La Máquina **ORGA1**: Arquitectura y Seguimiento

Organización del Computador I

Verónica Coy

Departamento de Computación - FCEyN
UBA

2^{do} Cuatrimestre 2017

Outline

Arquitectura

Descripción General

¿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*.

Descripción General

¿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:

Descripción General

¿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.

Descripción General

¿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.

Descripción General

¿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

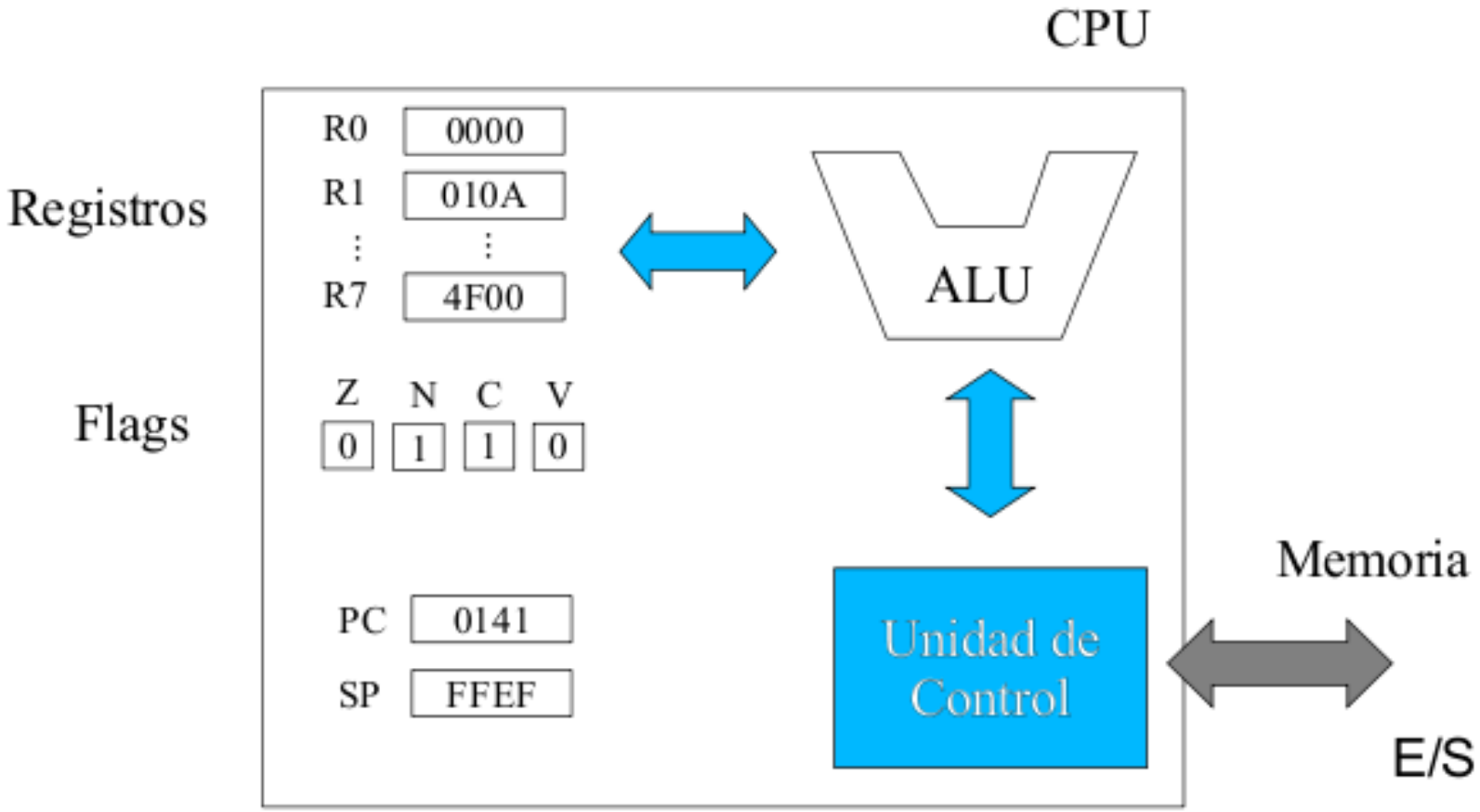
- 3 registros de **propósito específico** de 16 bits.
  - **PC** (Program Counter): Apunta a la dirección que contiene la próxima instrucción a ejecutar.
  - **SP** (Stack Pointer): Apunta a la próxima posición libre de la pila. [1]
  - **IR** (Instruction Register): Guarda la instrucción a ser inmediatamente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

[1] El **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.

