Organización de computadoras

Clase 9

Universidad Nacional de Quilmes

Lic. Martínez Federico

¿Qúe pasó?

- Limitaciones de Q3

¿Qúe pasó?

- Limitaciones de Q3
- Flags:
  - ¿Qué?
  - ¿Cómo?
  - ¿Para qué?

¿Qúe pasó?

- Limitaciones de Q3
- Flags:
  - ¿Qué?
  - ¿Cómo?
  - ¿Para qué?
- Saltos:
  - ¿Qué?
  - Absolutos vs relativos
  - Condicionales vs incondicionales

Lo que viene, lo que viene

- Mascaras

Lo que viene, lo que viene

- Mascaras
- Repeticiones controladas

Lo que viene, lo que viene

- Mascaras
- Repeticiones controladas
- Arreglos

Lo que viene, lo que viene

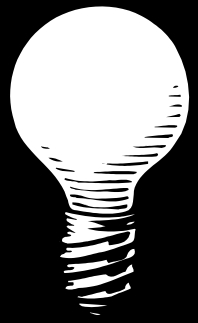
- Mascaras
- Repeticiones controladas
- Arreglos
- Modo indirecto

Lo que viene, lo que viene

- Mascaras
- Repeticiones controladas
- Arreglos
- Modo indirecto
- Q5

Lógica

Lógica

0 →  → Falso

1 →  → Verdadero

Operaciones lógicas

- Se aplican sobre cadenas
- Actúan Bit a Bit

Operaciones lógicas

- AND: Realiza el “Y” lógico entre los bits

Operaciones lógicas

- AND: Realiza el “Y” lógico entre los bits

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Operaciones lógicas

- AND: Realiza el “Y” lógico entre los bits

```
  1010101010
and
  1100101100
```

Operaciones lógicas

- AND: Realiza el “Y” lógico entre los bits

```
  1010101010
and
  1100101100
  ----------
  1000101000
```

Operaciones lógicas

- OR: Realiza el “O” lógico entre los bits

Operaciones lógicas

- OR: Realiza el “O” lógico entre los bits

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Operaciones lógicas

- OR: Realiza el “O” lógico entre los bits

```
  1010101010
or
  1100101100
```

Operaciones lógicas

- OR: Realiza el “O” lógico entre los bits

or

1010101010
1100101100
1110101110

Operaciones lógicas

- Not: Realiza la negación de los bits

Operaciones lógicas

- Not: Realiza la negación de los bits

A	NOT A
0	1
1	0

Operaciones lógicas

- Not: Realiza la negación de los bits

not 1100101100

Operaciones lógicas

- Not: Realiza la negación de los bits

not 1100101100

0011010011

Operaciones lógicas

- XOR: Realiza el “O exclusivo” lógico entre los bits

Operaciones lógicas

- XOR: Realiza el “O exclusivo” lógico entre los bits

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Operaciones lógicas

- XOR: Realiza el “O exclusivo” lógico entre los bits

```
          1010101010
xor       1100101100
```

Operaciones lógicas

- XOR: Realiza el “O exclusivo” lógico entre los bits

```
      1010101010
xor   1100101100
      ----------
      0110000110
```

Mascaras

Mascaras

- Cadenas binarias que se combinan con otras mediante operaciones lógicas

Mascaras

- Cadenas binarias que se combinan con otras mediante operaciones lógicas
- Sirven para analizar características de las cadenas

Mascaras

- Con AND:
  - Si queremos preservar el bit original: 1
  - Si queremos dejar un bit en 0: 0

Mascaras

- Con AND:
  - Si queremos preservar el bit original: 1
  - Si queremos dejar un bit en 0: 0
- Con OR:
  - Si nos interesa el bit original: 0
  - Si queremos que quede un 1: 1

Mascaras

Ej: ¿La cadena en R0 es impar?

