# PRÁCTICA 2

## OPERACIONES ARITMÉTICAS DE NÚMEROS CON SIGNO NÚMEROS REALES. DETECCIÓN Y CORRECCIÓN DE ERRORES

### OBJETIVOS

Una vez finalizada la práctica debemos ser capaces de:

- Sumar y restar números binarios con signo representados en Complemento a 2.
- Definir el concepto de desbordamiento (Overflow).
- Realizar tareas de codificación de valores reales expresados en decimal a la representación en coma flotante mediante el estándar IEEE-754.
- Realizar tareas de conversión de valores representados en coma flotante mediante el estándar IEEE=754 a su representación decimal.
- Comprender las limitaciones a la representación de valores derivadas del uso de un espacio material de representación en un computador.
- Identificar los diferentes métodos de detección de errores en sistemas que operan con codificación binaria.
- Entender el funcionamiento de los métodos de detección de errores.

### REFERENCIAS

- T.L. Floyd, *Fundamentos de los Sistemas Digitales*, 9ª Edición, Capítulo 2. ”Sistemas de numeración, operaciones y códigos”; secciones 2-5 a 2-7 y 2-12.
- P. de Miguel Anasagasti, *Fundamentos de los computadores*, 9ª Edición, Capítulo 2. “Representación de la Información”; Sección 2.5: Representaciones Numéricas. Resolución. Sección 2.7.2.
- W. Stallings, *Organización y Arquitectura de Computadores*, 5ª Edición, Capítulo 8. “Aritmética del Computador”; Capitulo 8. Secciones 8.4 “Representación en Coma Flotante”. Capítulo 5. Sección 5-2. “Corrección de errores”.
- Transparencias Tema 2 “Representación de la información”. Fundamentos de los Computadores.

### ELEMENTOS NECESARIOS

Se deberán realizar las operaciones sobre papel y, posteriormente, comprobar los resultados obtenidos haciendo uso de asistentes informáticos:

- Calculadora de software
- Hojas de cálculo
- http://babbage.cs.qc.edu/IEEE-754
- http://www.etsimo.uniovi.es/pub/uned/etcI/floatlab.zip

## INTRODUCCIÓN TEÓRICA

Cuando se realizan operaciones de cualquier tipo con números binarios, es necesario fijar previamente el sistema de representación que vamos a utilizar para los números negativos. Según el sistema que utilicemos una misma cadena de unos y ceros equivaldrá a un número u otro. Además, nos aseguraremos siempre de operar con datos que tengan la misma longitud.

Normalmente, trabajando con números enteros, siempre se suele utilizar el complemento a 2 si se ha de operar con ellos. El hecho de poder representar números negativos hace que una operación de resta (A – B) se reduzca a sumar a un número la representación negativa del otro [A + (-B)].

Los pasos necesarios para sumar dos números, tanto si son positivos como si son negativos, en complemento a 2 se reducen a:

- Representar correctamente los números a operar con el número de bits necesarios.
- Realizar su suma según las reglas de la aritmética binaria
- Si se produce un bit de acarreo final, se desecha y no formará parte del resultado.

Se produce *desbordamiento* (u *overflow*) cuando el resultado de la suma excede de la capacidad de representación del número de bits disponible. La consecuencia del desbordamiento es que el bit de signo del resultado es incorrecto. Lógicamente, solo se puede producir desbordamiento cuando sumamos dos números positivos o dos negativos.

La representación de valores reales se realiza utilizando un espacio de representación formado por varios campos. Cada uno de los campos se representa mediante códigos binarios de una longitud determinada en número de bits. Existen dos sistemas de representación:

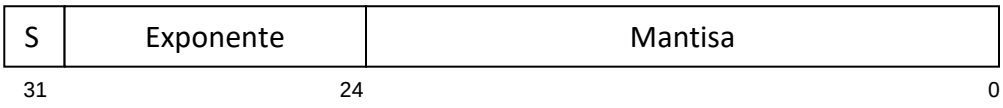
- En *coma fija*: Utiliza los dos campos: signo y mantisa. La mantisa se divide en dos partes para la representación binaria de la parte entera y fraccionaria del número.
- En *coma flotante*: Utiliza tres campos: signo, mantisa y exponente. El valor N del número representado se calcula como:

$$N = (-1)^S \; M \cdot B^E$$

donde *S* representa el signo, *M* el valor de la mantisa, *E* el valor del exponente y *B* la base empleada.

