Ejercicio 1 Realizar la traza de multiplicación con signo según el algoritmo de Booth resultante de multiplicar los números decimales 30 y 13 con 6 bits. Dar el resultado en binario y en hexadecimal

30 multiplicando: **011110**

13 multiplicador: **001101**

1. Se inicializan tres registros M, S y Q con el multiplicando, su complemento a dos y el multiplicador.
2. Se inicializa un registro A a 0 y un registro de un único bit Q_{-1} a 0.

2. Si 1,0: Se suma A + S y se almacena en A. El acarreo se pierde.

   Se desplaza un bit a la derecha A, Q, y Q_{-1}.

   Si 0,1: Se suma A + M y se almacena en A. El acarreo se pierde

   Se desplaza un bit a la derecha A, Q, y Q_{-1}.

   Si Q_0 y Q_{-1} son iguales (1-1, 0-0): sólo se realiza desplazamiento de A, Q, y Q_{-1}

A	Q	Q_{-1}	M	S	Acción	Vez
000000	001101	0	011110	100010	Inicio	
100010	001101	0	011110	100010	1-0	1
110001	000110	1			Desp	1
001111	000110	1			0-1	2
000111	100011	0			Desp	2
101001	100011	0			1-0	3
110100	110001	1			Desp	3
111010	011000	1			1-1 solo desp	4
011000	011000	1			0-1	5
001100	001100	0			Desp	5

000110	**000110**	**0**			**Desp**	**6**

000110000110 = 390 = 0001 1000 0110= 186H

Ejercicio 2 Realizar la traza de multiplicación con signo según el algoritmo de Booth resultante de multiplicar los números decimales -5 y -6 con 5 bits. En binario y en hexadecimal

5 en binario= 00101

Ca2(5)=11011= -5

6 en binario= 00110

Ca2(6)=11010=--6

1. Se inicializan tres registros M, S y Q con el multiplicando, su complemento a dos y el multiplicador.
2. Se inicializa un registro A a 0 y un registro de un único bit Q_{-1} a 0.
3. Si 1,0: Se suma A + S y se almacena en A. El acarreo se pierde.

   Se desplaza un bit a la derecha A, Q, y Q_{-1}.

   Si 0,1: Se suma A + M y se almacena en A. El acarreo se pierde

   Se desplaza un bit a la derecha A, Q, y Q_{-1}.

   Si Q_0 y Q_{-1} son iguales (1-1, 0-0): sólo se realiza desplazamiento de A, Q, y Q_{-1}

A	Q	Q_{-1}	M	S	Acción	Vez
00000	11010	0	11011	00101	Inicio	
00000	01101	0			0-0 solo desp	1
00101	01101	0			1-0 A+S	2
00010	10110	1			Desp	2
11101	10110	1			0-1 A+M	3
11110	11011	0			Desp	3
00011	11011	0			1-0 A+S	4
00001	11101	1			Desp	4
0000	**11110**	**1**			Desp	**5**

11110= 30 =0001 1110=1EH

Ejercicio 3 Realizar la traza de multiplicación con signo según el algoritmo de Booth resultante de multiplicar los números decimales -2 y -0 con 3 bits

2 en binario= 010

Ca2(2)= 110

0 en binario 000

A	Q	Q_{-1}	M	S	Acción	Vez
000	000	0	110	010	Inicio	
000	000	0			0-0 solo desp	1
000	000	0			0-0 solo desp	2
000	000	0			0-0 solo desp	3

Ejercicio 3 Realizar la traza de multiplicación con signo según el algoritmo de Booth resultante de multiplicar los números decimales 1 y -2 con 3 bits

2 en binario= 010

Ca2(2)= 110

A	Q	Q_{-1}	M	S	Acción	Vez
000	110	0	001	111	Inicio	
000	011	0			0-0 solo desp	1
111	011	0			1-0 A+S	2
111	101	1			desp	2
111	110	1			1-1 solo desp	3

111110 es -2 en binario Ca2 solo que con 6 dígitos

Ejercicio 4 Realizar la traza de división con signo según el algoritmo visto en clase resultante de dividir los números decimales -11 y -5 con 5 bits

11= 01011

Ca2(11) =-11= 10101

5= 00101

Ca2(-5) =-5= 11011

Se inicializan dos registros M y S con el divisor y su **complemento a dos**.