Mascaras

Ej: ¿La cadena en R0 es impar?

```
AND R0, 0x0001
JNE TerminabaEnCero
```

Mascaras

Ej: Copiar los 3 bits mas significativos de R0 a R1

Mascaras

Ej: Copiar los 3 bits mas significativos de R0 a R1

```
MOV R1, 0xE000
AND R1, R0
```

Mascaras

- Permisos de archivos en Linux:
  - 3 bits: leer (r) , escribir (w), ejecutar (x)
  - Usuario, Grupo y Otros
  - 111 111 111 le da permisos a todos para hacer cualquier cosa
  - ¿Cómo saber si otro usuario del grupo del archivo puede escribirlo?
    - AND 000 010 000

Repeticiones controladas

- Mi trabajo es ser repetitivo
- Mi trabajo
- Mi trabajo
- Repetir es mi trabajo

Repeticiones controladas

- Los saltos condicionales nos permiten flujos alternativos de ejecución

Repeticiones controladas

- Los saltos condicionales nos permiten flujos alternativos de ejecución

- Además podemos usarlos para repetir un bloque de código varias veces

Repeticiones

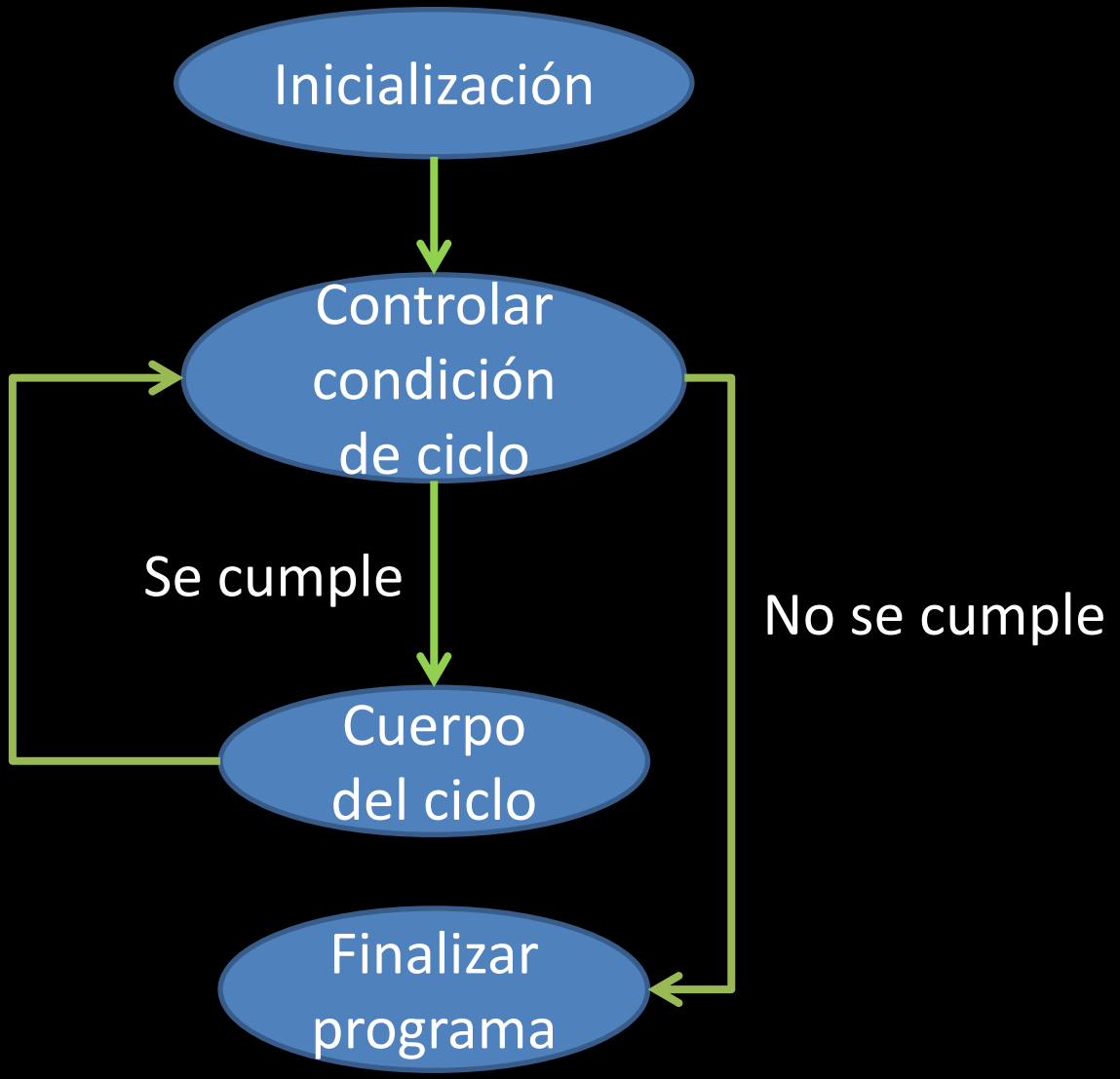
- A está en R0, y B en R1. Hacer A*B sin usar MUL

