
 - ► R0
 - **•** ...
 - ▶ R7
- Se utilizan para poder traer datos desde Memoria hacia el CPU y poder operar con ellos.
- Están disponibles para que el programador los use a su antojo.

Registros de Propósito Específico

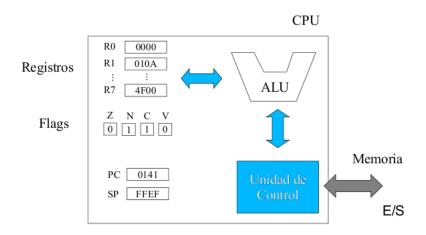
- ▶ 3 registros de **propósito específico** de 16 bits.
 - ▶ **PC** (Program Counter): Apunta a la dirección que contiene la próxima instrucción a ejecutar.
 - ▶ SP (Stack Pointer): Apunta a la próxima posición libre de la pila. ¹
 - IR (Instruction Register): Guarda la instrucción a ser inmediatamente ejecutada por el CPU.
- No son directamente accesibles por el programador. Sólo pueden ser modificados por la ejecución de instrucciones.

¹El **Stack** es una estructura de datos en **Memoria** que veremos hacia el final de la clase.

FLAGS

- Los FLAGS también son registros pero de 1 bit.
- Sus valores cambian en función de la ejecución de algunas operaciones. No todas las operaciones alteran los FLAGS.
- Son los mismos que vimos en los Talleres de Lógica Digital.
 - **► Z** (zero)
 - ▶ N (negative)
 - ► C (carry)
 - ▶ **V** (overflow)

Nuestra máquina por el momento...



Memoria

Memoria

- ▶ La **Memoria** tiene un tamaño fijo con 65520 posiciones.
- Cada posición de Memoria tiene asociada una dirección de 16 bits. Entonces:
 - Las posiciones de Memoria se numeran en el rango de 0x0000 a 0xFFFF.
 - Quedan 16 direcciones libres (0xFFF0 a 0xFFFF) que no apuntan a posiciones de **Memoria**. En cambio, usan para mapear registros de E/S. ²
- ▶ La **Memoria** tiene direccionamiento a palabra. Eso significa que:
 - ► Cada posición de **Memoria** puede guardar una **palabra**.
 - Las palabras en la **ORGA1** son datos de 16 bits.

²Más de esto en la segunda mitad de la materia. ←□ → ←② → ←② → ←② → → ② → ◆② ←

Set de Instrucciones

4 bits

Tipo 1: Instrucciones de dos operandos

6 bits 6 bits

cod. op.	destino	fuente	constante destino (opcional)	constante fuente (opcional)
operación	cod. op.	efecto		modifica flags
MOV d , f	0001	$d \leftarrow$	f	no
ADD d , f	0010	$d \leftarrow$	d+f (suma binaria)	sí
SUB d , f	0011	$d \leftarrow$	d-f (resta binaria)	sí
ANID 7 C	0100	7 .	7 1 6	,

16 bits

MOV d , f	0001	$d \leftarrow f$	no
ADD d , f	0010	$d \leftarrow d + f$ (suma binaria)	sí
SUB d , f	0011	$d \leftarrow d - f$ (resta binaria)	sí
AND d , f	0100	$d \leftarrow d$ and f	SÍ (*)
OR d , f	0101	$d \leftarrow d \text{ or } f$	SÍ (*)
CMP d, f	0110	Modifica los $flags$ según el resultado de $d-f$.	sí
ADDC d , f	1101	$d \leftarrow d + f + carry$ (suma binaria)	sí

 $^{(\}star)$ dejan el flag de carry (C) y el de overflow (V) en cero.

16 bits

Tipo 2: Instrucciones de un operando

Tipo 2a: Instrucciones de un operando destino.

		$16\ bits$
cod. op. des	tino 000000	constante destino (opcional)

operacin	cod. op.	efecto	modifica flags
NEG d	1000	$d \leftarrow$ el inverso aditivo de d	S
NOT d	1001	$d \leftarrow \text{not } d \text{ (bit a bit)}$	S (*)

 (\star) deja el flag de carry (C) y el de overflow (V) en cero.

Tipo 2b: Instrucciones de un operando fuente.

4 bits	6 bits	6 bits	16 bits
cod. op.	000000	fuente	constante fuente (opcional)

operacin	cod. op.	efecto	modifica flags
$JMP\ f$	1010	$PC \leftarrow f$	no
CALL f	1011	$[SP] \leftarrow PC, SP \leftarrow SP - 1, PC \leftarrow f$	no

Tipo 3: Instrucciones sin operandos

$4 \ bits$	$6 \ bits$	$6 \ bits$
cod. op.	000000	000000

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

▶ No modifica *flags*.

Tipo 4: Saltos relativos condicionales

- la condición de salto correspondiente.
- ► PC ← PC + desplazamiento
- el desplazamiento se representa en complemento a 2 de 8 bits.
- No modifican flags.

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

Modos de Direccionamiento

Modos de Direccionamiento

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.

Programación de la ORGA1

El ciclo de vida de un programa para la máquina **ORGA1** consta de las siguientes 4 etapas:

 Programación: Se escribe un algoritmo usando las instrucciones en lenguaje ensamblador (o assembly) provistas por el set de instrucciones de la Arquitectura. Esto da origen a un código fuente.

- Programación: Se escribe un algoritmo usando las instrucciones en lenguaje ensamblador (o assembly) provistas por el set de instrucciones de la Arquitectura. Esto da origen a un código fuente.
- Ensamblado: Un programa llamado Ensamblador toma el código fuente y lo traduce a un código máquina comprensible para la máquina ORGA1. Para esto, realiza los siguientes pasos:

- Programación: Se escribe un algoritmo usando las instrucciones en lenguaje ensamblador (o assembly) provistas por el set de instrucciones de la Arquitectura. Esto da origen a un código fuente.
- Ensamblado: Un programa llamado Ensamblador toma el código fuente y lo traduce a un código máquina comprensible para la máquina ORGA1. Para esto, realiza los siguientes pasos:
 - Resuelve las directivas dirigidas al Ensamblador.

- Programación: Se escribe un algoritmo usando las instrucciones en lenguaje ensamblador (o assembly) provistas por el set de instrucciones de la Arquitectura. Esto da origen a un código fuente.
- Ensamblado: Un programa llamado Ensamblador toma el código fuente y lo traduce a un código máquina comprensible para la máquina ORGA1. Para esto, realiza los siguientes pasos:
 - Resuelve las directivas dirigidas al Ensamblador.
 - Calcula el tamaño de cada una de las instrucciones, y en base a ello resuelve los valores de las etiquetas.

- Programación: Se escribe un algoritmo usando las instrucciones en lenguaje ensamblador (o assembly) provistas por el set de instrucciones de la Arquitectura. Esto da origen a un código fuente.
- Ensamblado: Un programa llamado Ensamblador toma el código fuente y lo traduce a un código máquina comprensible para la máquina ORGA1. Para esto, realiza los siguientes pasos:
 - Resuelve las directivas dirigidas al Ensamblador.
 - Calcula el tamaño de cada una de las instrucciones, y en base a ello resuelve los valores de las etiquetas.
 - Traduce todas las instrucciones a 0s y 1s guiándose en el Formato de Instrucción.

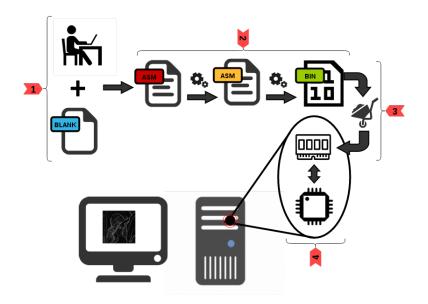


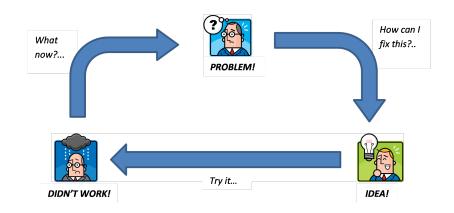
Ciclo de Vida de un Programa (cont.)

 Carga: Se le indica al Ensamblador una dirección inicial, y éste copia el código máquina en la Memoria desde esa posición en adelante. Luego, carga esa misma dirección en el PC (Program Counter).

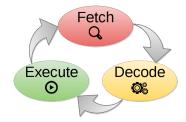
Ciclo de Vida de un Programa (cont.)

- Carga: Se le indica al Ensamblador una dirección inicial, y éste copia el código máquina en la Memoria desde esa posición en adelante. Luego, carga esa misma dirección en el PC (Program Counter).
- Ejecución: El CPU da inicio a su ciclo de ejecución comenzando por la posición indicada en el PC.

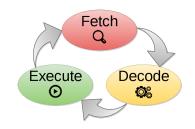




Ciclo de Ejecución

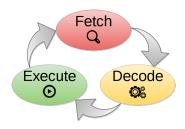


Ciclo de Ejecución



Fetch La UC (Unidad de Control)
obtiene una instrucción de la
posición de Memoria a la que
apunta el PC
y lo incrementa.

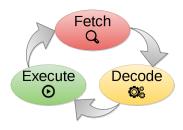
Ciclo de Ejecución



Fetch La UC (Unidad de Control)
obtiene una instrucción de la
posición de Memoria a la que
apunta el PC
y lo incrementa.

Decode La UC decodifica la instrucción.

Ciclo de Ejecución



Fetch La UC (Unidad de Control)
obtiene una instrucción de la
posición de Memoria a la que
apunta el PC
y lo incrementa.

Decode La UC decodifica la instrucción.

Execute La **UC** ejecuta la instrucción.

Ciclo de Ejecución (un poco más en detalle)

- La UC obtiene la primer palabra de la instrucción en Memoria (a partir de la dirección apuntada por el PC) e incrementa el PC.
- La UC decodifica la primer palabra de la instrucción.
 Si es necesario: busca más palabras de la instrucción usando el PC e incrementándolo cada vez.
- 3. La **UC** decodifica la instrucción completa.
- 4. La **UC** ejecuta la instrucción.
- 5. Ir a Paso 1

De acá en adelante van a escuchar muchos términos parecidos como ser: assembler, ensamblador, assembly, lenguaje ensamblador, programa ensamblador, etc.

De acá en adelante van a escuchar muchos términos parecidos como ser: assembler, ensamblador, assembly, lenguaje ensamblador, programa ensamblador, etc.

Lenguaje Ensamblador, Assembly y Assembler son términos que se utilizan para denotar al lenguaje de programación cuya sintaxis se compone del Set de Instrucciones de la Arquitectura de la máquina en la que se trabaje.

De acá en adelante van a escuchar muchos términos parecidos como ser: assembler, ensamblador, assembly, lenguaje ensamblador, programa ensamblador, etc.

