MANUAL DE USUARIO

MICROPROCESADOR Z80

Brian Julian Moreno Niampira

Lenguajes de Programacion

UNIVERSIDAD NACIONAL

2018-2

Contenido

INTRODUCCION

El objetivo de este proyecto es construir un emulador para el procesador Z80, iniciando con la parte del ensamblador, encargado de traducir un fichero fuente escrito en lenguaje ensamblador, a un fichero objeto que contiene código maquina, para posteriormente escribir este código en la memoria ROM y poder procesarla y generar resultados eficientes. En este procesador se pueden llevar acabó 156 instrucciones entre lectura, escritura, operaciones aritméticas, operaciones lógicas y entre otras presentes en la arquitectura de este microprocesador. Inicialmente se realiza un programa para emular el funcionamiento de este, el cual muestra una pantalla de inicio donde se encuentra alguna información, pero lo mas importante 3 botones, uno que enlaza al ensamblador, otro al enlazador y finalmente un botón que direcciona al procesador. Se deben ejecutar secuencialmente cada uno de estos para llevar a finalidad este proceso de compilación. Este programa tiene como usuarios finales personas con habilidades en lenguajes de programación interesados en conocer el proceso interno que realiza este microprocesador Z80. Este emulador esta creado mediante un proyecto en lenguaje java.

OBJETIVOS DEL SISTEMA

El objetivo principal como se menciona es realizar el proceso de compilación de un programa en lenguaje C pero que debe ser ingresado inicialmente en lenguaje ensamblador. Con el fin de crear la codificación necesaria para que el procesador pueda leer en memoria y procesar la información que se necesita.

OBJETIVOS ESPECIFICOS

- Realizar el ensamblado del código en lenguaje ensamblador a código maquina.
- Enlazar este código maquina en la memoria ROM junto con los datos necesarios.
- Realizar el procesamiento mediante el microprocesador Z80.

GUIA DE USO

Este proyecto fue realizado en lenguaje Java, el cual tendrá que ser importado en un entorno de desarrollo preferiblemente eclipse, debido a que puede generar algo de conflictivos con netbeans. Después de haberlo importado, se deberá ejecutar desde el archivo Init.java, para esto lo abrimos en el directorio del proyecto, al ejecutarlo abrirá esta pantalla como inicio del programa. Figura 1.