Repeticiones

- A está en R0, y B en R1. Hacer A*B sin usar MUL

```
tot=0
Repetir B veces:
    tot = tot + A
```

Repeticiones



Repeticiones

- Calcular N a la M. N en R0 y M en R1.

Repeticiones

- Calcular N a la M. N en R0 y M en R1.

- Calcular el resto de N divido M sin DIV

Arreglos

Arreglos

- Posiciones de memoria consecutivas

Arreglos

- Posiciones de memoria consecutivas
- Colección de datos

Arreglos

- Posiciones de memoria consecutivas
- Colección de datos
- Los elementos no necesariamente ocupan una celda

Arreglos

- Posiciones de memoria consecutivas
- Colección de datos
- Los elementos no necesariamente ocupan una celda
- El tamaño del arreglo es la cantidad de elementos

Arreglos

- Ejemplo:

Posición	Contenido
0xF000	0x1020
0xF001	0xDCE3
0xF002	0xE451
0xF003	0x29C8

Arreglos

- Ejemplo:

Posición	Contenido
0xF000	0x1020
0xF001	0xDCE3
0xF002	0xE451
0xF003	0x29C8

Un arreglo de 4 números de 16 bits

Arreglos

- Ejemplo:

Posición	Contenido
0xF000	0x1020
0xF001	0xDCE3
0xF002	0xE451
0xF003	0x29C8

Un arreglo de 4 números de 16 bits

Un arreglo de 8 números de 8 bits

Arreglos

- Ejemplo:

Posición	Contenido
0xF000	0x1020
0xF001	0xDCE3
0xF002	0xE451
0xF003	0x29C8

Un arreglo de 4 números de 16 bits
Un arreglo de 8 números de 8 bits
Un arreglo de 2 números de 32 bits

Arreglos

- ¿Cómo saber donde termina el arreglo?
  - Nos dicen el tamaño
  - Nos dicen que termina con algún valor especial (i.e: 0)

Ejercicio

- A partir de 0x000F hay un arreglo de números. El último de ellos es 0. Hacer un programa que sume todos esos números en R3

