Resumen programación avanzada

Comúnmente en los microprocesadores, se requiere hacer operaciones aritméticas ya sea hacer operaciones bit a bit o considerar al conjunto de bits como una entidad y operar entre conjuntos de bits, para ello es necesario hacer la conversión de códigos dentro del mismo microcontrolador. comúnmente dentro de los microprocesadores de 8 bits solo se disponen instrucciones para realizar la suma y la resta de números binarios, ya sean números de varios bits o codificados en BCD. Por lo tanto, si necesitamos realizar una multiplicación o división, es necesario hacer un arreglo de instrucciones para lograr realizar la operación.

Al realizar operaciones, podemos hacerlas con números con signo y sin signo, en el caso de que el número no cuente con signo, asumimos que es un numero positivo, por lo tanto, todos los bits que componen el numero son parte de la cifra. Para representar un numero decimal en binario se usa la ponderación, de tal manera que El valor en decimal de un número entero sin signo de n bits es igual a la suma de los productos de cada bit por el peso (potencias de 2) asociado a su posición. Cuando el numero se puede representar en un solo byte se le llama precisión sencilla, en cambio cuando el numero esta distribuido en varios bytes, se les llama números de precisión múltiple.

Antes de realizar una operación entre bytes, es necesario tener claro que sucede en una suma o resta bit a bit. En el caso de la suma, al momento de sumar 0+0 el resultado será cero, mientras que al sumar 1+0 ó 0+1 lo cual es lo mismo debido a que en la suma los términos pueden conmutar, el resultado será 1 y finalmente cuando tenemos 1+1 el resultado será 0 pero tendremos un “1” de acarreo. Al momento de realizar una suma de varios bits, se comienza por el bit menos significativo hacia el mas significativo, cabe resaltar que de igual manera que una suma decimal, en una suma binaria el acarreo se suma al siguiente bit. Así mismo si el resultado de la suma excede el número de bits que los sumandos, se genera un bit de acarreo. Para realizar una resta entre bytes, de igual manera se necesita primero identificar que sucede en la diferencia entre bits, al momento de restar 0-0 el resultado será 0, pero en la resta si importa el orden ya que no es lo mismo restar 0-1 que 1-0, en el primer caso se requiere un préstamo y el resultado seria 0, en el segundo caso el resultado sería 1. Finalmente 1-1es igual a 0. Para realizar una resta entre bytes, de igual manera que en la suma se empieza del bit menos al mas significativo y se toman en cuenta los préstamos en la realización de la resta. Pero hay una ligera excepción, puesto que, si el sustraendo es mayor que el minuendo, no podemos representar el resultado sucede de manera similar que en la aritmética decimal obtendremos un numero negativo, y en la aritmética binaria tendremos un numero con signo, por lo cual los bits con los que contamos para representar el resultado no serían suficientes.

En otro tema, es posible representar números binarios en código BCD, en este código solo contamos con 4bits para la representación de un número, por lo cual el número máximo para representar es 15_{10} y al solo abarcar 4bits, podemos tener 2 números en BCD dentro de un byte, así mismo algunas veces resulta conveniente manejar este código, por ello es necesario aprender en que consiste la aritmética en BCD. Sin embargo, el hecho de manejar el código BCD representa un problema, esto se debe a que por ejemplo se necesita sumar 05_{BCD} + 07_{BCD} estas cantidades están contenidas en un byte, por lo tanto, el primer sumando queda como 000000101 y el segundo queda como 00000111. Si sumamos bit a bit como en la suma binaria, el resultado queda como 00001100, sin embargo, como

podemos observar el 12 se almacena en un solo nibble, pero nosotros requerimos que el 1 se almacene en 1nibble y el 2 en otro nibble, por ello es necesario sumar 0000 0110 para que de esta manera podamos visualizar el resultado en BCD, una vez que tenemos en cuenta esta consideración, es necesario deducir el algoritmo para la suma. Sabemos que dentro del código BCD los números pueden estar en un intervalo de [0…..9] , además dentro de la aritmética, las operaciones son binarias , es decir solo se puede sumar entre 2 términos, por ende si sumamos 2 valores máximos en BCD, el número máximo que podemos obtener es 18. Teniendo en cuenta esto, se requiere idear una forma de tal manera que las unidades se separen de las decenas. En caso de que el resultado sea >10 sucede esto, para realizar esta labor restamos -10 al resultado de la suma, por ejemplo, si el resultado es 16, le restamos -10 y obtendremos el valor de las unidades que es 6, si sabemos que el numero 16 puede ser representado mediante dos nibbles R1 y R0, R0 representara las unidades y R1 representara las decenas, por ello podemos almacenar el resultado de 16-10 en R0 y a R1 cargarlo con 1, ambos números en BCD.