[2] Más de esto en la segunda mitad de la materia.

Set de Instrucciones

Formato de instrucción

Tipo 1: Instrucciones de dos operandos

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

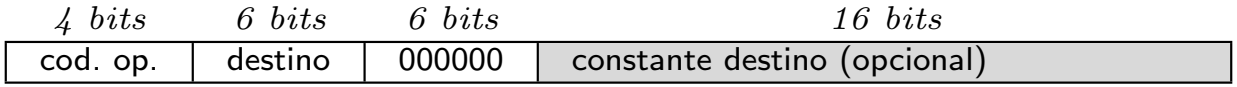
operación	cod. op.	efecto	modifica flags
MOV d, f	0001	$d \leftarrow f$	no
ADD d, f	0010	$d \leftarrow d + f$ (suma binaria)	sí
SUB d, f	0011	$d \leftarrow d - f$ (resta binaria)	sí
AND d, f	0100	$d \leftarrow d$ and f	sí $(\star)$
OR d, f	0101	$d \leftarrow d$ or f	sí $(\star)$
CMP d, f	0110	Modifica 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.

Formato de instrucción

Tipo 2: Instrucciones de un operando

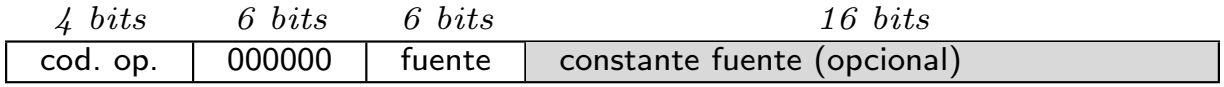
Tipo 2a: Instrucciones de un operando destino.



operacin	cod. op.	efecto	modifica flags
NEG d	1000	$d \leftarrow$ el inverso aditivo de d	s
NOT d	1001	$d \leftarrow$ not d (bit a bit)	s ($\star$)

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

Tipo 2b: Instrucciones de un operando fuente.



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

Formato de instrucción

Tipo 3: Instrucciones sin operandos

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

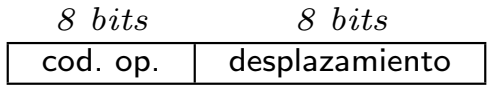
operación	cod. op.	efecto
RET	1100	PC $\leftarrow$ [SP+1], SP $\leftarrow$ SP + 1

- No modifica *flags*.

Formato de instrucción

Tipo 4: Saltos relativos condicionales

- el salto se produce si se cumple la *condición de salto* correspondiente.
- PC $\leftarrow$ PC + *desplazamiento*
- el desplazamiento se representa en *complemento a 2* de *8 bits*.
- No modifican *flags*.



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

Ciclo de Vida de un Programa

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

Ciclo de Vida de un Programa

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

1. 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**.

Ciclo de Vida de un Programa

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

1. 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**.

2. 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:

Ciclo de Vida de un Programa

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

1. 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**.

2. 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**.

Ciclo de Vida de un Programa

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

1. 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**.

2. 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.

Ciclo de Vida de un Programa

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

1. 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**.

2. 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.)

