Apuntes de Clases

Representación de números binarios en punto fijo y punto flotante.

Realizado por Sergio Noriega

Introducción a los Sistemas Lógicos y Digitales
Departamento de Electrotécnia
Facultad de Ingeniería
Universidad Nacional de La Plata
2003

INDICE

1 - Introducción.

Los números reales en base diez ó decimal que nosotros habitualmente utilizamos para realizar cálculos matemáticos ya sea para tareas cotidianas ó científicas generalmente son expresados en una de dos formas:
Una es en punto fijo (por ej: $345,70 , 230Kg , -10°C, etc.), donde se emplean tres campos para la representación: signo, parte entera y parte decimal.

Otra forma es en notación científica ó punto flotante, donde un número se expresa también con tres campos: signo, mantisa y exponente (por ej: 10E-09 seg. = 0,000000001 seg. , 12.74E04 metros = 127400 metros, etc.).

Generalmente esta última notación se emplea cuando el número a representar es muy grande ó muy pequeño y llevaría muchos dígitos su representación en punto fijo, lo cual puede resultar en errores de cálculo, de representación, etc.

Para el caso de números en formato binario, el análisis es similar.
Tanto punto fijo como punto flotante son los dos sistemas de representación de números binarios con signo.

2 - Representación de números binarios en punto fijo.

Un número binario con signo se puede representar por tres métodos diferentes, denominados: signo y módulo, complemento a la base disminuída (también llamado complemento a 1: Ca1) y complemento a la base (complemento a 2: Ca2).

Los empleados actualmente son los de complemento a 1 y complemento a 2, los cuales se utilizan para exprersar números binarios en formato de punto fijo, es decir, teniendo tres campos: uno para el signo, otro para la parte entera y el restante para la parte decimal.

El tratamiento en Ca1 y Ca2 hace que el signo queda embebido dentro del campo de la parte entera del número y empleando la definición de complemento de un número binario éste automáticamente queda establecido así como en el caso de realizar operaciones de suma y de resta.

La parte decimal (parte del número después de la coma), también queda embebida dentro de la representación del número completo, ya que cuando se deban realizar operaciones aritmética, al igual que como se procede en números de base 10, se toma todo el número para la operación.

En la suma y resta, la posición de la coma nunca se modifica.
Si hay que sumar o restar dos números, se debe primero hacer coincidir las posiciones de la coma y luego realizar la operación aritmética como si el número fuera entero.

En caso de realizar operaciones de multiplicación y división, el procedimiento es análogo; lo único que cambia es que se debe correr la coma al número binario resultado, según corresponda, de igual forma que en el caso de números decimales.

Ejemplo:

Suma en Ca2 de dos números: A = 0111001,110001 y B = 1110011,001

Se disponen los números de la siguiente forma y se suma:

```
  0111001,110001
  1110011,001000
-----------------
 10101100,111001
```

Un número expresado en punto fijo fuera cual fuere su base, tendrá un número limitado de símbolos, lo que nos limitará el rango de representación.

En general tendrá ***m*** símbolos en la parte entera y ***n*** símbolos en la parte decimal.

Suponiendo que tenemos un número binario expresado con 5 cifras significativas (5 bits después de la coma) y de la siguiente expresión obtenemos que:

La más pequeña representación es: $2^{-5} = 1/(2^5) = 1/32_{10} = 0.03125_{10}$

El error absoluto será constante e igual a:

$$eA = +/-1/2LSB = +=/-1/2\ (2^{-5}) = +/- 2^{-6}$$

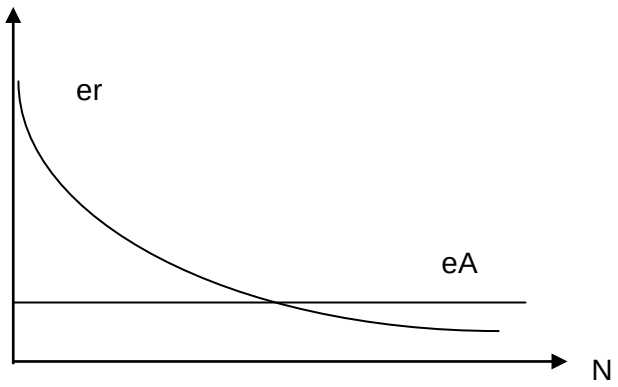
Por otro lado, el error relativo será variable dependiendo del valor del número a representar:

Este se define, como el error absoluto dividido dicho valor.

