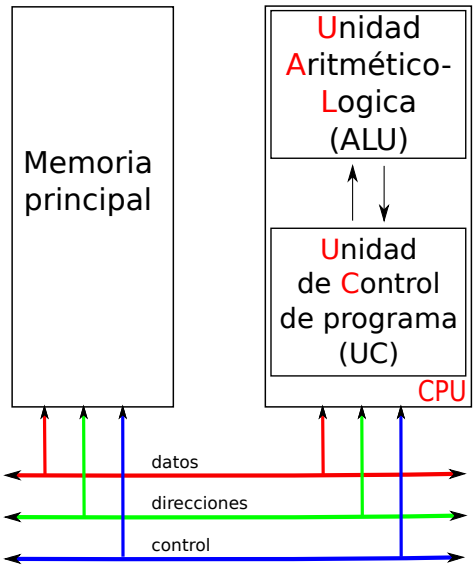
Operaciones lógicas y repetición

Organización de computadoras

Universidad Nacional de Quilmes

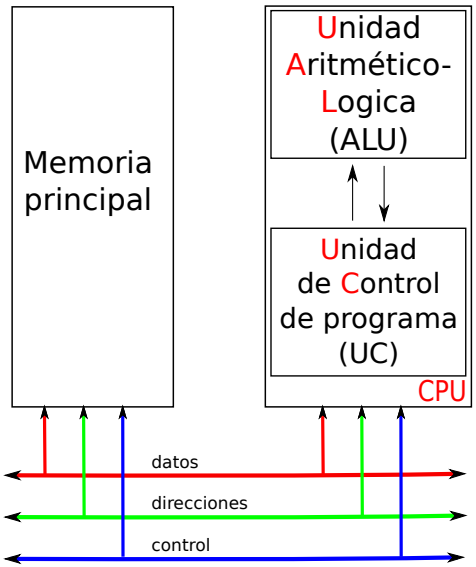
http://orga.blog.unq.edu.ar

Repaso



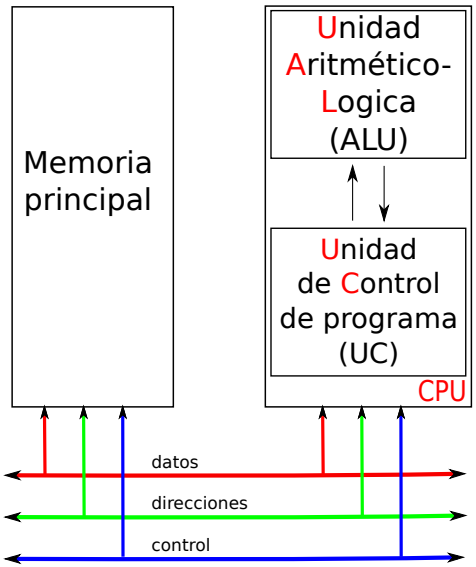
1. Flags

Repaso



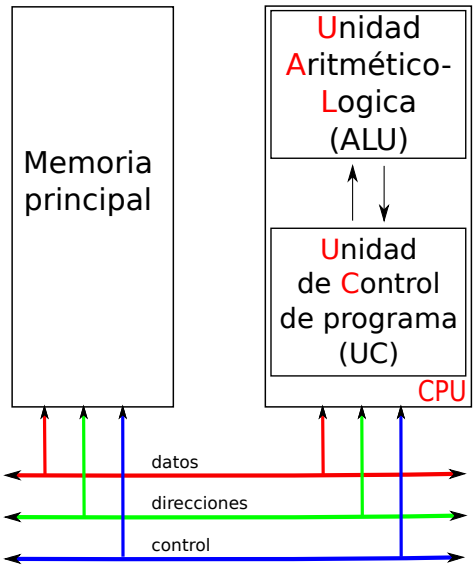
1. Flags
2. *Program Counter*

Repaso



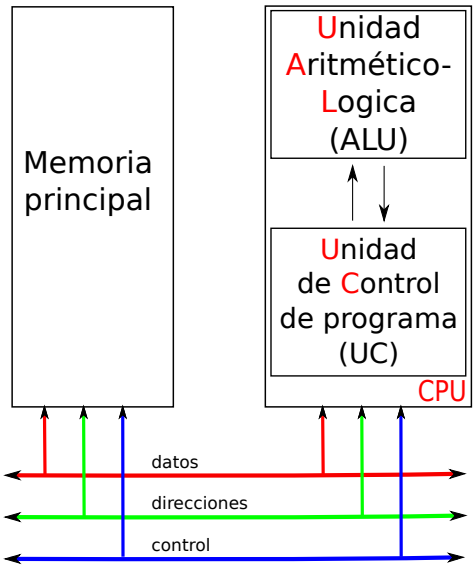
1. Flags
2. *Program Counter*
3. *Instruction Register*

Repaso



1. Flags
2. *Program Counter*
3. *Instruction Register*
4. **Q3**

Repaso



1. Flags
2. *Program Counter*
3. *Instruction Register*
4. **Q3**
   1. Saltos condicionales (con y sin signo)
   2. Saltos incondicionales: JMP

Algebra de Boole: Características

- Para capturar la matemática del pensamiento (1854)
- Se representa información relativa a los hechos, como el vaso está vacío o el vaso no está vacío
- Tiene variables y operaciones.