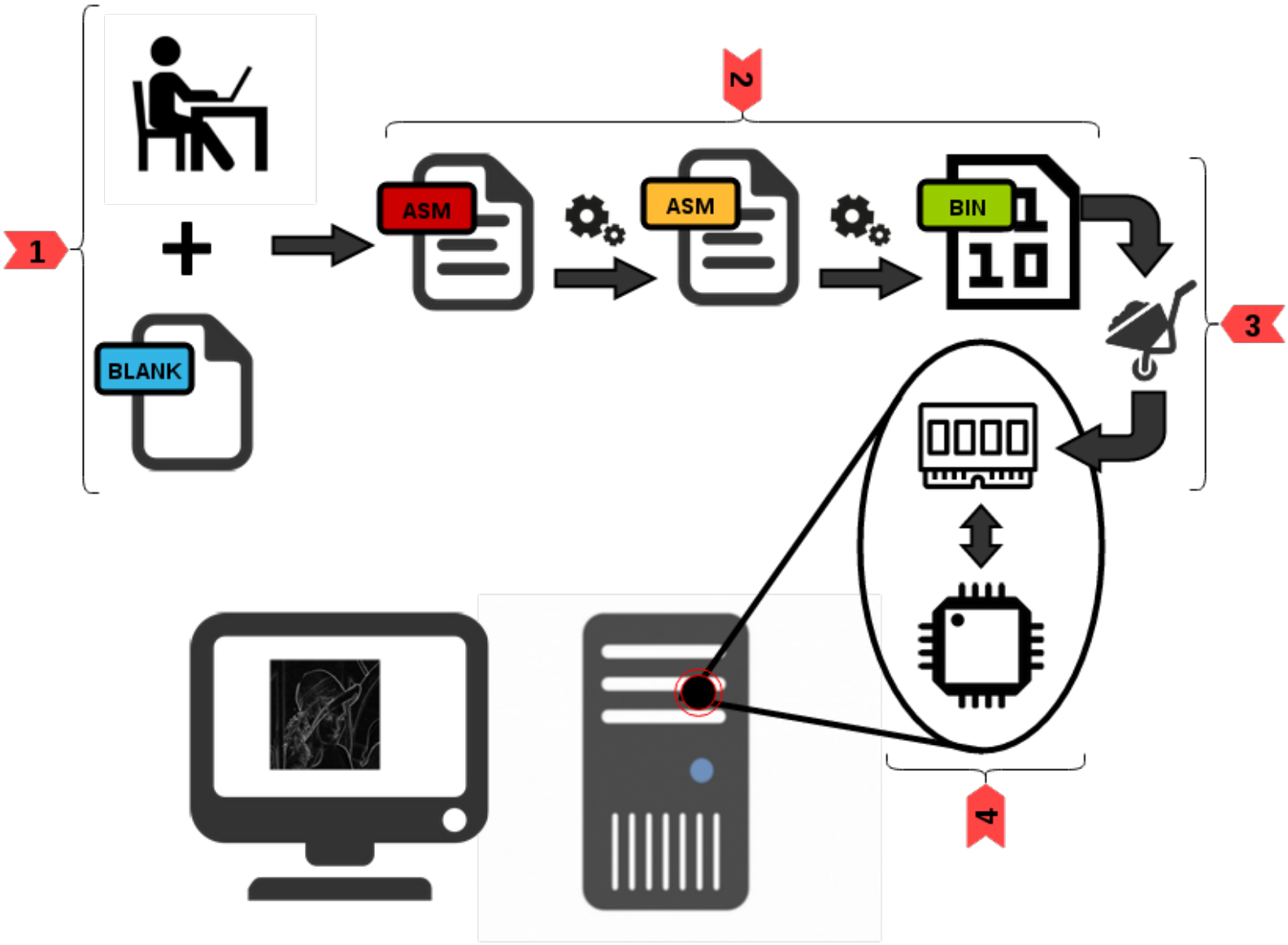
3. Carga: Se le indica al **Ensamblador** una dirección inicial, y éste copia el **código máquina** en la **Memoria** desde esa posición en adelante. Luego, carga esa misma dirección en el **PC (Program Counter)**.

Ciclo de Vida de un Programa (cont.)

