División en una ALU

RestoDividendo (RD)-RestoDivisor (Rd) >0 entonces q\_0=”2”

Desplazamos a la izquierda

RD’=RD-Rd y repetir otra vez

Si no cabe, es decir:

RD-Rd<0 q=”0”

RD’=RD-Rd+Rd

Desplazamos izquierda = 2(RD-Rd+Rd)-Rd = (2RD - 2Rd) + 2Rd – Rd = 2(RD - Rd) + Rd

Y volverlo a intentar.

Algoritmo de división sin restauración

Tenemos el dividendo y le restamos el divisor, repitiendo la secuencia de instrucción n veces, teniendo el dividendo 2n bits.

En el ejemplo tiene 8 bits, por lo que se realizarán 4 iteraciones.

Los 4 primeros bits del dividendo serán los correspondientes al resto y los 4 últimos(los de la derecha) serán los correspondientes al cociente.

Dividendo\_n=Dividendo\_n-Divisor

Si Dividendo\_n < 0 entonces

Desplazar dividendo a la izquierda

Dividendo\_n=Dividendo\_n + Divisor

Si no:

Desplazar dividendo a la izquierda

Dividendo\_n=Dividendo\_n - Divisor

Fin Si;

Si Dividendo\_n < 0 entonces

q\_0 = 0

Sino q\_0 = 1

Fin Si;

Fin Repetir;

Si Dividendo\_n < 0 entonces

Dividendo\_n=Dividendo\_n+Divisor

Fin Si;

IMPORTANTE

LOS NÚMEROS ESTÁN EN COMPLEMENTO A 2

Restar el divisor al dividendo equivale a sumarle su complemento a 2

Ejemplo restar el divisor 0010, es decir 2, lo haremos de modo que:

Número -2=> Lo pasamos a complemento a 2

-2=1101+1=1110, en este caso se sumaría al resto dicho valor y se volvería a comprobar según el esquema explicado en clase.

Ejemplo hacer algoritmo de división sin restricción para un número de 5 bits 11/5

Dividendo (D) = 11 = 01011

Divisor (d) = 5 = 00101

-5 = 11010+1=11011 // Cuando se resta el divisor se sumará este valor a la parte izquierda

q\_0🡪 es el valor que toma \_ después de desplazar

Dividendo Divisor Acción Iteración

00000 01011 00101 Valores iniciales 0

11011 01011 00101 D\_n=D\_n - d 0

10110 1011\_ 00101 D\_n<0 🡪 Despl. Izq. 1

11011 1011\_ 00101 D\_n=D\_n + d 1

11011 10110 00101 D\_n<0 🡪 q\_0=0 1

10111 0110\_ 00101 D\_n<0 🡪 Despl. Izq. 2

11100 0110\_ 00101 D\_n=D\_n + d 2

11100 01100 00101 D\_n<0 🡪 q\_0=0 2

11000 1100\_ 00101 D\_n<0 🡪 Despl. Izq. 3

11101 1100\_ 00101 D\_n=D\_n + d 3

11101 11000 00101 D\_n<0 🡪 q\_0=0 3

11011 1000\_ 00101 D\_n<0 🡪 Despl. Izq 4

00000 1000\_ 00101 D\_n=D\_n + d 4

00000 10001 00101 D\_n>0 🡪 q\_0=1 4

00001 0001\_ 00101 D\_n>0 🡪 Despl. Izq. 5

11100 0001\_ 00101 D\_n=D\_n – q 5

11100 00010 00101 D\_n<0 🡪q\_0=0 5

00001 00101 00101 D\_n=D\_n + d 5 🡪 En la última se hace la última comprobación y como es negativo se suma el divisor

IEEE 754

Simple precisión 8 bits 🡪 exceso a 127 en simple precisión 1111111=127

Conversión Decimal-754

1. Representar en coma fija el número decimal
2. Pasar el número decimal a binario
3. Normalizar Mantisa
4. Normalizar exponente
5. Representar en formato IEEE 754

-0.81357 x 10^2

1. -81.375

2. Parte entera: 81🡪1010001(binario)

Parte decimal: 0.375 🡪 0.011

1010001.011

3. 1.010001011 x 2^6

4. E= 6+127 = 133 🡪 10000101(binario)

1 Bit 8 Bits 23 Bits

1 10000101 01000101100000000000000

MIRAR REGLAS DE CÓMO HACER LA SUMA, LA RESTA, LA MULTIPLICACION Y LA DIVISIÓN DE NÚMEROS EN IEEE 754.