Lenguaje Ensamblador, Assembly y Assembler son términos que se utilizan para denotar al lenguaje de programación cuya sintaxis se compone del Set de Instrucciones de la Arquitectura de la máquina en la que se trabaje.

Programa Ensamblador y Assembler son términos que se utilizan para denotar al programa que traduce códigos fuente (escritos en Assembly) en código máquina.

De acá en adelante van a escuchar muchos términos parecidos como ser: assembler, ensamblador, assembly, lenguaje ensamblador, programa ensamblador, etc.

Lenguaje Ensamblador, Assembly y Assembler son términos que se utilizan para denotar al lenguaje de programación cuya sintaxis se compone del Set de Instrucciones de la Arquitectura de la máquina en la que se trabaje.

Programa Ensamblador y Assembler son términos que se utilizan para denotar al programa que traduce códigos fuente (escritos en Assembly) en código máquina.

Como ven, el término anglosajón **Assembler** es válido para ambas acepciones. Para evitar confusiones, cuando utilicemos los términos en Inglés lo haremos de la siguiente forma:

Assembly ← Lenguaje Ensamblador Assembler ← Programa Ensamblador



Ejercicios

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Tenemos que:

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Tenemos que:

 ver cuántas palabras necesita cada instrucción

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, [[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Tenemos que:

- ver cuántas palabras necesita cada instrucción
- calcular los valores de las etiquetas

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, [[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Tenemos que:

- ver cuántas palabras necesita cada instrucción
- calcular los valores de las etiquetas

DW (Define Word): directiva al ensamblador que provoca que en la posición de memoria que le corresponde, aparezca el valor indicado.

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

dirección ocupa

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

dirección ocupa

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

0×0000

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

 0×0000 dos palabras

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

 0×0000 dos palabras

0×0002

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

 0×0000 dos palabras

 0×0002 dos palabras

0x0004

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

0x0004 dos palabras

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

0x0000 dos palabras

 0×0002 dos palabras

 0×0004 dos palabras

0×0006

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

0x0000 dos palabras

 0×0002 dos palabras

 0×0004 dos palabras

0x0006 una palabra

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

 0×0000 dos palabras

0x0002 dos palabras

 0×0004 dos palabras

 0×0006 una palabra

0×0007

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1,[[et1]]

ADD [R1],0x6000

JMP main

et1: DW 0x0007

et2: DW 0x0004

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

0x0004 dos palabras

0x0006 una palabra

0x0007 una palabra

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R1,[[et1]]
	ADD [R1],0x6000
	JMP main

et1: DW 0x0007 et2: DW 0x0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R1,[[et1]]		
	ADD [R1],0x6000		
	JMP main		
et1:	DW 0x0007		
et2:	DW 0×0004		

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

La etiqueta main corresponde a la dirección de memoria 0x0000. La etiqueta et1 corresponde a la dirección de memoria 0x0006. La etiqueta et2 corresponde a la dirección de memoria 0x0007.

Reemplazando las etiquetas por valores de las direcciones estamos listos para codificar.



Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0×0006]]
ADD [R1],0×6000
JMP 0x0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0x0006]]
ADD [R1],0×6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:								
0000	0001	0002	0003	0004	0005	0006	0007	





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

	Traducidas las etiquetas lo codificamos y cargamos en memoria:							
	0000	0001	0002	0003	0004	0005	0006	0007
Ì								





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0x0006]]
ADD [R1],0×6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:							
0000	0001	0002	0003	0004	0005	0006	0007
1858	0006						





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Tradu	icidas las etic	iuetas lo cod	lificamos v ca	rgamos en m	emoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006						





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0×0006]]
ADD [R1],0×6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000				





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

ADD [R1],0×6000
ADD [KI],0x0000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000				





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

ADD [R1],0x6000 JMP 0x0000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000	A000	0000		





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

ADD [R1],0x6000 JMP 0x0000 DW 0x0007 DW 0x0004	MOV R1,[[0×0006]]
DW 0×0007	ADD [R1],0×6000
211 0/10001	JMP 0×0000
DW 0×0004	DW 0×0007
	DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000	A000	0000		





Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0x0006]]
ADD [R1],0×6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000	A000	0000	0007	





Ejercicio 1 - Ensamblando

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0x0006]]
ADD [R1],0×6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000	A000	0000	0007	

Y también cargamos la dirección inicial en el PC:





Ejercicio 1 - Ensamblando

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0x0006]]
ADD [R1],0×6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000	A000	0000	0007	0004

Y también cargamos la dirección inicial en el PC:





Ejercicio 1 - Ensamblando

Codifiquemos el siguiente programa y carguémoslo desde la posición de memoria 0x0000:

MOV R1,[[0×0006]]
ADD [R1],0x6000
JMP 0×0000
DW 0×0007
DW 0×0004

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	dos palabras
0×0006	una palabra
0×0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000							
1858	0006	2C40	6000	A000	0000	0007	0004

Y también cargamos la dirección inicial en el PC:





Empezamos a hacer el ciclo **Fetch-Decode-Execute** con el PC = 0x0000.

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006	2C40	6000	A000	0000	0007	0004

Ejecución

Ejecución

Ejecución

alores	P	C S	SP		R0	R1		R2	R3	R4	R5	R6	R7		Z	C	V	N
iciales	00	000 F	FEF		0000	00	00	0000	0000	0000	0000	000	0000		0	0	0	0
-	-0	+1	+2	+	-3 -	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
00 18	358	0006	2C40	60	00 A	000	0000	0007	0004	0000	0000	0000	0000	0000	00	00	0000	0000
10 00	000	0000	0000	00	00 00	000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
PC	:	SP	[SP+1]	IR	Instr	ucciór	1 - 1er p	alabra	PC	2da palabra	PC	3ra palabra	PC				
	00 18 10 00	+0 00 1858	+0	+0	+0	1000 FFEF 0000	+0	+0	+0	+0	+0	1000 1000	1000 1000	1000 1000	1000 1000	10	10	10

Flags

Flags

Flags

Ejecución

Ejecución

Ejecución

3

Va	alores F	PC S	SP	R0	R1	. F	R2	R3	R4	R5	R6	R7		Z C	V	N
ini	iciales 🛛 C	000 F	FEF	0000	00	00 C	0000	0000	0000	0000	0000	0000		0 0	0	0
+0												+F				
00	00 1858	0006	2C40 6	5000 A	4000	0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	Instr	rucción	- 1er pa	alabra	PC	2da palabra	PC	3ra palabra	PC	1	nstrucció ecodifica	
4	0000		1	1									1	1		-

Flags

Flags

Flags

FFEF 0000

0000 Ejecución

3

Ejecución

Ejecución

Valo	ores P	C	SP	R0	R:	1	R2	R3	R4	R5	R6	R7		Z	C	V	N
inic	iales 0	000 F	FEF	000	0 00	000	0000	0000	0000	0000	000	0000		0	0	0	0
	+0	. 1	12	+3	1.4	1	16	1 . 7	+8	+9	Ι.Λ	+B			D	ı E	
	+0	+1	+2	+3	+4	+5	+0	+1	+0	+9	+A	+6	+C	+	U	+=	+F
0000	1858	0006	2C40	6000	A000	0000	0007	0004	0000	0000	0000	0000	0000	00	00	0000	0000
0010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
	PC	SP	[SP+1]	ı IR	Inst	rucción	1 - 1er p	alahra	PC	2da	PC	3ra	PC			strucci	
		J1	[5, +1	, "`	11130	Lucciói	i Tei b	aiabia		palabra		palabra	1.0		dec	codific	ada

Flags

Flags

Flags

Valo	res P	C 5	SP	-	₹0	R1	. 1	₹2	R3	R4	R5	R6	R/		Z	C	V	N
inicia	ales 0	000 F	FEF	0	0000	00	00 (0000	0000	0000	0000	0000	0000		0	0	0	0
	+0	+1	+2	+3	3 -	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
0000	1858	0006	2C40	600	0 A	000	0000	0007	0004	0000	0000	0000	0000	0000	00	00	0000	0000
0010	0000	0000	0000	000	0 00	000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
					-													
	PC	SP	[SP+1	.]	IR	Instr	ucción	- 1er p	alabra	PC	2da	PC	3ra	PC			strucci	

Valor	es F	PC	SP		R0	R1	.	R2	R3	R4	R5	R6	R7		Z	C	V	N
inicia	les 0	000	FFEF	Ì	0000	00	00	0000	0000	0000	0000	0000	000	0	0	0	0	0
				,								•						
	+0	+1	+2	-	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
0000	1858	0006	2C40	6	000 A	4000	0000	0007	7 0004	0000	0000	0000	0000	0000	00	000	0000	0000
0010	0000	0000	0000	0	000 C	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	000	0000	0000
	D.C.	65	ren .	.,		١		_		5.0	2da	5.0	3ra		.	Ins	strucci	ón

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucció decodifica	
1	0000	FFEF	0000	1858	0001 1000 0101 1000							
	Ejecución										Flags	
2											•	
	Ejecución										Flags	
3												
	Ejecución										Flags	

Valor	es	PC	SP		R0	R1	. [R2	R3	R4	R5	R6	R7		Z		N
inicia	les [0000	FFEF] [0000	00	00 (0000	0000	0000	0000	0000	000	00	0 (0	0
	+0	+1	+2	+	-3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	1858	0006	2C40	60	000 A	٥٥٥٧	0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
0010	0000	0000	0000	00	000 0	000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC		stru codit		
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001								
	Ejecución										Flags		\Box	
2														
	Ejecución										Flags		П	
3														
	Ejecución										Flags			

Valor	es F	C	SP	RO	R:	1 F	R2	R3	R4	R5	R6	R7		Z	V	N
inicia	les 0	000	FFEF	00	00 00	000 0	000	0000	0000	0000	0000	000	00	0 0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	1858	0006	2C40	6000	A000	0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
0010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC		struc codif	
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002					
	Ejecución										Flags		
2													
	Ejecución										Flags		
3													
	Ejecución										Flags		

Valor	es	PC	SP] [R0	R1		R2	R3	R4	R5	R6	R7		Z	С	V	N
inicia	les [0000	FFEF] [000	00 00	00	0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	-	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+0)	+E	+F
0000	1858	0000	2C40	6	000	A000	0000	0007	0004	0000	0000	0000	0000	0000	000	0	0000	0000
0010	0000	0000	0000	0	000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Valor	es	PC	SP] [R0	R1		R2	R3	R4	R5	R6	R7		Ζ	С	V	N
inicia	les [0000	FFEF		000	00 00	00	0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	-	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+1	D	+E	+F
0000	1858	0006	2C40	6	000	A000	0000	0007	0004	0000	0000	0000	0000	0000	000	00	0000	0000
0010	0000	0000	0000	0	000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	00	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Valor	es	PC	SP]	R0	R1		R2	R3	R4	R5	R6	R7		Ζ	С	V	N
inicia	les 🛭	0000	FFEF		000	00 00	00	0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2		+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
0000	1858	0006	2C40	6	000	A000	0000	0007	0004	0000	0000	0000	0000	0000	00	00	0000	0000
0010	0000	0000	0000	0	000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002										
	Ejecución										Flags
3											
	Ejecución										Flags

