Tecnológico de Monterrey

Instituto Tecnológico de Estudios Superiores de Monterrey

Campus Estado de México (CEM)

Ingeniería en Sistemas Computacionales

Segundo Semestre

Materia: Organización computacional

Profesor: Patricia Chávez, Víctor Ferman

Proyecto final: Emulador IAS

Integrantes del equipo:

A01372812 José Javier Rodríguez Mota

A01373240 Miguel Angel Elizalde Cruz

A01373264 Carlos E. Carbajal Nogués

A01373302 Ian Gonzalez Palmanes

La máquina IAS (Institute for Advanced Study) recibe su nombre de la institución donde se construyó durante seis años (1945-1951) en Princeton, esta fue la primera computadora electrónica construida con base en la arquitectura Von Neumann que es hoy en día la base de la computación moderna. Este tipo de arquitectura para computadoras digitales establece que deben existir en un computador digital tres unidades principales: Procesamiento, Memoria y I/O. El procesamiento está a cargo del CPU (Central Processing Unit) que a su vez se divide en una Unidad de Control (UC) y una Unidad Aritmética-Lógica (ALU).

La máquina de Von Neumann contaba con 5.1KB de memoria dividida en palabras de 40 bits cada una, pudiendo almacenar hasta 1024 palabras. Cada una de las palabras se dividía a su vez en dos instrucciones (derecha o izquierda) donde los primeros 8 bits eran los bits de operación (también llamado Opcode) y los 12 restantes se utilizaban para la dirección en memoria a la que apuntaba el ordenador. Esta máquina podía representar dos tipos de enteros (positivos y negativos) en binario y complemento a 2 respectivamente, queriendo decir que el número más grande que se podía almacenar era el 549 755 813 888 (7FFFFFFFFF) y el más pequeño el -549 755 813 888 (80000000000). Es importante mencionar que la máquina IAS podía utilizar únicamente la parte derecha o la parte izquierda de la memoria a conveniencia del programador.

Este proyecto es justamente un emulador de la máquina de Von Neumann, escrito en Javascript, HTML y CSS que busca emular las capacidades de dicho computador e interpretar las instrucciones como si fuese la misma máquina, es por ello que cuenta con las características en cuanto al tamaño de la palabra, el modo de lectura de instrucciones y los opcodes de la misma.

IASemu Manual de uso

Contenido

Interfaz general

Al iniciar IASemu podrás observar una serie de botones e información actual del computador (véase ilustración 1.1). En el selector de base podremos elegir el tipo de números que queremos observar durante la ejecución del programa, es importante mencionar que todo dato en memoria se guarda en formato hexadecimal y esta opción únicamente afecta a los números que el usuario utiliza durante la ejecución del programa deseado.

