Divisor con residuo

Contreras R Orlando[1*]

[3]Departamento de Sistemas Electrónicos, UAA, Av. Universidad 940, Aguascalientes, 20131, Aguascalientes, México.

Corresponding author(s). E-mail(s): al348390@edu.uaa.mx;

Abstract

El presente documento busca implementar de forma eficiente un divisor en el lenguaje ensamblador, en esta práctica se abordarán temas como el manejo de registros, manejo de memoria, saltos condicionales y operaciones aritméticas en la arquitectura ARM

Keywords: ASM, Banderas, Carry

1 Introducción

Una de las operaciones derivadas más básicas es la division, pudiendose explicar como la operación inversa de la multiplicación, basándonos en esta misma relación podemos deducir como expresarlo de una forma mas esencial, mientras que el producto son sumas iteradas, la división se puede expresar como restas iteradas.

2 Metodología

2.0.1 Banderas

La forma en que la lógica del codigo se construyó fue la siguiente. Se analizaron dos casos una división exacta y otra que no quedará de forma exacta. Se realizaron restas continuas hasta obtener el valor deseado, se observo que si en ambos casos se realizaba una resta adicional se encendia la bandera del negativo y carry, tomando esto en cuenta se decidió considerar este como el caso de limite, realizando un pequeño ajuste en el resultado como en el residuo.
Otra opción que también se considero es anticipar la resta, es decir restar doble para de esta forma mantener el valor en el que debe, esta opción se descarto porque cada resta

que se realice se deberia de hacer el doble de tiempo, mientras que la otra solución solo aplicaría para la última resta.

2.0.2 Lógica y análisis

Teniendo en cuenta que una division es una resta iterada, el contador iterado en nuestro caso i contendrá las veces que se requiera restar numero a de b para poder obtenerlo. Otro concepto tambien a considerar es cuando llegamos a un punto en el que no es suficiente el valor b para ser restado de a es decir $a < b$, en éste caso podemos expresar que queda un residuo o un remanente, dicho residuo es importante considerarlo porque nos demuestra que no es un resultado exacto el que estamos obteniendo.

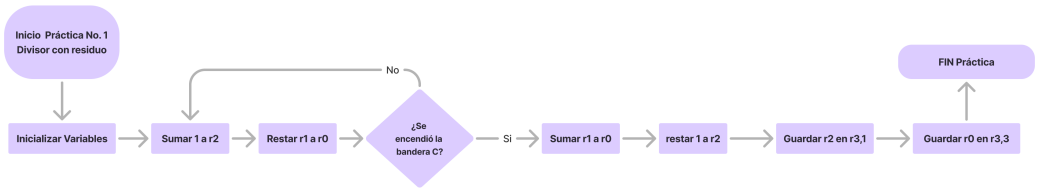

Fig. 1 Diagrama de flujo

Algo importante a mencionar es que se llego a hacer una corrección en el código. Para que éste pudiera aceptar valores en el bit más significativo, se cambió el brinco condicional de negativo por el de carry, siendo que cuandose hace una resta de $a - b$siendo $b > a$ se enciende la bandera del negativo y carry. La estructura de la sentencia se puede apreciar en la linea 4 del siguiente código.

```
1  brinqitos
2      add     times,times,#1
3      subs    r0,r0,r1
4      bhs     brinqitos
5      add     r0,r0,r1
6      subs    times,times, #1
7      strb    times, [r3,#1]
8      strb    r0, [r3,#3]
```