Valor	es	PC	SP] [R0	R1		R2	R3	R4	R5	R6	R7		Z	С	V	N
inicia	les [0000	FFEF] [000	00 00	00	0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	-	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+0)	+E	+F
0000	1858	0000	2C40	6	000	A000	0000	0007	0004	0000	0000	0000	0000	0000	000	0	0000	0000
0010	0000	0000	0000	0	000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC		icción ificada	
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1,	[[0x000)6]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags		
2	0002			2C40									
	Ejecución										Flags		П
3													
	Ejecución										Flags		\Box

Valor	es	PC	SP		R0	R1	.	R2	R3	R4	R5	R6	R7		Z	С	V	N
inicia	les [0000	FFEF		000	00 00	00 (0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2		+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+1	D	+E	+F
0000	1858	0000	5 2C4	1 6	5000	A000	0000	0007	0004	0000	0000	0000	0000	0000	000	00	0000	0000
0010	0000	0000	0000	(0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	00	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000						
	Ejecución										Flags
3											
	Ejecución										Flags

Valor	es	PC	SP] [R0	R1		R2	R3	R4	R5	R6	R7		Z	С	V	N
inicia	les [0000	FFEF] [000	00 00	00	0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	-	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+0)	+E	+F
0000	1858	0000	2C40	6	000	A000	0000	0007	0004	0000	0000	0000	0000	0000	000	0	0000	0000
0010	0000	0000	0000	0	000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instru decod	icción ificada	
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1,	[[0×0006]]	
	Ejecución	R1=[[0	×0006]]=	0×0007]	=4						Flags		٦
2	0002			2C40	0010 1100 0100 0000	0003							\neg
	Ejecución										Flags		
3													\neg
	Ejecución										Flags		\neg

Valor	es	PC	SP		R0	R1	.	R2	R3	R4	R5	R6	R7		Z	V	N
inicia	les [0000	FFE		000	00 00	00 (0000	0000	0000	0000	0000	000	00	0 0	0	0
	١.0	1	1.0		1.2		1.5	1.6	1.7	1.0	١.0	ι Λ	+B	1.0	I D	E	ı E
	+0	+1	+4		+3	+4	+5	+6	+7	+8	+9	+A	+6	+C	+D	+=	+-
0000	1858	0006	5 2C4	0 6	5000	A000	0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
0010	0000	0000	000	0 (0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC		rucci difica		
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1	, [[0:	×000	6]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags			\Box
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			·			
	Ejecución										Flags			
3														
	Ejecución										Flags			П

Valor	es l	PC	SP	F	R0	R1	R	2	R3	R4	R5	R6	R7		Z		N
inicia	les (0000	FFEF		0000	000	0 0	000	0000	0000	0000	0000	000	00	0 (0	0
	+0	+1	+2	+3	+	4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	1858	0006	2C40	600	0 A0	00 (0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
0010	0000	0000	0000	000	0 000	00 (0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución										Flags
3											
	Ejecución										Flags

Valor	es F	C	SP		R0	R1	F	R2	R3	R4	R5	R6	R7		Ζ	C	V	N
inicia	les 0	0000	FFEF		0000	00	00 0	000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	+3	3 +	-4	+5	+6	+7	+8	+9	+A	+B	+C	+1	D	+E	+F
0000	1858	0006	2C40	600	0 AC	000	0000	0007	0004	0000	0000	0000	0000	0000	000	00	0000	0000
0010	0000	0000	0000	000	00	00	0000	0000	0000	0000	0000	0000	0000	0000	000	00	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x000	4]=0×A00	$0+0\times600$	00=0×0000						Flags
3											
	Ejecución						Flags				

Valor	es l	PC	SP	R) R	1 F	R2	R3	R4	R5	R6	R7		Z	V	N
inicia	les (0000	FFEF	00	00 00	000	0000	0000	0000	0000	0000	000	00	0 (0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	1858	0006	2C40	6000	0000	0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
0010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x000	4]=0×A00	$0+0\times600$	00=0×0000						Flags 1 1 0 0
3											
	Ejecución										Flags

Valor	es	PC	SP	F	R0	R1	F	R2	R3	R4	R5	R6	R7		Z	С	V	N
inicia	les 🛭	0000	FFEF		0000	00	00 (0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	+3	3 +	-4	+5	+6	+7	+8	+9	+A	+B	+C	+E)	+E	+F
0000	1858	0006	2C40	600	0 00	000	0000	0007	0004	0000	0000	0000	0000	0000	000	0	0000	0000
0010	0000	0000	0000	000	0 00	000	0000	0000	0000	0000	0000	0000	0000	0000	000	0	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004	4]=0×A00	$0+0\times600$	00=0×0000						Flags 1 1 0 0
3	0004										
	Ejecución										Flags

Valor	es	PC	SP	F	R0	R1	F	R2	R3	R4	R5	R6	R7		Z	C V	N	
inicia	les 🗌	0000	FFEF		0000	000	0 0	000	0000	0000	0000	0000	000	00	0	0 0	0	
	+0	+1	+2	+3	+	4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+		+F
0000	1858	0006	2C40	600	0 00	00	0000	0007	0004	0000	0000	0000	0000	0000	0000	000	0 0	0000
0010	0000	0000	0000	000	0 00	00	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0 0	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004	4]=0×A00	$0+0\times600$	00=0×0000						Flags 1 1 0 0
3	0004			0000							
	Ejecución						•				Flags

Valor	es l	PC	SP	R) R	1 F	R2	R3	R4	R5	R6	R7		Z	V	N
inicia	les (0000	FFEF	00	00 00	000	0000	0000	0000	0000	0000	000	00	0 (0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	1858	0006	2C40	6000	0000	0000	0007	0004	0000	0000	0000	0000	0000	0000	0000	0000
0010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Г	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da	PC	3ra	PC	Instrucci	
		-	1				palabra		palabra		decodifica	ida
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x	:0006]]
	Ejecución	R1=[[0)×0006]]=	0×0007]	=4						Flags	
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6	000
	Ejecución	[0x000	4]=0×A00	0+0×60	00=0×0000						Flags 1 1	0 0
3	0004			0000	0000 0000 0000 0000							
	Ejecución										Flags	

Valor	es	PC	SP	F	R0	R1	F	R2	R3	R4	R5	R6	R7		Z	C V	N	
inicia	les 🗌	0000	FFEF		0000	000	0 0	000	0000	0000	0000	0000	000	00	0	0 0	0	
	+0	+1	+2	+3	+	4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+		+F
0000	1858	0006	2C40	600	0 00	00	0000	0007	0004	0000	0000	0000	0000	0000	0000	000	0 0	0000
0010	0000	0000	0000	000	0 00	00	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0 0	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	×0006]]=[0×0007]	=4						Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004	4]=0×A00	0+0×600	00=0×0000						Flags 1 1 0 0
3	0004			0000	0000 0000 0000 0000						
	Ejecución										Flags

Valor	es	PC	SP		R0	R1	. [R2	R3	R4	R5	R6	R7		Z	C V	N	
inicia	les 🛭	0000	FFEF		0000	00	00 (0000	0000	0000	0000	0000	000	00	0	0 0	0	
	+0	+1	+2	+	3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+1		+F
0000	1858	0006	2C40	600	00 0	0000	0000	0007	0004	0000	0000	0000	0000	0000	0000	000	0 0	0000
0010	0000	0000	0000	000	00 0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0 0	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	R1=[[0x0006]]=[0x0007]=4								
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004	4]=0×A00	$0+0\times600$	00=0×0000						Flags 1 1 0 0
3	0004			0000	0000 0000 0000 0000	0005					
	Ejecución										Flags

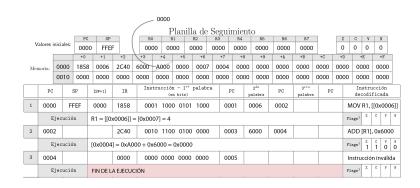
Valor	es	PC	SP	F	R0	R1	F	R2	R3	R4	R5	R6	R7		Ζ	С	V	N
inicia	les 🛭	0000	FFEF		0000	00	00 (0000	0000	0000	0000	0000	000	00	0	0	0	0
	+0	+1	+2	+3	3 +	-4	+5	+6	+7	+8	+9	+A	+B	+C	+D)	+E	+F
0000	1858	0006	2C40	600	0 00	000	0000	0007	0004	0000	0000	0000	0000	0000	000	0	0000	0000
0010	0000	0000	0000	000	0 00	000	0000	0000	0000	0000	0000	0000	0000	0000	000	0	0000	0000

DC.	CD	[SP+1]	ID	Instrucción Ior palabra	DC.	2da	DC.	3ra	DC.	Instrucción		
r C	31		"	ilistruccion - 1er palabra	' `	palabra	r C	palabra	1	decodificada		
0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]		
Ejecución	R1=[[0	×0006]]=[Flags									
0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000		
Ejecución	[0x000	4]=0×A00	0+0×60	00=0×0000						Flags 1 1 0 0		
0004			0000	0000 0000 0000 0000	0000 0000 0000 0000 0005 1							
Ejecución					•					Flags		
	Ejecución 0002 Ejecución 0004	0000 FFEF Ejecución R1=[[0 0002 Ejecución [0×000- 0004	0000 FFEF 0000 Ejecución R1=[[0x0006]]= 0002 Ejecución [0x0004]=0xA00	0000 FFEF 0000 1858 Ejecución R1=[[0x0006]]=[0x0007] 0002 2C40 Ejecución [0x0004]=0xA000+0x600 0000	0000 FFEF 0000 1858 0001 1000 0101 1000 Ejecución R1=[[0x0006]]=[0x0007]=4 0002 2C40 0010 1100 0100 0000 Ejecución [0x0004]=0xA000+0x6000=0x0000 0004 0000 0000 0000 0000 0000 0000	0000 FFEF 0000 1858 0001 1000 0101 1000 0001 Ejecución R1=[[0x0006]]=[0x0007]=4 0002 2C40 0010 1100 0100 0000 0003 Ejecución [0x0004]=0xA000+0x6000=0x0000 0000 0000 0000 0000 0000 0000 0005	PC SP SP+1 IR Instrucción - 1er palabra PC palabra	PC SP SP+1 IR Instrucción - 1er palabra PC palabra PC	PC SP SP+1 IR Instrucción - 1er palabra PC palabra	PC SP SP+1 IR Instrucción - 1er palabra PC palabra		

Valor	es	PC	SP		R0	R1	. [R2	R3	R4	R5	R6	R7		Z	C V	N	
inicia	les 🛭	0000	FFEF		0000	00	00 (0000	0000	0000	0000	0000	000	00	0	0 0	0	
	+0	+1	+2	+	3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+1		+F
0000	1858	0006	2C40	600	00 0	0000	0000	0007	0004	0000	0000	0000	0000	0000	0000	000	0 0	0000
0010	0000	0000	0000	000	00 0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	000	0 0	0000

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0	R1=[[0x0006]]=[0x0007]=4								Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004	4]=0×A00	0+0×600	00=0×0000						Flags 1 1 0 0
3	0004			0000	0000 0000 0000 0000	0005					Instrucción Inválida
	Ejecución	FIN DE	LA EJE		Flags						

Ejercicio 1 - Solución



Ejercicio 1 - Observaciones ¿Qué pasó con el programa anterior?