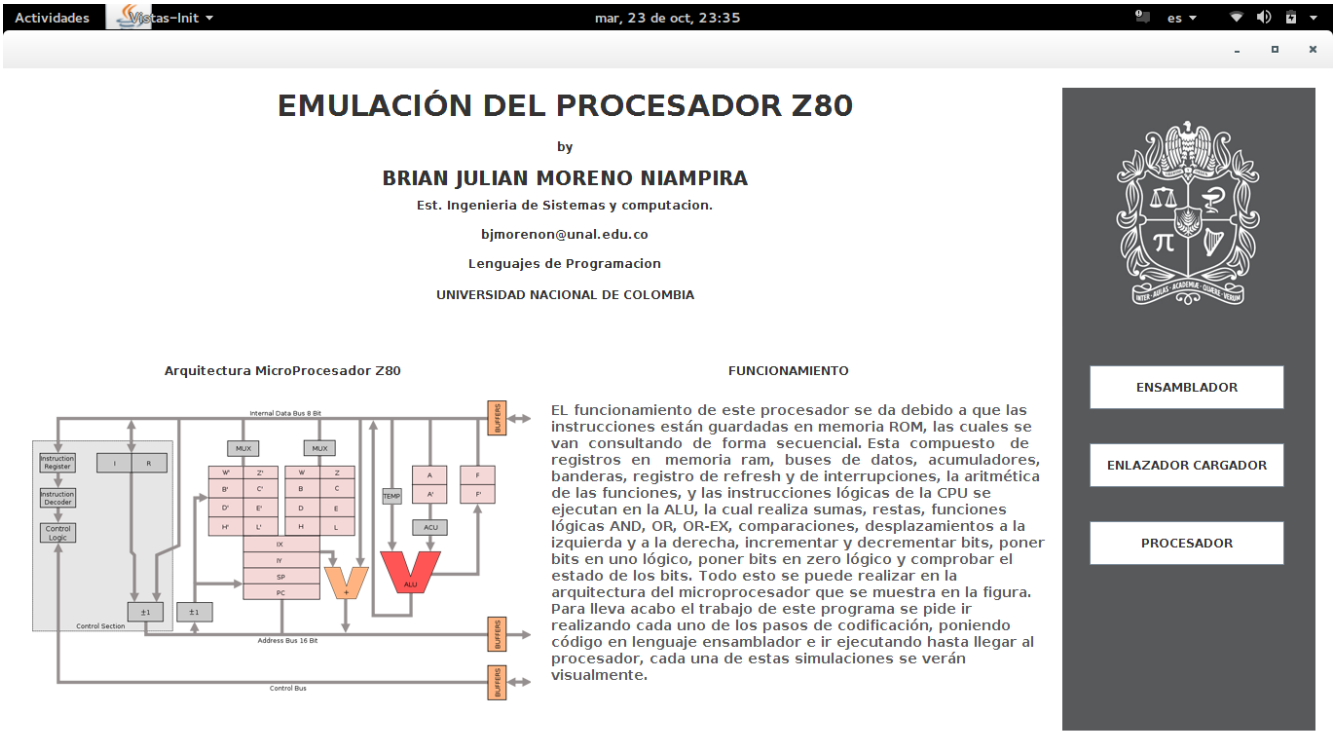

Figura 1. Init.java

Como se puede observar, estará en esta pantalla inicial algo de información, inicialmente con los datos personales del autor y programador de este programa, como también una imagen de la arquitectura del microprocesador Z80 como también algo de información de funcionamiento y que mas se puede realizar en este programa. Para realizar un funcionamiento adecuado, se hará de forma secuencial, iniciando con en ensamblado, luego con el procesamiento y terminaremos con la parte de procesamiento. Para esto podemos encontrar 3 botones en la parte derecha, cada uno encargado de direccionarnos a una funcionalidad de las ya mencionadas en este programa. Como se indico iniciaremos abriendo el ensamblador presionando el botón que ya indicamos, que nos direccionara a la siguiente pantalla que se muestra en la figura 2.

Como podemos observar en la pantalla del ensamblador se muestran dos paneles, donde en el primero se ingresara el código en lenguaje ensamblador cargándolo linea por linea como se muestra en el ejemplo que esta puesto en esta imagen(Figura 2), el cual para ser ensamblado y llevarlo a código maquina en su parte inferior estará un botón el cual realizara el ensamblado y nos mostrara en el panel de la derecha ya los códigos maquina de cada instrucción. Debemos estructurarlo bien en cuanto a sus etiquetas, instrucciones y parámetros para que el programa realice bien el ensamblado.

