Enteros Negativos en SM, CA1 y CA2

Sitio: Agencia de Habilidades para el Futuro
Curso: Lógica Computacional 1°D
Libro: Enteros Negativos en SM, CA1 y CA2

Imprimido por: RODRIGO PINTO
Día: martes, 26 de noviembre de 2024, 18:29

Tabla de contenidos

1. Enteros negativos - Introducción

En la clase anterior vimos como utilizar binario para representar números naturales.

Esta semana veremos como trabajar con números enteros negativos.

En decimal, solemos utilizar el signo "-"para indicar que un número es negativo, pero este no es representable en las computadoras como tal, donde solo se cuenta con 0s y 1s. Veremos entonces, como salvar esto de distintas maneras, cada una con sus particularidades.

2. Signo Magnitud

La idea detrás de este sistema es cubrir la incapacidad de escribir el signo '-' en el contexto de una computadora, indicando de alguna forma si esta presente o no. Dicho en otras palabras, **indicar mediante un bit la polaridad del valor.**

Por convención se suele usar el primer bit de una cadena (aquel del extremo izquierdo) como indicador y se lo denomina **bit de signo**. Si el bit de signo es un 1 se trata de un número negativo, y en caso contrario es positivo. Los bits restantes de la cadena reciben el nombre de magnitud y su valor se determina con el mecanismo de interpretación del sistema binario sin signo (BSS).

Por lo anterior, este sistema recibe el nombre Signo-Magnitud (SM). Cuando se restringe la cantidad de bits a n, se lo denota SM(n), donde el primer bit es el signo, y la magnitud es de n - 1 bits.

2.1. Interpretación de cadenas en SM (n)

Para interpretar una cadena en Signo Magnitud se utiliza la interpretación del sistema Binario Sin Signo sobre los bits de la magnitud.
Por ejemplo, si se considera la cadena 1010, el primer paso para interpretarla es separar el bit de signo, en este caso 1, de la magnitud, en este caso: 010.
El bit de signo se interpreta según lo indicado mas arriba, en la sección anterior: El bit **1** indica un **valor negativo**, y **0** un **valor positivo**. En el ejemplo, la cadena comienza con 1, de modo que se trata de un negativo.

La magnitud, se interpreta como BSS: $IBSS(010) = 0*2^0 +1*2^1 +0*2^2 = 2$

De esta manera, la interpretación en SM se puede expresar en términos de la función de interpretación de BSS:

$$I sm(4)(1010) = -1 \times IBSS(010) =$$

$$= -1 \times (0 * 2^0 + 1 * 2^1 + 0 * 2^2) = -1 \times 2 = -2$$

2.2. Representación de cadenas en SM(n)

Del mismo modo, la representación en signo magnitud se apoya sobre el mecanismo de representación del sistema Binario Sin Signo, pero este último permite representar sólo números positivos, por lo que es necesario tomar el valor absoluto del valor dado para representar.

Suponiendo como ejemplo que se se necesita representar el valor -5 en SM(4). Este sistema destina 1 bit para el signo y 3 bits para la magnitud.

Lo primero a resolver es definir el signo, y como en este caso el número es negativo el valor del bit de signo que va a tener la cadena resultante es 1. Luego se toma el valor absoluto del número y se lo representa como en BSS de tres bits, obteniendo como resultado 101

La cadena final en SM(4) se obtiene componiendo ambas partes, es decir:

$$R_{SM(4)}(\text{-}5) = \mathbf{1}R_{bss(3)}(|5|) = \mathbf{1101}$$

2.3. Rango del sistema SM(n)

Como se definió en apartados anteriores, el rango es el intervalo de números representables en un sistema con una cierta cantidad de bits (n bits).
El número mínimo que es posible representar en un sistema de signo magnitud, va a ser negativo, es decir va a comenzar con 1. Su magnitud va a ser la mas grande posible. De esta manera, para n bits, el mínimo va a ser:

$$\underbrace{1}_{signo}\ \underbrace{1...,1}_{n-1 magnitud}$$

$$-(2^0 + ... + 2^{n-2})$$

$$-(2^{n-1} - 1)$$

El numero máximo se obtendrá utilizando signo positivo, es decir 0; y nuevamente la mayor magnitud.

$$\underbrace{0}_{signo}\ \underbrace{1...,1}_{n-1 magnitud}$$

$$2^0 + ... + 2^{n-2}$$

$$2^{n-1} - 1$$

Por lo tanto, que en este caso, el rango es $[-(2^{n-1} - 1); 2^{n-1} - 1]$.