Ejercicio 1 - Observaciones

¿Qué pasó con el programa anterior?

En la ejecución del programa cambió una posición de memoria donde estaba cargado... **el mismo programa!**

Ejercicio 1 - Observaciones

¿Qué pasó con el programa anterior?

En la ejecución del programa cambió una posición de memoria donde estaba cargado... **el mismo programa!**

Eso hizo que nuestro programa se modificara, haciendo que la máquina llegase a una instrucción inválida.

Ejercicio 1 - Observaciones

¿Qué pasó con el programa anterior?

En la ejecución del programa cambió una posición de memoria donde estaba cargado... **el mismo programa!**

Eso hizo que nuestro programa se modificara, haciendo que la máquina llegase a una instrucción inválida.

Moraleja:



Programen con cuidado y atención.



Ejercicio 2 - Ensamblando

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

Ejercicio 2 - Ensamblando

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

0×0000

Ejercicio 2 - Ensamblando

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

0x0000 dos palabras

 0×0002

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

0x0002 dos palabras

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

 $0x0002 \quad {\rm dos\ palabras}$

0×0004

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

 0×0002 dos palabras

0x0004 una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

 $0x0002 \quad {\rm dos\ palabras}$

0x0004 una palabra

0×0005

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

 $0x0002 \quad {\rm dos\ palabras}$

0x0004 una palabra

0x0005 una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

 0×0002 dos palabras

0x0004 una palabra

0x0005 una palabra

0×0006

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

0x0004 una palabra

0x0005 una palabra

0x0006 una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R7, [0x0004]

CALL subs

DW 0x0FE0

subs: SUB R6, R7

RET

dirección ocupa

 0×0000 dos palabras

 0×0002 dos palabras

0x0004 una palabra

 0×0005 una palabra

 0×0006 una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R7, [0×0004]		
	CALL subs		
	DW 0×0FE0		
subs:	SUB R6, R7		
	RET		

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

La etiqueta main corresponde a la dirección de memoria 0x0000. La etiqueta subs corresponde a la dirección de memoria 0x0005.

Reemplazando las etiquetas por valores estamos listos para codificar.



Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

	dirección	ocupa
MOV R7, [0x0004]	0×0000	dos palabras
CALL 0×0005	0×0002	dos palabras
DW 0x0FE0	0×0004	una palabra
SUB R6, R7	0×0005	una palabra
RET	0×0006	una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0×0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
direction	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006





MOV R7, [0×0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
direction	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0×0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004					





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0×0004]
CALL 0×0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004					





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0×0004]
CALL 0×0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra
0×0004 0×0005	una palabra una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005			





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV	R7	[0×0004]
IVIOV	$I \times I$,	JUXUUU 4

CALL 0x0005

DW 0x0FE0

SUB R6, R7

RET

dirección	ocupa
-----------	-------

0x0000 dos palabras

 0×0002 dos palabras

0x0004 una palabra

 0×0005 una palabra

0x0006 una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005			





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0×0004]
CALL 0×0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005	0FE0		





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV I	R7. [[0×0004]
IVIOVI	`', [UNUUUT

CALL 0x0005

DW 0x0FE0

SUB R6, R7

RET

ра

 0×0000 dos palabras

0x0002 dos palabras

0x0004 una palabra

0x0005 una palabra

0x0006 una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005	0FE0		





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0x0004
CALL 0×0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005	0FE0	39A7	





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0x0004	MOV	R7,	[0×0004
-----------------	-----	-----	---------

CALL 0x0005

DW 0x0FE0

SUB R6, R7

RET

dirección	ocupa
direction	ocupa

0x0000 dos palabras

 0×0002 dos palabras

0x0004 una palabra

0x0005 una palabra

0x0006 una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005	0FE0	39A7	





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0x0004]
CALL 0x0005
DW 0×0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005	0FE0	39A7	C000





Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

MOV R7, [0×0004
CALL 0×0005
DW 0x0FE0
SUB R6, R7
RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006
19C8	0004	B000	0005	0FE0	39A7	C000





Empezamos a hacer el ciclo **Fetch-Decode-Execute** con el **PC** = 0x0000.

0000	0001	0002	0003	0004	0005	0006	0007
19C8	0004	B000	0005	0FE0	39A7	C000	0000

SP

R0

R1

R2

Valores

Ejecución

Ejecución

Ejecución

Ejecución

Ejecución

2

3

4

5

ini	ciales 0	000 F	FEF	000	0 00	00 00	000	0000	0000	0000	0000	0000)	0 0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
001	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	Inst	rucción -	- 1er pa	alabra	PC	2da palabra	PC	3ra palabra	PC		strucció codifica	- 1
1																

R3

R4

R5

R6

R7

Flags

Flags

Flags

Flags

SP

FFEF

R0

0000

R1

0000

R2

0000

PC

0000

Valores

iniciales

Ejecución

Ejecución

5

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]] IR	Inst	Instrucción - 1er palabra				2da palabra	PC	3ra palabra	PC		strucció	
1	0000															
	Ejecución													Flags		
2																
	Ejecución			Ī									•	Flags		
3																
	Ejecución			Ī									•	Flags		
4																

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

0

0

Ν

Flags

SP

R0

R1

0000

R2

PC

Valores

	ciaics _o	100			00	00 00	700 0	1000	0000	0000	0000	, 000		0 0	1 0 1	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00		0004		0005	0FE0	39A7	C000	0000	0000		0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
		I		Т						2da		3ra		In	strucció	in l
	PC	SP	[SP+1]	IR	Insti	rucción -	1er pal	abra	PC	palabra	PC	palabra	PC		codifica	
1	0000	FFEF	0000													
	Ejecución								•					Flags		
2																
	Ejecución													Flags		
3																
	Ejecución													Flags		
4																
	Ejecución													Flags		
5																
	Ejecución													Flags		

R3

0000 0000

R4

0000

R6

R7

SP

FFEF

R0

0000

R1

0000

R2

0000

PC

0000

Valores

iniciales

4

5

Ejecución

Ejecución

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	Inst	Instrucción - 1er palabra			PC	2da palabra	PC	3ra palabra	PC		strucció codifica	
1	0000	FFEF	0000	19CI	3											
	Ejecución													Flags		
2																
	Ejecución			•									•	Flags		
3																
	Ejecución			-				-					•	Flags		

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

0

0

Ν

Flags

SP

FFEF

R0

0000

R1

0000

R2

0000

PC

0000

Valores

iniciales

4

5

Ejecución

Ejecución

		•								•	•					
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
	PC.	SP	[SP+1]	IR	Inct	Instrucción - 1er palabra				2da	PC.	3ra	PC	In	strucció	in
			[31 +1]			Instrucción - 1er palabra			PC	palabra	'	palabra	1 0	de	codifica	da
1	0000	FFEF	0000	19C	3 000	1 1001 1	.100 100	10								
	Ejecución												•	Flags		
2																
	Ejecución												•	Flags		
3																
	Ejecución							-	•				•	Flags		

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

0

0

Ν

Flags

SP

0000 FFEF

R0

0000

R1

0000

R2

0000

PC

Valores

iniciales

5

Ejecución

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
				_								_				
	PC	SP	[SP+1]	IR	Inst	rucción -	- 1er pal	abra	PC	2da	PC	3ra palabra	PC		strucció codifica	
			1 .			•				palabra		paiabra		ae	codifica	aa
1	0000	FFEF	0000	19C	B 000	1 1001 1	1100 100	0	0001							
	Ejecución													Flags		
2																
	Ejecución				•								•	Flags		
3																
	Ejecución		•											Flags		
4																
	Eiecución													Flags		

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

SP

R0

R1

R2

PC

Valores

Ejecución

Ejecución

Ejecución

Ejecución

3

5

ini	ciales 0	000 F	-FEF	0000	00 0	00 00	000 0	0000	0000	0000	0000	000	0	0 0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0 0000	0000	0000	0000	0000	0000	0000
003	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
$\overline{}$				1	_					0.1		2	_			
	PC	SP	[SP+1]	IR	Inst	rucción -	- 1er pa	labra	PC	2da	PC	3ra	PC		strucció	
										palabra		palabra		de	codifica	da
1	0000	FFEF	0000	19CE	000	1 1001 1	1100 100	00	0001	0004	0002					
	Ejecución		•					•						Flags		
2																

R3

R4

R5

R6

Flags

Flags

Flags

SP

R0

R1

0000

R2

PC

Valores

5

Ejecución

	Cidicso				,0 00	00 0	000 0	,000	0000	0000	0000	, 000		0 0	1 0 1	•
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
									•							
	PC.	SP	[SP+1]	l IR	Inst	rucción -	- 1er pa	labra	PC	2da	PC	3ra	PC		strucci	
			[Instrucción - 1er palabra				palabra		palabra	1	de	codifica	ıda
1	0000	FFEF	0000	19C	B 000	0001 1001 1100 1000				0004	0002			MOV	R7, [0x	0004]
	Ejecución				·									Flags		
2																
	Ejecución				·				•					Flags		
3																
	Ejecución				·				•					Flags		
4																
	Ejecución													Flags		

R3

0000 0000

R4

0000

R5

R6

R7

in	iciales	0000	0 F	FEF	000	00 00	000 0	000	0000	0000	0000	0000	000	0	0	0	0	0
	+	0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
00	00 190	CB 0	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
00	10 000	0 0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	000	0000	0000
	PC		SP	[SP+1]	IR	Inst	Instrucción - 1er palabra				2da palabra	PC	S 3ra PC palabra		decodificada			
1	0000	F	FEF	0000	19C	B 000	0001 1001 1100 1000				0004	0002			M	OV F	R7, [0:	<0004]
	Ejecuci	ón R	R7=[0:	$\times 0004] = 0$	×0FE0										Fla	ags		
2																		
	Ejecuci	ón													Fla	ags		
3																		
	Ejecuci	ón													Fla	ags		
4																		
	Ejecuci	ón				•									Fla	ags		
5																		
	Ejecuci	ón		-	•	-									Fla	ags		T