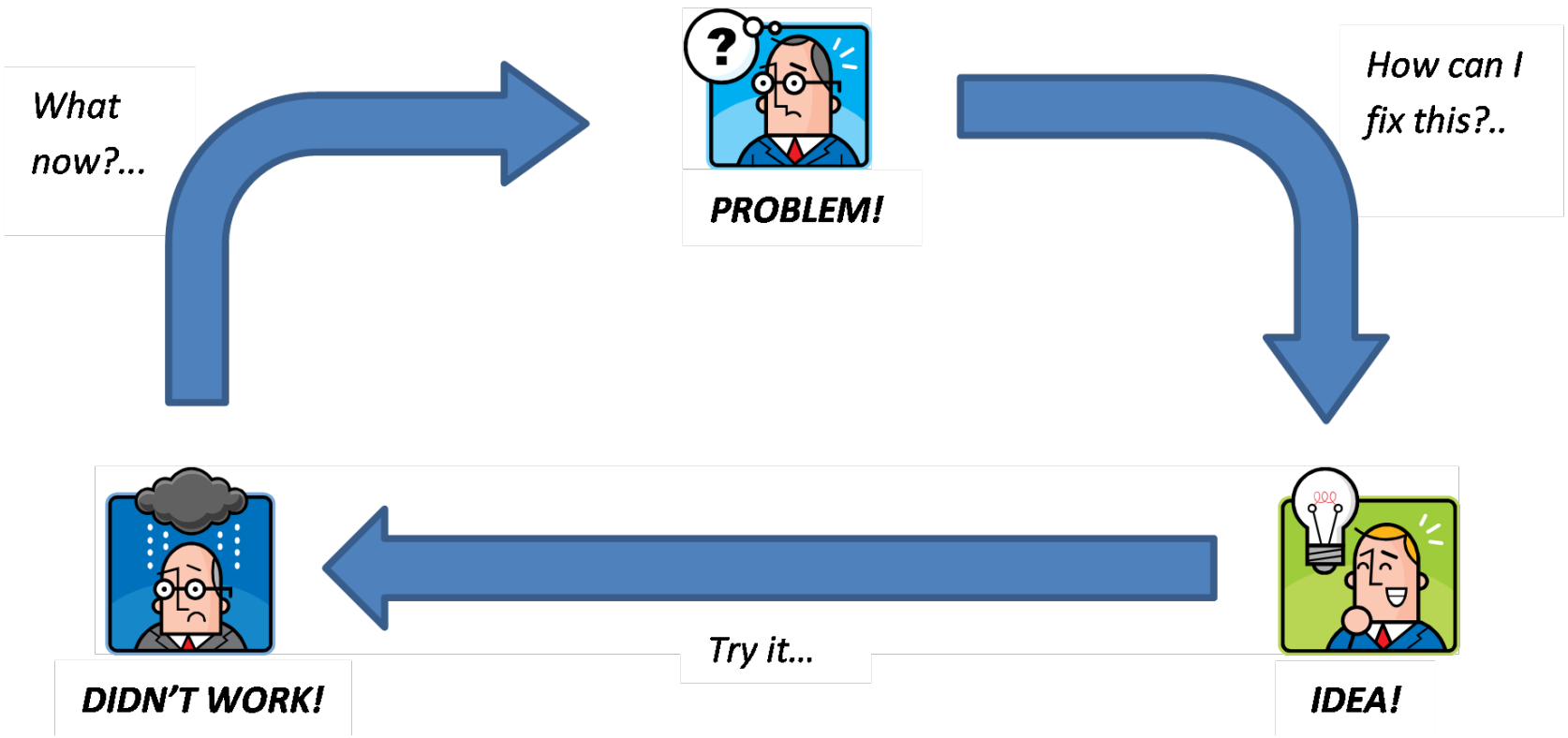
3. Carga: Se le indica al **Ensamblador** una dirección inicial, y éste copia el **código máquina** en la **Memoria** desde esa posición en adelante. Luego, carga esa misma dirección en el **PC (Program Counter)**.

