



Práctica 3

Objetivo

Distinguir las características de la organización y arquitectura del microprocesador de una computadora de propósito general, analizando sus recursos de hardware y software, para conocer capacidades y limitaciones de forma organizada y responsable.

Desarrollo

Responda los siguientes cuestionamientos.

1. Explique las diferencias entre una unidad de control por hardware y por microprograma.

Control por hardware	Microcódigo
Implementado por medio de un circuito	Implementado por medio de programación
Instrucciones estilo RISC	Instrucciones estilo CISC
Difícil modificación ya que requiere modificación de circuitería	Modificaciones sencillas ya que solo requiere cambio en el código
Funciona bien con instrucciones simples	Funciona bien con instrucciones simples y complejas
No necesita memoria de control	Necesita memoria de control
Ejecución rápida	Ejecución relativamente lenta

2. Una computadora tiene una unidad de memoria con 32 bits por palabra. Su conjunto de instrucciones consiste en 110 operaciones diferentes. Todas las instrucciones se componen de un código de operación (opcode) y dos campos operandos: uno para una dirección de memoria y otro para una dirección de registro. La computadora tiene 8 registros de propósito general. Los registros pueden cargar datos de memoria, y la memoria se puede actualizar con datos de los registros. No hay soporte para movimientos de datos de memoria a memoria. Cada instrucción se almacena en una palabra de memoria.
 - a) ¿Cuántos bits de instrucción se necesitan para el opcode? **7**
 - b) ¿Cuántos bits de instrucción se necesitan para especificar el registro? **3**
 - c) ¿Cuántos bits de instrucción quedan para especificar la dirección de memoria? **22**
 - d) ¿Cuál es el tamaño máximo de memoria que puede tener el sistema? **16MB**
 - e) ¿Cuál es número sin signo más grande que se puede representar con una palabra de memoria? **4,294,967,295**

3. Escriba el código máquina del siguiente programa del simulador MARIE.

Dirección (HEX)	Etiqueta	Instrucción	Código Maquina (HEX)
100		LOAD A	1108
101		ADD ONE	3109
102		JUMP S1	9106
103	S2,	ADD ONE	3109
104		STORE A	2108
105		HALT	7000
106	S1,	ADD A	3108
107		JMP S2	9103
108	A,	HEX 0023	0023
109	One,	HEX 0001	0001

Tabla 1. Programa en lenguaje ensamblador de MARIE.

4. Escriba el lenguaje ensamblador (mnemónicos) equivalente a las siguientes instrucciones en código máquina de MARIE.
- | | |
|---------------------|--|
| a) 0111000000000000 | HALT |
| b) 1011001100110000 | ADDI X, donde "x" está en la dirección 330h |
| c) 0100111101001111 | SUBT Y, donde "y" está en la dirección 4F4h |

5. Escriba un programa que contenga la subrutina **Pirámide**, la cual despliega el patrón que se muestra en la Figura 1. El usuario ingresa en el código principal la cantidad de niveles a desplegar.

```
/Piramide
Input
store x          /Tamaño de la piramide
Subt TEN         /Si es mayor que 10
Skipcond 000
Jump @End        /Terminamos el programa
@Loop, Load number /número de ciclos
Store cycles/Numero a ciclar
@Loop2, Load number /Cargamos el número a ciclar
Add Unicode /Convertimos dec a Unicode
Output
Load cycles /Decrementamos numero ciclos
Subt ONE
Store cycles
Skipcond 800 /Si es positivo
Jump @EndLoop2 /Si no, cerramos ciclo
Jump @Loop2 /Ciclamos
@EndLoop2, Load lineFeed
Output
Load number /Aumentamos el numero
Add ONE
Store number
Load x /Decrementamos x
Subt ONE
Store x
Skipcond 800 /Si x es positivo
jump @End /si no, terminamos programa
Jump @Loop /Volvemos al ciclo

@End, Halt
lineFeed, HEX 0A /Salto de línea
Unicode, HEX 30 /Convertir un digito a su
valor Unicode
number, dec 1 /Numero a ciclar (Comenzamos en 1)
cycles, dec 0 /Numero de ciclos
ONE, DEC 1
TEN, DEC 10
x, DEC 0
```

OUTPUT MODE:

1
22
333
4444
55555

OUTPUT MODE:

1
22
333
4444
55555
666666
7777777
88888888

OUTPUT MODE:

1
22
333
4444
55555
666666
7777777
88888888
999999999

6. Escriba un programa que contenga la subrutina **Distancia Manhattan**, la cual recibe dos coordenadas en el plano cartesiano y despliega su distancia Manhattan en base a la fórmula:

$$|x_1 - x_2| + |y_1 - y_2|$$

El usuario ingresa en el código principal las coordenadas (x_1, y_1) y (x_2, y_2) .

```
/Distancia Manhattan  
/ |x1 - x2| + |y1 - y2|
```

```
Input  
Store x1  
Input  
Store y1
```

```
Input  
Store x2  
Input  
Store y2  
Load x1  
Subt x2  
Store tempX
```

```
Skipcond 000 /Si la resta da negativo  
Jump @Next  
Load x2 /Cambiamos el orden de los  
Subt x1 /operandos para obtener un valor  
positivo  
Store tempX
```

```
@next, Load y1  
Subt y2  
Skipcond 000 /Si la resta da negativo  
Jump @End  
Load y2 /Cambiamos el orden de los operandos para obtener un valor positivo  
Subt y1  
@end, Store tempY  
Add tempX  
Store distance  
Output  
Halt
```

```
distance, DEC 0  
tempX, DEC 0  
tempY, DEC 0  
x1, DEC 1  
x2, DEC 2  
y1, DEC 3  
y2, DEC 4
```

$x_1 = 10, y_1 = 12, x_2 = 20, y_2 = 24$

OUTPUT MODE: DEC ▼

6 ▲

$x_1 = 10, y_1 = 30, x_2 = -20, y_2 = 40$

OUTPUT MODE: DEC ▼

40 ▲

$x_1 = 50, y_1 = -50, x_2 = 60, y_2 = -60$

OUTPUT MODE: DEC ▼

20 ▲

Conclusiones y comentarios

RISC y CISC son las arquitecturas inicialmente utilizadas para facilitar el trabajo de los microprocesadores y homogeneizar las computadoras, debido a que RISC nació unos cuantos años después que CISC y que tenía un conjunto de instrucciones más sencilla, esta mejora el rendimiento y la capacidad de crear programas más largos y difíciles de desarrollar.

Dificultades en el desarrollo

La parte que más se me dificultó fue intentar obtener el valor absoluto de la resta en la distancia Manhattan ya que normalmente se hace (si $x < 0$, $x * = -1$), sin embargo, en Marie no tenemos la opción de multiplicar por negativos así que tuve que tomar una idea completamente diferente. Terminé optando por cambiar el orden de los operandos si el resultado fue negativo. Ej. $5-9 = -4$; $9-5 = 4$.

Referencias

T, N. (2022). Difference Between Hardwired and Microprogrammed Control Unit - Binary Terms. Retrieved 28 February 2022, from <https://binaryterms.com/difference-between-hardwired-and-microprogrammed-control-unit.html>

Anexos

Carpeta google drive con los códigos fuentes y videos de ejecución de código.

<https://drive.google.com/drive/folders/1yQKPpDuD6ZsblUTMpRfRJLEzhnNxixPuj?usp=sharing>