SP

0000 FFEF

R0

0000

R1

0000

R2

0000

PC

Valores

iniciales

Ejecución

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	FE0 39A7 C000 0000		0000	0000	0000	0000	0000	0000	0000	0000	
00	10 0000	0000	0000	0000	0000	00 0000 0000 0000		000	0000	0000	0000	0000	0000	0000	0000	
	PC.	PC SP		I ID	Locati	1			PC	2da	PC	3ra	PC	Instrucción		
	PC	3P	[SP+1]	IR	Inst	trucción - 1er palabra			PC	palabra	PC	palabra	PC	decodificada		
1	0000	FFEF	0000	19C	B 000	0001 1001 1100 1000				0004	0002			MOV I	R7, [0x0	0004]
	Ejecución	R7=[0	×0004]=0	×0FE0				•					•	Flags		
2	0002															
	Ejecución												•	Flags		
3																
	Ejecución			-										Flags		
4																
	Ejecución			-										Flags		
5																

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

SP

FFEF

R0

0000

R1

0000

Valores

iniciales

Ejecución

PC

0000

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	E0 39A7 C000 0000		000	0000	0000	0000	0000	0000	0000	0000	
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
	PC	C SP	[SP+1]	IR	Inst	strucción - 1er palabra		ahra	PC	2da	PC	3ra	PC	Instrucción		
						•				palabra		palabra		decodificada		
1	0000	FFEF	0000	19C	3 000	0001 1001 1100 1000				0004	0002			MOV	R7, [0x0	0004]
	Ejecución	R7=[0:	×0004]=0	×0FE0										Flags		
2	0002			B00)											
	Ejecución		•					•					•	Flags		
3																
	Ejecución		•					•					•	Flags		
4																
	Ejecución													Flags		
5																

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

0

R2

0000

SP

FFEF

R0

0000

R1

0000

R2

0000

PC

0000

Valores

iniciales

5

Ejecución

	_																
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000	
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000	
	PC.	SP [SP+1]		IR	Inst	Instrucción - 1er palabra				2da	PC	3ra	PC	Instrucción			
	- 0		[51 11]			·			PC	palabra		palabra	' `	decodificada			
1	0000	FFEF	0000	19CF	3 000	0001 1001 1100 1000			0001	0004	0002			MOV I	₹7, [0x0	0004]	
	Ejecución	R7=[0	×0004]=0	x0FE0										Flags			
2	0002			B000	101	1 0000 0	000 000	0									
	Ejecución													Flags			
3																	
	Ejecución													Flags			
4															Ť		
	Ejecución													Flags			

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

SP

FFEF

R0

0000

R1

0000

R2

0000

PC

0000

Valores

iniciales

3

4

5

Ejecución

Ejecución

Ejecución

							•			•	•	•				
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	lund		- 1er pal		PC	2da	PC.	3ra	PC	In	strucci	ón
	PC	35	[37+1]	II.	Insi	ruccion -	- 1er pai	abra	PC	palabra	[palabra	PC	de	codifica	ada
1	0000	FFEF	0000	19C	B 000	1 1001 1	100 100	0	0001	0004	0002			MOV	R7, [0x	0004]
	Ejecución	R7=[0	×0004]=0	x0FE0	•									Flags		
2	0002			B00	0 101	1 0000 0	000 000	0	0003							
	Ejecución		•					•					•	Flags		

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

Flags

Flags

SP

R7=[0x0004]=0x0FE0

R0

B000

R1

R2

1011 0000 0000 0000

PC

Valores

Ejecución 2 0002

Ejecución

Ejecución

Ejecución

Ejecución

3

4

5

ini	ciales 0	000 F	FEF	0000	00	00 0	000	0000	0000	0000	0000	0000		0	0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
00	00 19CB	0004	B000	0005	FE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
			•	·										•			
_					_												
	PC	SP	[SP+1]	IR	Instr	rucción	- 1er p	alabra	PC	2da	PC	3ra	PC			struccio	
		-	[0. 1-]							palabra		palabra			ded	codifica	da
1	0000	FFEF	0000	19CB	0001	1 1001 :	1100 10	000	0001	0004	0002			MC)V F	R7, [0x0	0004

R3

R4

0003 0005

R5

R6

R7

VN

Flags

Flags

Flags

Flags

Flags

◆□▶ ◆□▶ ◆□▶ ◆□▶ □ のQで

SP

R7=[0x0004]=0x0FE0

R0

B000

R1

R2

1011 0000 0000 0000

Valores

Ejecución 2 0002

Ejecución

Ejecución

Ejecución

Ejecución

3

4

5

PC

	ini	ciales 0	000 F	FEF	0000	00 00	00 00	000 0	000	0000	0000	0000	0000		0 0	0	0
		+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
	000	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
Ī	001	.0 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
		PC.	SP	[SP+1]	IR	Inst	rucción .	- 1er pala	ahra	PC	2da	PC	3ra	PC		strucció	
l			٥.	[51 1]	"`	50	raccion	ici paic	JUIG	٠.	palabra		palabra	١.		codifica	
	1	0000	FFEF	0000	19CE	000	1 1001 1	100 100	0	0001	0004	0002			MOV I	₹7, [0×0	004]

R3

R4

0003 0005

R5

R6

0004

R7

Flags

Flags

Flags

Flags

Flags

CALL 0x0005

SP

FFEF

R0

0000

R1

0000

R2

0000

Valores

iniciales

Ejecución

PC

														- -		-
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
				,												
	PC.	SP	[SP+1]	IR	Innt		- 1er pala		PC	2da	PC.	3ra	PC	In	strucció	n
	FC	31	[31 +1]	"	IIISU	de	codifica	da								
1	0000	FFEF	0000	19CI	3 000	1 1001 1	100 100	0	0001	0004	0002			MOV	R7, [0x0	004]
	Ejecución	R7=[0	×0004]=0	×0FE0										Flags		
2	0002			B000	0 101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0×FFE	F]=0x00	04 SP=	=0×FFE	E PC=0	×0005							Flags		
3																
	Ejecución													Flags		
4																
	Ejecución													Flags		
5																
-		$\overline{}$		-				_						,		

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

SP

R0

R1

R2

PC

Valores

Ejecución

	iciaicso			000	00	00 00	000 0	000	0000	0000	0000	000		0 0	"	U
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004		0005	0FE0	39A7	C000	0000			0000	0000	0000	0000	0000	0000
	10 0000	0000		0000	0000	0000	0000	0000			0000	0000	0000	0000	0000	0000
									1 000	-						1
	PC.	SP	[SP+1]	ID	R Instrucción - 1er palabra PC 2da PC 3ra PC										strucci	ón
	PC	35	[3F+1]	l IK	Inst	ruccion -	- 1er pai	abra	PC	palabra	PC	palabra	PC	de	codific	ada
1	0000	FFEF	0000	19CE	000	1 1001 1	100 100	0	0001	0004	0002			MOV I	R7, [0x	0004]
	Ejecución	R7=[0	×0004]=0	x0FE0	•						•		•	Flags		
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0×FFE	F]=0x000	04 SP=	0×FFE	E PC=0	×0005							Flags		
3	0005															
	Ejecución													Flags		
4																
	Ejecución													Flags		
г			T	т —												

R3

R4

R6

R7

R1

Valores

ını	iciales 0	J00 F	FEF	000	0 00	00 00	000 0	000	0000	0000	0000	0000	U	0 0	0	0	
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000	
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000	
					_					2.1-		2		1	-411		
	PC	SP	[SP+1]	, balapta balapta decoditicada													
1	0000	FFEF	0000	19CF	· palabra palabra decodificada												
	Ejecución	R7=[0	×0004]=0	x0FE0									1	Flags			
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005		
	Ejecución	[0xFFE	F]=0x000)4 SP=	0×FFE	E PC=0	×0005							Flags			
3	0005	FFEE															
	Ejecución													Flags			
4															·		
	Ejecución													Flags			
5																	
	Ejecución													Flags			

Ejecución

R1

ını	iciales 0)00 F	FEF	000	0 00	00 00	000 0	000	0000	0000	0000	000	0	0 0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000 (0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	Inst	rucción -	1er pal	abra	PC	2da palabra	PC	3ra palabra	PC	ı	struccio codifica	
1	0000	FFEF	0000	19CE	3 000	1 1001 1	100 100	0	0001	0004	0002			MOV	R7, [0x0	0004]
	Ejecución	R7=[0	×0004]=0:	x0FE0										Flags		
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0xFFE	F]=0x000)4 SP=	0×FFE	E PC=0	×0005							Flags		
3	0005	FFEE	0004	39A7	'											
	Ejecución													Flags		
4																
	Ejecución													Flags		
5																

R3

R4

R6

SP

FFEF

R0

0000

R1

0000

R2

0000

PC

0000

Valores

iniciales

Ejecución

	0.0.00				-					1000		000		- -		
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
			I	l	1.			. 1	PC.	2da	PC	3ra	PC	In	strucci	ón
	PC	SP	[SP+1]	IR	Inst	rucción -	1er pal	de	codific	ada						
1	0000	FFEF	0000	19CF	3 000	1 1001 1	100 100	0	0001	0004	0002			MOV	R7, [0x	0004]
	Ejecución	R7=[0:	$\times 0004] = 0$	x0FE0										Flags		
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0×FFE	F]=0x00	04 SP=	0×FFE	E PC=0	×0005							Flags		
3	0005	FFEE	0004	39A	001	1 1001 1	.010 011	1								
	Ejecución		•											Flags		
4																
	Ejecución													Flags		
5																

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

SP

0000 FFEF

R0

0000

R1

0000

R2

0000

PC

Valores

iniciales

5

Ejecución

										-	-	_				
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
	•									•	•					•
- 1		ı		1						2da		3ra		In	strucci	ín.
	PC	SP	[SP+1]	IR	Inst	rucción -	- 1er pal	PC		codifica						
1	0000	FFEF	0000	19C	3 000	1 1001 1	100 100	0	0001	0004	0002			MOV	R7, [0x	0004]
	Ejecución	R7=[0:	×0004]=0	x0FE0										Flags		
2	0002			B00	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0xFFE	F]=0x00	04 SP=	=0×FFE	E PC=0	×0005							Flags		
3	0005	FFEE	0004	39A	7 001	1 1001 1	.010 011	1	0006							
	Ejecución													Flags		
4																
	Ejecución													Flags		