Variable Binaria
(o de Boole) que toma un valor del conjunto {0,1}

Algebra de Boole: Características

Valores de verdad

1  Verdadero

0  Falso

Algebra de Boole: Características

Operaciones Básicas

Conjunción  **AND**

Disyunción  **OR**

Negación  **NOT**

Algebra de Boole: Operaciones

Conjunción  **AND**

A	B	A ∧ B
0	0	0
0	1	0
1	0	0
1	1	1

Algebra de Boole: Operaciones

Disyunción  **OR**

A	B	A $\vee$ B
0	0	0
0	1	1
1	0	1
1	1	1

Algebra de Boole: Operaciones

Negación  **NOT**

A	$\overline{A}$
0	1
1	0

Algebra de Boole: Características

Operaciones derivadas

Disyunción Exclusiva  **XOR**

Negación de la conjunción  **NAND**

Negación de la disyunción  **NOR**

Algebra de Boole: Operaciones derivadas

Negación de la conjunción  **NAND**

$$A \uparrow B \iff \overline{(A \wedge B)}$$

Algebra de Boole: Operaciones derivadas

Negación de la conjunción  **NAND**

$$A \uparrow B \iff \overline{(A \wedge B)}$$

A	B	A $\wedge$ B	$\overline{(A \wedge B)}$
0	0	0	
0	1	0	
1	0	0	
1	1	1	

Algebra de Boole: Operaciones derivadas

Negación de la conjunción  **NAND**

$$A \uparrow B \iff \overline{(A \wedge B)}$$

Algebra de Boole: Operaciones derivadas

Negación de la conjunción  **NAND**

$$A \uparrow B \iff \overline{(A \wedge B)}$$

A	B	$A \uparrow B$
0	0	1
0	1	1
1	0	1
1	1	0

Algebra de Boole: Operaciones derivadas

Negación de la disyunción  **NOR**

$$A \downarrow B \iff \overline{(A \vee B)}$$

Algebra de Boole: Operaciones derivadas

Negación de la disyunción  **NOR**

$$A \downarrow B \iff \overline{(A \vee B)}$$

A	B	A $\vee$ B	$\overline{(A \vee B)}$
0	0	0	
0	1	1	
1	0	1	
1	1	1	

Algebra de Boole: Operaciones derivadas

Negación de la disyunción  **NOR**

$$A \downarrow B \iff \overline{(A \vee B)}$$

Algebra de Boole: Operaciones derivadas

Negación de la disyunción  **NOR**

$$A \downarrow B \iff \overline{(A \vee B)}$$

A	B	$A \downarrow B$
0	0	1
0	1	0
1	0	0
1	1	0

Algebra de Boole: Operaciones derivadas

Disyunción Exclusiva XOR

$$A \oplus B \iff (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$$

Algebra de Boole: Operaciones derivadas

Disyunción Exclusiva XOR

$$A \oplus B \iff (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$$

A	$\overline{A}$	B	$\overline{B}$	$(A \wedge \overline{B})$	$(\overline{A} \wedge B)$	$(A \wedge \overline{B}) \vee (\overline{A} \wedge B)$
0	1	0	1			
0	1	1	0			
1	0	0	1			
1	0	1	0			

Algebra de Boole: Operaciones derivadas

Disyunción Exclusiva XOR

$$A \oplus B \iff (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$$

Algebra de Boole: Operaciones derivadas

Disyunción Exclusiva  XOR

$$A \oplus B \iff (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$$

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Operaciones sobre cadenas

Operaciones sobre cadenas: AND bit a bit

$$\begin{array}{rr} & 1010 \\ \text{AND} & 0101 \\ \hline \end{array}$$

Operaciones sobre cadenas

Operaciones sobre cadenas: AND bit a bit

$$\begin{array}{rr} & 1010 \\ \text{AND} & 0101 \\ \hline & 0000 \end{array}$$

Operaciones sobre cadenas

Operaciones sobre cadenas: OR bit a bit

$$\begin{array}{rr} & 1010 \\ \text{OR} & 0101 \\ \hline \end{array}$$

Operaciones sobre cadenas

Operaciones sobre cadenas: OR bit a bit

$$\begin{array}{rr} & 1010 \\ \text{OR} & 0101 \\ \hline & 1111 \end{array}$$

Operaciones sobre cadenas

Operaciones sobre cadenas: NOT bit a bit

$$\begin{array}{r} 1010 \\ \text{NOT} \quad \underline{} \end{array}$$

Operaciones sobre cadenas

Operaciones sobre cadenas: NOT bit a bit

$$\begin{array}{rr} & 1010 \\ \text{NOT} & \\ \hline & 0101 \end{array}$$

Operaciones sobre cadenas

Ejercicios

1. NAND
   ```
     1100
   & 0100
   ------
     ?
   ```
2. NOR
   ```
     1100
   | 0100
   ------
     ?
   ```
3. XOR
   ```
     1100
   ^ 0100
   ------
     ?
   ```

Máscaras

Máscaras

Máscara
Cadena binaria que se aplica sobre otra mediante una operación lógica para descubrir características sobre esa cadena

$$\begin{array}{ll} & 0101 \text{ cadena} \\ \text{AND} & 1111 \text{ máscara} \\ \hline & 0101 \end{array}$$

Máscaras

Usando AND:

- Si se quiere preservar el bit: usar 1
- Si se quiere poner un cero: usar 0

```
  ????
AND 0001
```

Máscaras

Usando AND:

- Si se quiere preservar el bit: usar 1
- Si se quiere poner un cero: usar 0

```
    ????
AND 0001
-------
    000?
```

Máscaras

Usando OR:

- Si se quiere preservar el bit: usar 0
- Si se quiere poner un uno: usar 1

```
  ????
OR 0001
-------
```

Máscaras

Usando OR:

- Si se quiere preservar el bit: usar 0
- Si se quiere poner un uno: usar 1

$$\begin{array}{rr} & ???? \\ \text{OR} & 0001 \\ \hline & ???1 \end{array}$$

Máscaras: ejemplos de uso

Example

Determinar si la cadena en R0 es impar

Máscaras: ejemplos de uso

Example

Determinar si la cadena en R0 es impar

```
AND R0, 0x0001
JNE saltarAEsPar
```

Máscaras: ejemplos de uso

Example

Copiar el byte mas significativo de la celda 0348 en el registro R1

Máscaras: ejemplos de uso

Example

Copiar el byte mas significativo de la celda 0348 en el registro R1

```
MOV R1, [0348]
AND R1, 0xFF00
```

Máscaras: ejemplos de uso

Ejercicio (NO se entrega): Si la celda [CCCC] contiene un número par, sumar 30 al valor de R3. En caso contrario sumar 70 al valor de R4

Máscaras: ejemplos de uso

Ejercicio (NO se entrega): Si la celda [CCCC] contiene un número par, sumar 30 al valor de R3. En caso contrario sumar 70 al valor de R4

```
        MOV R1, [CCCC]
        AND R1, 0x0001
        JNE noespar
        ADD R3, 0x001E
        JMP sigue
noespar:ADD R4, 0x0046
  sigue:
```

Máscaras: ejemplos de uso

Si las celdas CCCC y CCCD contienen números impares, restarles 0x0001 a ambas

Máscaras: ejemplos de uso

Si las celdas CCCC y CCCD contienen números impares, restarles 0x0001 a ambas

Ayudita:

```
si ([CCCC] es impar)
   si ([CCCD] es impar)
      [CCCC] <-- [CCCC]-1
      [CCCD] <-- [CCCD]-1
   fin si
fin si
```

Máscaras: ejemplos de uso

Permisos de acceso sobre archivos

- Con 3 bits se indica:
  1. ¿puedo leer? (r)
  2. ¿puedo escribir? (w)
  3. ¿puedo ejecutar? (x)
- Con 3 cadenas se describen permisos de usuario, grupo y otros

Example

La cadena 111111111 le da todos los permisos a todos

Máscaras: ejemplos de uso

Permisos de acceso sobre archivos

¿Cómo saber si otro usuario del grupo puede escribirlo?

Máscaras: ejemplos de uso

Permisos de acceso sobre archivos

¿Cómo saber si otro usuario del grupo puede escribirlo?

```
   ?????????
AND 000010000
-------------
   0000?0000
```

Ejercicio: Calcular el resto de la división entera: $14 \,\%3$

Ejercicio: Calcular el resto de la división entera: 14 %3

Idea

```
resto <- 14
mientras(resto > 3)
        resto <- resto-3
fin mientras
```

Repeticiones

Example (¿Cómo encender el piloto del calefón?)

1. poner la perilla en posición piloto
2. acercar un fósforo mientras se presiona la perilla
3. mantener presionando aproximadamente 20 segundos
4. Si al liberar la perilla el piloto se apaga, volver al paso (1), sino seguir con el paso (5)
5. ...

