Práctica 1: Seno y Coseno*

Christian David Pocol Franco, 201807325,[1,**] Héctor Fernando Carrera Soto, 201700923,[1,***] and Nicole Alejandra López Calderón , 201800683 sección: D[1,****]

[1] *Facultad de Ingeniería, Escuela de ingeniería mecánica eléctrica, Universidad de San Carlos, Edificio T1, Ciudad Universitaria, Zona 12, Guatemala.*

El lenguaje ensamblador es el lenguaje de programación utilizado para escribir programas informáticos de bajo nivel, y constituye la representación más directa del Código máquina específico para cada arquitectura de computadoras legible por un programador. La práctica consiste en la realización de un programa que calcula la serie coseno de un número, utilizando lenguaje ensamblador.

I. OBJETIVOS

A. Generales

* Realizar un programa capaz de calcular la serie coseno de un número.

B. Específicos

* Aprender a utilizar lenguaje ensamblador.

* Investigar y dominar los principales mnemónicos para su utilización posterior.

II. MARCO TEÓRICO

A. Lenguaje ensamblador

El único lenguaje que entienden los microcontroladores es el código máquina formado por ceros y unos del sistema binario. El lenguaje ensamblador expresa las instrucciones de una forma más natural al hombre a la vez que muy cercana al microcontrolador, ya que cada una de esas instrucciones se corresponde con otra en código máquina. El lenguaje ensamblador trabaja con nemónicos, que son grupos de caracteres alfanuméricos que simbolizan las órdenes o tareas a realizar. La traducción de los nemónicos a código máquina entendible por el microcontrolador la lleva a cabo un programa ensamblador. El programa escrito en lenguaje ensamblador se denomina código fuente (*.asm). El programa ensamblador proporciona a partir de este fichero el correspondiente código máquina, que suele tener la extensión *.hex.

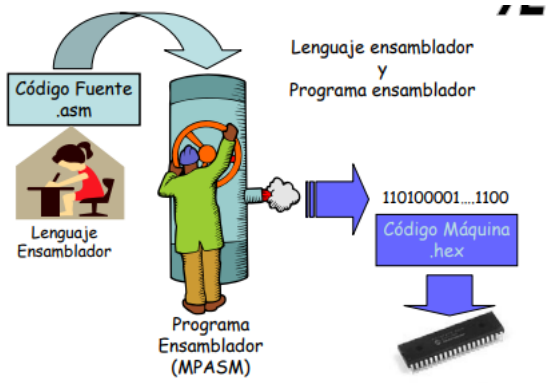

Figura 1: Estructura assembly

Un programa escrito en lenguaje ensamblador consiste en una serie de instrucciones que corresponen al flujo de órdenes ejecutables que pueden ser cargadas en la memoria de un sistema basado en microprocesador. Por ejemplo, un procesador x86 puede ejecutar la siguiente instrucción binaria como se expresa en código máquina:

- Binario: 10110000 01100001 (Hexadecimal: 0xb061)

La representación equivalente en lenguaje ensamblador es más fácil de recordar:

- MOV al, 061h

Esta instrucción significa:

- Asigna el valor Hexadecimal 61 (97 Decimal) al registro .[a]l".

Se le llama **mnemónico** a un símbolo que representa a una instrucción de lenguaje de máquina. Los mnemónicos se definen por el lenguaje ensamblador.

Una **directiva** es una operación que se procesa en el ensamblado del programa, no en su ejecución. Las directivas se definen también por el programa ensamblador. Éstas no se van transformar a lenguaje de máquina, sino que modifican el comportamiento del ensamblador mientras procesa los mnemónicos a lenguaje de máquina. Las directivas permiten extender el lenguaje ensamblador para manejar conceptos como variables, constantes, etc. A las directivas también se les conoce como pseudoinstrucciones. Las directivas varían

* Laboratorios de electrónica 5
** e-mail: correo1@dominio1
*** e-mail: 3505043180101@ingenieria.usac.edu.gt
**** e-mail: 3639934600101@ingenieria.edu.gt

dependiendo del programa ensamblador utilizado. Las directivas utilizadas por el ensamblador Arm no son las mismas que las del ensamblador GNU.

B. TIVA C

Es una plataforma de prototipos electrónicos basados en una familia de microcontroladores creada por Texas Instruments. Las placas de prototipos son del tamaño aproximado de una tarjeta de crédito. Están equipadas con un microcontrolador ARM Cortex-M4F, fabricado por Texas Instruments, con una CPU de 32 bits funcionando de 80 a 120 MHz. La TM4C Series TM4C123G LaunchPad. es una mejora de la TI de la Stellaris LaunchPad añadiendo opciones de soporte de PWMs para control de movimiento y con soporte de USB Host.

Están equipados con 40 o 80 pines multifunción, es decir, pueden ser configurados como entradas o salidas, digitales o analógicas u otras funciones, permitiendo una gran variedad de aplicaciones. Sus pines son compatibles con el estándar de 3,3 V.

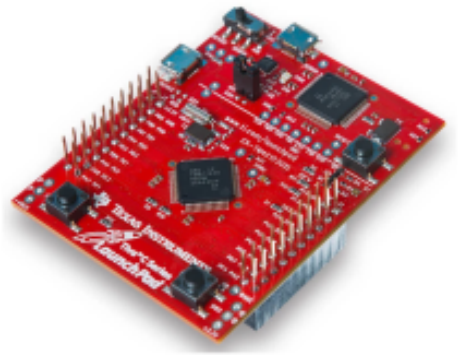

Figura 2: Placa Tiva-C.