4. Ejecución: El **CPU** da inicio a su **ciclo de ejecución** comenzando por la posición indicada en el **PC**.

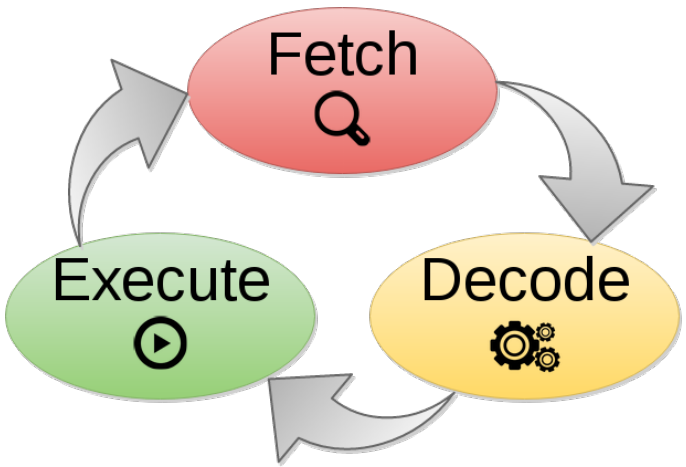
Ciclo de Vida de un Programa



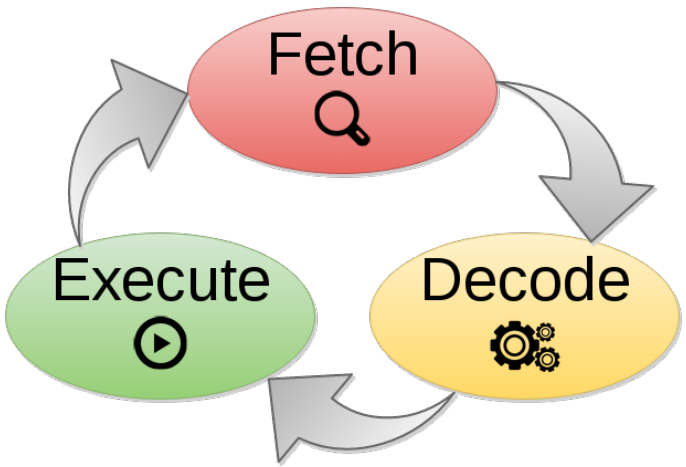
Ciclo de Vida de un Programador



Ciclo de Ejecución

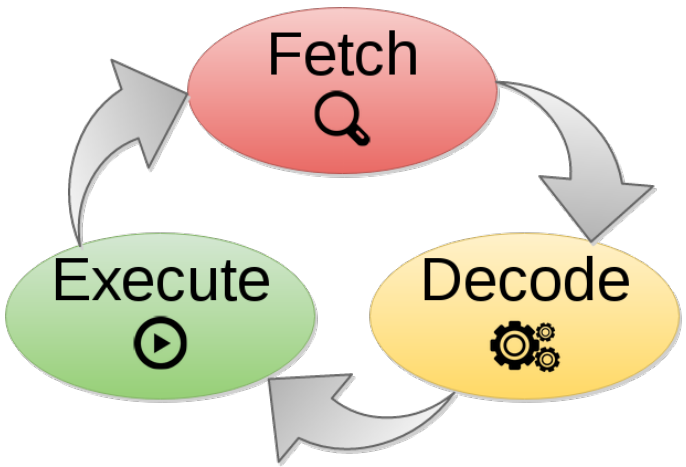


Ciclo de Ejecución



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

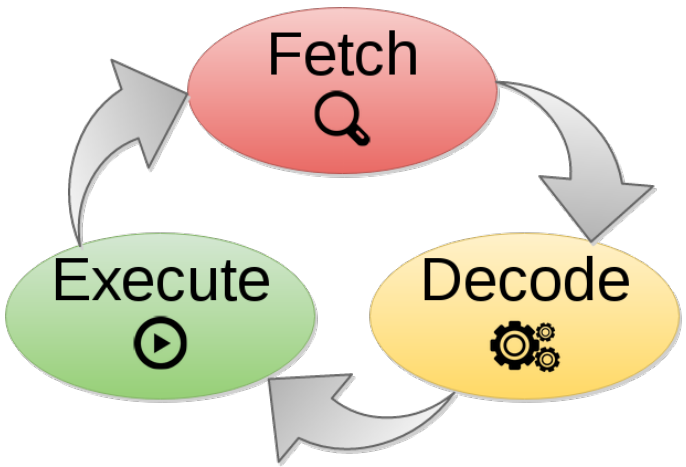
Ciclo de Ejecución



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)

1. La **UC** obtiene la primer palabra de la instrucción en **Memoria** (a partir de la dirección apuntada por el **PC**) e incrementa el **PC**.

2. 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

Algunas cuestiones semánticas...

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

Algunas cuestiones semánticas...

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.

Algunas cuestiones semánticas...

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.

Algunas cuestiones semánticas...

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:

$$\textbf{Assembly} \iff \textbf{Lenguaje Ensamblador}$$
$$\textbf{Assembler} \iff \textbf{Programa Ensamblador}$$

Ejercicios

Ejercicio 1 - Ensamblando

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
```

Ejercicio 1 - Ensamblando

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:

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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.

Ejercicio 1 - Ensamblando

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		

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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	

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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	

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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	

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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	

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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	

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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

Ejercicio 1 - Ensamblando

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

		dirección	ocupa
main:	MOV R1,[[et1]]	0x0000	dos palabras
	ADD [R1],0x6000	0x0002	dos palabras
	JMP main	0x0004	dos palabras
et1:	DW 0x0007	0x0006	una palabra
et2:	DW 0x0004	0x0007	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.

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006						

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

0000	0001	0002	0003	0004	0005	0006	0007
1858	0006						

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

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

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	una palabra

Traducidas las etiquetas lo codificamos y cargamos en memoria:

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

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	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		

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	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		

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	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**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	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**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	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**:

PC	

Ejercicio 1 - Ensamblando

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

```
MOV R1,[[0x0006]]
ADD [R1],0x6000
JMP 0x0000
DW 0x0007
DW 0x0004
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	dos palabras
0x0006	una palabra
0x0007	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**:

PC	0000

Ejercicio 1 - Seguimiento

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

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1											
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000										
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000								
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858							
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000						
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001					
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

	PC	SP
Valores iniciales	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

	PC	SP
Valores iniciales	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada				
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]				
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags				
2	0002														
	Ejecución										Flags				
3															
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada				
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]				
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40											
	Ejecución										Flags				
3															
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000						
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada				
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]				
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40	0010 1100 0100 0000	0003									
	Ejecución										Flags				
3															
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada				
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]				
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004							
	Ejecución										Flags				
3															
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución										Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución	R1=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags 1 1 0 0
3											
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000				
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags	1	1	0	0
3	0004														
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000				
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags	1	1	0	0
3	0004			0000											
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags 1 1 0 0
3	0004			0000	0000 0000 0000 0000						
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000				
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags	1	1	0	0
3	0004			0000	0000 0000 0000 0000										
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags				
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000				
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags	1	1	0	0
3	0004			0000	0000 0000 0000 0000	0005									
	Ejecución										Flags				