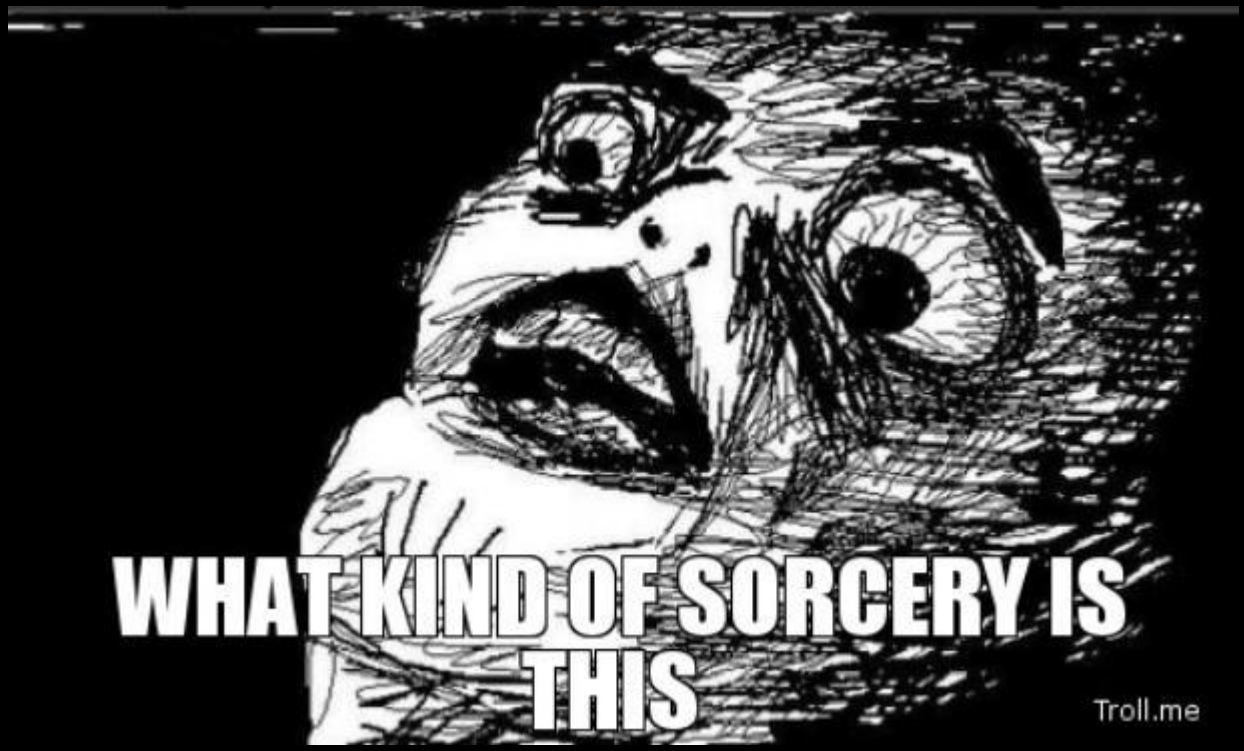

¡¡ISABEL!!

Ejercicio

- Se puede hacer si el programa modifica su propio código a medida que se ejecuta

Ejercicio

- Se puede hacer si el programa modifica su propio código a medida que se ejecuta



Ejercicio

- Nos gustaría poder referenciar a memoria sin usar una constante

Ejercicio

- Nos gustaría poder referenciar a memoria sin usar una constante

- Necesitamos un nuevo modo de direccionamiento

Modo indirecto

- Especificamos la celda o el registro en el que se encuentra la dirección del operando

Indirecto registro

- Se denota $[R_i]$

Indirecto registro

- Se denota $[R_i]$

- Semántica: El valor que está contenido en la celda cuya dirección está en R_i

Indirecto registro

R5 = AA00

Dirección	Contenido
AA00	AA02
AA01	5647
AA02	4FFA

¿Qué valor queda en R0 si hacemos MOV R0, [R5] ?

Indirecto

- Se denota [[*Cte*]]

Indirecto

- Se denota [[*Cte*]]

- Semántica: El valor que está contenido en la celda cuya dirección está en contenida en la celda *Cte*

Indirecto

Dirección	Contenido
B100	B102
B101	5647
B102	EDAB

¿Qué valor queda en R0 si hacemos MOV R0, [[B100]]?

Indirecto

- Requiere 2 accesos a memoria para obtener el operando

- Para guardarlo se necesita 1 solo, porque ya queda calculada la dirección de la celda

Ejercicio

Dirección	Contenido
0x0000	0x0007
0x0001	0x0000
0x0002	0x2345
0x0003	0xFEDE
0x0004	0x0002
0x0005	0x0006
0x0006	0x0003
0x0007	0x0001

R1 = 0x0005 R2 = 0x0003

¿Qué valor queda en R0?