Si, por ejemplo, se restringe el sistema a 8 bits, el rango para el sistema SM(8) tendría, como mínimo la cadena 11111111, y como máximo la cadena 01111111

Es interesante notar que en dicho intervalo no hay 2n números distintos. Por ejemplo, si n = 3, el rango del sistema SM es:

$[-(2^{3-1} - 1), 2^{3-1}] = [-3, 3]$

y en dicho intervalo hay 7 números: {-3; -2; -1; 0; 1; 2; 3}. En binario sin signo, con 3 bits se tenían 8 números representables diferentes. ¿Dónde está el número que falta? El número que falta dentro del rango se genera a causa de que el sistema posee una doble representación del 0 , ya que tanto la cadena 000 como doble representación la cadena 100 representan dicho valor del 0.

Esto trae consigo dos desventajas: la primera es el hecho de desaprovechar una cadena, y la segunda es que esta doble representación complica la aritmética (y los circuitos que la implementan) al tener que considerar dos cadenas que
representan el mismo valor.

Una característica que posee el rango de este sistema (y otros) es que, a diferencia del BSS(), el rango en SM() es un rango equilibrado. Esto significa que, partiendo desde el 0, se tienen n cantidad de números positivos y negativos.
Tomando el ejemplo anterior de un sistema SM(3), al ver los números posibles a representar se ve que se pueden representar 3 números positivos [1, 2, 3] y 3 números negativos [-3, -2, -1].

2.4. Aritmética - Suma

La suma en SM considera diferentes casos en función de los signos de las cadenas a sumar. Si las cadenas a sumar tienen el mismo signo (ambas negativas o ambas positivas), la suma se realizará sumando las magnitudes como vimos para BSS y tomando como signo el signo del resultado.

Considerar como ejemplo la suma 1101+1001. Dado que el signo es el mismo en ambos operandos (1) y la magnitud se obtiene al sumar en BSS es 101 + 001 = 110. Por lo tanto la cadena resultado de la suma en SM es 1110.

Si las cadenas a sumar tienen diferente signo, se debe primero identificar qué cadena tiene la mayor magnitud (sea A la cadena de mayor magnitud y B la de menor magnitud). El signo del resultado va a ser el signo que tenga A, y la
magnitud resultado se obtiene restando la magnitud de B a la magnitud de A.

Considerar el siguiente ejemplo: 1101 + 0001. La mayor magnitud es la de la cadena 1101, ya que su magnitud es 101 (5) y es mayor que la magnitud 001 (1). Es por esta razón que ya podemos afirmar que el signo del resultado va a ser 1.

Tenemos ahora que restar las magnitudes (a la mayor magnitud, la menor).

Esta resta auxiliar la tenemos que hacer en BSS y nos queda: 101 - 001 = 100.

Entonces el resultado final es 1100

2.5. Artimética - Resta

El algoritmo para calcular la resta entre las cadenas C1 y C2, puede pensarse como una suma, aplicando la siguiente equivalencia:

$$C1 - C2 = C1 + (-C2)$$

En el sistema SM, es posible construir el inverso de la cadena C2 (-C2) a partir de invertir el bit de signo sobre C2.
Luego de ese paso, la operación se traduce en una suma y se deben analizar los casos que se describieron en el apartado anterior.
Si se considera, por ejemplo, la resta 1101 - 1001, entonces como primer paso se la traduce en la suma 1101 + 0001 y dicha suma se resuelve considerando los signos adecuadamente, según el algoritmo de la sección anterior.

3. Complemento a 1 (Ca1)

El complemento a 1 es un concepto fundamental en la teoría de números y la lógica binaria. En un sistema numérico binario, cada número puede ser representado como una secuencia de bits, donde cada bit puede tener un valor de 0 o 1.

El complemento a 1 de un número binario se refiere a la secuencia de bits que se obtiene al invertir todos los bits de ese número. Por ejemplo, el complemento a 1 de la secuencia binaria 0101 es 1010.

El complemento a 1 es útil en la computación y la electrónica, donde se utiliza para representar números negativos en un sistema de representación de números con signo. En este sistema, el complemento a 1 se utiliza para representar el valor negativo de un número positivo, permitiendo la representación de todos los valores numéricos en un rango de números negativos y positivos.