Ejercicio 1 - Seguimiento

	PC	SP
Valores iniciales	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags 1 1 0 0
3	0004			0000	0000 0000 0000 0000	0005					Instrucción Inválida
	Ejecución										Flags

Ejercicio 1 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
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	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=[[0x0006]]=[0x0007]=4									Flags
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1],0x6000
	Ejecución	[0x0004]=0xA000+0x6000=0x0000									Flags 1 1 0 0
3	0004			0000	0000 0000 0000 0000	0005					Instrucción Inválida
	Ejecución	FIN DE LA EJECUCIÓN									Flags

Ejercicio 1 - Solución

0000

Planilla de Seguimiento

Valores iniciales:

PC	SP
0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

Memoria:

	+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 - 1^{er} palabra (en bits)	PC	2^{da} palabra	PC	3^{era} palabra	PC	Instrucción decodificada
1	0000	FFEF	0000	1858	0001 1000 0101 1000	0001	0006	0002			MOV R1, [[0x0006]]
	Ejecución		R1 = [[0x0006]] = [0x0007] = 4								Flags[1] Z C V N
2	0002			2C40	0010 1100 0100 0000	0003	6000	0004			ADD [R1], 0x6000
	Ejecución		[0x0004] = 0xA000 + 0x6000 = 0x0000								Flags[1] Z 1 C 1 V 0 N 0
3	0004			0000	0000 0000 0000 0000	0005					Instrucción inválida
	Ejecución		FIN DE LA EJECUCIÓN								Flags[1] Z C V N

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.

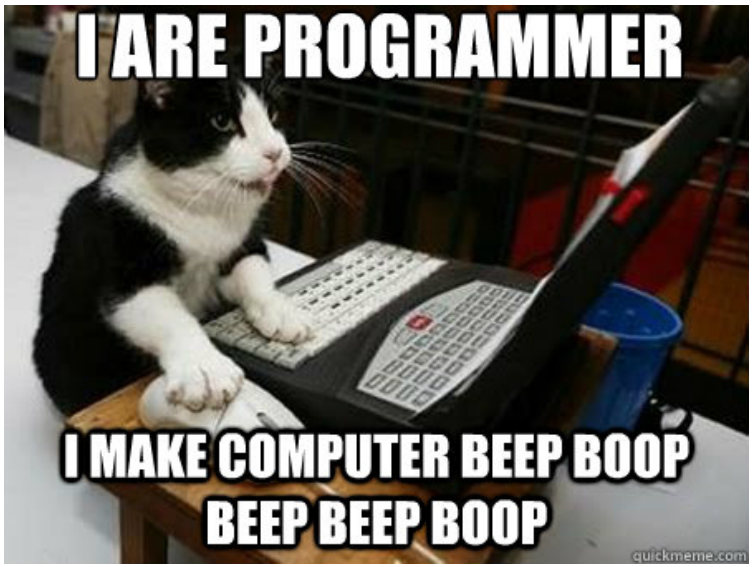
Ejercicio 1 - Observaciones

¿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
0x0000	

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
0x0000	dos palabras

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
0x0000	dos palabras
0x0002	

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
0x0000	dos palabras
0x0002	dos palabras

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	una palabra

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	una palabra
0x0006	

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	una palabra
0x0006	una palabra

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	una palabra
0x0006	una palabra

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
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	una palabra
0x0006	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.

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	dos palabras
0x0002	dos palabras
0x0004	una palabra
0x0005	una palabra
0x0006	una palabra

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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					

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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					

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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			

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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			

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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		

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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		

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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	39A7	

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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	39A7	

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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	39A7	C000

Y también cargamos la dirección inicial en el **PC**:

PC	

Ejercicio 2 - Ensamblando

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

```
MOV R7, [0x0004]
CALL 0x0005
DW 0x0FE0
SUB R6, R7
RET
```

dirección	ocupa
0x0000	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	39A7	C000

Y también cargamos la dirección inicial en el **PC**:

PC	0000

Ejercicio 2 - Seguimiento

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

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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											
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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										
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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								
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB							
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000						
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001					
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución										Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2											
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002										
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000							
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000						
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003					
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución										Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3											
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005										
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE									
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7							
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111						
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución										Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags
4											
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]				
	Ejecución	R7=[0x0004]=0x0FE0									Flags				
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005				
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags				
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7				
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags	0	1	0	1
4															
	Ejecución										Flags				
5															
	Ejecución										Flags				