Se expresa generalmente en [%]:

$$er[\%] = eA/N \times 100$$

Las gráficas para el caso de los errores absolutos y relativos se pueden observar en la siguiente figura:



Se puede notar que para valores de N pequeños, el error relativo puede ser muy grande.

El peor caso práctico es cuando el número es N es igual al error absoluto, por ejemplo: 2^{-5}, donde tendremos un er = 100%.

Está también el caso límite que es cuando N=0, donde el <er> es infinito.

3 - Representación de números binarios en punto flotante.

Como en el caso de números de base decimal, también es necesario en algunos casos trabajar con números muy grandes ó pequeños. Para ello resulta mas cómodo poder expresarlos en un formato que permita operar con números de diferente posición de la coma. A esto se le denomina números de coma flotante.

Su uso se hace imprescindible cuando hay que realizar por ejemplo cálculos entre números muy grandes, donde el formato permite poder describirlos con mayor comodidad para su operativa y visualización del resultado.

La forma usual es expresarlos con tres campos: uno de signo del número total, otro con la mantisa (ó significando) y el tercero con el exponente del número.

En base diez tenemos por ejemplo:

+10E-07 +4.322E+122 -1.003E-98

El primer signo corresponde al número, mientras que el segundo al exponente.

Si el signo del exponente es positivo, entonces el número es mayor que 1, caso contrario es menor que 1.

En binario es igual. Se han propuesto varias formas de representación de números de coma flotante. El que más ha prevalecido es el definido por la IEEE bajo la norma IEEE P754.

En ella se define el formato que se debe respetar para la representación de un número binario con signo en punto flotante como así también el tratamiento de errores, operaciones, etc.

Según esta norma un número binario en punto flotante normalizado se puede expresar de tres formas diferentes, según el rango del mismo y son de menor a mayor:

Punto flotante de simple precisión.
Punto flotante de doble precisión.
Punto flotante de precisión extendida.
Punto flotante en formato BCD empaquetado.

La razón de tener diferentes formatos es debido a que hay casos donde se necesitan realizar operaciones con números de rango similar y cuyo resultado se puede volver a expresar en ese mismo formato sin errores apreciables.

En otros casos, por ejemplo si los números son de diferente rango (uno con exponente positivo y otro con exponente negativo), la diferencia en módulo entre ellos puede ser significativa y si bien pueden ser expresados en el mismo formato, puede que no alcance los dígitos para representar al resultado ó que el error sea inadmisible.

Punto Flotante Simple Precisión:

Formato de Simple Precisión:

Signo	Exponente	Fracción del Significando
1 bit:: "0" positivo	8 bits	23 bits

El número expresado en este formato consta de un total de 32 bits.
El primero es para expresar el signo del número y como convención se adopta: "0" positivo y "1" negativo.

A continuación se pone el campo del exponente con 8 bits y por último el de la fracción del significando con 23 bits, donde se considera que el significando siempre es de la forma 1.x....x, donde x....x es la fracción del significando.

Los pasos a seguir para determinar el significando y el exponente del número para expresarlo en este formato son los siguientes:

1 - Dado el número binario a formatear, se lo debe expresar de tal forma que la parte entera sea igual a 1 debiendo correr para ello si es necesario, la coma **n** veces ya sea para la derecha (cuando el número es menor que 1) ó para la izquierda (cuando el número es mayor que la unidad).

2 - El número n de veces que se ha corrido la coma hasta que quede expresado como 1. x....x (donde n es en realidad el exponente de 2^n: <léase 2 elevado a la potencia n>), servirá para calcular el número que deberá colocarse en el campo del exponente.

