Universidad Autónoma de Baja California

Luis Eduardo Galindo Amaya
1274895

13 de agosto de 2022

Asignatura:	Arquitectura de Computadoras
Docente:	Arreola Alvarez Arturo
Grupo:	361

Objetivo

El alumno se familiarizará con la herramienta Marie.js para la ejecución de código en lenguaje ensamblador.

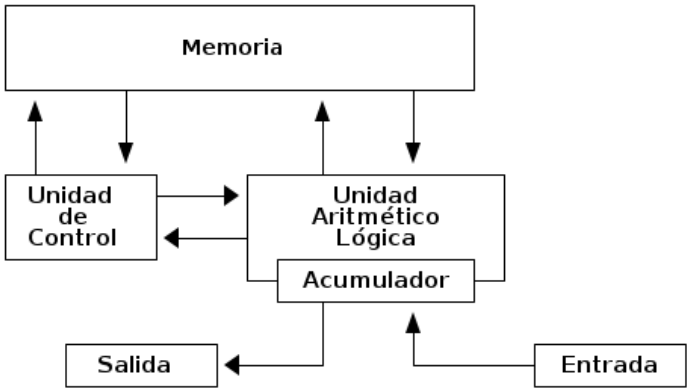
Equipo

Computadora personal con acceso a Internet.

Desarrollo

1. Ingrese a la página donde se encuentra la herramienta.
2. Consulte la documentación y tutoriales de Marie, en especial las secciones Introduction to Marie y Marie Codes.
3. Escriba un código que solicite al usuario 2 números y despliegue el resultado de la ecuación $2x + 3y - 5$.
4. Escriba un código que solicite 2 números y los reste. Desplegar un 1 si el resultado fue negativo o un 0 en caso contrario.
5. Realice capturas de pantalla donde se muestre el funcionamiento de los programas.
6. Realizar un reporte que incluya:
   - Diagrama de bloques de una máquina de von Neumann y una breve descripción de cada componente.
   - Lista de las instrucciones de Marie y su y su función
   - Describir el funcionamiento de los registros del Acumulador, Registro de instrucción, Contador del Programa, Registro de Acceso a Memoria (MAR), y Registro de Buffer de Memoria (MBR).
7. Al entregar su práctica, adjunte los códigos de sus programas y su reporte.

Máquina De Von Neumann



Memoria Son un conjunto de celdas usadas para cualquier proceso, se utiliza para almacenar los programas qu

Memoria Son un conjunto de celdas usadas para cualquier proceso, se utiliza para almacenar los programas que va a ejecutar el procesador e información que el programa necesite almacenar.

Unidad Aritmético Lógica es un circuito digital electrónico que realiza las operaciones aritméticas y lógicas bit a bit en n´umeros binarios enteros.

Unidad de Control es un componente que dirige las operaciones del procesador. Le dice a la memoria, ALU y los dispositivos de entrada y salida cómo responder a las instrucciones de un programa.

Entrada/Salida Es la comunicación entre la computadora y un humano u otro sistema de procesamiento de información. Inputs son las señales o datos recibidos por el sistema y outputs son las señales o datos enviados por éste.

Registros son unidades de almacenamiento pequeñas que son típicamente dirigidas por mecanismos distintos de la memoria principal y a los que se puede acceder más rápido.

Programa 1

```
INPUT
Store X
INPUT
Store Y

Clear    / limpiar Acumulador

Add X    / 2x
Add X
Add Y    / 3y
Add Y
Add Y
Subt R   / -5

Output
Halt

X, DEC 0
Y, DEC 0
R, DEC 5
```

Programa 2

```
/ Escriba un código que solicite 2 números y los reste.
/ Desplegar un 1 si el resultado fue negativo o un 0 en caso contrario.


INPUT                           / Captura el valor de X
Store X
INPUT                           / Captura un valor de Y
Store Y

load X                          / Carga el valor de X en el acumulador
Subt Y                          / Resta el valor de Y a X

                                / Para ese punto, el valor en AC es 'X - Y'

Skipcond 000                    / si AC es mayor o igual a 0
clear                           / el valor en AC se vuelve 0

Skipcond 400                    / si es diferente a 0
Load verdadero_negativo         / el valor en AC se vuelve 1

Output
Halt

X, DEC 0
Y, DEC 0
verdadero_negativo, dec 1
```

Lista De Las Instrucciones

Type	Instruction	Hex Opcode	Summary
Arithmetic	Add X	3	Adds value in AC at address X into AC, AC ← AC + X
	Subt X	4	Subtracts value in AC at address X into AC, AC ← AC - X
	AddI X	B	Add Indirect: Use the value at X as the actual address of the data operand to add to AC
	Clear	A	AC ← 0
Data Transfer	Load X	1	Loads Contents of Addr. X into AC
	Store X	2	Stores Contents of AC into Addr. X
I/O	Input	5	Request user to input a value
	output	6	Prints value from AC
Branch	Jump X	9	Jumps to Address X
	Skipcond (C)	8	Skips the next instruction based on C: if (C) = - 000: Skips if AC < 0 - 400: Skips if AC = 0 - 800: Skips if AC > 0
Subroutine	JnS X	0	Jumps and Store: Stores value of PC at address X then increments PC to X+1
	JumpI X	C	Uses the value at X as the address to jump to
Indirect Addressing	LoadI	D	Stores value in AC at the indirect address.
	StoreI	E	Stores value in AC at the indirect address.
	Halt	7	End the program