R3

0000

R4

0000

R5

0000

R6

0000

R7

0000

R1

R2

Valores

Ejecución

in	iciales 00	000 F	FEF	000	00 00	00 00	000 0	0000	0000	0000	0000	000	0	0 0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	Inst	rucción -	- 1er pal	abra	PC	2da palabra	PC	3ra palabra	PC		strucció codifica	- 1
1	0000	FFEF	0000	19CI	B 000	1 1001 1	100 100	0	0001	0004	0002			MOV	R7, [0x0	0004]
	Ejecución	R7=[0:	×0004]=0	x0FE0									•	Flags		
2	0002			B000	0 101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0xFFE	F]=0x00	04 SP=	=0×FFE	E PC=0	×0005						•	Flags		
3	0005	FFEE	0004	39A	7 001	1 1001 1	1010 011	1	0006					SUB R	R6, R7	
	Ejecución													Flags		
4																
	Ejecución							·						Flags		
5																

R3

R4

R6

in	iciales	000	0 F	FEF	000	0 00	00 00	000	0000	0000	0000	0000	000	0	0	0	0	0	
	+0)	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+	F
00	00 19C	ВС	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	000	0000	000	00
00	10 0000) (0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	000	0000	000	00
																			_
	PC		SP	[SP+1]	IR	Inst	rucción -	ler pal	abra	PC	2da palabra	PC	3ra palabra	PC			struco codific		
1	0000	F	FEF	0000	19C	3 000	1 1001 1	100 100	00	0001	0004	0002			M	OV F	R7, [0	<0004]	
	Ejecució	n F	R7=[0>	<0004]=C	x0FE0	•									Fla	ags			
2	0002				B00	101	1 0000 0	000 000	00	0003	0005	0004			CA	ALL ()×000	5	
	Ejecució	n [0xFFE	F]=0x00	04 SP=	=0×FFEI	E PC=0	×0005							Fla	ags			
3	0005	F	FEE	0004	39A	7 001	1 1001 1	.010 011	.1	0006					SU	JB R	6, R7		
	Ejecució	n F	R6=0x	0000-0×0	FE0=0	xF020									Fla	ags			
4											, and the second								
	Ejecució	n								•					Fla	ags			
5																			
	Ejecució	in													Fla	ags			ı T

R4

	alores P		FEF	R0	R1			R3 0000	R4 0000	R5 00000	R6 F020	R7	10	Z C 0 0	V 0	N 0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0 0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0 0000	0000	0000	0000	0000	0000	0000
	PC	SP [SP+1] IR Instrucción - 1er palabra PC 2da PC 3ra													strucció	ón
	1.0	31	[31 +1]	"`	11130	i uccioni -	Tel b	alabia	1 C	palabra	' C	palabra	PC	de	codifica	da
1	0000	FFEF	0000	19C	B 000	1 1001 1	100 10	000	0001	0004	0002			MOV	R7, [0x0	0004]
	Ejecución	R7=[0	×0004]=0	x0FE0				•					•	Flags		
2	0002			B00	0 101	1 0000 0	000 00	000	0003	0005	0004			CALL	0×0005	$\overline{}$
	Ejecución	[0×FFE	F]=0x00	04 SP=	=0×FFE	E PC=0	×0005							Flags		
3	0005	FFEE	0004	39A	7 001	1 1001 1	.010 01	111	0006					SUB F	R6, R7	$\overline{}$
	Ejecución	R6=0×	0000-0×0	FE0=0)×F020									Flags	0 1	0 1
4																
_	Ejecución													Flags		
5																
_	Ejecución		1	-	-			-						Flags		

SP

R0

R1

R2

PC

Valores

in	iciales 00	000 F	FEF	0000	00	00 00	000 0	000	0000	0000	F020	000	0	0 0	0		0	
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+	E	+	F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0 0000	0000	0000	0000	0000	000	00	00	00
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	000	00	00	00
				·														_
	PC	SP	[SP+1]	IR	Inst	rucción -	- 1er pal	abra	PC	2da palabra	PC	3ra palabra	PC		stru codi			
1	0000	FFEF	0000	19CE	000	1 1001 1	100 100	0	0001	0004	0002			MOV I	₹7, [0×0	004	
	Ejecución	R7=[0:	×0004]=0	x0FE0	•									Flags				
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×00	05		
	Ejecución	[0xFFE	F]=0x000	04 SP=	0xFFE	E PC=0	×0005							Flags				
3	0005	FFEE	0004	39A7	001	1 1001 1	.010 011	.1	0006					SUB R	6, R	7		
	Ejecución	R6=0×	0000-0x0	FE0=0:	×F020									Flags	0	1	0	1
4	0006																	
	Ejecución													Flags				
5																		
	Ejecución													Flags				

R3

R4

R5

R6

R7

SP

0000 FFEF

R0

0000

R1

0000

R2

0000

PC

Valores

iniciales

0006 Ejecución

Ejecución

5

											1					
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
			·	,												
	PC	SP	[SP+1]	IR	Inst	rucción	- 1er pal	labra	PC	2da palabra	PC	3ra palabra	PC		strucció	
1	0000	FFEF	0000	19CI	3 000	1 1001 1	1100 100	00	0001	0004	0002	paraure.		MOV	R7, [0x0	004]
	Ejecución	R7=[0	×0004]=0	x0FE0				-	-				-	Flags		
2	0002			B000	0 101	1 0000 (000 000	00	0003	0005	0004			CALL	0×0005	
	Ejecución	[0xFFE	F]=0x00	04 SP=	=0×FFE	Flags										
3	0005	FFEE	0004	39A	7 001	1 1001 1	1010 011	11	0006					SUB R	6, R7	
	Ejecución	R6=0×	(0000-0x0	FE0=0	xF020			•						Flags	0 1	0 1

R3

0000

R4

0000

R5

0000

R6

F020

R7

0000

Flags

SP

FFEE 0004

R0

39A7

C000

R6=0x0000-0x0FE0=0xF020

R1

0000

0011 1001 1010 0111

1100 0000 0000 0000

Valores

3 0005

5

0006

Ejecución

Ejecución

Ejecución

PC

""	iciales 0	000 1	I LI	000	0 00	00 00	000 0	000	0000	0000	1 020	000		0 0	0	U	
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E		+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0 0000	0000	0000	0000	0000	000	0	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	000	0	0000
				_	_			-								.,	
	PC.	SP	[SP+1]	l IR	Inst	rucción .	- 1er pal	ahra	PC	2da	PC.	3ra	PC.		struc		
		J.	[5, 11]	"`	""50	raccion	ICI Pai	abia		palabra	' ~	palabra	1, 0	de	codifi	cada	a
1	0000	FFEF	0000	19CE	3 000	1 1001 1	100 100	0	0001	0004	0002			MOV I	R7, [0	x00	04]
	Ejecución	R7=[0:	×0004]=0	×0FE0	•								•	Flags			
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×000)5	
	Ejecución	[0xFFE	F]=0x00	04 SP=	0xFFE	E PC=0	×0005						•	Flags			

R3

0000 0000

R4

0006

R5

R6

0000 0000 F020 0000

R7

SUB R6, R7

Flags

Flags

Flags 0 1

0 1

SP

R0

C000

1100 0000 0000 0000

R6=0x0000-0x0FE0=0xF020

R1

PC

Valores

Ejecución

Ejecución

Ejecución

0006

5

111	iciales 0	100 1	I LI	000	0 00	00 00	000 0	000	0000	0000	1 020) 000	<u> </u>	0 0	U	U
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	000	0000	0000	0000	0000	0000	0000	0000
	D.C.	CD.	[CD . 1]	l ID	Τ	• , ,		. 1	DC	2da	DC	3ra	DC	In	strucció	in
	PC	SP	[SP+1]	IR	Inst	ruccion	- 1er pala	abra	PC	palabra	PC	palabra	PC	de	codifica	da
1	0000	FFEF	0000	19CF	3 000	1 1001 1	1100 100	0	0001	0004	0002			MOV I	R7, [0x0	004]
	Ejecución	R7=[0	×0004]=0	x0FE0										Flags		
2	0002			B000	101	1 0000 0	000 000	0	0003	0005	0004			CALL	0×0005	
	Ejecución	[0xFFE	F]=0x00	04 SP=	=0×FFE	E PC=0	×0005							Flags		
3	0005	FFEE	0004	39A	7 001	1 1001 1	1010 011	1	0006					SUB R	6, R7	

R3

0000

R4

0007

R5

R6

R7

Flags

Flags

Flags 0 1 0 1

R7=[0x0004]=0x0FE0

R6=0x0000-0x0FE0=0xF020

FFEE 0004

B000

39A7

C000

[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005

Ejecución

Ejecución

Ejecución

Ejecución

Ejecución

0006

2 0002

3 0005

5

ini	iciales 0	000 F	FEF	0000	000	0 00	000 0	0000	0000	0000	F020	0000)	0	0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
00	00 19CB	0004	B000	0005	DFE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
	PC	SP	[SP+1]	IR	Instru	cción -	- 1er pal	labra	PC	2da	PC	3ra	PC			strucci	
			["`			pu			palabra		palabra			ded	codifica	ıda
1	0000	FFEF	0000	19CB	0001	1001 1	100 100	00	0001	0004	0002			M	OV F	7, [0xl	0004]