Example (Calcular el resto de la división entera 14 %3)

1. Inicializar la variable resto con el valor 14
2. Si resto es menor a 3, seguir por el paso (4)
3. Restar el valor 3 a la variable resto y volver al paso (2)
4. ...

Example (Calcular el resto de la división entera 14 %3)

1. Inicializar la variable resto con el valor 14
2. Si resto es menor a 3, seguir por el paso (4)
3. Restar el valor 3 a la variable resto y volver al paso (2)
4. ...

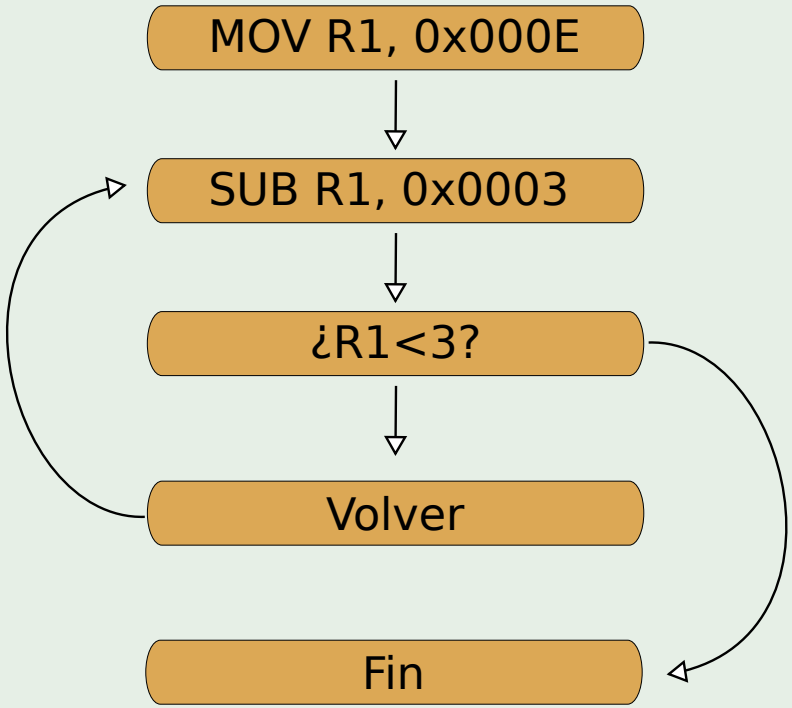
¿Se necesitan herramientas nuevas?

Example (Calcular el resto de la división entera 14 %3 en **Q3**)

1. Inicializar la variable resto con el valor 14
2. Si resto es menor a 3, seguir por el paso (4)
3. Restar el valor 3 a la variable resto y volver al paso (2)
4. ...

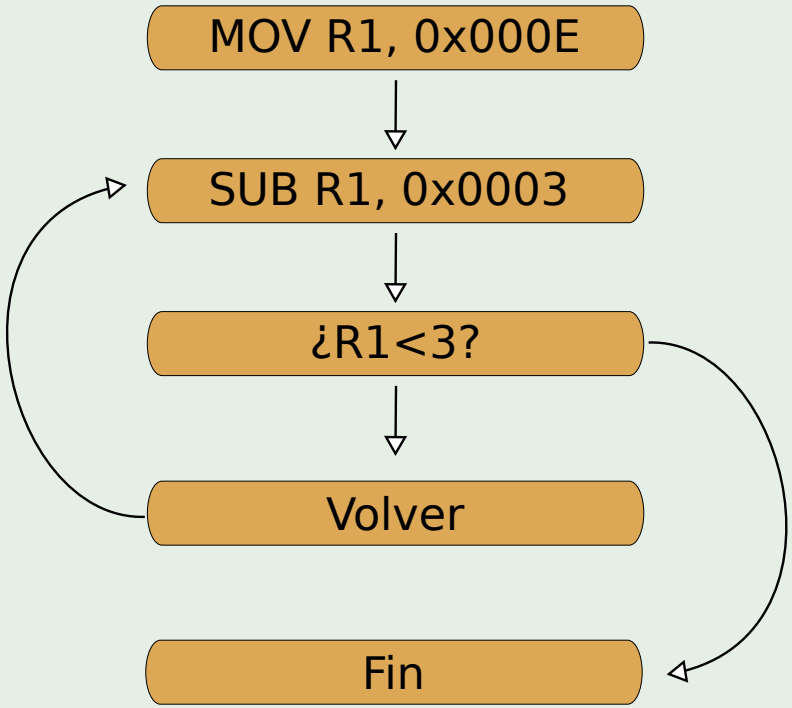
Example (Calcular el resto de la división entera 14 %3 en **Q3**)

1. Inicializar la variable resto con el valor 14
2. Si resto es menor a 3, seguir por el paso (4)
3. Restar el valor 3 a la variable resto y volver al paso (2)
4. ...



Example (Calcular el resto de la división entera 14 %3 en **Q3**)

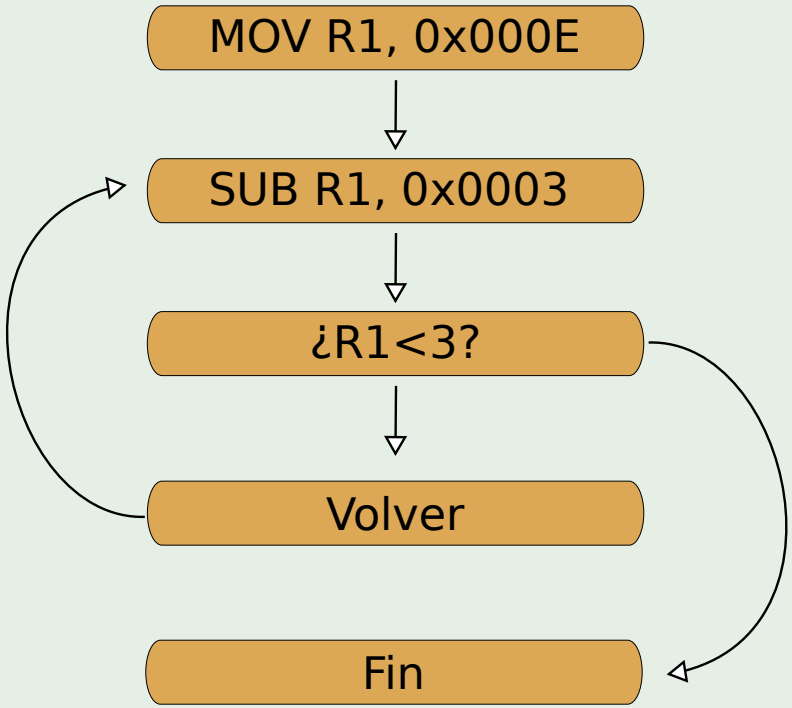
1. Inicializar la variable resto con el valor 14
2. Si resto es menor a 3, seguir por el paso (4)
3. Restar el valor 3 a la variable resto y volver al paso (2)
4. ...




```
        MOV R1, 0x000E
arriba: SUB R1, 0x0003
        JLE fin
        JMP arriba
fin:
```

Example (Calcular el resto de la división entera 14 %3 en **Q3**)

1. Inicializar la variable resto con el valor 14
2. Si resto es menor a 3, seguir por el paso (4)
3. Restar el valor 3 a la variable resto y volver al paso (2)
4. ...




```
        MOV R1, 0x000E
arriba: SUB R1, 0x0003
        JLE fin
        JMP arriba
fin:
```

Recorrido de arreglos

Modos indirectos

Arreglo de valores

Posiciones de memoria consecutivas que contienen una colección de elementos. Cada elemento puede ocupar mas de una celda.