- MOV R0, [[0x0006]]
- MOV R0, [[0x0001]]
- MOV R0, [[0x004]]
- MOV R0, [R1]
- MOV R0, [R2]

Ejercicio

- A partir de 0x000F un arreglo de números. El último de ellos es 0. Hacer un programa que sume todos esos números en R3

(In)directamente te va a encantar

- Nuevos modos de direccionamiento

Modo	Código
Inmediato	000000
Registro	100RRR
Directo	001000
Indirecto	011000
Registro Indirecto	110RRR

- Formato de instrucción:
  - Instrucciones tipo 1:

Cod Op (4bits)	Modo Destino (6 bits)	Modo origen (6 bits)	Destino (16 bits)	Origen (16 bits)

Operación	Código	Efecto
MUL	0000	Dest ←Dest * Origen
MOV	0001	Dest← Origen
ADD	0010	Dest ← Dest + Origen
SUB	0011	Dest← Dest - Origen
DIV	0111	Dest← Dest% Origen
CMP	0110	Modifica los Flags según el resultado de Dest – Origen

- Formato de instrucción:
  - Instrucciones tipo 2:

Cod Op (4 bits)	Relleno (000000)	Modo Origen (6 bits)	Origen (16 bits)

- Operaciones tipo 2:

Operación	Código	Efecto
CALL	1011	[SP] ←PC; SP←SP-1; PC ←Origen
JMP	1010	PC ← Origen

- Formato de instrucción:
  - Instrucciones tipo 3:

Cod Op (4 bits)	Relleno (000000000000)

- Operaciones tipo 3:

Operación	Código	Efecto
RET	1100	PC ←[SP+1]; SP←SP + 1

- Formato de instrucción:
  - Instrucciones tipo 4:

Prefijo (1111)	Cod Op (4 bits)	Desplazamiento (8 bits)

- Operaciones tipo 4:

Salto	Codop	Descripción	Condición
JE	0001	Igual / Cero	Z
JNE	1001	No igual	¬ Z
JLE	0010	Menor o igual con signo	Z + (N ⊕ V)
JG	1010	Mayor con signo	¬(Z + (N ⊕ V))
JL	0011	Menor con signo	N ⊕ V
JGE	1011	Mayor o igual con Signo	¬ (N ⊕ V)
JLEU	0100	Menor o igual sin signo	C + Z
JGU	1100	Mayor sin signo	¬(C + Z)
JCS	0101	Menor sin signo	C
JNEG	0110	Negativo	N
JVS	0111	Overflow	V

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. Considerar que 0 no es una temperatura en la localidad.

Ejercicio

- Ensamblar:
  - MOV R0, [R1]
  - ADD R0, [[0xF0CA]]
  - SUB [0x1111],][0x1111]]

Ejercicio

- Completar la cantidad de accesos a memoria en la siguiente tabla:

Instrucción	FI	FO	ST
MOV R0, [R1]			
ADD R0, [[0xF0CA]]			
SUB [0x1111],][0x1111]]			
MUL [[0x0010]], [R5]			
DIV [R1], [[0x43AE]]			
MOV [[0xFEDE]], R1			

Ejercicio

- Hacer un programa que arme un arreglo a partir de la celda 0099 solo con las posiciones pares de otro arreglo que comienza en 1099 y que finaliza con el valor 0

Que aconteció en el día de la fecha

Que aconteció en el día de la fecha

- Mascaras

Que aconteció en el día de la fecha

- Mascaras
- Repeticiones controladas

Que aconteció en el día de la fecha

- Mascaras
- Repeticiones controladas
- Arreglos

Que aconteció en el día de la fecha

- Mascaras
- Repeticiones controladas
- Arreglos
- Modo indirecto

Que aconteció en el día de la fecha

- Mascaras
- Repeticiones controladas
- Arreglos
- Modo indirecto
- Q5


Graciela
ACIELA ALFANO

¿Preguntas?


NO ESTOY SEGURO DE
SI NO ENTENDI NADA
O ME QUEDE DORMIDO
memegenerator.es