0003 0005

0006

0007

1011 0000 0000 0000

0011 1001 1010 0111

1100 0000 0000 0000

R6

0004

Flags Ox0005

Flags

Flags

Flags

SUB R6, R7

Flags 0 1

0 1

SP

0000 FFEF

R0

R6=0x0000-0x0FE0=0xF020

C000

PC=[0xFFEF]=0x0004 SP=0xFFEF

0000

R1

0000

R2

1100 0000 0000 0000

0000

PC

Valores

iniciales

Ejecución

Ejecución

Ejecución

0006

5

0000 19CB 0004 B000 0005 0FE0 39A7 C000 0000 0000 0000 0000 0000 0000 00						- -													
0000 19CB 0004 B000 0005 0FE0 39A7 C000 0000 <th< th=""><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></th<>																			
PC SP [SP+1] IR Instrucción - 1er palabra PC 2da palabra PC 3ra palabra PC Instrucción decodif 1 0000 FFEF 0000 19CB 0001 1001 1100 1000 0001 0004 0002 MOV R7, [6 Ejecución R7=[0x0004]=0x0FE0 Flags Flags CALL 0x000 Ejecución [0xFFEF]=0x0004 SP=0xFFEE PC=0x0005 Flags Flags	+F	+E	+D	+C	⊦B	+1	+A	+9	+8	-7 ∣	+6	-5	+4	+3	+2	+1	+0		
PC SP [SP+1] IR Instrucción - 1er palabra PC 2da palabra PC palabra	000	0000	0000	0000	000	000	0000	0000	0000	000	000	A7	FEO 3	005 0	B000 (004	19CB	000	00
PC SP SP+1 IR Instrucción - 1er palabra PC palabra PC palabra PC palabra PC decodif	000	0000	0000	0000	000	000	0000	0000	0000	000	000	00	000	0000	0000	000	0000	010	00
PC SP SP+1 IR Instrucción - 1er palabra PC palabra PC palabra PC palabra PC decodif														•	•				
PC SP SP+1 IR Instrucción - Ier palabra PC palabra PC palabra PC palabra PC decodif																			
1 0000 FFEF 0000 19CB 0001 1001 1100 1000 0001 0004 0002 MOV R7, [0]	ión	strucció	In	PC	3ra	3r	PC	2da	-		r nala	ión	Instru	IR	[SP_1]	SP	oc		
Ejecución R7=[0x0004]=0x0FE0 Flags Plags 2 0002 B000 1011 0000 0000 0000 0003 0005 0004 CALL 0x000 Ejecución [0xFFEF]=0x0004 SP=0xFFEE PC=0x0005 Flags Flags				1 C	labra	pala	1 0	alabra	~ p	1	п рата	1011 -	mstruv		[31 +1]	٥'			
2 0002 B000 1011 0000 0000 0000 0003 0005 0004 CALL 0x000 Ejecución [0xFFEF]=0x0004 SP=0xFFEE PC=0x0005 Flags	(0004]	R7, [0x0	MOV F				0002	1004	0 0	0	0 1000	01 11	0001	19CB	0000	FEF	0	000	1
Ejecución [0xFFEF]=0x0004 SP=0xFFEE PC=0x0005 Flags			Flags											:0FE0	0004]=0	7=[0>	cución	Eje	
	5	0×0005	CALL				0004	005	0 (0	0	0000	00 00	1011 (B000			2	000	2
3 0005 FFEE 0004 39A7 0011 1001 1010 0111 0006 SUB R6, R	\top		Flags							•	05	C=0x	FFEE	4 SP=0]=0×000)×FFE	cución	Eje	
		6, R7	SUB R)6	0	0 0111	01 10	0011	39A7	0004	FEE	5	000	3

R3

0000

R4

0007

0000

R5

0000

R6

F020

R7

0000

Flags 0 1

RET

Flags

Flags

0

ini	iciales 0	000 F	FEF	0000	0000	000	00 0	000	0000	0000	F020	0000	1	0	0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
000	, ,	0004				9A7	C000	0000			0000	0000	0000	00	00	0000	0000
00	10 0000	0000	0000	0000	000 0	000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
			1							2da		3ra		Т	Ins	strucci	ón
	PC	SP	[SP+1]	IR	Instruc	ción -	1er pal	abra	PC	palabra	PC	palabra	PC			odifica	
1	0000	FFEF	0000	10CD	0001 1	001 11	100 100	10	0001	0004	0000			D.4.0)\/ E	7 [0	00041

in	iciales 0	000 F	FEF	0000	00	00 0	000	0000	0000	0000	F020	0000)	0	0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	-D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	000	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	000	0000	0000
			I	Τ	Τ.			1		2da		3ra	Ι	T	In	strucci	ón
	PC	SP	[SP+1]	IR	Insti	rucción	- 1er p	alabra	PC	palabra	PC	palabra	PC			codifica	
1	0000	FFFF	0000	19CB	000	1 1001	1100 1	000	0001	0004	0002			M	OV F	77 [0x	00041

ini	iciales 0	000 F	FEF	0000	00	00 00	000	0000	0000	0000	F020	0000)	0	0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
	•									•							-
					_					0.1							,
	PC	SP	[SP+1]	IR	Inst	rucción	- 1er p	alabra	PC	2da	PC	3ra	PC			strucci	
										palabra		palabra			ded	codifica	ıda
1	በበበበ	FFFF	nnnn	10CP	000	1 1001	1100 10	กกก	0001	0004	0002			MAC)\/ F	77 [0~1	1000/1

R6

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da	PC	3ra	PC.	Instrucció	ón	
	r C	31	[31 +1]	1111	mstruccion - 1er palabra	r C	palabra	1	palabra	FC	decodifica	da	
1	0000	FFEF	0000	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0	0004]	٦
	Ejecución	R7=[0:	<0004]=0>	0FE0				•			Flags		٦
2	0002			B000	1011 0000 0000 0000		CALL 0x0005		٦				
	Ejecución	[0xFFE	F]=0x000	4 SP=0:	FFEE PC=0x0005			Flags		٦			
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7		٦
	Ejecución	R6=0x	0000-0×0F	E0=0xF	020			•			Flags 0 1	0 1	.]
4	0006			C000	1100 0000 0000 0000	0007					RET		٦
	Ejecución	PC=[0	xFFEF]=0	×0004 S		Flags		٦					
5	0004	FFEF	0000	0FE0									7
	Ejecución										Flags		

ini	iciales 0	000 F	FEF	0000	00	00 00	000	0000	0000	0000	F020	0000)	0	0	0	0
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+	D	+E	+F
00	00 19CB	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
00	10 0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	00	00	0000	0000
	•									•							-
					_					0.1							,
	PC	SP	[SP+1]	IR	Inst	rucción	- 1er p	alabra	PC	2da	PC	3ra	PC			strucci	
										palabra		palabra			ded	codifica	ıda
1	በበበበ	FFFF	nnnn	10CP	000	1 1001	1100 10	กกก	0001	0004	0002			MAC)\/ F	77 [0~1	1000/1

R6

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC.	2da	PC.	3ra	PC.	Instrucción
	r C	31	[31 +1]	1111	mstruccion - 1er palabra	r C	palabra	1	palabra	FC	decodificada
1	0000	FFEF	0000	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0:	<0004]=0>	0FE0				•			Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFE	F]=0x000	4 SP=0:	FFEE PC=0x0005			Flags			
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x	0000-0×0F	E0=0xF	020			•			Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0	xFFEF]=0		Flags						
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000						
	Ejecución										Flags

R7=[0x0004]=0x0FE0

R6=0x0000-0x0FE0=0xF020

FFEE 0004

FFEF 0000

B000

39A7

C000

0FE0

PC=[0xFFEF]=0x0004 SP=0xFFEF

[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005

R1

1011 0000 0000 0000

0011 1001 1010 0111

1100 0000 0000 0000

0000 1111 1110 0000

Valores PC

Eiecución

Ejecución

Ejecución

Ejecución

Ejecución

0006

2 0002

3 0005

5 0004

ini	ciales 0	000 F	FFEF		00	00 00	000	0000	0000	0000	F020	0000		0 0	0	0
	+0	+1	+2	+3	+4	+5 +6		+7	+8	+9	+A	+B	+C	+D	+E	+F
000	00 19CB	0004	B000	0005	OFE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
00	10 0000	0000	0000	0000	0000	0000 000		0000	0000	0000	0000	0000	0000	0000	0000	0000
	PC	SP	[SP+1]	IR	Insti	rucción -	- 1er pa	alabra	PC	2da palabra	PC	3ra palabra	PC		strucció codifica	
1	0000	FFEF	0000	19CB	000	1 1001 1	100 10	00	0001	0004	0002			MOV	0004]	

R4

0003 0005

0006

0007

R6

0004

Flags

Flags

Flags 0 1 0

RET

Flags

Flags

CALL 0x0005

SUB R6, R7

iniciales		PC	SP	R0	R	1	R2	R3	R4	R5	R6	R7		Z	С	V	N	
iniciales		0000	FFEF	000	00 00	000	0000	0000	0000	0000	F020	0000	0	0	0	0	0	
	+0	+1	. +2	+3	+4	+5	5 +6	5 +7	+8	+9	+A	+B	+C	+	D	+E	+F	F
0000	19CE	3 000	4 B000	0005	0FE0	39A	7 C00	000	0000	0000	0000	0000	0000	00	000	0000	000	00
0010	0000	000	0 0000	0000	0000	000	0 000	000	0000	0000	0000	0000	0000	00	000	0000	000)0
																		_

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada			
1	0000	FFEF	0000	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]			
	Ejecución	R7=[0:	Flags											
2	0002		CALL 0x0005											
	Ejecución	[0×FFE	Flags											
3	0005	FFEE	SUB R6, R7											
	Ejecución	R6=0x	Flags 0 1 0 1											
4	0006			C000	1100 0000 0000 0000	0007					RET			
	Ejecución	PC=[0:												
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000	0005					Instrucción Inválida			
	Ejecución		C000 1100 0000 0000 0000 0007 PC=[0xFFEF]=0x0004 SP=0xFFEF											

Val	ores	PC	SP		R0	-		R2	R3		K4	K5	R6	R/		_ Z	C	V	IN
iniciales (0000	FFE	EF	0000		0000 00		0000)	0000	0000	F020	000	0	0	0	0	0
+0			-						_								_		
	+0	+:	1 +	+2	+3	+4	+5	+6) +	-7	+8	+9	+A	+B	+C	+	-D	+E	+F
0000	0000 19CB)4 B(000	0005	0FE0	39A	7 C00	00 00	000	0000	0000	0000	0000	0000	00	000	0000	0000
0010	0010 0000		00 00	000	0000	0000	000	000	0 00	000	0000	0000	0000	0000	0000	00	000	0000	0000
	0000 0000 0000																		
			_		_					_		0.1				_			,

	PC	SP	[SP+1]	IR	Instrucción - 1er palabra	PC	2da palabra	PC	3ra palabra	PC	Instrucción decodificada			
1	0000	FFEF	0000	19CB	0001 1001 1100 1000	0002	paradia		MOV R7, [0x0004]					
	Ejecución	R7=[0:	Flags											
2	0002		CALL 0x0005											
	Ejecución	[0xFFE	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005 FFEE 0004 39A7 0011 1001 1010 0111 0006											
3	0005	FFEE	SUB R6, R7											
	Ejecución	R6=0x	R6=0x0000-0x0FE0=0xF020											
4	0006			C000	1100 0000 0000 0000	0007					RET			
	Ejecución	PC=[0:	xFFEF]=0	×0004 S	P=0xFFEF						Flags			
5	0004	FFEF	0000		Instrucción Inválida									
	Ejecución	FIN DE	LA EJEC	CUCIÓN							Flags			