⋮	⋮
000A	1er valor
000B	2do valor
000C	3er valor
000D	4to valor
000E	5to valor
000F	6to valor
⋮	⋮

Tamaño de un arreglo

Cantidad de elementos

Desafío

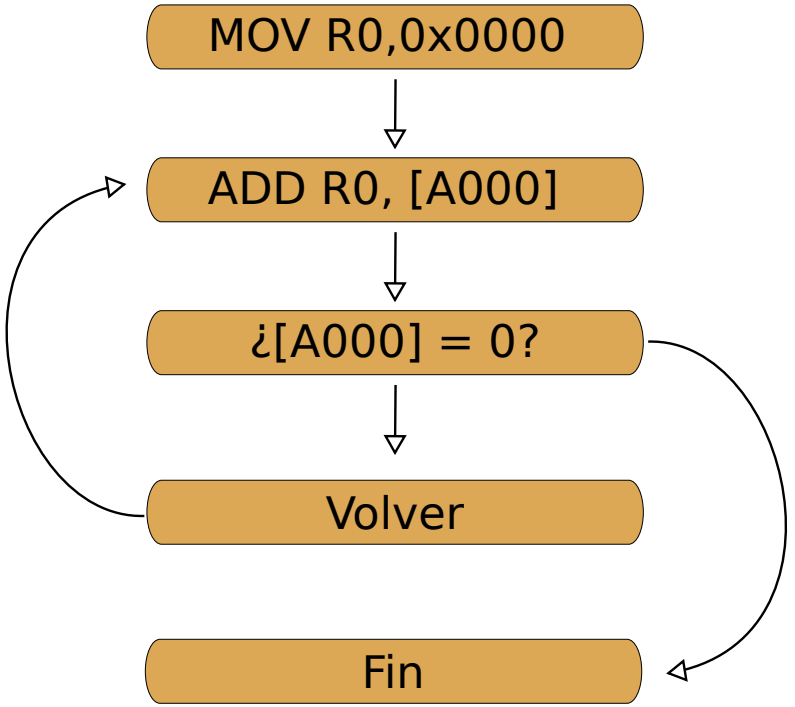
Desafío

A partir de la celda A000 hay un arreglo que contiene los pedidos de empanadas de una rotisería, y que finaliza con el primer valor 0. Sumar todos los valores

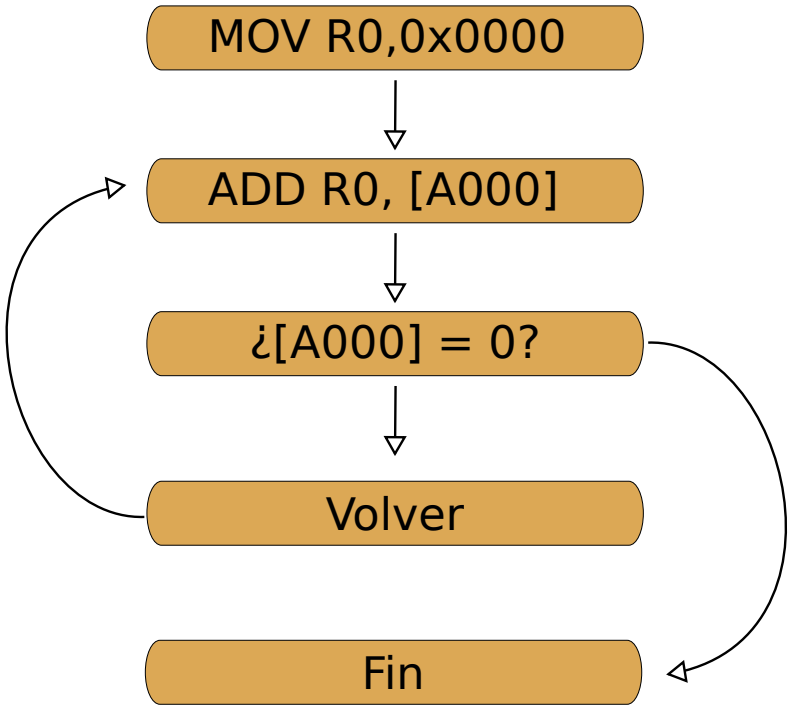
⋮	⋮
A000	0010
A001	001A
A002	0014
A003	0018
A004	0000
⋮	⋮

A partir de la celda A000 hay un arreglo que contiene los pedidos de empanadas de una rotisería, y que finaliza con el primer valor 0. Sumar todos los valores

A partir de la celda A000 hay un arreglo que contiene los pedidos de empanadas de una rotisería, y que finaliza con el primer valor 0. Sumar todos los valores



A partir de la celda A000 hay un arreglo que contiene los pedidos de empanadas de una rotisería, y que finaliza con el primer valor 0. Sumar todos los valores



¿Que limitación encontramos?

Lo que estabas necesitando es...

Modos indirectos

Modo de direccionamiento indirecto

Se especifica una dirección de memoria que contiene la dirección de memoria que contiene el operando

Modos indirectos

Modo de direccionamiento indirecto

Se especifica una dirección de memoria que contiene la dirección de memoria que contiene el operando

MOV R0, [[FFFF]]

Modos indirectos

Modo de direccionamiento indirecto
Se especifica una dirección de memoria que contiene la dirección de memoria que contiene el operando

MOV R0, [[FFFF]]

¿Dónde está el operando?

Modos indirectos

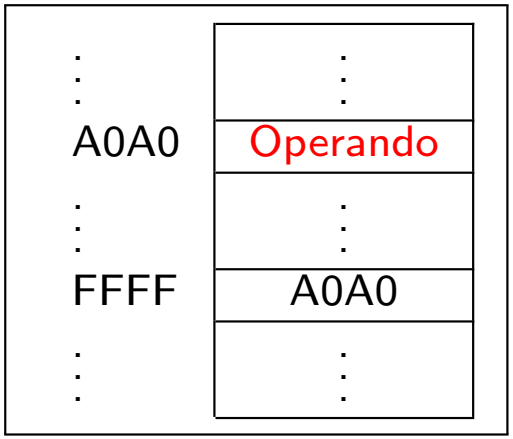
MOV R0, [[FFFF]]

¿Dónde está el operando?

Modos indirectos

MOV R0, [[FFFF]]

¿Dónde está el operando?



Modos indirectos

Modo de direccionamiento registro indirecto

Se especifica un número de registro que contiene la dirección de memoria que contiene el operando

Modos indirectos

Modo de direccionamiento registro indirecto
Se especifica un número de registro que contiene la dirección de memoria que contiene el operando

MOV R0, [R5]

Modos indirectos

Modo de direccionamiento registro indirecto
Se especifica un número de registro que contiene la dirección de memoria que contiene el operando

MOV R0, [R5]

¿Dónde está el operando?