En el estándar para coma flotante IEEE754, en simple precisión, se emplean 32 bits para representar cualquier número según la siguiente distribución:

- 1 bit para el signo
- 8 bits para el exponente
- 23 bits para la mantisa




En el formato IEEE754, no todos los patrones de bits se interpretan de la manera habitual. Algunas combinaciones se emplean para representar valores especiales. Los valores extremos del exponente consistentes en todo ceros y todo unos definen valores especiales. Por tanto, si nos centramos en la representación en simple precisión, se nos pueden presentar las siguientes clases de números:

- Para valores del exponente desde 1 hasta 254, se representan números en coma flotante normalizados distintos de cero. El exponente está sesgado, siendo el rango de exponentes desde (-126) a (+127). Un número normalizado debe contener un bit 1 a la izquierda de la coma binaria; este bit está implícito, danto una mantisa efectiva de 24 bits (parte fraccionaria).
- Un exponente todo unos junto con una mantisa (parte fraccionaria) cero representa, dependiendo del bit de signo, el infinito positivo o negativo. Siempre resulta útil tener una representación del infinito, ya que nos dejará libertad para tratar el desbordamiento como un error o no.
- Un exponente cero junto con una mantisa distinta de cero representa un número desnormalizado. En este caso, el bit a la izquierda de la coma binaria es cero y el exponente original es -126. El número será positivo o negativo dependiendo del bit de signo.
- A un exponente todo unos junto con una mantisa distinta de cero se le da el nombre de NaN (*Not a Number*, "No representa un número"), y se emplea para señalar condiciones de excepción.

El método más simple para detectar errores, y que muchos sistemas emplean, es el bit de paridad. Cualquier grupo de bits contiene un número par o impar de unos. Un bit de paridad se añade al grupo de bits, al principio o al final, dependiendo del sistema, para hacer que el número total de unos en el grupo, incluido el bit de paridad, sea siempre par en el caso del bit de paridad par o impar en el caso del bit de paridad impar.

Los códigos correctores permiten detectar errores múltiples e incluso corregir el error. La idea básica es añadir p bits de paridad, cada uno de los cuales afecta a una parte del código. El código de Hamming, para corregir un error en un dato de longitud n necesita añadir p bits de paridad de forma que se cumpla que:

$$2^p \geq n + p + 1$$

La utilización de códigos de Hamming para detectar y corregir errores en una memoria principal es frecuente.

Los códigos de redundancia cíclica, también conocidos como códigos polinomiales constituyen el método de detección de errores más empleado en comunicaciones. Se utiliza con esquemas de transmisión orientados a tramas (o bloques). Permiten sustanciales mejoras en fiabilidad respecto a los métodos anteriores, siendo a la vez una técnica de fácil implementación. Imponiendo condiciones bastante simples sobre los polinomios divisores es posible detectar un gran número de errores.

## REALIZACIÓN PRÁCTICA

1. Sean los números decimales siguientes:

- A = + 159
- B = + 224
- C = - 99
- D = - 187

Obtén el resultado de las siguientes operaciones en Complemento a 2 empleando en todos los casos diez bits. Indica siempre si el resultado es correcto o no.

a) A + B

b) C – A

c) B + D

d) D – A

3. Se dispone de un sistema de representación para números reales en coma fija compuesto por 16 bits según la siguiente distribución:

- 1 bit de signo
- 10 bits para la parte entera
- 5 bits para la parte fraccionaria

4. Indica el rango de representación de dicho formato.

5. Identifica los números:

i. 0 1011110000 00000

ii. 1 1010100001 10000

iii. 1 0101101100 00110

6. Obtén la representación que se tendría para los siguientes números decimales y explica las situaciones particulares que ser produzcan:

i. 118.75

ii. -71.25

iii. 0.15

iv. -11.1

4. Realiza las siguientes conversiones al formato de representación IEEE754 en simple precisión.

a) 112.415†

b) $-2.76*10^2$

c) $10^{20}$

d) $2^{-42}$

5. Obtén razonadamente el equivalente decimal de los siguientes números expresados en el formato IEEE 754 en simple precisión.

a) 1 01110001 10111001000000000000000
b) 0 11010010 10001110110000000000000
c) 0 00000000 00000000000000010111001
d) 0 11111111 00110000000000000000000
e) 1 11111111 00000000000000000000000

6. Se desea transmitir una información que está compuesta por la codificación en binario con 4 bits de cada uno de los dígitos decimales.

a) Obtén el código de Hamming que nos posibilite la detección y corrección de un error.
b) Si al receptor llega la cadena 1010110, comprueba si es correcta o no y en caso que no lo sea indica cómo se puede obtener el bit erróneo y corrígelo.

7. Para la transmisión de un bloque de información se desea emplear un código polinomial para prevenir posibles errores. El polinomio empleado es $x^5 + x^4 + 1$.

a) Indica qué se transmitirá (y que permitirá detectar errores en el receptor) si la información original fuera $051F_H$.
b) Si se ha recibido la cadena 11000001110101 ¿ha habido error en la transmisión?

†Obtén únicamente los 5 primeros dígitos significativos de la parte fraccionaria