Se inicializan dos registros A y Q con el dividendo extendido al tamaño 2n.

Desplazar A y Q un bit a la izquierda

Si M y A tienen el mismo signo: Se suma A + S y se almacena en A. El acarreo se pierde

Si no tiene éxito , entonces hacer $Q_0 = 0$ y restablecer el valor anterior de A.

Si tiene éxito hacer $Q_0 = 1$.

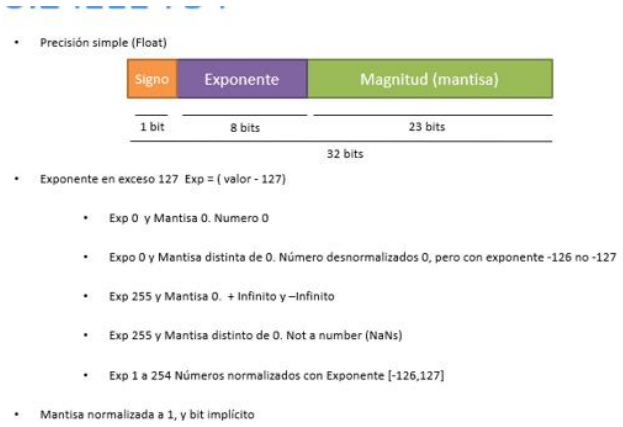
A	Q	M	S	Acción	Vez
11111	**10101**	**11011**	**00101**	Inicio	
11111	01010	**11011**	**00101**	Desp a la izq de Q y A	1
00100	01010	**11011**	**00101**	M y A mismo signo 1 A+S	1
11111	01010	**11011**	**00101**	No exito	1
11110	10100	**11011**	**00101**	Desp a la izq de Q y A	2
00011	10100	**11011**	**00101**	M y A mismo signo 1 A+S	2

11110	10100	**11011**	**00101**	No exito	2
11101	01000	**11011**	**00101**	Desp a la izq de Q y A	3
00010	01000	**11011**	**00101**	M y A mismo signo 1 A+S	3
11101	01000	**11011**	**00101**	No exito	3
11010	10000	**11011**	**00101**	Desp a la izq de Q y A	4
11111	10000	**11011**	**00101**	M y A mismo signo 1 A+S	4
11111	10001	**11011**	**00101**	Éxito Q0=1	4
11111	00010	**11011**	**00101**	Desp a la izq de Q y A	5
00100	00010	**11011**	**00101**	M y A mismo signo 1 A+S	5
11111	00010	**11011**	**00101**	No exito	5

Resto: A= **11111 = 00001 en complemento a 2 = 1**

Cociente Q= 00010= 2

Ejercicio 5. Represente en el estándar IEEE 754 de simple precisión el valor -36.
Solución:
El valor 36 en binario es 100100. 100100 = 1.00100 x 2^{5} . Por tanto:
• El bit de signo es 1, porque el número es negativo.
• El exponente es 5, por tanto el exponente que se almacena es 5 + 127 = 132, que en binario es 10000100
• La mantisa es 001000000 00000
Por tanto, el número -36 se representa como 1 10000100 00100000000000000000000 en binario.



Ejercicio 6 Indique el valor decimal del siguiente número hexadecimal 0x00600000 que representa un número en coma flotante según IEEE 754 (precisión simple)

Solución:
0x00600000 en binario es 0000 0000 0110 0000 0000 0000 0000 0000
Signo = 0, número positivo
Exponente = 00000000
Mantisa = 1100000...0000
Se trata de un número no normalizado cuyo valor es 0,11 x 2^{-126} = 0,75 x 2^{-126}

Ejercicio 6 Represente el número -24,50 utilizando el estándar de coma flotante de simple precisión IEEE 754. Exprese dicha representación en binario y en hexadecimal.

24,5 $_{(10}$ = 11000,1 $_{(2}$ = 1,10001 x 2^{4}
Signo = 1, número negativo
Exponente = 4 + 127 = 131 = 10000011
Mantisa = 1000100000 .. 00000
En binario es: 11000000111000100000 00000
En Hexadecimal es: 0xC1C40000

Ejercicio 7 ¿Cuál es el número positivo normalizado más pequeño que se puede representar utilizando el estándar de simple precisión IEEE 754? Justifique su respuesta. Indique también el número positivo no normalizado más pequeño que se puede representar. Justifique de igual forma su respuesta.

