UNIVERSITAT POLITÈCNICA DE VALÈNCIA

# Representación de números enteros: el convenio “signo y magnitud”


| | |
|---|---|
| **Apellidos, nombre** | Martí Campoy, Antonio (amarti@disca.upv.es) |
| **Departamento** | Informàtica de Sistemes i Computadors |
| **Centro** | Escola Tècnica Superior d’Enginyeria Informàtica |

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

# 1 Resumen de las ideas clave

En este artículo se trata la problemática de la representación de los números enteros en los computadores. Así mismo, se presentará una posible solución a este problema, que recibe el nombre de representación en signo y magnitud. Los conocimientos previos que necesitas para abordar este artículo se presentan en la tabla 1.

**Tabla 1. Conocimientos previos**

| Conocimientos previos |
|---|
| 1. Sistemas de numeración posicionales |
| 2. Sistema de numeración binario |
| 3. Cambios de base, especialmente binario |
| 4. Aritmética básica en base 2 |

# 2 Objetivos

Una vez acabes de leer este artículo docente y reproduzcas los ejemplos presentados, deberás ser capaz de **representar** números enteros en binario **aplicando** el convenio llamado signo y magnitud. Además podrás **calcular** el rango de representación para un tamaño de bits determinado. También serás capaz de **realizar** operaciones aritméticas de suma y resta de números enteros en binario y de extensión de signo utilizando la representación en signo y magnitud. Por último, podrás **razonar** sobre las ventajas y desventajas de este convenio convenio de representación de números enteros.

# 3 Introducción

En la vida cotidiana los números enteros se representan mediante los 10 símbolos (del 0 al 9) de la base decimal, junto con los símbolos "+" y "-" para identificar a los números positivos y negativos, respectivamente.

A la hora de representar números enteros en un computador (para almacenarlos, operarlos o comunicarlos) el problema que surge es que en los circuitos digitales sólo se pueden utilizar dos valores, normalmente representados por los símbolos 0 y 1. No cabe la posibilidad de representar un tercer y cuarto símbolo para distinguir un número positivo de otro negativo.

Así surge la necesidad de crear y definir convenios para codificar el signo de un número entero utilizando únicamente los símbolos 0 y 1 disponibles en los circuitos digitales.

Antes de explicar el convenio signo y magnitud, objeto de este artículo, recordarte que los números se almacenan en circuitos digitales llamados registros, y que su

longitud es fija. Es decir, cuando hablemos de un número entero representado en binario y en signo y magnitud deberemos indicar el número total de bits utilizados.

# 4 El convenio signo y magnitud

Su nombre se debe a que se representa por un lado el signo del número y por otro su magnitud.

## 4.1 Definición

Según este método, si se utilizan n bits para representar un número, se reserva un bit (normalmente el de mayor peso) para indicar el signo, y el resto de bits se utilizan para representar la magnitud. El convenio, un acuerdo arbitrario, dice que se utiliza la siguiente codificación para un número entero:

- Si el número es **positivo** su bit de mayor peso será 0:

| 0 | *Magnitud* (n-1 bits) |
|---|---|

- Si el número es **negativo** su bit de mayor peso será 1:

| 1 | *Magnitud* (n-1 bits) |
|---|---|

El bit de mayor peso que indica el signo del número recibe el nombre de **bit de signo**.

Ejemplo: utilizando 8 bits (n = 8), representa los números +25 y –25, siguiendo el convenio de signo y magnitud.

En primer lugar se convierte la magnitud o valor absoluto, 25, a binario natural con n-1=7 bits, completando con ceros los bits de mayor peso si fuera necesario:

$$25_{10} = 0011001_2$$

En segundo lugar se añade el bit de signo:

$$+25_{10} = \mathbf{0}0011001_2$$

$$-25_{10} = \mathbf{1}0011001_2$$

Ejemplo: obtén el valor decimal correspondiente a $011010_2$ y $100010_2$ sabiendo que están representados en signo y magnitud utilizando 6 bits (n = 6).

$011010_2 \longrightarrow$ dado que su bit de mayor peso (bit de signo) es 0, sabemos que se trata de un número positivo con magnitud $11010_2 = 26_{10}$ por lo que $011010_2 = +26_{10}$

$100010_2 \longrightarrow$ dado que su bit de mayor peso (bit de signo) es 1, sabemos que se trata de un número negativo con magnitud $00010_2 = 2_{10}$ por lo que $100010_2 = -2_{10}$

## 4.2 Rango

El rango de un sistema o convenio de representación es el conjunto de valores diferentes que pueden representarse. Para signo y magnitud con n bits es fácil ver que el rango es:

| | | | | |
|---|---|---|---|---|
| Rango en binario: | [111···111, | 100···000, | 000···000, | 011···111] |
| Rango en decimal: | $[-(2^{n-1}-1)$, | -0, | +0, | $+(2^{n-1}-1)]$ |

El rango es simétrico, es decir, incluye la misma cantidad de valores positivos que negativos, y además incluye dos representaciones para el cero, una positiva y otra negativa.