Modos indirectos

MOV R0, [R5]

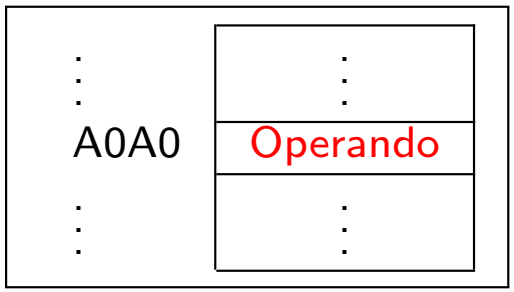
¿Dónde está el operando?

Modos indirectos

MOV R0, [R5]

¿Dónde está el operando?

R5 = A0A0

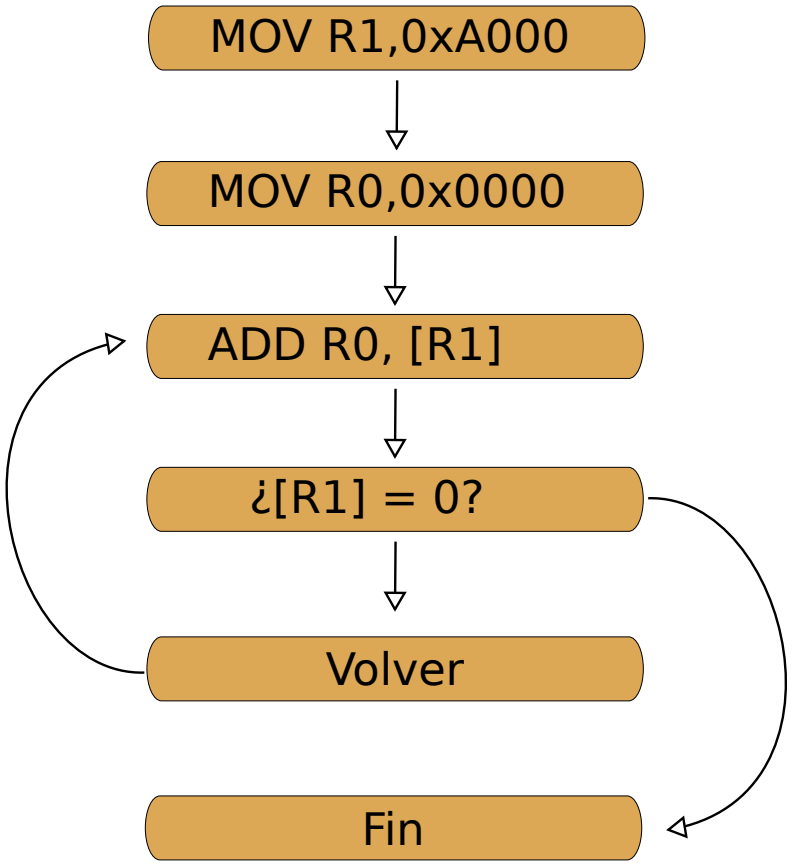


Revisando el programa de la rotisería

A partir de la celda A000 hay un arreglo que contiene los pedidos de empanadas de una rotisería, y que finaliza con el primer valor 0. Sumar todos los valores

Revisando el programa de la rotisería

A partir de la celda A000 hay un arreglo que contiene los pedidos de empanadas de una rotisería, y que finaliza con el primer valor 0. Sumar todos los valores



Ejercicio: Completar el programa

Ejercicio

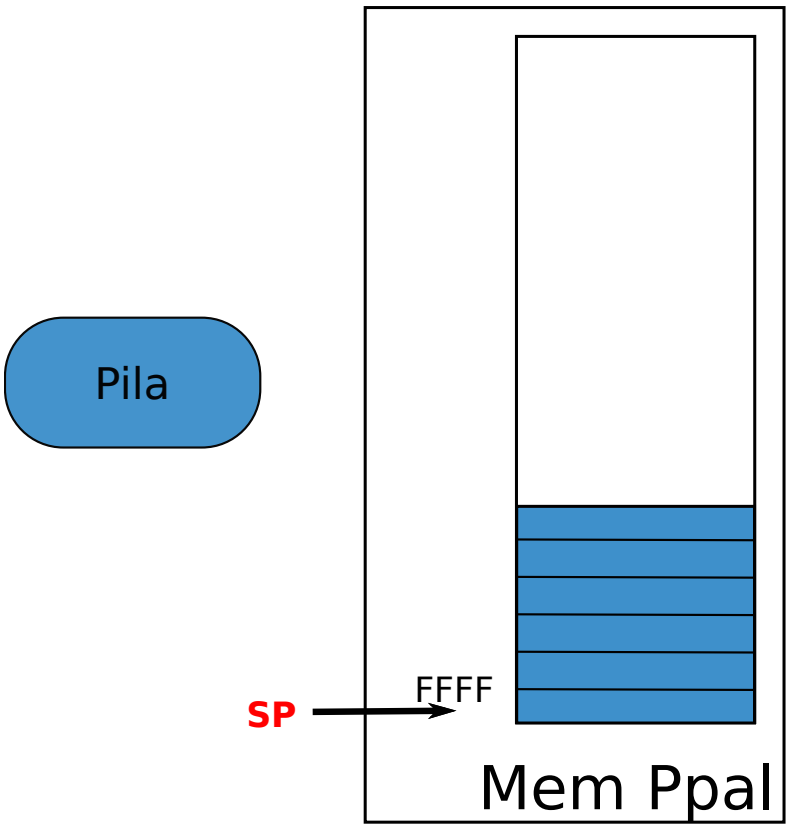
Ejercicio: A partir de la celda B0B0 hay un arreglo con las temperaturas de una cierta localidad, y que finaliza con el primer valor 0. Calcular el promedio

La Pila

Estructura de Pila (Stack)

- La pila es un sector especial de la memoria
- Los datos se organizan *apilados*:
  1. Cuando se escribe en la pila, se lo agrega "sobre" el último agregado
  2. Cuando se lee de la pila, se lo saca del "tope" de la pila
- El seguimiento del tope de pila se lleva mediante un registro especial SP (Stack Pointer)