III. RESULTADOS

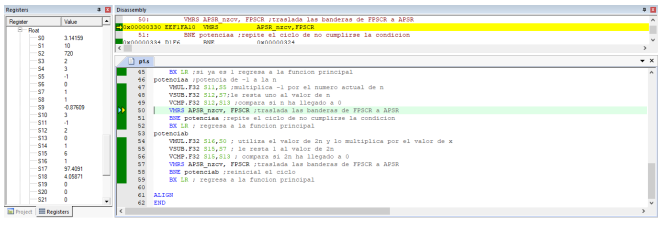

Figura 3: Captura de resultados del IDE.

IV. DISCUSIÓN DE RESULTADOS

En este apartado se deben analizar los resultados obtenidos, contrastándolos con la teoría expuesta en la sección del Marco Teórico. Corresponde explicar el comportamiento de las tablas y gráficas expuestas en la sección de Resultados, tomando en cuenta el análisis estadístico apropiado.

V. CONCLUSIONES

1. En el lenguaje ensamblador es necesario especificar cada uno de los procesos a realizar, por lo que para una operación matemática sencilla, puede resultar en algo muy grande.

2. Es importante saber diferenciar y aprender los mnemónicos básicos, ya que son base en la programación en lenguaje ensamblador.

3. Se logró programar la función coseno en su expresión dado por una serie de Taylor y Maclaurin, en lenguaje ensamblador.

VI. CÓDIGO

```
    AREA codigo, CODE, READONLY,ALIGN=2
THUMB
EXPORT Start
Start
    VLDR.F32 S0, = 3.141592654;
    VLDR.F32 S1, = 10;
    VLDR.F32 S2, = 1;
    VLDR.F32 S3, = 2;
    VLDR.F32 S4, = 0;
    VLDR.F32 S5, = -1;
    VLDR.F32 S7, = 1;
    VLDR.F32 S9,=0;
    VLDR.F32 S10, = 3;
    VLDR.F32 S11,=1
    VLDR.F32 S13,=0;
    VLDR.F32 S16,=1
sumatoria
    VADD.F32 S4,S7;
    VMUL.F32 S8,S4,S3;
    VADD.F32 S15,S8,S13;
    BL factorial;
    VADD.F32 S12,S4,S13 ;
    BL potenciaa
    VADD.F32 S14,S11,S13;
    BL potenciab
    VADD.F32 S17,S16,S13;
    VDIV.F32 S18,S14,S2;
    VMUL.F32 S18,S17;
    VADD.F32 S9,S18;
    VLDR.F32 S2, =1;
    VLDR.F32 S16,=1;
    VLDR.F32 S11,=1;
    VCMP.F32 S4,S1;
    VMRS APSR_nzcv, FPSCR
```

```
    BNE sumatoria
    VADD.F32 S9,S7;
Loop
    B Loop
factorial
    VMUL.F32 S2,S8;
    VSUB.F32 S8,S7;
    VCMP.F32 S8,S7;
    VMRS APSR_nzcv, FPSCR ;
    BNE factorial ;
    BX LR ;
potenciaa ;potencia de -1 a la n
    VMUL.F32 S11,S5 ;
    VSUB.F32 S12,S7;
    VCMP.F32 S12,S13 ;
    VMRS APSR_nzcv, FPSCR ;
    BNE potenciaa ;
    BX LR ;
potenciab
    VMUL.F32 S16,S0 ;
    VSUB.F32 S15,S7 ;
    VCMP.F32 S15,S13 ;
    VMRS APSR_nzcv, FPSCR ;
    BNE potenciab ;
    BX LR ;

ALIGN

END
```

VII. DIAGRAMA DE FLUJO

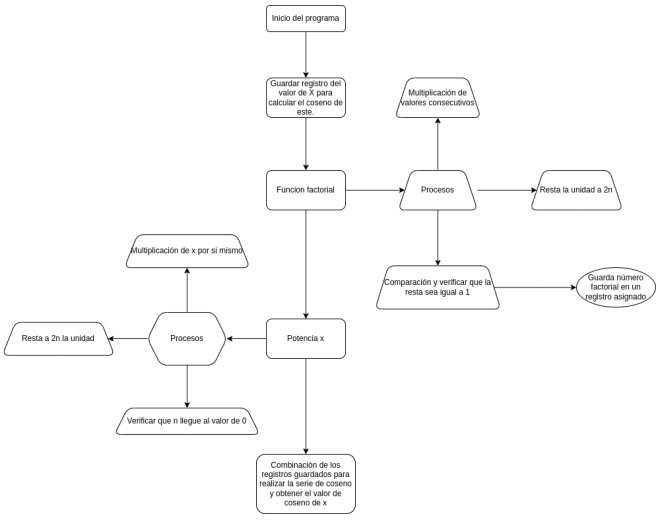

Figura 4: Diagrama de flujo sobre el funcionamiento del código.

[1] Grossman, S. (Segunda edición). (1987). *Álgebra lineal*. México: Grupo Editorial Iberoamericana.

[2] Gastón, Salazar *Lenguaje ensamblador - instrucciones, mnemónicos y directivas* [En linea][25 de octubre de 2012]. Disponible en: https://ghsalazar.github.io/2020/11/18/Lenguaje-ensamblador-instruciones-y-directivas.html#:~:text=Un%20mnem%C3%B3nico%20es%20un%20s%C3%ADmbolo,definen%20por%20el%20programa%20ensamblador.