**PONTIFICIA UNIVERSIDAD CATÓLICA MADRE Y MAESTRA**

**FACULTAD DE CIENCIAS DE LA INGENIERÍA**

**DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN**

****

**Tarea de teoría #1:**

Números flotantes en binario

**Presentado a:**  
ÁLVARO A. REYES P.

**Materia:**

Arquitectura Computacional

ST-ISC-359-T-001

**Presentado por:**

Adonis A. Veloz García

**Matrícula:**

2014-0452

**Fecha de entrega:**

Miércoles 19 de Octubre del 2016

Números flotantes

En programación, los números flotantes son una representación de los números racionales los cuales están conformados por una parte entera y una parte decimal. La precisión que este puede tomar suele depender del lenguaje de programación usado, ya que este especifica la cantidad de bytes que se le dedica en la memoria, y por lo tanto, cuanto puede crecer su parte decimal.

Categorías:

* Normalizados: Se refiere a los valores normales que incluye a todos los valores binarios cuyos exponentes tengan al menos un valor diferente, lo que significa que no todos sus bits de exponentes pueden ser 0 o 1 al mismo tiempo. Para el caso del cero

Ejemplo:

* Binario: **1 10000010 11110000000000000000000**
* Decimal: -15.5
* Desnormalizados: Los valores desnormalizados pueden verse cuando el exponente del valor binario tiene todos los bits en cero. Cuando esto ocurre se deben tomar ciertas medidas a la hora de hacer conversiones. Estos valores suelen ser muy cercanos al cero.

Ejemplo:

* Binario: 0 00000000 11110000000000000000000
* Decimal: 1.1479 \*10-41
* Not a Number (NaN): Son usados para representar valores que no son considerados un número real, viéndolo así como un valor indefinido como la división por 0. Para representarlo en binario flotante, se ve cuando la parte exponencial tiene todos sus bits en 1 y al menos uno de los bits de la parte fraccionaria es diferente de 0.

Ejemplo:

* Binario: 1 11111111 00000000000000000000001
* Decimal: NaN

Estándar IEEE 754

Es un estándar que define el formato que se usa en números flotantes. Este formato es el más extendido sobre números de coma flotante, al mismo tiempo que abarca valores especiales como el infinito y el NaN y otros valores como los desnormalizados y el cero. En lenguajes de programación como en C utilizan este formato para variables de tipo float y double en la precisión simple y doble respectivamente.

Este formato divide al valor binario en tres secciones:

* **El signo:** Este ocupa la posición más a la izquierda del número y se encarga de indicar el signo que tiene, ya sea 1 para negativo y 0 para positivo. En flotantes de 32 bits al igual que en 64 bits, ocupa solo una posición.
* **El exponente:** Esta parte se sitúa en la parte siguiente al signo e indica si el valor es normalizado o desnormalizado. El exponente también indica cuantas veces se rodará el punto flotante. Este valor ocupa 8 bits en 32 bits y 11 en 64 bits según el estándar.
* **El fraccionario o mantisa:** Esta parte ocupa la parte siguiente al exponente hasta llegar al final del número binario. Representa la parte significativa de número. Para los valores desnormalizados, indica el valor que tendrán y saber si son infinitos o NaN. Según la IEEE754, estos ocupan los últimos 23 bits de los 32 bits de un flotante y 52 bits en el flotante doble de 64 bits.

Conversiones

* **Proceso de conversión de flotante decimal al estándar binario flotante**

Para realizar la transformación del binario flotante al flotante decimal se necesita saber lo siguiente:

* **Valor de BIAS:** Este puede ser calculado con la fórmula BIAS = 2k-1 -1, donde k representa la cantidad de bits que tiene el exponente. Por lo tanto, el mismo BIAS puede ser utilizado para calcular el valor de varios flotantes siempre y cuando sigan el mismo formato de bits en el exponente. En el caso de los de 32 Bits que cuentan con 8 bits en el exponente, se calcularía como BIAS=28-1-1 = 127, mientras que en 64 Bits, que cuenta con 11 bits en el exponente, sería BIAS=211-1-1 = 1023
* **Valor de E:** Este valor es calculado con E = exponente (convertido a decimal) – BIAS. Para el caso de los desnormalizados cuyo exponente es siempre 0, se toma como 1, haciéndolo E=1-BIAS
* **Valor de M:** Para este valor, se tiene que tomar en cuenta si es normalizado o desnormalizado. El valor de este se toma 1 (Para normalizados) o 0 (Para desnormalizados) y se le suma el valor de 2-n según la posición de cada bit en 1 después del punto donde n es la posición iniciando en 1 con el bit más cercano al punto.

