IMPLEMENTACIÓN DE UNA FUNCIÓN DE MULTIPLICACIÓN ENTERA SIN SIGNO DE 8 BITS LaTeX

Royner Baltodano Sánchez[1], *Angello Vividea García*[2], *Luis Ortiz Rua*[3]

[1]royner.baltodano@hotmail.com, [2]luis092994@gmail.com, [3]markusvividea@gmail.com

ABSTRACT

Abstract— In modern math implementations requires the use of math operators to perform calculations, This is the case of multiplications. Within the processor, multiple mathematical operations are performed, one of them being binary multiplication. The purpose of this research is to implement a MASM code that satisfies the need to multiply 8-bit binary numbers, using SHL's Irvine algorithm, set to test under the processor and compared to the results using.

1. INTRODUCCIÓN

El siguiente informe describe la implementación del algoritmo para realizar multiplicaciones de 1 byte sin signo. Dentro del procesador, se encuentra la ALU, Unidad Aritmética Lógica, que se encarga de efectuar los procesos matemáticos, pero para realizar una multiplicación o cualquier operación matemática, no la hace con el método convencional, usa métodos mas prácticos para realizar multiplicaciones de forma mas rápida y eficiente, para ello usa diversos algoritmos, SHL, SHR, SHRD, SHLD, SAL, etc. Teniendo eso en cuenta, el procesador, para multiplicar 2 números de 1 byte, lo efectúa mediante Corrimiento lógico a la izquierda o corrimiento a la izquierda, este recorrido de bits a la izquierda de un byte se realizan en un registro de memoria (el operando también puede ser una constante), este corrimiento lógico también tratan a los bits con signos.Estas multiplicaciones son realizadas diariamente por el procesador, pero en este proyecto le daremos instrucciones al procesador, mediante el lenguaje Microsoft Macro Assembler (MASM), con el que programamos a muy bajo nivel para poder comprender y conocer el funcionamiento del procesador. Le daremos valores específicos, y realizaremos el algoritmo SHL para llevar a cabo una multiplicación de un número de un byte, esperando como respuesta un número de 16 bytes.

2. DESARROLLO

Explicación del algoritmo: Mediante la técnica de corrimientos lógicos a la izquierda efectuaremos cualquier tipo de multiplicación de números de 1 byte sin signo.para introducir el algoritmo tengamos en cuenta que recibiremos 2 números como parámetros iniciales, luego mediante SHL recorreremos los bits a la izquierda un numero especifico de veces y llenaremos con ceros las posiciones vacates a la derecha, posteriormente, guardaremos los valores de los corrimientos para después hacer una suma entre estos, de esta forma el procesador realiza las multiplicaciones. En nuestro presente código declaramos mediante PROTO el prototipo de nuestro algoritmo. recibimos 2 valores iniciales a los cuales les llamaremos multiplicador de 2 bytes, multiplicando de 1 byte, agregaremos una variable llamada resultado y un contador que solo tendrá 2 bytes de espacio, además de colocar los distintos impust para solicitar datos por consola. Mediante la palabra reservada PROC realizaremos el siguiente procedimiento.Con estos valores establecidos procedemos a efectuar la multiplicación, agarro los valores de multiplicador, multiplicando y resultado y las muevo a los registros AX,AH,BX respectivamente, luego dentro de un ciclo, analizaremos: si el multiplicando tiene un 1, en caso de ser así, tomo el registro AX(multiplicador) le lo desplazo con SHL las posiciones que tengamos guardada en contador, luego sumamos este resultado al registro BX(resultado) y movemos con un SHR(movimiento lógico a la derecha) de 1 espacio al registro AH(multiplicando) para continuar con el siguiente dígito a multiplicar, pasado esto volveremos a asignarle el valor multiplicador a nuestro registro AX(multiplicador), y volveremos a repetir el ciclo, agregándole 1 a nuestro contador para ver si aun no cumplimos con nuestra condición de parada contador < 8. Si no el multiplicando tiene un 0, solo auto-sumaremos a nuestro contador un 1, y moveremos 1 dígito a nuestro multiplicando para continuar analizando todos los dígitos del multiplicando. Para retornar el resultado usaremos INVOKE para llamar a nuestro proceso que se encargara de realizar la multiplicación ya explicada, de esta forma obtendremos el resultado.

Ejemplo:

Eso funciona básicamente así.

10001010.Multiplicador

00000010. Multiplicando

Entonces el último bit de multiplicando, Si es 0 entonces agrega 1 a un contador. Mueve a la derecha multiplicando. Entonces queda así

10001010

00000001

En este caso quedaría 100010100

Explicacion del codigo en prosa:

Mueve los valores a los registros

mov ax, multiplicador // mov ah, multiplicando // mov bx, resultado

while. contador < 8 // Ciclo, nuestra condicion de parada es si contador es < a 8

if. ah 1 //si el ultimo dígito de multiplicando es un 1

shl ax, contador // aplico SHL al registro AX(multiplicador) la cantidad q contador indique

add bx, ax // sumo AX(multiplicador) al registro BX(resultado)

add contador, 1 // Sumo 1 al contador

mov ax, multiplicador // muevo multiplicador al registro AX

shr ah, 1 // muevo el multiplicado hacia la derecha SHR para verificar si el ese dígito es un 1 o 0

else. // si en caso el dígito del multiplicando no es 1

add contador,1 // sumo 1 al contador

shr ah, 1 // muevo el multiplicado hacia la derecha SHR para verificar si el ese dígito es un 1 o 0

3. CONCLUSIÓN

Como resultado de este proyecto puede señalarse que se presento la solución para la resolución del problema planteado usando SHL, y código MASM para efectuar multiplicaciones en el procesador.logramos comprender las ventajas y limitaciones que el procesador tiene a la hora de efectuar operaciones aritméticas, y conocer así más del lenguaje.De igual forma, se concluye el aprendizaje los temas presentados.

4. REFERENCIAS

IRVINE, K. R. (2008). Lenguaje ensamblador para computadoras basadas en Intel. Monterrey.