Estructura de Pila



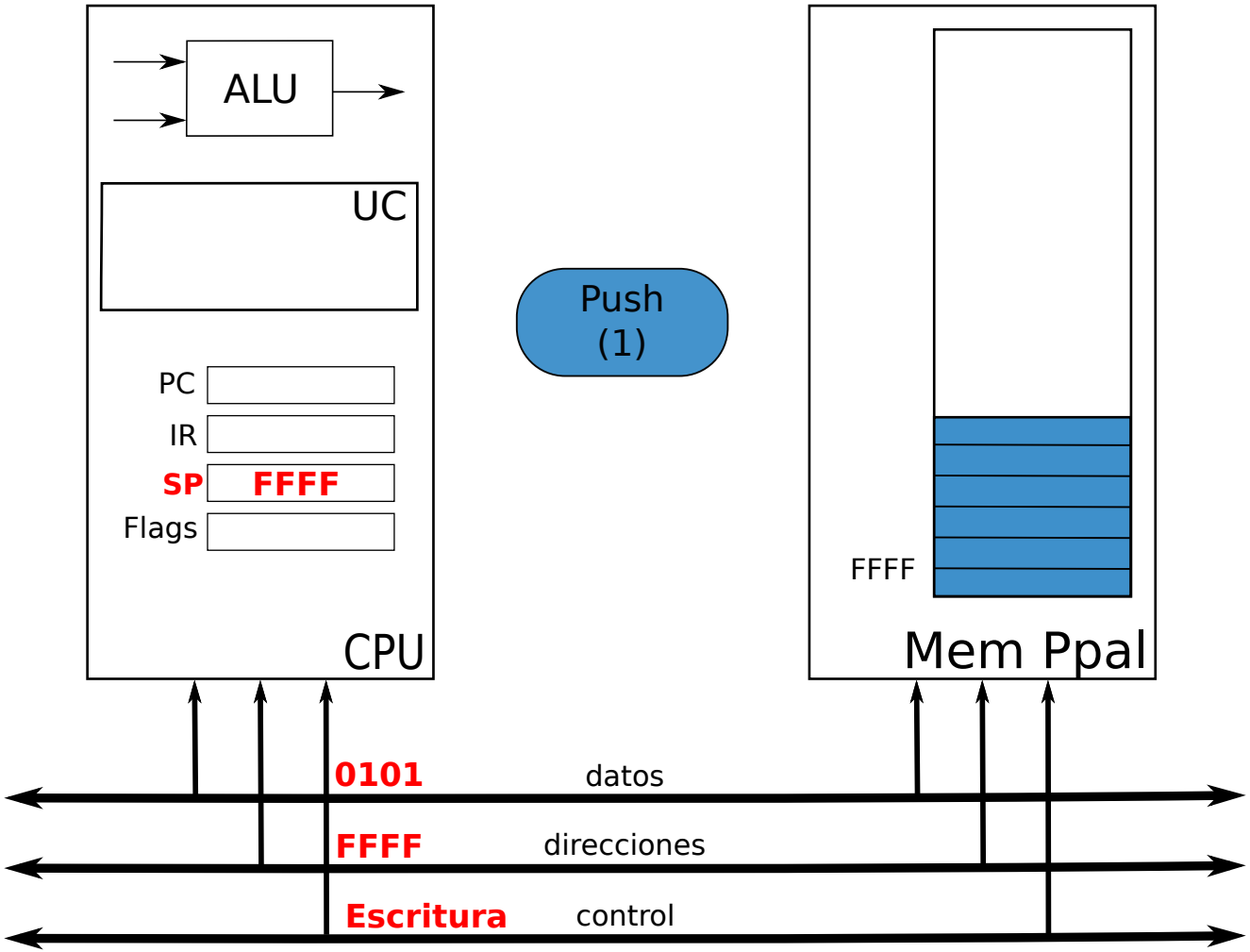
SP (Stack Pointer) contiene la dirección de la primer celda de memoria **disponible** de la pila.

Estructura de Pila: Push

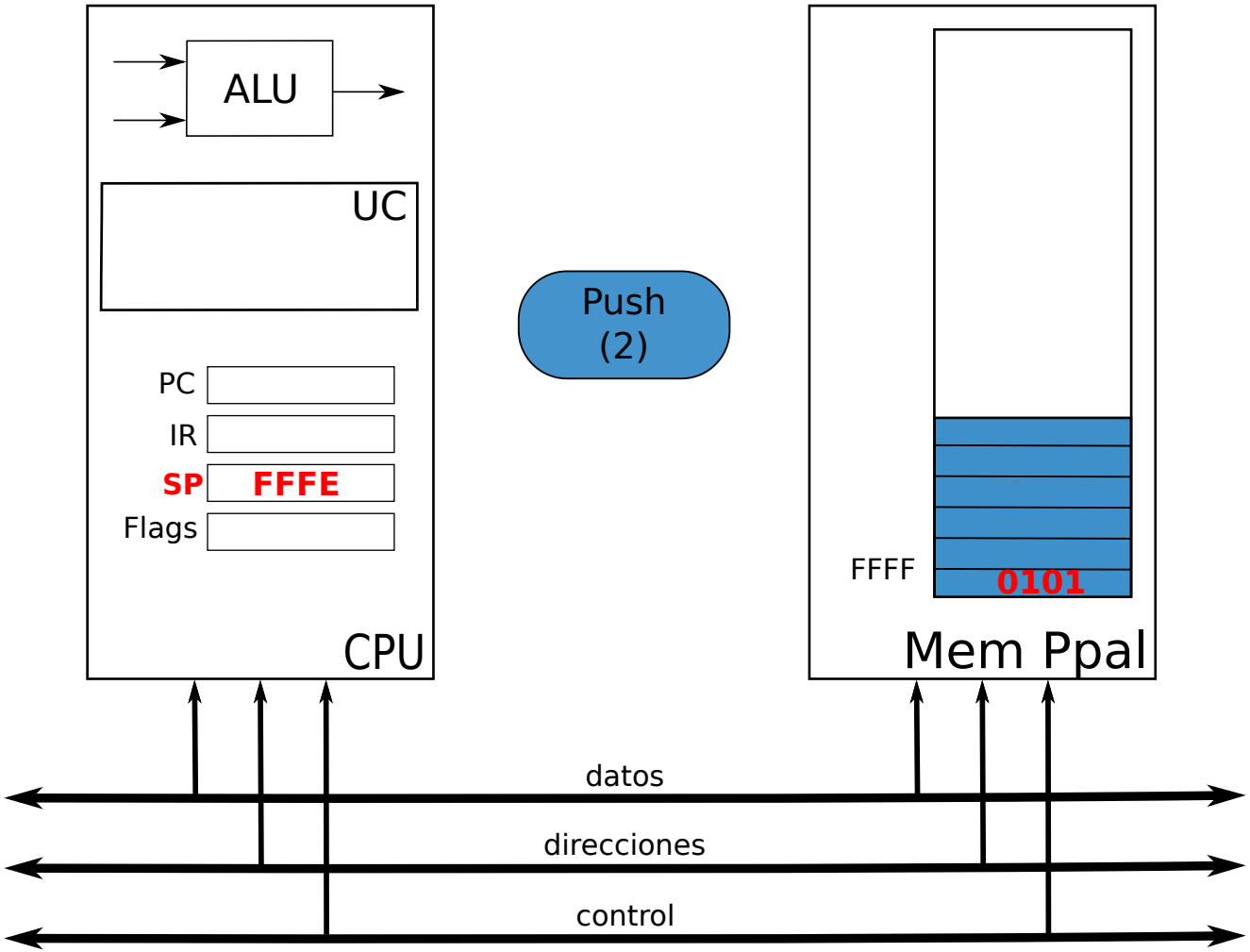
Push

1. Se hace una escritura del dato que está en el bus de datos en la dirección que está en SP
2. Se decrementa SP (así sigue cumpliendo la condición)