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006										
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000							
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000						
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución										Flags
5											
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]				
	Ejecución	R7=[0x0004]=0x0FE0									Flags				
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005				
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags				
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7				
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags	0	1	0	1
4	0006			C000	1100 0000 0000 0000	0007					RET				
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags				
5															
	Ejecución										Flags				

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags
5	0004										
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]				
	Ejecución	R7=[0x0004]=0x0FE0									Flags				
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005				
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags				
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7				
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags	0	1	0	1
4	0006			C000	1100 0000 0000 0000	0007					RET				
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags				
5	0004	FFEF	0000												
	Ejecución										Flags				

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags
5	0004	FFEF	0000	0FE0							
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000						
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000	0005					
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000	0005					Instrucción Inválida
	Ejecución										Flags

Ejercicio 2 - Seguimiento

Valores iniciales	PC	SP
	0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	F020	0000

Z	C	V	N
0	0	0	0

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19CB	0004	B000	0005	0FE0	39A7	C000	0000	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	19CB	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]
	Ejecución	R7=[0x0004]=0x0FE0									Flags
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005
	Ejecución	[0xFFEF]=0x0004 SP=0xFFEE PC=0x0005									Flags
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7
	Ejecución	R6=0x0000-0x0FE0=0xF020									Flags 0 1 0 1
4	0006			C000	1100 0000 0000 0000	0007					RET
	Ejecución	PC=[0xFFEF]=0x0004 SP=0xFFEF									Flags
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000	0005					Instrucción Inválida
	Ejecución	FIN DE LA EJECUCIÓN									Flags

Ejercicio 2 - Solución

Planilla de Seguimiento F020

Valores iniciales:

PC	SP
0000	FFEF

R0	R1	R2	R3	R4	R5	R6	R7
0000	0000	0000	0000	0000	0000	0000	0000

Z	C	V	N
0	0	0	0

Memoria:

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	19C8	0004	B000	0005	0FE0	39A7	C000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0001	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

	PC	SP	[SP+1]	IR	Instrucción - 1^{er} palabra (en bits)	PC	2^{da} palabra	PC	3^{era} palabra	PC	Instrucción decodificada				
1	0000	FFEF	0000	19C8	0001 1001 1100 1000	0001	0004	0002			MOV R7, [0x0004]				
	Ejecución		R7 = [0x0004] = 0x0FE0								Flags[1]	Z	C	V	N
2	0002			B000	1011 0000 0000 0000	0003	0005	0004			CALL 0x0005				
	Ejecución		[0xFFEF] = 0x0004 SP = 0xFFEE PC = 0x0005								Flags[1]	Z	C	V	N
3	0005	FFEE	0004	39A7	0011 1001 1010 0111	0006					SUB R6, R7				
	Ejecución		R6 = 0x0000 - 0x0FE0 = 0xF020								Flags[1]	Z 0	C 1	V 0	N 1
4	0006			C000	1100 0000 0000 0000	0007					RET				
	Ejecución		PC = [0xFFEF] = 0x0004 SP = 0xFFEF								Flags[1]	Z	C	V	N
5	0004	FFEF	0000	0FE0	0000 1111 1110 0000	0005					Instrucción Inválida				
	Ejecución		FIN DE LA EJECUCIÓN.								Flags[1]	Z	C	V	N

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.

Ejercicio 3

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
```

Ejercicio 3

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	

Ejercicio 3

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

Ejercicio 3

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	

Ejercicio 3

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

Ejercicio 3

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	

Ejercicio 3

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

Ejercicio 3

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
0x0005	

Ejercicio 3

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
0x0005	una palabra

Ejercicio 3

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
0x0005	una palabra
0x0006	

Ejercicio 3

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
0x0005	una palabra
0x0006	dos palabras

Ejercicio 3

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
0x0005	una palabra
0x0006	dos palabras
0x0008	

Ejercicio 3

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
0x0005	una palabra
0x0006	dos palabras
0x0008	una palabra

Ejercicio 3

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
0x0005	una palabra
0x0006	dos palabras
0x0008	una palabra
0x0009	

Ejercicio 3

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
0x0005	una palabra
0x0006	dos palabras
0x0008	una palabra
0x0009	una palabra

Ejercicio 3

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

		dirección	ocupa
main:	MOV R1, 0x0000	0x0000	dos palabras
loop:	CMP R1, 0x0001	0x0002	dos palabras
	JL sumoUno	0x0004	una palabra
	JGE fin	0x0005	una palabra
sumoUno:	ADD R1, 0x0001	0x0006	dos palabras
	JNE loop	0x0008	una palabra
fin:	RET	0x0009	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.

Ejercicio 3

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

Ejercicio 3

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.

Ejercicio 3

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

desplazamiento loop:

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

desplazamiento loop:

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

desplazamiento loop:
0x0002 - 0x0009

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

desplazamiento loop:
0x0002 - 0x0009 = -7

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

desplazamiento loop:
0x0002 - 0x0009 = -7 = 0xF9

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.

Ejercicio 3

0x0000:	MOV R1, 0x0000
0x0002:	CMP R1, 0x0001
0x0004:	JL sumoUno
0x0005:	JGE fin
0x0006:	ADD R1, 0x0001
0x0008:	JNE loop
0x0009:	RET

Despl = Dir_Etiqueta - $Dir\_Instr_{postfetch}$

desplazamiento sumoUno:
0x0006 - 0x0005 = 1 = 0x01

desplazamiento fin:
0x0009 - 0x0006 = 3 = 0x03

desplazamiento loop:
0x0002 - 0x0009 = -7 = 0xF9

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.

Ejercicio 3