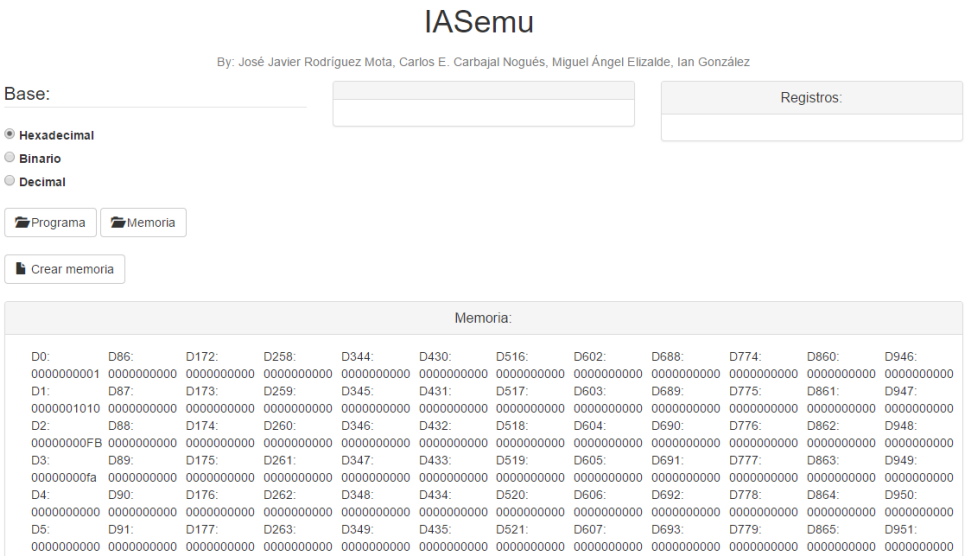

Ilustración 1.1

Justo debajo de la selección de base, se pueden apreciar tres botones principales: Programa, Memoria y Crear Memoria. Los primeros dos sirven para subir el programa que se desea ejecutar y alterar la memoria predeterminada, esta memoria únicamente servirá durante la ejecución del programa y si se desea guardar para un uso posterior, se deberá hacer clic en el botón de crear memoria como se explica en la sección “Escritura de datos en memoria”.

Ilustración 1.2

En la siguiente columna se encuentra el apartado donde aparecerá el programa en mnemónicos y las opciones de ejecución de programa (explicadas en la sección “Lectura de programa”) como se

muestra en la ilustración 1.2. En la última columna de la primera fila se ve la sección de registros donde aparecerán el Contador de Programa (PC) el Acumulador (AC) y el Multiplicador/Cociente (MQ) como se muestra en la ilustración 1.3.

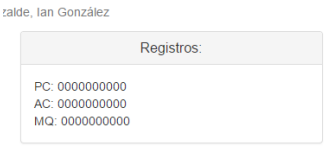

Ilustración 1.3

Toda la segunda fila está dedicada a la memoria, donde se puede leer “Dx:” siendo x el bloque de memoria que se está observando, seguido del contenido de la memoria en hexadecimal como se muestra en la ilustración 1.4.

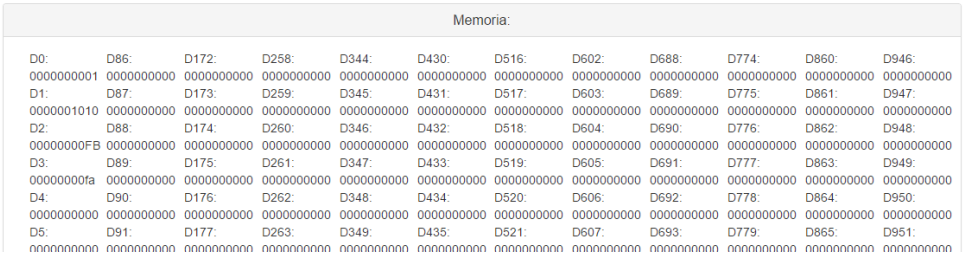

Ilustración 1.4

Lectura de programa

En esta sección se hablará de cómo cargar un programa a IASemu así como las opciones de ejecución del mismo programa. La máquina IAS contaba con 21 instrucciones distintas, IASemu cuenta con interpretación para las 21 instrucciones, más 1 instrucción extra como se muestra en la siguiente tabla:

Opcode	Mnemónico	Descripción
0A	LOAD MQ	Carga en el registro AC el contenido del registro MQ.
09	LOAD MQ, M(x)	Carga en el registro MQ el contenido de la ubicación en memoria M(X).
21	STOR M(X)	Guarda el contenido deL registro AC en la ubicación en memoria M(X).
01	LOAD M(X)	Carga en el registro AC el contenido de la ubicación en memoria M(X).

02	LOAD -M(X)	Carga en el registro AC el contenido de la ubicación en memoria M(X) y lo multiplica por -1.
03	LOAD |M(x)|	Carga en el registro AC el valor absoluto del contenido de la ubicación en memoria M(X).
04	LOAD -|M(x)|	Carga en el registro AC el valor absoluto del contenido de la ubicación en memoria M(X) y lo multiplica por -1.
0D	JUMP M(X,0:19)	Cambia el registro PC al número X y establece que la siguiente instrucción a ejecutar es la del lado izquierdo.
0E	JUMP M(X,20:39)	Cambia el registro PC al número X y establece que la siguiente instrucción a ejecutar es la del lado derecho.
0F	JUMP M(X,0:19)	Si AC es positivo, cambia el registro PC al número X y establece que la siguiente instrucción a ejecutar es la del lado izquierdo.
10	JUMP M(X,20:39)	Si AC es positivo, cambia el registro PC al número X y establece que la siguiente instrucción a ejecutar es la del lado derecho.
05	ADD M(X)	Suma el contenido de la ubicación en memoria M(X) al registro AC.
07	ADD |M(X)|	Suma el valor absoluto del contenido de la ubicación en memoria M(X) al registro AC.
06	SUB M(X)	Resta el contenido de la ubicación en memoria M(X) al registro AC.
08	SUB |M(X)|	Resta el valor absoluto del contenido de la ubicación en memoria M(X) al registro AC.
0B	MUL M(X)	Multiplica el contenido de la ubicación en memoria M(X) por el registro MQ y almacena