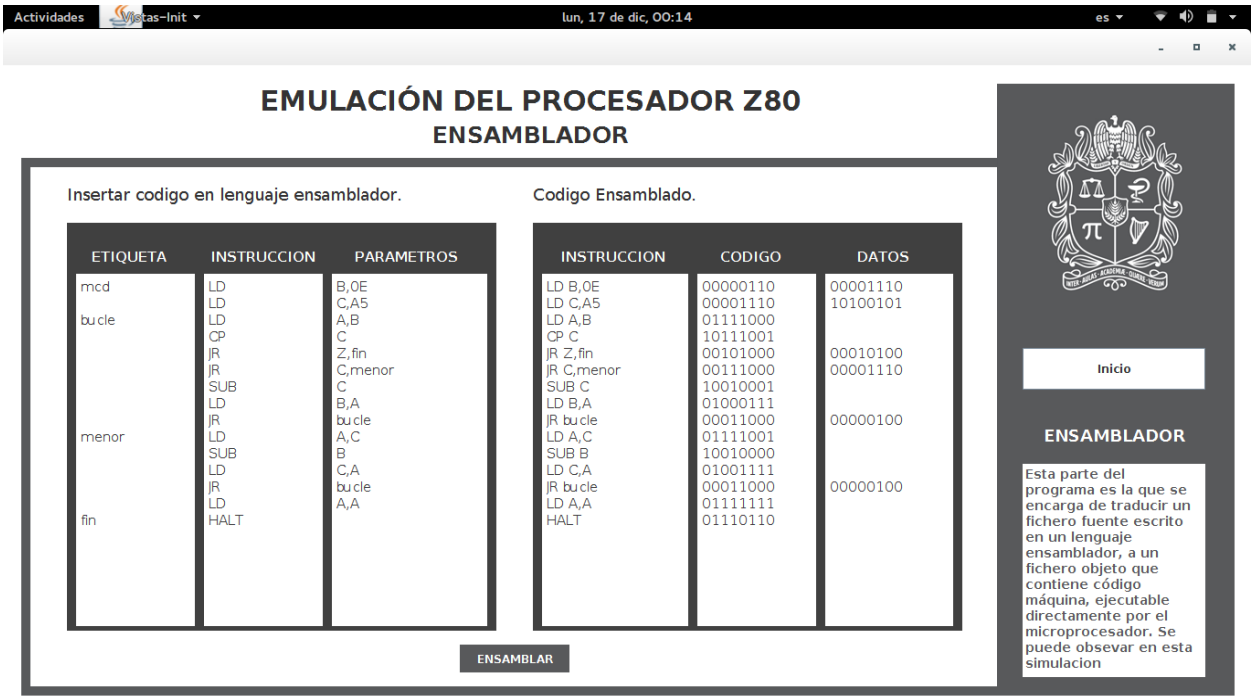

Figura 2. Ensamblador.java

Luego de haber realizado el proceso de ensamblado tenemos que regresar a la pantalla de inicio con el botón en la parte derecha. Estando en la pantalla de inicio(Figura1) realizaremos el segundo paso y consiste en el enlazado, donde podemos ir con botón de enlazador y cargador en la parte derecha que nos direccionara a la pantalla siguiente(figura 3).

Figura 3. Enlazador.java

Como podemos ver se encuentra en la parte superior un panel donde motrará la pisicion en memoria donde colocara el primer dato, el cual puede ser modificado, esto con el fin de cargar el programa en un espacio de memoria especifico, también se pueden ver cada uno de los códigos maquina que va cargar, y al lado un botón que añade en tiempo real cada uno de los registros con sus datos asociados en una memoria, pero si no desean cargar cada uno a la vez, esta un botón para cargar todos los códigos a la memoria ROM, la cual se puso el panel inferior mostrando el contenido de la memoria que se va ejecutando y actualizando cada vez que realizamos el cargado de un código, este se realizara de forma hexadecimal. Este proceso lo haremos hasta que se carguen todos los códigos maquina, que confirmara con un mensaje “Fin” que nos indicara que ha terminado su trabajo. Cuando lo terminemos regresaremos a la pantalla de inicio.

Finalmente en la pagina de inicio iremos con el botón de procesador, que nos llevara a realizar el procesamiento de la información que se encuentra en memoria. Y lo podremos evidenciar en la siguiente pantalla. Figura 4.