```
0x0000:  MOV R1, 0x0000
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 seguimiento!

Ejercicio 3 - Observaciones

Moraleja del ejercicio anterior:

Ejercicio 3 - Observaciones

Moraleja del ejercicio anterior:

- Los desplazamientos de los saltos condicionales *siempre son relativos*.

Ejercicio 3 - Observaciones

Moraleja del ejercicio anterior:

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

Ejercicio 3 - Observaciones

Moraleja del ejercicio anterior:

- 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.

Ejercicio 3 - Observaciones

Moraleja del ejercicio anterior:

- 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.

Ejercicio 3 - Observaciones

Moraleja del ejercicio anterior:

- 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.

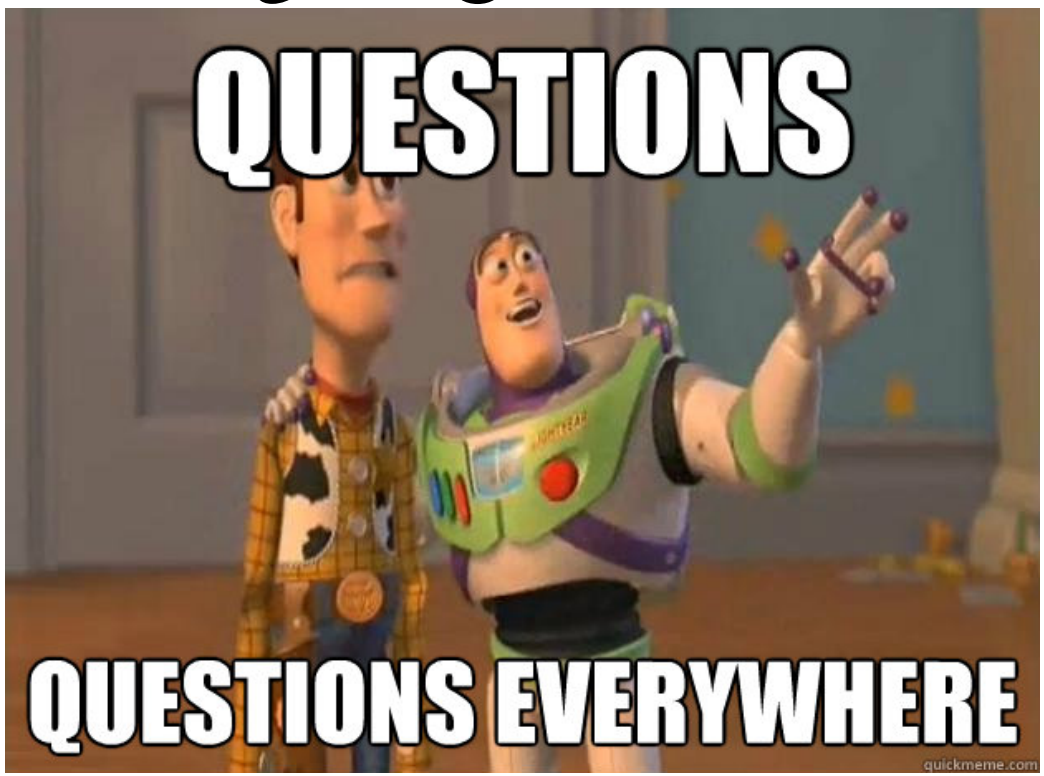
Resumen

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?

¿Preguntas?



Avisos

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

IMPORTANTE: Traer planillas de Seguimiento

¡Eso es todo amigos!