		las cifras más significativas en AC y las menos significativas en MQ.
0C	DIV M(X)	Divide el registro AC entre el contenido en memoria M(X) y pone el cociente en MQ y el remanente en AC.
14	LSH	Multiplica AC por dos.
15	RSH	Divide AC entre dos.
12	STOR M(x8:19)	Reemplaza los dígitos de dirección de la instrucción izquierda del contenido en memoria M(X) por los 12 primeros dígitos de AC.
13	STOR M(x28:39)	Reemplaza los dígitos de dirección de la instrucción derecha del contenido en memoria M(X) por los 12 primeros dígitos de AC.
00	HALT	Termina la ejecución del programa.

Escribir el programa

Para realizar un programa, se recomienda fijarse en la tabla anterior, así como en el formato de los archivos de ejemplo, recordando que cada línea del archivo deberá tener exactamente 10 caracteres en hexadecimal, representando los primeros cinco la instrucción izquierda y los restantes la derecha como se muestra en la ilustración 2.1 que es el ejemplo de programa 2. De igual manera, se recomienda observar la memoria predeterminada que sigue el mismo formato y si así se desea editar la misma como se explica en la sección “Escritura de datos en memoria” para poder iniciar el programa, ya que para cada instrucción los primeros dos caracteres hexadecimales son el código de operación (opcode) de la tabla anterior y los tres caracteres restantes indican la posición de memoria a la que dicho opcode recurrirá (M(X)), el emulador NO cuenta con carga directa a los registros, es decir la instrucción LOAD MQ, 5 carga en MQ el contenido de la memoria 5 y no el número 5 como en otros ensambladores.

```
1  0900101000
2  0500005002
3  060010C002
4  0000000000
5
```

Ilustración 2.1

Una vez que se tiene el programa y la memoria que se desea ejecutar podemos avanzar a la siguiente subsección.

Cargar archivos

El único archivo necesario de cargar es el del programa que se desea ejecutar, sin embargo, si se han realizado cambios a la memoria, también se deberá subir este archivo para un funcionamiento óptimo del emulador.

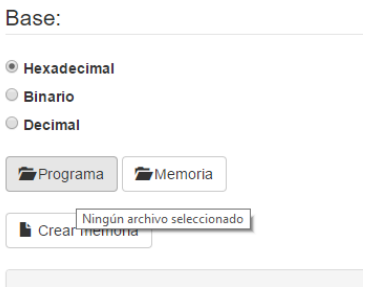

Ilustración 2.2

Para cargar el archivo de programa, se deberá hacer clic en el botón programa como se muestra en la Ilustración 2.2. Lo cual abrirá una ventana emergente donde podremos seleccionar la ubicación donde se encuentra el programa y posteriormente seleccionar abrir como se muestra en la ilustración 2.3.

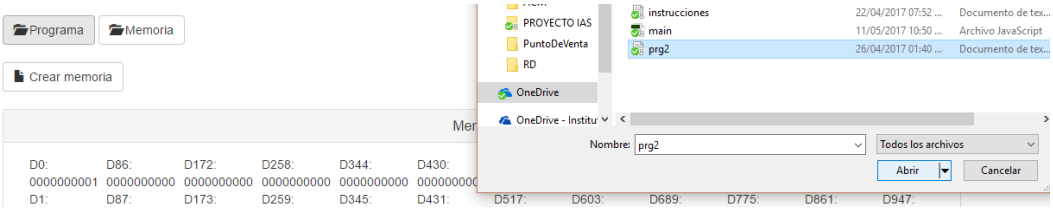

Ilustración 2.3

Una vez cargado el archivo podremos observar el programa en la columna de programa como se explicó en la sección anterior con la ilustración 1.2.