Estructura de Pila: Push



Estructura de Pila: Push

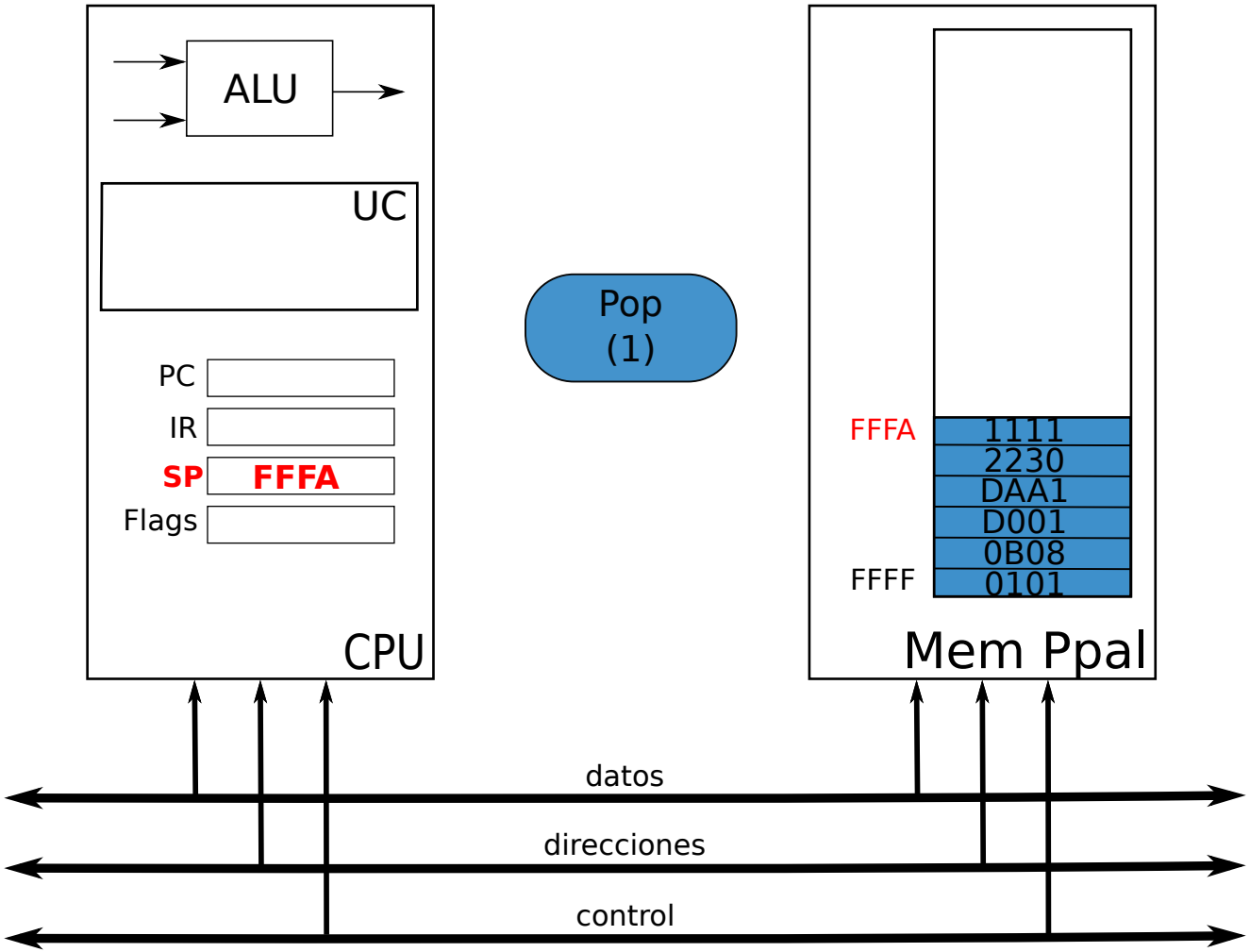


Estructura de Pila: Pop

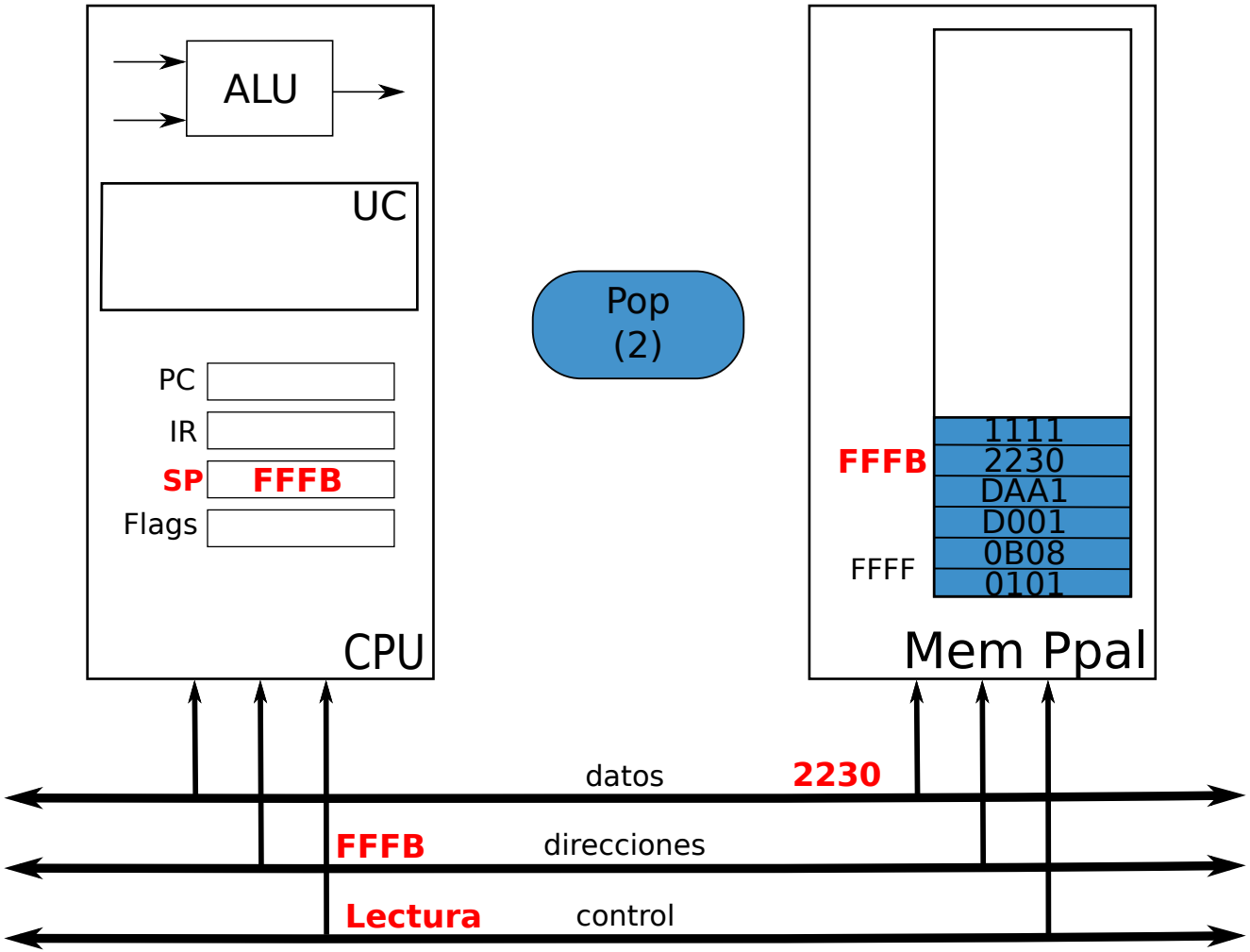
Pop

1. Se incrementa SP (para que haga referencia a un dato dentro de la pila)
2. Se hace una lectura de la dirección que está en SP