Al final, el valor del bit será

Número = (-1) signo \* (M \* 2BIAS)

Ejemplo:

En 32 bits

**1 10000010 11110000000000000000000**

10000010 = 130

E = 130 – BIAS = 130 – 127 = 3

Exponente diferente de 0, entonces M = 1.1111 = 1 + 2-1 + 2-2+ 2-3++ 2-4 = 1 + (15/16) = 31/16

Valor = -11 \* (31/16) \* 23 = -15.5

En 64 bits

**1 10000000010 11110000000000000000000000000000**

10000000010 = 1026

E = 1026 – BIAS = 1026 – 1023 = 3

Exponente diferente de 0, entonces M = 1.1111 = 1 + 2-1 + 2-2+ 2-3++ 2-4 = 1 + (15/16) = 31/16

Valor = -11 \* (31/16) \* 23 = -15.5

Para algunos valores estas reglas no se aplican. Estos valores son considerados especiales por tener un valor fijo sin importar la cantidad de bits que represente el binario siempre y cuando cumplan una condición.

* **Infinito:** Ocurre cuando todos los bits del exponente sean igual a 1 y tenga todos los bits del fraccionario en 0. Este puede ser infinito o negativo según el bit del signo.

**Ejemplo:**

0 11111111 00000000000000000000000

0 11111111111 00000000000000000000000000000000

* **NaN:** Parecido al infinito, necesita que todos los bits de su exponente sean igual a 1, pero a diferencia del infinito, tiene que tener un fraccionario que tenga al menos un bit diferente de 0. Ya que tienen bits diferentes, estos valores no pueden ser comparados entre ellos ya que no se consideran iguales al menos que tenga la misma combinación de bits.

**Ejemplo:**

0 11111111 10000000000000000000000

0 11111111111 10000000000000000000000000000000

* **Cero**: Para el cálculo de este valor, tiene que tener tanto el exponente como el fraccionario en 0. Este valor puede ser calculado por el método, pero al presentar ningún cambio se puede tomar como un valor estático.

**Ejemplo:**

0 00000000 00000000000000000000000

0 00000000000 00000000000000000000000000000000

* **Para el proceso de llevar el binario flotante a un decimal se hace lo siguiente:**

Se tendrá un valor M que iniciará siendo el valor de la parte entera del número y, divido por un punto, la parte después del punto convertida a su literal flotante binario, el cual puede variar según la precisión que se tendrá, pues algunos valores solo podrán verse expresados como una aproximación del mismo. Tras esta transformación, se debe rodar el punto hasta que te solo quede un 1, el valor de lo que se rodó se verá afectado por la precisión ya que si se excede del tamaño, se pierden parte de los dígitos obtenidos.

Además, se tendrá un valor E que tendrá el número de veces que se rodó el punto en M. Este valor se usará para calcular el exponente al sumarle el BIAS. Tras esto, se lleva el exponente a binario para ser usado en la conversión.

Ya con el signo del número original, el exponente calculado y el valor después del punto calculado en M se forma el valor binario que se quiera obtener.

Signo-Exponente-Fraccionarios sacados de M

Ejemplo

1.5

00000010.10000000000000000000000  
M = 1. 01000000000000000000000

Se rodó una sola ves  
E = 1

Exp = 1 + 127 = 128 = 10000000

Signo positivo

**Valor = 0 10000000 01000000000000000000000**

**En 64 bits (Solo varia la precisión y por el BIAS, el exponente)**

**Valor = 0 10000000000 01000000000000000000000000000000**