Introducción teórica

Teorema:

Para cada $x \in$ R, (de signo *s*) existe un par $(m \in R,\ exp \in Z)$ que verifica: $x = (-1)^s * 1.m * 2^{exp}$.

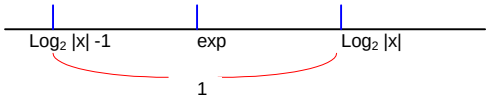
Demostración:

Vamos a calcular *m* y *exp*; y así al mismo tiempo:
- Demostraremos que existen, es decir, que el teorema es cierto.
- Proporcionaremos un método para calcularlos.

a) Cálculo de *exp*.

$$|x| = 1.m \cdot 2^{exp} \Rightarrow \left\{ \begin{array}{l} |x| \geq 1.0 \cdot 2^{exp} = 2^{exp} \quad \Rightarrow \quad exp \leq Log_2 |x| \\ |x| < 1.\hat{9} \cdot 2^{exp} < 2 \cdot 2^{exp} = 2^{exp+1} \quad \Rightarrow \quad exp+1 > Log_2 |x| \quad \Rightarrow \quad exp > Log_2 x \ - \ 1 \end{array} \right\} \Rightarrow (Log_2 |x|) \ - \ 1 \ < exp \leq Log_2 |x|$$

Hay que tener en cuenta que en cualquier intervalo $(r\text{-}1, r]$, con $r \in$ R, hay sólo un entero de Z.



Así pues, *exp* es el mayor entero de Z que está entre los reales $Log_2 |x| - 1$ y $Log_2 |x|$

Conclusión: la forma de encontrar *exp* es:

1º: Calcular $Log_2 |x| = \frac{Ln\,|x|}{Ln\,2}$. Lo más probable es que sea un real no entero.

2º: El *exp* buscado es el mayor de los enteros menores o iguales que $Log_2 |x|$, es decir: $exp = floor(Log_2 |x|)$

b) Cálculo de *m*:

$$|x| = 1.m \cdot 2^{exp} \Rightarrow \quad 1.m = \frac{|x|}{2^{exp}} \quad \Rightarrow \quad m = FRAC\left(\frac{|x|}{2^{exp}}\right)$$

Corolarios:

1º.-

$|x| = 1.m \cdot 2^{exp} \Rightarrow$ En formato IEEE 754 :

1.a) Campo exponente = exp+ Exceso = exp+ $(2^{n-1} - 1)$ = $floor(Log_2 |x|) + (2^{n-1} - 1)$

1.b) Campo mantisa = $m = FRAC\left(\frac{|x|}{2^{exp}}\right)$

2º.-

$$|x| = 1.m \cdot 2^{exp} \Rightarrow |x| = (1 + 0.m) \cdot 2^{exp} = 2^{exp} + 0.m \cdot 2^{exp} \Rightarrow \left\{ \begin{array}{l} |x| = 2^{exp} + 0.m \cdot 2^{exp} \geq 2^{exp} \\ |x| = 2^{exp} + 0.m \cdot 2^{exp} < 2^{exp} + 1.0 \cdot 2^{exp} = 2 \cdot 2^{exp} = 2^{exp+1} \end{array} \right\} \Rightarrow$$

$$2^{exp} \leq |x| < 2^{exp+1}$$

$$\Downarrow$$

La mayor potencia de 2 que se puede restar de $|x|$ es 2^{exp}.

$$\Downarrow$$

El primer bit significativo de $|x| = b_{n-1}\, b_{n-2} \ldots b_1\, b_0\, .\, b_{-1}\, b_{-2}\, b_{-3} \ldots$ es b_{exp}.

Algoritmo para un número fraccionario de decimal a binario:

Recordemos que en la igualdad:

$X = 0.D_{-1}\, D_{-2}\, D_{-3} \ldots = 0.b_{-1}\, b_{-2}\, b_{-3} \ldots$

tenemos dados los dígitos D_i; y se trata de encontrar los dígitos b_j que cumplen dicha igualdad. Cuando el número decimal tiene muchos dígitos fraccionarios y carece de parte entera, es preferible, por su rapidez, este algoritmo:

```
Acumulador := X ;
i := -1 ;
REPEAT
   IF Acumulador * 2 > 1
      THEN
         bi := 1 ;
         i := i - 1 ;
         Acumulador := Acumulador - 1 ;
      ELSE
         bi := 0 ;
   END
UNTIL  Acumulador  = 0
```

Su justificación puede encontrarse, por ejemplo, en el capítulo 2 del libro de ETC I (Gestión).

Ejercicios.

Convierta el número decimal 1227 a complemento a dos de 16 bits.