Se debe entender que no se coloca directamente el exponente en los 8 bits asignados para el campo llamado "exponente", sino un número relacionado con éste siguiendo la siguiente regla:

Como se quiere expresar tanto exponentes positivos como negativos de los 256 números posibles que se pueden poner en el campo mencionado con 8 bits, se considera que el exponente n = 0 tiene asociado al número 127 = 01111111, denominado "***bias***", el cual debe ir en los 8 bits del campo "exponente".

2.1 Si el exponente **n** es positivo, el número que se debe poner en el campo "exponente" estará comprendido entre 128=10000000 y 254 = 11111110 (se reserva el 255 = 11111111).

El número que se debe poner en el campo "exponente" será la suma del exponente real **n** y 127 sin signo.

Ej: Si n = +6, entonces el número es 127 + 6 = 133 = 10001111.

2.2 Si el exponente **n** es negativo, dicho número estará comprendido entre 126 = 01111110 y 1 = 00000001 (se reserva el 0 = 00000000).

El número que se debe poner en el campo "exponente" será igual a la resta entre 127 y el exponente real n.

Ej: Si n = - 40, entonces el número es 127 - 40 = 87 = 01010111.

Como regla de primera comprobación, se puede ver que si el exponente real es positivo, el número formateado que lo representa en el campo "exponente" comienza con "1", mientras que si el exponente real es negativo, comenzará con "0".

Esto es importante de aclararlo ya que la convención que se utiliza en esta norma para representar al exponente de un número binario ***es diferente*** de la empleada en las representaciones de números con signo en signo y módulo y en Ca1 y Ca2 para punto fijo.

3 - Con respecto al campo de la fracción del significando, en punto flotante simple precisión se considera, como se dijo, que la mantisa tiene como parte entera al número 1.

Los bits de la parte decimal son los que se deben colocar en el campo "fracción del significando", siendo de 23 bits.

Si la parte decimal del número tiene menos de 23 bits de longitud, se deberá completar con ceros.

Rango de representación:

El módulo de la mantisa que es 1.x....x tendrá como valores mínimo y máximo:

Valor mínimo = $1.00000000000000000000000 = 1_{10}$
Valor máximo = $1.11111111111111111111111 \cong 2_{10}$

Con respecto al exponente:

Valor máximo positivo = +127 ≡ (11111110 formateado)
Valor máximo negativo = -126 ≡ (00000001 formateado)

Por lo tanto el número mas grande que se puede representar en simple precisión es:

±(2-2^-23) 2^+127 ≅ ± 2 2^+127

El número más pequeño será:

±(1.0) 2^-126

Ejemplo 1: Dado el número decimal -115.25 expresarlo en punto flotante simple precisión según norma IEEE P754.

Paso 1: Pasar el número a binario punto fijo.

La parte entera es igual a: 115 = 1110011.
La parte decimal es igual a: 0,25 = 0,010.....0.

El número 115,25 en base 10, es igual al número binario 1110011,01 en punto fijo.

Es importante aclarar que si bien el signo del número es negativo, éste se identifica poniendo en "1" el campo de signo. Siempre se trabaja en la conversión de decimal a binario con números positivos.

Paso 2: Pasar el número binario de punto fijo a punto flotante.

El signo es negativo, por lo tanto el primer campo se pone a "1".

1		
s	exponente	fracción del significando

El módulo del número que es 1110011.01 se debe formatear como
1.xxxxx 2^n donde xxxxx son los 23 bits de la parte decimal y n el exponente que luego hay que formatear.

De esto se tiene que para que quede el módulo del número expresado de esa forma se deberá correr la coma hacia la izquierda 6 lugares, por lo tanto tendremos que el módulo se puede poner temporariamente de la forma:

1.11001101 2^+6

La parte decimal: 110011010.....0 se debe poner en el campo fracción del significando. Nos queda parcialmente:

1		11001101000000000000000
s	exponente	fracción del significando

El exponente +6 se debe formatear según la regla anteriormente citada.
Sumando 6 al bias (127), tendremos:

```
   6 = 00000110
 127 = 01111111
-----------------
 133 = 10000101
```

El número 10000101 es el que se debe poner en el campo "exponente".