Ejercicio 2 - Solución

								Plan	illa de	S	egui:	miei	nto	F020								
			PC	SP		RC		R1	R2	3	R3	R4	R5	R6	R7		Z	C	V)		
V	alores i	niciales:	0000	FFEF	:	000	00 0	0000	0000	0	000	0000	0000	0000	0000		0	0	0	()	
	[+0	+1	+2	+3	+4	+5	+6	T	+7	+8	+9	+A	+B	+C	+I		+E	4	F	
Mer	noria:	0000	19C8	0004	B000	0005	005 OFE0 39/		A7 C000		0000	000	0000	0000	0000	0000	000	00 00	000	00	000	
		0001	0000	0000	0000	0000	0000	000	0000 0		0000	000	0000	0000	0000	0000	000	00 00	000	00	000	
	PC		SP	[SP+1]	IR	Ins		ón – 1 (en bits	- 1 ^{er} palabra		P	С	2 ^{do} palabra	PC	3 ^{cro} palabra	Pi	2		stru codi			
1	000) FI	FEF	0000	19C8	000	1 10	01 11	00 100	0	00	01	0004	0002				MOV	R7,	(0x0	000	1]
	Ejecución R7 = [0x0004] = 0x0FE0										Flags ¹	Z	С	٧	N							
2	000	2			B000	101	1 00	00 00	00 000	0	00	03	0005	0004				CALL 0x0005				
	Е	jecució	n	[0xFFEF]	= 0x00	04 9	SP = 0:	ĸFFEE	PC	= 0x	c0005							Flags ¹	Z	С	٧	м
3	000	5 FF	EE	0004	39A7	001	1 10	01 10	10 011	1	00	06						SUB R6, R7				
	Е	jecució	n	R6 = 0x0	0000 - 0	k0FE0 =	0xF0	20										Flags ¹	z 0	c 1	۷ 0	1 1
4	0006 C000 1100 0000 0000 0000 0007										RET											
	Ejecución PC = [0xFFEF] = 0x0004 SP = 0xFFEF										Flags ¹	Z	с	٧	м							
5	000	1 FI	FEF	0000	0FE0	000	00 11	11 11	10 000	0	00	05						Instru	ıcció	n In	váli	la
	Ejecución FIN DE LA EJECUCIÓN.									Flags ¹	Z	С	٧	20								

Ejercicio 2 - Observaciones

¿Qué pasó con el programa anterior?

Ejercicio 2 - Observaciones

¿Qué pasó con el programa anterior?

- Utilizamos la instrucción CALL.
- El valor del PC de retorno se almacena en la pila durante de la ejecución de la instrucción.
- Recuerden que la pila crece en el sentido de las direcciones de memoria menores
- ▶ Al regresar con RET, se "saca" la dirección de retorno de allí.

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

 0×0000

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

 0×0000 dos palabras

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

0x0002

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

 0×0002 dos palabras

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

0×0004

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

0x0004 una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

 $\mathsf{JL}\ \mathsf{sumoUno}$

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

0x0002 dos palabras

 0×0004 una palabra

 0×0005

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0x0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

 $0x0002 \quad \ {\rm dos\ palabras}$

 0×0004 una palabra

 0×0005 una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

 $main: \quad MOV \ R1, \ 0x0000$

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

 0×0002 dos palabras

 0×0004 una palabra

 0×0005 una palabra

0×0006

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main: MOV R1, 0×0000

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección ocupa

0x0000 dos palabras

 0×0002 dos palabras

 0×0004 una palabra

 0×0005 una palabra

 0×0006 dos palabras

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R1,	0×0000
-------	---------	--------

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	dos palabras

0x0008

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R1	, 0×0000
-------	--------	----------

loop: CMP R1, 0x0001

JL sumoUno

JGE fin

sumoUno: ADD R1, 0x0001

JNE loop

fin: RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	dos palabras
0×0008	una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R1, 0×0000
loop:	CMP R1, 0x0001
	JL sumoUno
	JGE fin
sumoUno:	ADD R1, 0x0001
	JNE loop
fin:	RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	dos palabras
8000×0	una palabra
0×0009	

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

main:	MOV R1, 0x0000
loop:	CMP R1, 0x0001
	JL sumoUno
	JGE fin
sumoUno:	ADD R1, 0x0001
	JNE loop
fin:	RET

dirección	ocupa
0×0000	dos palabras
0×0002	dos palabras
0×0004	una palabra
0×0005	una palabra
0×0006	dos palabras
8000×0	una palabra
0×0009	una palabra

Codifiquemos el siguiente código y carguémoslo desde la posición de memoria 0x0000:

		dirección	ocupa
main:	MOV R1, 0x0000	0×0000	dos palabras
loop:	CMP R1, 0x0001	0×0002	dos palabras
	JL sumoUno	0×0004	una palabra
	JGE fin	0×0005	una palabra
sumoUno:	ADD R1, 0×0001	0×0006	dos palabras
	JNE loop	0×0008	una palabra
fin:	RET	0×0009	una palabra

La etiqueta main corresponde a la dirección de memoria 0x0000. La etiqueta loop corresponde a la dirección de memoria 0x0002. La etiqueta sumoUno corresponde a la dirección de memoria 0x0006. La etiqueta fin corresponde a la dirección de memoria 0x0009. Reemplazando las etiqutas por valores estamos listos para codificar.

Buenísimo, tenemos las direcciones de las etiquetas, pero... *los saltos condicionales son* **relativos**.

Buenísimo, tenemos las direcciones de las etiquetas, pero... *los saltos condicionales son* **relativos**.

Esto significa que las etiquetas de los saltos son reemplazadas por el desplazamiento necesario para "llegar" desde la dirección en la que estamos parados, hasta la de la etiqueta de destino.

Buenísimo, tenemos las direcciones de las etiquetas, pero... *los saltos condicionales son* **relativos**.

- Esto significa que las etiquetas de los saltos son reemplazadas por el desplazamiento necesario para "llegar" desde la dirección en la que estamos parados, hasta la de la etiqueta de destino.
- Vamos a tener que calcular estos desplazamientos.

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} \mathsf{_Etiqueta} - Dir \mathsf{_Instr}_{postfetch}$

desplazamiento sumoUno:

0×0000: MOV R1, 0×0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

0×0000: MOV R1, 0×0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} \mathsf{_Etiqueta} - Dir \mathsf{_Instr}_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} \mathsf{_Etiqueta} - Dir \mathsf{_Instr}_{postfetch}$

desplazamiento sumoUno:

 $0 \times 00006 - 0 \times 0005 = 1$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} _\mathsf{Etiqueta} - Dir _Instr_{postfetch}$

desplazamiento sumoUno: $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0×0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 00006 - 0 \times 00005 = 1 = 0 \times 01$

desplazamiento fin:

0×0000: MOV R1, 0×0000

0x0002: CMP R1, 0x0001

0×0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0×0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0×0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0×0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin: $0 \times 0009 - 0 \times 0006$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0×0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0×0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin: $0 \times 0009 - 0 \times 0006 = 3$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0×0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0×0009: RET

 $\mathsf{Despl} = \mathsf{Dir}_\mathsf{Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} _\mathsf{Etiqueta} - Dir _Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

desplazamiento loop:

0x0000: MOV R1, 0x0000

0×0002: CMP R1, 0×0001

0×0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir_Etiqueta} - Dir_Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

desplazamiento loop:

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0x0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} _\mathsf{Etiqueta} - Dir _Instr_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

desplazamiento loop:

 $0 \times 0002 - 0 \times 0009$

0x0000: MOV R1, 0x0000

0×0002: CMP R1, 0×0001

0×0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} \mathsf{_Etiqueta} - Dir \mathsf{_Instr}_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

desplazamiento loop:

 $0 \times 0002 - 0 \times 0009 = -7$

0x0000: MOV R1, 0x0000

0×0002: CMP R1, 0×0001

0×0004: JL sumoUno

0x0005: JGE fin

0x0006: ADD R1, 0x0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} \mathsf{_Etiqueta} - Dir \mathsf{_Instr}_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

desplazamiento loop:

 $0 \times 0002 - 0 \times 0009 = -7 = 0 \times F9$

0x0000: MOV R1, 0x0000

0x0002: CMP R1, 0x0001

0×0004: JL sumoUno

0x0005: JGE fin

0×0006: ADD R1, 0×0001

0x0008: JNE loop

0x0009: RET

 $\mathsf{Despl} = \mathsf{Dir} \mathsf{_Etiqueta} - Dir \mathsf{_Instr}_{postfetch}$

desplazamiento sumoUno:

 $0 \times 0006 - 0 \times 0005 = 1 = 0 \times 01$

desplazamiento fin:

 $0 \times 0009 - 0 \times 0006 = 3 = 0 \times 03$

desplazamiento loop:

 $0 \times 0002 - 0 \times 0009 = -7 = 0 \times F9$

0×0000: MOV R1, 0×0000

0x0002: CMP R1, 0x0001

0x0004: JL 0x01

0x0005: JGE 0x03

0x0006: ADD R1, 0x0001

0x0008: JNE 0xF9

0x0009: RET

Ahora sí: nuestro programa esta listo para ser ensamblado!

Tarea: ensamblado y segui-

miento!

Moraleja del ejercicio anterior:

 Los desplazamientos de los saltos condicionales siempre son relativos.

- Los desplazamientos de los saltos condicionales siempre son relativos.
- Siempre van a ser traducidas las etiquetas primero, y luego calculados los desplazamientos.

- Los desplazamientos de los saltos condicionales siempre son relativos.
- Siempre van a ser traducidas las etiquetas primero, y luego calculados los desplazamientos.
- ▶ Para calcular los desplazamientos, tengan en cuenta que el salto se ejecuta *después* de incrementar el PC.

- Los desplazamientos de los saltos condicionales siempre son relativos.
- Siempre van a ser traducidas las etiquetas primero, y luego calculados los desplazamientos.
- ▶ Para calcular los desplazamientos, tengan en cuenta que el salto se ejecuta *después* de incrementar el PC.
- ▶ Los desplazamientos se representan en complemento a dos.

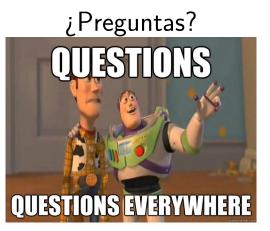
- Los desplazamientos de los saltos condicionales siempre son relativos.
- Siempre van a ser traducidas las etiquetas primero, y luego calculados los desplazamientos.
- ▶ Para calcular los desplazamientos, tengan en cuenta que el salto se ejecuta *después* de incrementar el PC.
- ▶ Los desplazamientos se representan en complemento a dos.
- Siempre extender los desplazamientos con signo a la hora de aplicarlos.

Resumen

Hoy vimos:

- Máquina Orga 1 y su arquitectura
 - ▶ ISA: formato de instrucción, modos de direccionamiento
 - ciclo de ejecución
- Ciclo de vida de un programa
 - lenguaje ensamblador vs. programa ensamblador
 - uso de etiquetas
 - directivas
- Ensamblamos programas
- Seguimos programas dentro de la máquina Orga1

¿Preguntas?



Avisos

Jueves 28/09: Taller de Ciclo de Instrucción en los labos.



IMPORTANTE: Traer planillas de Seguimiento



¡Eso es todo amigos!