## 4.3 Suma y resta

Las operaciones de suma y resta de números representados en signo y magnitud se realizan de igual forma a como las hacemos en base diez. En este proceso es necesario comparar por un lado los signos de los operandos, y por otro lado sus magnitudes. Después de esta comparación, se decide si se realiza una suma o una resta de las magnitudes, y si se intercambian los operandos o no. Finalmente, también se decide el signo del resultado. Todo este proceso lo realizamos los humanos de forma inconsciente ya que hemos sido entrenados desde niños, pero es complicado de implementar en un circuito digital. A continuación se muestra como ejemplo una suma de números representados en binario signo y magnitud con 4 bits (y sus equivalentes en decimal), que se convierte en una resta con los operandos intercambiados:

| | | | |
|---|---|---|---|
| Binario SyM con 4 bits | **0**100<br>+ **1**111 | se convierte en | 111<br>- 100<br>**1**011 |
| Decimal | +4<br>+ -7 | se convierte en | 7<br>- 4<br>-3 |

## 4.4 Desbordamiento en la suma y resta

Al realizar una suma o resta de números enteros es posible que el resultado exceda el rango de representación. En este caso se dice que no hay resultado o que el resultado no es representable.

Con operandos representados en signo y magnitud se produce desbordamiento si al operar las magnitudes, tanto en una suma como una resta, el último bit de acarreo toma valor 1.

## 4.5 Extensión de signo

En algunos casos es necesario operar datos con diferentes tamaños. Para aumentar el número de bits con que se representa un dato se realiza la operación llamada extensión de signo.

En el caso de la representación en signo y magnitud, la extensión de signo se realiza insertando ceros entre el bit de signo y la magnitud.

Ejemplo: dados los números enteros $0010_2$ y $1110_2$ representados en signo y magnitud con 4 bits, extiende el signo para representarlos con 8 bits.

$$\mathbf{0}010_2 \quad = \quad \mathbf{0}\underline{0000}010_2$$
$$\mathbf{1}110_2 \quad = \quad \mathbf{1}\underline{0000}110_2$$

# 5 Ejercicios

A continuación tienes unos pocos ejercicios. Es muy conveniente que cojas lápiz y papel y los resuelvas. Recuerda que estas aprendiendo, por lo que puedes, y aún diría más, debes consultar las secciones anteriores de este documento para resolver los ejercicios. También tienes las soluciones de los ejercicios, pero te pido encarecidamente que no las mires hasta que no hayas intentado resolver todos los ejercicios.

## 5.1 Enunciados

1. Representa el número $-67_{10}$ en binario signo y magnitud con 8 bits.
2. Representa el número $+68_{10}$ en binario signo y magnitud con 8 bits.
3. Indica la representación decimal de $10010111_2$ sabiendo que está representado en signo y magnitud de 8 bits.
4. Indica la representación decimal de $00110101_2$ sabiendo que está representado en signo y magnitud de 8 bits.
5. ¿Cuál es el rango de representación de signo y magnitud con 10 bits? Expresa el rango en decimal.
6. Realiza la extensión de signo a 16 bits de $11110111_2$ sabiendo que está representado en signo y magnitud de 8 bits.
7. Realiza la extensión de signo a 16 bits de $01110100_2$ sabiendo que está representado en signo y magnitud de 8 bits.

## 5.2 Soluciones

1. Representa el número $-67_{10}$ en binario signo y magnitud con 8 bits. Sol: $11000011_2$
2. Representa el número $+68_{10}$ en binario signo y magnitud con 8 bits. Sol: $01000100_2$
3. Indica la representación decimal de $10010111_2$ sabiendo que está representado en signo y magnitud de 8 bits. Sol: $-23_{10}$
4. Indica la representación decimal de $00110101_2$ sabiendo que está representado en signo y magnitud de 8 bits. Sol: $+53_{10}$
5. ¿Cuál es el rango de representación de signo y magnitud con 10 bits? Expresa el rango en decimal. Sol: $[-(2^9-1), +(2^9-1)] = [-511, +511]$
6. Realiza la extensión de signo a 16 bits de $11110111_2$ sabiendo que está representado en signo y magnitud de 8 bits. Sol: $1000000011110111$
7. Realiza la extensión de signo a 16 bits de $01110100_2$ sabiendo que está representado en signo y magnitud de 8 bits. Sol: $0000000001110100_2$

# 6 Conclusiones

Los circuitos digitales sólo pueden almacenar dos símbolos, por lo que es necesario establecer un acuerdo o convenio para utilizar estos dos símbolos, el 0 y el 1, para representar el signo de un número entero. El convenio llamado Signo y Magnitud es

sencillo y es el utilizado por los seres humanos. Sin embargo, la complejidad de la aritmética hace que no sea utilizado de forma general para representar números enteros, y su uso queda limitado a algunos pocos casos, como la representación de la mantisa en el estándar IEEE754 de representación de números reales.

# 7 Bibliografía

## 7.1 Libros:

[1] Pedro de Miguel Anasagasti. "*Fundamentos de los computadores*", 9ª ed. Madrid, Thomson-Paraninfo. 2004, 2007

[2] John F. Wakerly. "*Diseño digital : principios y prácticas*". Madrid. Pearson Educación. 2001

## 7.2 Recursos electrónicos:

[3] Martí Campoy, Antonio. "*Representación de enteros: Signo y Magnitud*", Universitat Politècnica de València, 2009. http://riunet.upv.es/handle/10251/5235