Ejecutar el programa

Para la ejecución del programa contamos con dos distintos modos de ejecución: Completa y Pausada. La ejecución completa es aquella que corre de principio a fin el programa y posteriormente nos muestra el resultado del mismo, por otra parte, si se desea observar el cambio de los registros en cada instrucción, se recomienda el uso de la ejecución pausada. Para seleccionar el tipo de ejecución utilizaremos los controles proporcionados en la interfaz como se puede ver en la ilustración 3.1.

Ilustración 3.1

Ejecución completa

Este tipo de ejecución se logra haciendo clic al ícono de “play” ubicado en la segunda posición de nuestros controles, con este tipo de ejecución como ya se mencionó, los registros se actualizarán

cada instrucción, mas estas se ejecutarán de principio a fin sin pausa y mostrarán el contenido de los registros al final de la ejecución como se puede observar en la imagen 3.2.

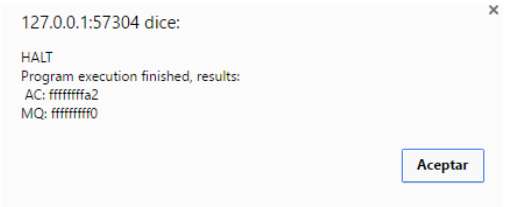

Ilustración 3.2

Ejecución pausada

Este tipo de ejecución se logra utilizando cualquiera de los controles ubicados a los extremos, es decir el ícono de siguiente y de anterior, cada uno logrando la ejecución de dichas instrucciones. Una vez que se inicia este tipo de ejecución, se podrá observar con cada clic cómo se actualizan los registros del emulador y al finalizar se mostrará la misma ventana que en la ejecución completa. Para este tipo de ejecución se utiliza la columna de registros mencionada en la sección anterior y es ahí donde se puede observar la actualización de los registros. Por otra parte, se marcará en letras con mayor peso la siguiente instrucción a realizar por el emulador como se puede observar en la ilustración 3.3.

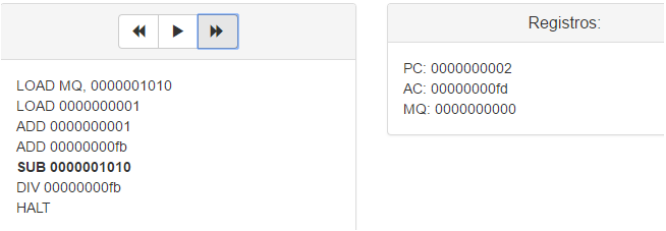

Ilustración 3.3

Una vez finalizada la ejecución del programa podemos pasar a la siguiente sección para la escritura de datos en memoria.

Escritura de datos en memoria

En esta sección se hablará brevemente de cómo guardar los datos arrojados por el programa para su uso en ocasiones posteriores. Como se puede observar en la tabla de la sección anterior, el emulador tiene instrucciones que guardan contenido en memoria, sin embargo, estas instrucciones son ejecutadas y reflejadas en el emulador, mas una vez cerrado la memoria regresa a ser la predeterminada. Para solucionar esta situación, el emulador cuenta con un botón llamado “Crear memoria”, donde podremos descargar una copia de la memoria obtenida al finalizar el programa, misma que podremos cargar en ocasiones posteriores con el botón “Memoria” del mismo emulador.

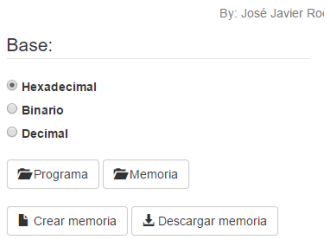

Ilustración 4.1

Para realizar la copia de memoria es muy simple, únicamente se debe hacer clic en el botón “Crear memoria” para que aparezca un nuevo botón como se muestra en la figura 4.1 llamado “Descargar memoria”, al hacer clic en este nuevo botón, el ordenador descargará un archivo llamado datos.txt que incluirá las 1024 palabras de la memoria para su uso en otra ocasión.

Agradecemos que hayas elegido utilizar IASemu y esperamos que la experiencia con el emulador sea de tu agrado.

Referencias

Stallings, W. Computer Organization and Architecture: Designing for Performance, 8th edition, Prentice Hall, 2010.

Von Neumann, J. First Draft on a Report on the EDVAC. Moore School, University of Pennsylvania, 1945.