**Clase de repaso 15/03/2016**

1. Dado A que representa la configuración en base 10 de un número almacenado en formato empaquetado de 3 bytes, se pide:
2. Indicar cuál es el número almacenado.
3. Almacenar el el número en formato BPFlotante IEEE 754 de precisión simple, dando su configuración hexadecimal.

A = (69659)10

1a) Para averiguar cuál es el número, necesito pasar a la configuración que trabaja el formato empaquetado (Hexadecimal)

6965910 -> 16 = 1101B16 (divisiones sucesivas de 16)

Expandimos a 3 Bytes: 01 10 1B16

Por la letra B, podemos afirmar que se trata de un número negativo (B,D: Negativos)

El número almacenado en base 10 es: -110110

1b) Para pasar al formato BPFlotante IEEE 754 es necesario:

* Pasar el número al Binario
* Tiene que ser en Positivo (ya que el formato cuenta con 1 Bit para el signo del número)

Para pasar -110110 al binario, podemos hacer las divisiones sucesivas de 2 (más cuentas), o podemos usar como pivot la base 16, y luego usando la regla de potencia (por cada símbolo que tenemos en base 16, obtenemos 4 símbolos en binario).

110110 -> 16 = 44D16 -> 2 = 0100 0100 11012

Para pasar al formato BPFlotante IEEE 754, primero normalizamos el número a la notación científica, dejando un 1 a la izquierda de la coma:

0100 0100 11012 🡪 [1,0001001101 \* 101010 ]2

(El exponente es 1010, ya que se corrieron 10 lugares la coma, y el número 1010 en base 2 corresponde al 10102)

Signo: Cómo se trata de un número negativo, el bit de signo es 1

Exp. Exceso: 1010 + 12710 = 13710 -> 2 = 1000 10012

Mantisa: 0001 0011 0100 0000 0000 0002

Pasamos limpio al formato:

1 10001001 0001 0011 0100 0000 0000 0002

Signo Exp. Exceso Mantisa

Finalmente pasamos el número a la configuración hexadecimal, agrupamos de a 4 dígitos desde derecha hacia izquierda: 1100 0100 1000 1001 1010 0000 0000 00002

C 4 8 9 A 0 0 016

1. Dado B que representa la configuración hexadecimal de un número almacenado en formato BPFlotante IEEE 754 de precisión simple, se pide:
2. Indicar cuál es el número almacenado.
3. Restar ambos números (el almacenado en A menos el almacenado en B) en formato BPF c/signo de 16 bits indicando la validez del resultado.

B = (C5002000)16

2a) Para averiguar cuál es el número, necesito pasar a la configuración que trabaja el formato (Binario)

Por la regla de potencia (cada símbolo de la base 16, se obtiene 4 símbolos en binario):

C 5 0 0 2 0 0 016

1100 0101 0000 0000 0010 0000 0000 00002

Acomodamos el número en el formato BPFlotante IEEE 754:

1 10001010 000000000100000000000002

Signo Exp. Exceso Mantisa

Del bit de Signo, podemos afirmar que se trata de un número negativo.

Para obtener el Exponente original, debemos restar el Exp. Exceso 12710

1000 10102 -> 10 = 13810

13810 - 12710 = 1110 (Exponente original)

Escribimos el número en la notación científica:

[-1,00000000010…..0 \* 101011 ]2 (1110 -> 2 = 10112)

Ahora para obtener el número original, corremos la coma tantas veces como nos indica el exponente

(En caso de que el exponente sea negativo, se corre hacia la izquierda)

El número original sería: -1000 0000 00102 -> 10 = - (211 + 21)10 = -205010

2b) Para la resta en formato BPF c/signo entre un número A contra un número B, se utiliza la regla:

A – B = A + (Complemento B)

Como B ya lo teníamos en binario, calculamos primero el Complemento B:

B = -1000 0000 00102

Expandimos B a 16 bits: B = - 0000 1000 0000 00102

Como el número es negativo, para pasar el número B al BPF c/signo, se necesita complementarlo.

Podríamos complementar el número B para pasar al BPF c/signo, pero si observamos la regla

mencionada [A – B = A + (Complemento B)], entonces al pasar el número al BPF c/signo y luego

volver a complementar el número, nos daría exactamente el mismo número sin el signo -.

Entonces, el Complemento B = 0000 1000 0000 00102

En 1b) vimos que el número A = - 0100 0100 11012

Expandimos A a 16 bits: A = - 0000 0100 0100 11012

Cómo el número A es negativo, para pasar al BPF c/signo, necesitamos complementarlo:

1111 1011 1011 00102

+ 12

A = 1111 1011 1011 00112 en BPF c/signo

11111 0000 0000 0102

Finalmente, A – B = A + (Complemento B) = 1111 1011 1011 00112

+ 0000 1000 0000 00102

0000 0011 1011 01012

Para concluir, vemos que los últimos 2 bits de acarreo (112) son iguales, entonces el resultado es válido.

1. Almacenar el resultado de 2b) en BPFlotante IEEE 754 de precisión simple, dando su configuración octal.

El resultado de 2b) es: 0000 0011 1011 01012

Cómo el número ya se encuentra en binario, y además por el resultado del 2b) sabemos que es un número positivo, entonces ya podemos empezar a normalizar el número:

0000 0011 1011 01012 = [1,110110101 \* 101001 ]2

(El exponente es 910, ya que se corrieron 9 lugares la coma, y el número 910 en base 2 corresponde al 10012)

Signo: Como se trata de un número positivo, el bit de signo corresponde un 0.

Exp. Exceso: 910 + 12710 = 13610 -> 2 = 1000 10002

Mantisa: 1101 1010 1000 0000 0000 0002

Pasamos limpio al formato:

0 10001000 1101 1010 1000 0000 0000 0002

Signo Exp. Exceso Mantisa

Finalmente pasamos el número a la configuración octal, donde agrupamos de a 3 dígitos desde derecha hacia izquierda: 001 000 100 011 011 010 100 000 000 000 0002

1 0 4 3 3 2 4 0 0 0 08