Por lo tanto, el número completo en punto flotante normalizado simple precisión que representa al -115.25 es:

1	10000101	11001101000000000000000
s	exponente	fracción del significando

Ejemplo 2: Dado el número decimal 0.0000129 expresarlo en punto flotante simple precisión según norma IEEE P754.

Paso 1: Pasar el número a binario punto fijo.

La parte entera es igual a: 0 = 0.
La parte decimal es igual a:

0,0000129 = 0,00000000000000001101100001101101001001111 1...

Este número no es divisible por potencia de dos, por lo tanto, su representación en binario puede requerir de infinitos dígitos binarios.

Por razones de resolución, con este formato no tiene sentido de ir mas allá de la posición número 23 después del primer uno que se encuentre, barriendo de izquierda a derecha, ya que todo dígito después de ese tope no podrá entrar en el campo "fracción del significando"en la representación de punto flotante simple precisión. Esto implica un redondeo por defecto.

El signo del número es positivo, éste se identifica poniendo en "0" el campo de signo.

Paso 2: Pasar el número binario de punto fijo a punto flotante.

El signo se denota de la siguiente manera:

0		
s	exponente	fracción del significando

Se deberá correr la coma hacia la derecha 17 lugares, por lo tanto tendremos que el módulo se puede poner temporariamente de la forma:

1.10110000110110100010011111... 2^-17

La parte decimal: 10110000110110100010011111... se debe poner en el campo "fracción del significando" ajustado a 23 dígitos. Nos queda parcialmente:

0		10110000110110100010011
s	exponente	fracción del significando

El exponente -17 se debe formatear según la regla anteriormente citada.
Restando 17 al bias (127), tendremos:

127 = 01111111
017 = 00010001
110 = 01101110

El número 01101110 es el que se debe poner en el campo "exponente".

Por lo tanto, el número completo en punto flotante normalizado simple precisión que representa al número decimal +0.0000129 es:

0	01101110	10110000110110100010011
s	exponente	significando

Punto Flotante Precisión Doble:

Formato de Doble Precisión:

Signo	Exponente	Fracción del Significando
1 bit:: "0" positivo	11 bits	52 bits

Este formato es similar al de precisión simple, con la excepción de disponer de mas bits para la representación del campo de exponente y fracción del significando.

El bias en este caso es representado por el número 1023 que equivale al exponente "0".

Se puede representar desde el exponente +1023 hasta el -1022.

El número máximo a representar es el +/-(2-2^-52) 2^+1023.

El mínimo número a representar es el +/- (1.0) 2^-1022.

Aquí se reservan también los números 11111111111 y 00000000000 del campo de exponente para casos especiales como se verá luego.

Punto Flotante Precisión Extendida:

Signo	Exponente	Campo nulo	Significando
1 bit: "0" positivo	15 bits	16 bits: todos "0"	64 bits: x,x.....x

Este formato de representación tiene como diferencia sustancial con los anteriores que hay un campo denominado "significando" y no "fracción del significando", dado que en el primero se permite poder expresar números no normalizados ya que se dispone de la representación de la parte entera que puede valer "1"como el los dos primeros casos ó "0", que no puede ser realizado en los mismos.

Punto Flotante BCD empaquetado:

Este formato es en realidad para la representación de números decimales en formato binario pero empaquetándolos según el código BCD.

Dicho formato tiene la siguiente configuración:

Signo Signific.	Signo exp	Ind. esp	Exponente	Todos "0"	Significando
1 bit	1 bit	2 bits	12 bits	12 bits	68 bits

Signo del significando: "0": positivo, "1": negativo.
Signo del exponente: Idem anterior.
Indicador especial.
Exponente: 3 dígitos BCD (12 bits).
Campo nulo: Todos "0".
Significando: 17 dígitos BCD de la forma x.x.....x con el primer dígito en la parte entera y los 16 restantes como fracción.