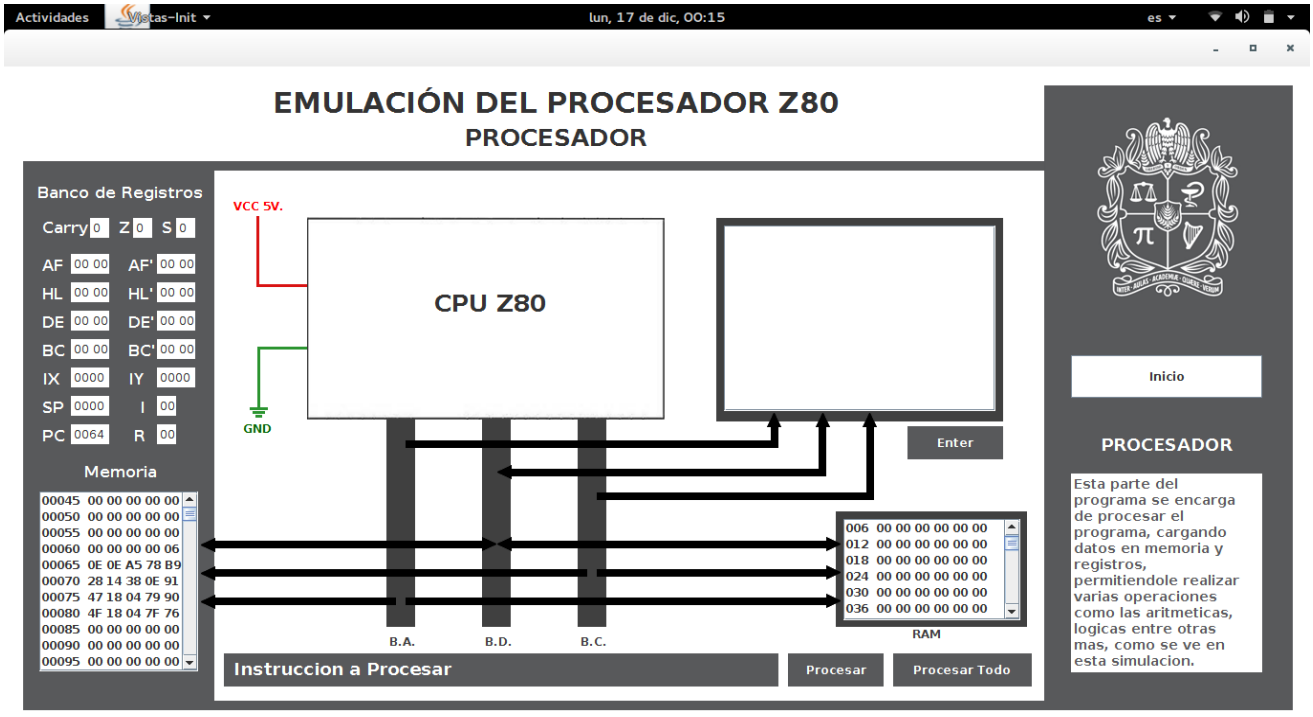

Figura 4. Procesador.java

En esta pantalla podremos ver el procesamiento de los opcodes en tiempo real, debido a que tenemos un panel en la parte inferior izquierda con un segmento de memoria ROM, que nos permitirá observar lo que hay en ella. En la parte superior izquierda estarán los registros de la RAM, en la parte inferior central los registros que se van a ejecutar y un botón que procesara el opcode. Como ya se había indicado estos registros están guardados en su forma hexadecimal. Como se trabaja con etiquetas, si el código ensamblador estuvo bien definido podremos ver el contador del programa(PC) saltar en las memoria realizando operaciones hasta dar resultados. En la parte central la imagen de la pastilla del Z80 en la parte superior haciendo visual la comunicación de los buses de direcciones, datos y control. En la parte

derecha tenemos una memoria de 256 puertos, forman 16 líneas de direcciones, proporcionan las direcciones correspondientes a intercambios de datos entre la memoria, la CPU y los puertos de los periféricos. En la parte derecha superior pusimos una entrada de texto para simular la entrada y salida de datos mediante IN/OUT, opcodes presentes en la arquitectura del microcontrolador. Debemos contarles que aunque no estuvieron presentes visualmente los pines del microprocesador Z80, hablaremos algunos de ellos que son importantes como los de direcciones, pines 30-40, 1-5, de los cuales ya hablamos, pines 14, 15, 12, 8, 7, 9, 10 y 13 que conforman la linea de datos, y otros pines de corriente, parada, reloj y demás actividades presentes en este procesador.

Como ejemplo y ver el funcionamiento correcto del programa pueden trabajar con el siguiente código en lenguaje ensamblador para calcular el máximo común divisor de dos números y como entradas cuando las pida en la pantalla te entrada de texto en el procesador, poner los números 110 y 165 que les dará como resultado de procesamiento el numero 55, calculado de manera precisa el resultado.

```
Mcd     IN    A,(01H)
        LD    B,A
        IN    A,(02H)
        LD    C,A
bucle   LD    A,B
        CP    C
        JR    Z,salida
        JR    C,menor
        SUB   C
        LD    B,A
        JR    bucle
menor   LD    A,C
        SUB   B
        LD    C,A
        JR    bucle
salida  OUT   (03H),A
fin     HALT
```

Deberá funcionar de manera adecuada dando como resultado 55 a las entradas 110 y 165 que en si el máximo común divisor de 110 y 165 es 55.

SOPORTE

Soporte: bjmorenon@unal.edu.co