Solución:
a) El número positivo normalizado más pequeño representable en el estándar IEEE 754 (32 bits) es:
0 00000001 00000000000000000000000
positivo normalizado más pequeño
Cuyo valor es $1.0 * 2^{1-127} = 2^{-126}$

b) El número positivo no normalizado más pequeño representable en el estándar es:
0 00000000 00000000000000000000001
positivo no normalizado más pequeño
Cuyo valor es $2^{-23} * 2^{-126} = 2^{-149}$

Ejercicio 8 Indique el valor decimal del siguiente número representado en el estándar IEEE 754 de simple precisión: 0xBF400000.

El valor decimal de 0xBF400000 es el siguiente:
En Binario: 1011 1111 0100 0000...00
Signo= Negativo
Exponente = $01111110_{(2} = 126_{(10}$ = Exponente = -1
Mantisa = 1
Número en binario: $-1,1 \times 2^{-1}{}_{(2} = -0,11_{(2} = -0,75_{(10}$

Ejercicio 9 En relación al estándar IEEE 754 responda a las siguientes preguntas:

a) En una representación IEEE 754 de 32 bits, indique de forma razonada el número de valores no normalizados que se pueden representar.

b) En un computador de 32 bits, ¿Se puede representar de forma exacta el valor 2 27 +1 en una variable de tipo float? ¿y en una variable de tipo int? Razone su respuesta. Sabiendo que que los float se guardan en coma flotante IEEE754 y los int en complemento a 2

c) Represente en el estándar IEEE 754 de doble precisión el valor 12.5. Exprese el resultado en hexadecimal.

a) Un valor no normalizado se corresponde con un exponente (8 bits) cuyo valor es 0 y una mantisa (23 bits) con valor distinto de cero. El número de elementos que se pueden representar es, por tanto, 2 (positivo y negativo) x 2^{23} -1 = 2 × (2^{23} -1)

b) El valor $2^{27} + 1 = 10000000000000000000000001_{(2}$ =
$1.00000000000000000000000001 \times 2^{27}{}_{(2}$

Dado que en la mantisa de un número de tipo `float` (IEEE 754 de 32 bits) solo se pueden almacenar 23 bits, aparte del bit implícito, para el número anterior no se podrían almacenar los 4 bits menos significativos del número, por lo que el número no se podría representar de forma exacta.

En una variable de tipo `int`, que
emplea complemento a 2, el rango de representación es [-2^{31} , 2^{31} -1. En este caso, sí que se puede representar el número de forma exacta

c)
$12.5_{(10} = 1100.1_{(2} = 1.1001 \times 2^{3}{}_{(2}$
Signo = 1
Exponente = 1023+3 = $1026_{(10}$ 1024+2 (10 = $10000000010_{(2}$
Mantisa = 1001.... 000000 (52 bits)
La representación del valor 12.5 es:
0 100 0000 0010 1001 00000000 00000
Expresado en hexadecimal queda: 0x4029000000000000

Ejercicio 10 En una representación IEEE 754 de 32 bits, ¿cuál es el número de valores que hay comprendidos entre el número 8 y el 9?

Para poder calcular el número de valores comprendidos entre el 8 y el 9 es necesario representar ambos números en coma flotante:

$8_{(10} = 1000_{(2} = 1.000 \times 2^{3}{}_{(2}$
Signo = 1
Exponente = 127+3 = $130_{(10}$ = $10000010_{(2}$
Mantisa = 000000.... 000000 (23 bits)

$9_{(10} = 1001_{(2} = 1.001 \times 2\ 3_{(2}$
Signo = 1
Exponente = 127+3 = $130_{(10}$ = $10000010_{(2}$
Mantisa = 001000.... 000000 (23 bits)

La representación del valor 8 es: 0 10000010 00000000000000000000000
La representación del valor 9 es: 0 10000010 00100000000000000000000

Lo único que hay que hacer es calcular el número de valores comprendidos entre ambas representaciones. Elprimer bit (comenzando desde la derecha) con un valor distinto es el que se encuentra en la posición 21, portanto el número de valores es 2^{20}.