Solución:

Al tratarse de un número positivo, nos basta con calcular su módulo, pues coinciden número y módulo.

1227	÷ 2 = 613.5	⇒	b_0 = 1
613	÷ 2 = 306.5	⇒	b_0 = 1
306	÷ 2 = 153.0	⇒	b_0 = 0
153	÷ 2 = 76.5	⇒	b_0 = 1
76	÷ 2 = 38.0	⇒	b_0 = 0
38	÷ 2 = 19.0	⇒	b_0 = 0
19	÷ 2 = 9.5	⇒	b_0 = 1
9	÷ 2 = 4.5	⇒	b_0 = 1
4	÷ 2 = 2.0	⇒	b_0 = 0
2	÷ 2 = 1.0	⇒	b_0 = 0
1	÷ 2 = 0.5	⇒	b_0 = 1
0	÷ 2 = 0.0	⇒	b_0 = 0

&1227 = % 0000 0100 1100 1011 = $04CB

Convierta el número decimal -1227 a complemento a dos de 16 bits.

Solución:

Al tratarse de un número negativo, primero convertimos su módulo de decimal a binario y después los complementamos a dos.

1º Cálculo del módulo. Es un problema ya resuelto anteriormente: &1227 = % 0000 0100 1100 1011
2º Complemento a dos:
- 2º.1 Inversión de todos los bits: %1111 1011 0011 0100
- 2º.2 Suma de 1 en el bit menos significativo %1111 1011 0011 0101 = $FB35

Convierta el número decimal -1227 a complemento a dos de 20 bits.

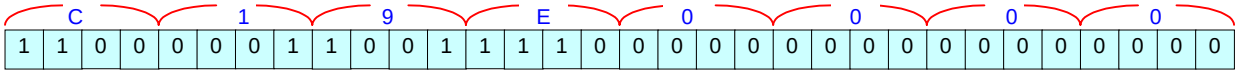
Solución:

Si aprovechamos el resultado de un problema resuelto anteriormente, -1227 en complemento a dos de 16 bits es
%1111 1011 0011 0101
Para almacenarlo en un registro de 20 bits debemos hacer una extensión de signo:
%1111 1111 1011 0011 0101 = $FFB35

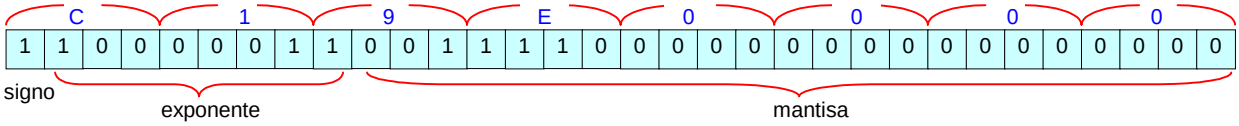
Encuentre el número decimal cuya representación en formato IEEE 754 en coma flotante de 32 bits, compactada en hexadecimal es C19E0000.

Solución:

Paso 1º: Obtención de la información almacenada en el registro a partir de la expresión compactada en hexadecimal



Paso 2º: Identificación de los diferentes campos presentes en el registro



Paso 3º: Cálculo de los diferentes componentes de la representación en punto flotante

Bit de signo = 1 $\Rightarrow$ El número es negativo

Mantisa = %1.001111

Exponente auténtico = Exponente representado - Exceso

$$\begin{array}{rr} & \%10000011 \\ - & \%1111111 \\ \hline & \%00000100 = 4 \end{array}$$

Paso 4º: Cambio de representación del número desde punto flotante a punto fijo

Número = $-(\%1.001111) \cdot (2^4) = -1.234375 \cdot 16 = -19.75$

Obtenga la representación del número 1540 en formato normalizado IEEE 754 para coma flotante de 32 bits.

Solución:

Paso 1º: Cálculo del campo exponente

Según el apartado a) del primer corolario:

$$\text{exp} = \text{floor}\left(\text{Log}_2 |\, x \,|\right) = \text{floor}\left(\frac{\text{Ln}\,|\, x \,|}{\text{Ln}\,2}\right) = \text{floor}\left(\frac{\text{Ln}\,|1540\,|}{\text{Ln}\,2}\right) = \text{floor}(10.58871464) = 10$$

Campo exponente = exp + Exceso = 10 + (2^7 - 1) = 10 + 127 = 137 = %10001001

Paso 2º: Cálculo del campo mantisa

Según el apartado b) del primer corolario:

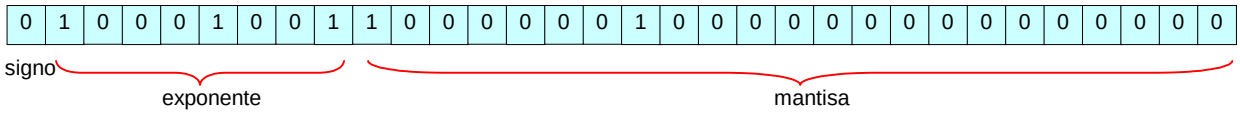
$$m = \text{FRAC}\left(\frac{|x|}{2^{\text{exp}}}\right) = \text{FRAC}\left(\frac{|1540|}{2^{10}}\right) = \text{FRAC}(1.050390625) = 0.050390625$$

Mantisa &1.050390625 = % 1.10000001

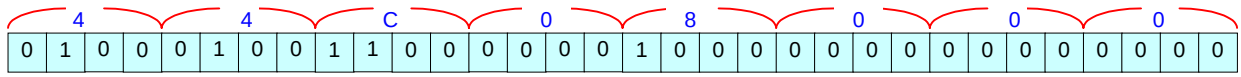
Paso 3º: Cálculo del campo de signo

1540 ≥ 0 ☞ Bit de signo = 0

Paso 4º: Almacenamiento en un registro de 32 bits de todos los campos anteriormente calculados



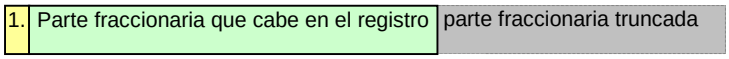
Paso 5º: Expresión compactada en hexadecimal de la información almacenada en el registro



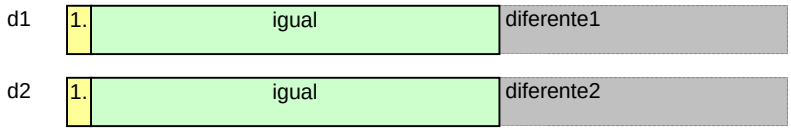
Calcule el error, en valor absoluto, que se comete al representar el número decimal 291.072 con el número 4391 en formato IEEE754 (de 16 bits).

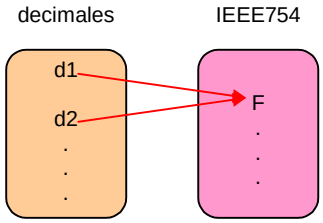
Solución:

Debido a la limitación de la longitud del registro donde se almacena el número IEEE754 (en este caso es 16 bits, pero siempre existirá algún límite), puede suceder que al convertir de decimal a IEEE754 haya que truncar la mantisa del número decimal.



Por este motivo, existe la posibilidad de que diferentes números decimales tengan la misma representación IEEE754:



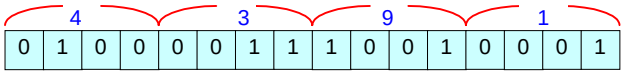


En este caso, el número en IEEE754 se corresponde exactamente sólo con aquel decimal cuya parte truncada es nula:

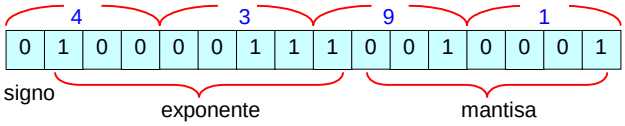
Cuando se quiere conocer el error (en valor absoluto) cometido al representar un decimal d_i con el número F en IEEE754, por definición se calcula el valor absoluto de su resta: $Error\ absoluto = |d_i - F|$ (en esencia, toda comparación consiste en una substracción). Por supuesto, las comparaciones sólo se pueden hacer entre entidades homogéneas, por ello ambos números deben estar en el mismo formato. Como las operaciones aritméticas se realizan de forma más directa en decimal que en IEEE754, convertimos el IEEE754 a decimal; y luego calculamos el valor absoluto de la resta. Observe el lector que al hacer esto, estamos comparando el número d_i con el número d_{exacto}

Conversión del IEEE754 a decimal

Paso 1º: Obtención de la información almacenada en el registro a partir de la expresión compactada en hexadecimal



Paso 2º: Identificación de los diferentes campos presentes en el registro



Paso 3º: Cálculo de los diferentes componentes de la representación en punto flotante

Bit de signo = 0 $\Rightarrow$ El número es positivo
Mantisa = %1.0010001
Exponente auténtico = Exponente representado - Exceso

%10000111
- %1111111

%00001000 = 8

Paso 4º: Cambio de representación del número desde punto flotante a punto fijo

Número = (%1.0010001) * (2^8) = %100100010= &290

Comparación

Error absoluto = $|d_i - d_{exacto}| = |291.072 - 290| = 1.072$