Para implementar este algoritmo en un diagrama de flujo mas general, es necesario tener en cuenta una decisión ya que podemos realizar el procedimiento anterior solo si el resultado de la suma global es mayor que 10, es decir al momento de restar con el -10 obtenemos un numero positivo, ya que de lo contario será un numero negativo, entonces a partir de esto podemos analizar la bandera “s” del microcontrolador que se activa en “1” si el resultado es negativo, por lo tanto el resultado se almacena directamente en R0 de lo contrario si no es negativo será necesario poner en marcha el procedimiento que mencionamos arriba. En el caso de la resta no tendremos este problema debido a que el minuendo debe de ser siempre menor que el sustraendo, además como se menciona arriba los números en BCD van de 0 a 9 por lo tanto no podremos tener resultados arriba de 10, esto simplifica en gran medida el algoritmo para la resta ya que solo se resta las variables en BCD y se almacena en R0.

Finalmente, las multiplicaciones las podemos definir como un numero consecutivo de sumas y por ende se utilizará el mismo algoritmo que la suma. En el caso de las divisiones también utilizaremos retas sucesivas, pero en este caso al numerador se le ira restando el denominador y el resultado se almacena en otra variable, así sucesivamente hasta que lleguemos a 0 y el numero de veces que se restó será el resultado de la división.

Ejemplos

Sumar:
$$\begin{array}{r} 9 \\ +\ 6 \\ \hline 7 \end{array}$$

- “carga r1 y r0 con 00

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	xx
W	xx

S	0

- “carga w con X_0”

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	xx
W	09
S	0

- “suma w con $Y_{0,}$ el resultado almacena en W”

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	xx
W	15
S	0

- “carga var_temp con w”

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	15
W	15
S	0

- “carga w con 10”

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	15
W	10
S	0

- “Resta w con var_temp y el resultado dejalo en w”

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	15
W	05
S	0

- “El resultado es(-) S=1?”

RAM	
X_0	09
Y_0	06
r0	00
r1	00
Var_temp	15
W	05
S	0

- “Carga a r0 con w”

RAM	
X_0	09
Y_0	06
r0	05
r1	00
Var_temp	15
W	05
S	0

- “Carga a w con 1”

RAM	
X_0	09
Y_0	06
r0	05
r1	00
Var_temp	15
W	01
S	0

- “Carga r1 con w”

RAM	
X_0	09
Y_0	06
r0	05
r1	01
Var_temp	15
W	01
S	0

Sumar: 3
+ 6
9

- “carga r1 y r0 con 00

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	xx
W	xx

S	0

- “carga w con X_0”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	xx
W	03
S	0

- “suma w con $Y_{0,}$ el resultado almacena en W”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	xx
W	09
S	0

- “carga var_temp con w”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	09
W	09
S	0

- “carga w con 10”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	09
W	10
S	0

- “Resta w con var_temp y el resultado dejalo en w”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	09
W	01
S	1

- “El resultado es(-) S=1?”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	09
W	01
S	1

- “Carga a w con var_temp”

RAM	
X_0	03
Y_0	06
r0	00
r1	00
Var_temp	09
W	09
S	0

- “Carga a r0 con w”

RAM	
X_0	03
Y_0	06
r0	09
r1	00
Var_temp	09
W	09
S	0

Restar: 6
- 5
1

- “carga r1 y r0 con 00

RAM	
X_0	06
Y_0	05
r0	00
r1	00
Var_temp	xx
W	xx

- “carga w con Y_0”

RAM	
X_0	06
Y_0	05
r0	00
r1	00
Var_temp	Xx
W	05

- “Resta X_0 con W, el resultado almacena en W”

RAM	
X_0	06
Y_0	05
r0	00
r1	00
Var_temp	xx
W	01

- “carga R0 con w”

RAM	
X_0	06
Y_0	05
r0	01
r1	00
Var_temp	xx
W	01

Restar: 9
- 4
5

- “carga r1 y r0 con 00

RAM	
X_0	09
Y_0	04
r0	00
r1	00
Var_temp	xx
W	xx

- “carga w con Y_0”

RAM	
X_0	09
Y_0	04
r0	00
r1	00
Var_temp	Xx
W	04

- “Resta X_0 con W, el resultado almacena en W”

RAM	
X_0	09
Y_0	04
r0	00
r1	00
Var_temp	xx
W	05

- “carga R0 con w”

RAM	
X_0	06
Y_0	05
r0	05
r1	00
Var_temp	xx
W	05

Multiplicación

5x3= 5+5+5 =15

Se realiza el proceso de suma 3 veces 5+5 y después 10+5

3x3=3+3+3=9

Se realiza el proceso de suma 3 veces 3+3 y después 6+3

División

8/2 = 8-2

6-2=4

4-2=2

2-2=0

Como podemos observar el proceso de resta se realiza 4 veces, este número corresponde al resultado de la división

1) 99/9= 99-9
2) 90-9=81

3) 81-9=72
4) 72-9=63
5) 63-9=54
6) 54-9=45
7) 45-9=36
8) 36-9=27
9) 27-9=18
10) 18-9=9
11) 9-9=0

Como podemos observar el proceso de resta se realiza 11 veces, este número corresponde al resultado de la división