Estructura de Pila: Pop



Estructura de Pila: Pop



Estructura de Pila

- El tamaño y la ubicación de la pila está definido por la arquitectura.
- El pop no blanquea el tope de la pila.
- Cuando se hace push se pierde el valor que tenía la celda (por definición de escritura)

Estructura de Pila

sumar los 2 números al tope de la pila y apilar el resultado

Estructura de Pila

sumar los 2 números al tope de la pila y apilar el resultado

```
POP R1
POP R2
ADD R1,R2
PUSH R1
```

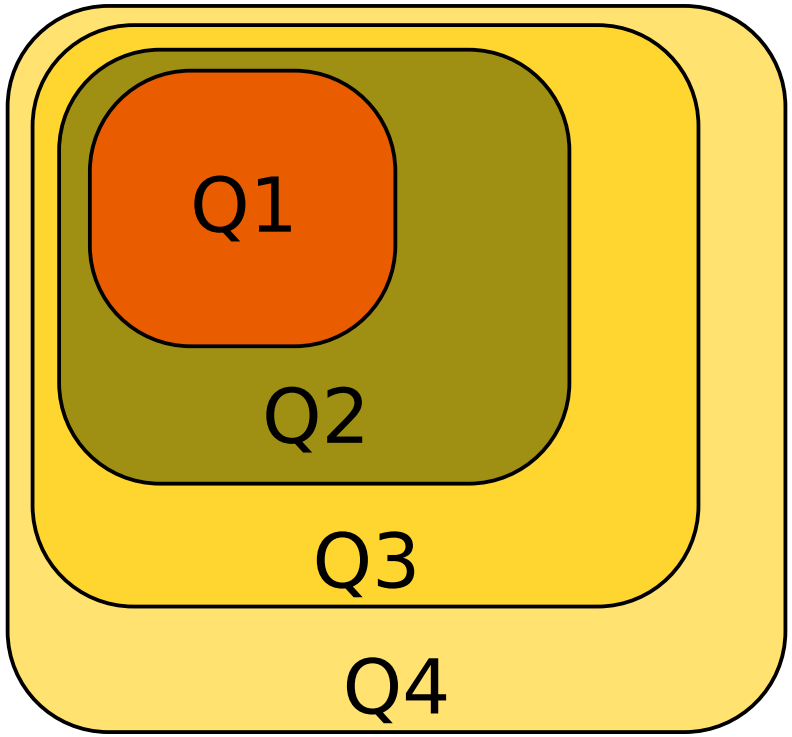
Arquitecturas **Q**

... Si todavía tenés ganas de más

...

Arquitectura **Q4**

Arquitectura **Q3**



Arquitectura **Q3**

- Tiene 8 registros de uso general de 16 bits: R0..R7
- Tiene direcciones de 16 bits
- Tiene registros no visibles al programador:
  - *Program counter* de 16 bits.
  - Registro de flags (**ZNCV**) de 16 bits.
  - *Stack Pointer* de 16 bits. Comienza en la dirección FFEF .
- permite 3 modos de direccionamiento:
  - modo registro: el valor buscado está en un registro
  - modo inmediato: el valor buscado está codificado dentro de la instrucción
  - modo directo: el valor buscado está contenido en una celda de memoria
  - modo indirecto: la dirección del valor buscado está contenido en una celda de memoria
  - modo registro indirecto: la dirección del valor buscado está contenido en un registro

Arquitectura **Q4**: formato de instrucciones

- Operaciones de tipo 1 (MUL,MOV,ADD,SUB,CMP,DIV,AND,OR)

Cod_Op (4b)	Modo Destino (6b)	Modo Origen (6b)	Operando Destino (16b)	Operando Origen (16b)

- Operaciones de tipo 2 (Un operando Origen)

Cod_Op (4b)	Relleno (000000)	Modo Origen (6b)	Operando Origen (16b)

- Operaciones de tipo 2 (Un operando Destino)

Cod_Op (4b)	Modo Destino (6b)	Relleno (000000)	Operando Origen (16b)

- Operaciones de tipo 3 (Saltos incondicionales y relativos)

Prefijo (1111)	Cod_Op (4)	Desplazamiento(8) (8b)

Arquitectura **Q4**: Operaciones de tipo 1

Tipo 1: Aritméticas y lógicas

Cod_Op (4b)	Modo Destino (6b)	Modo Origen (6b)	Operando Destino (16b)	Operando Origen (16b)

Operación	**CodOp**
MUL	0000
MOV	0001
ADD	0010
SUB	0011
CMP	0110
DIV	0111
AND	0100
OR	0101

Arquitectura **Q4**: Operaciones de tipo 2

Tipo 2: Un operando Origen

Cod_Op (4b)	Relleno (000000)	Modo Origen (6b)	Operando Origen (16b)

Operación	**CodOp**	**Efecto**
JMP	1010	PC ← dirección
PUSH	1110	[SP] ← Origen; SP ← SP - 1

Arquitectura **Q4**: Operaciones de tipo 3

Tipo 3: Un operando Destino

Cod_Op (4b)	Modo Destino (6b)	Relleno (000000)	Operando Origen (16b)

Operación	**CodOp**	**Efecto**
NOT	1001	Dest $\leftarrow$ NOT Dest (bit a bit)
POP	1101	SP $\leftarrow$ SP + 1; Dest $\leftarrow$ [SP]

Arquitectura **Q4**: Operaciones de tipo 4

Tipo 4: Salto condicional (relativo) - 1 de 2

Prefijo (1111)	Cod_Op (4b)	Desplazamiento(8b)

Salto	**Codop**	**Descripción**	**Condición**
JE	0001	Igual / Cero	Z
JNE	1001	No igual	$\overline{Z}$
JLEU	0100	Menor o igual sin signo	$C \vee Z$
JGU	1100	Mayor sin signo	$\overline{(C \vee Z)}$
JCS	0101	Menor sin signo	C
JNEG	0110	Negativo	N

Arquitectura **Q4**: Operaciones de tipo 4

Tipo 4: Salto condicional (relativo) - 2 de 2

Prefijo (1111)	Cod_Op (4b)	Desplazamiento(8b)

Salto	**Codop**	**Descripción**	**Condición**
JVS	0111	Overflow	V
JLE	0010	Menor o igual con signo	$Z^{\vee}(N \oplus V)$
JG	1010	Mayor con signo	$\overline{(Z^{\vee}(N \oplus V))}$
JL	0011	Menor con signo	$N \oplus V$
JGE	1011	Mayor o igual con signo	$\overline{(N \oplus V)}$

¿Preguntas?