Este formato además de la particularidad que puede representar un número decimal en formato BCD empaquetado, dispone de un campo de signo para el exponente y otro para iindicar situaciones especiales.

Representación de Casos especiales para los tres primeros formatos:

Números no normalizados:

Los números no normalizados representan valores reales próximos a cero, tanto positivos como negativos.

Se expresan con el campo de exponente con todos ceros ("0...0") y además para el caso de precisión extendida con la parte entera en "0".

Ceros:

Se representa tanto el cero positivo como el negativo, donde el campo de exponente es cero ("0...0") y la fracción de significando en cero ("0...0"), además de la parte entera en cero para precisión extendida.

Infinitos:

Se representa tanto el $+\infty$ como el $-\infty$, que indican el exceso de representación con el formato seleccionado.

Se expresa con el campo de exponente con todos "1" y el campo de fracción de significando igual a "0", además de poder estar la parte entera del significando en precisión extendida tanto en "1"como en "0".

No Número (NAN: Not A Number):

Se emplea para indicar operaciones indefinidas como por ejemplo expresar el resultado de la operación ∞/∞, x/0, etc.

Se indica con el campo de exponente con todos "1" y el campo de fracción de significando distinto de cero. En ell caso de precisión extendida la parte entera puede ser "0" ó "1".

Errores en representación de punto flotante:

A diferencia de punto fijo, el error absoluto en la representación de punto flotante no es constante sino variable en forma lineal con el número que representa.

Si consideramos por ejemplo el caso de simple precisión, tenemos 23 bits en la fracción de significando y 8 del exponente.

El error absoluto del número es la incerteza de la cantidad finita de representación de esa fracción truncada en el bit 23, por lo tanto tendremos un error de 2^-24 que será "pesada" además por el exponente, siendo entonces eA del número total N:

$$eA = 2^{-24}\ 2^{n}$$

donde n es el exponente del número N.

Por otro lado el error relativo está definido como vimos como:

$$er[\%] = eA/N\ 100$$

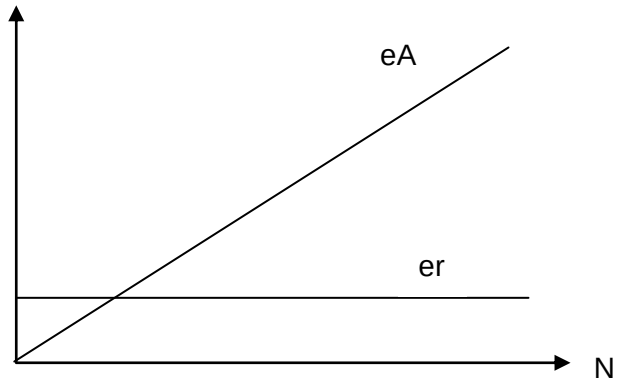
donde si tomamos como referencia N = 1 2^n tendremos:

$$er[\%] = 2^{-24}\ 2^{n} / (\ 1\ 2^{n})\ 100 = 2^{-24}$$

Se obtiene como es razonable un error constante ya que el efecto del exponente pesa siempre por igual en el error absoluto de Ncomo en el número N.

A diferencia de punto fijo donde en este caso el error relativo es una exponencial decreciente

Gráficamente podemos ver esto como:



La ventaja de punto flotante respecto de punto fijo es la capacidad de poder manejar números muy pequeños ó muy grandes.

La desventaja es que en general las operaciones realizadas con microprocesadores son mas lentas, además de tener que realizar reiterados cambios de la posición de la coma en las operaciones por ejemplo de suma y resta ó de multiplicación y división cuando el resultado en el significando excede de 2 y hay que volver a acomodarlo para cumplir con el formato normalizado.

4 - Bibliografía.

1 Circuitos Digitales y Microprocesadores. Herbet Taub.
2 Sistemas Digitales: Principios y aplicaciones. Ronald Tocci.
3 Norma IEEE P754.