Además, el complemento a 1 también se utiliza en la realización de operaciones aritméticas, como la suma y la resta, en el procesamiento de datos y la transmisión de información.

En resumen, el complemento a 1 es un concepto fundamental en la teoría de números y la lógica binaria, con aplicaciones importantes en la computación, la electrónica y la transmisión de información.

3.1. Ventajas y desventajas

Ventajas del complemento a 1:

- **Representación de números negativos:** El complemento a 1 permite la representación de números negativos en un sistema de números binarios con signo, lo que facilita la realización de operaciones aritméticas con números negativos.
- **Eficiencia en operaciones aritméticas:** El complemento a 1 permite realizar operaciones aritméticas, como la suma y la resta, de manera más eficiente y rápida en comparación con otros sistemas de representación de números negativos.
- **Facilidad de implementación en hardware:** El complemento a 1 es fácil de implementar en hardware, lo que lo hace adecuado para su uso en sistemas electrónicos y de computación.

Desventajas del complemento a 1:

- **Dificultad en la comprensión:** El complemento a 1 puede ser difícil de entender para aquellos que no están familiarizados con la teoría de números y la lógica binaria.
- **Representación de números decimales:** El complemento a 1 no es adecuado para la representación de números decimales, lo que limita su uso en aplicaciones que requieren una representación precisa de números decimales.
- **Dificultad en la realización de operaciones de comparación:** Las operaciones de comparación, como la igualdad y la mayoría, pueden ser más difíciles de realizar con números representados con complemento a 1.

3.2. ¿Cómo calcular el complemento a 1?

Invertir los bits: Invierta todos los bits de la secuencia binaria original. Por ejemplo, si la secuencia binaria original es 0101, invierta cada bit para obtener 1010.

Añadir 1 al resultado: Añada 1 al resultado obtenido en el paso anterior. Por ejemplo, si el resultado del paso anterior es 1010, añada 1 para obtener 1011.

Eso es todo. El resultado final es el complemento a 1 de la secuencia binaria original. Por ejemplo, el complemento a 1 de la secuencia binaria 0101 es 1011.

Es importante tener en cuenta que el complemento a 1 se utiliza en sistemas numéricos con un número fijo de bits. Por lo tanto, si la secuencia binaria original tiene un número de bits diferente al utilizado en el sistema numérico, deberás realizar los ajustes necesarios antes de calcular el complemento a 1.

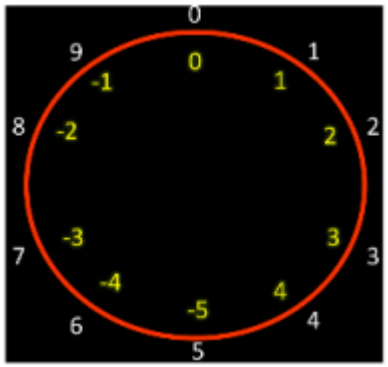
4. Complemento a 2 (Ca2)

La idea de este sistema es lograr tener números negativos pero manteniendo la aritmética del sistema BSS, que se implementa con los circuitos aritméticos presentados anteriormente.

Esto es muy deseable porque permite que una maquina no necesite dentro de su ALU (unidad aritmético lógica, que se encuentra dentro del microprocesador) diferentes circuitos para sumar números naturales y enteros; y esto se traduce en abaratar costos.

Suponer que se tiene un solo dígito en el sistema decimal (denotado dec(1)) y se tiene que poder representar negativos (sin usar el signo "-"). Para resolverlo, se ubica las cadenas del sistema dec(1) en el exterior de una rueda como se indica en la figura que se preseta abajo (con color blanco), y se le asignan valores en la parte interna de la rueda (en color amarillo) de manera que cada cadena (en este caso, dígito) quede aparejada con el opuesto a su complemento a la base. En este escenario, el complemento a la base es el valor 10. Así, el 9 queda asociado al -1 pues:

$$10 + (-1) = 9$$



Este aparejamiento tiene la particularidad que preserva las propiedades de las operaciones aritméticas, pues la suma entre cadenas sigue el mismo mecanismo que en decimal, aunque las cadenas estén representando otros números y esa es una gran ventaja.
Por ejemplo en el sistema decimal restringido a un dígito, 8+2 = 0 (con acarreo de 1). Entonces, si se tiene en cuenta la interpretación de cada

cadena:

$$I(8) + I(2) = -2 + 2 = 0$$

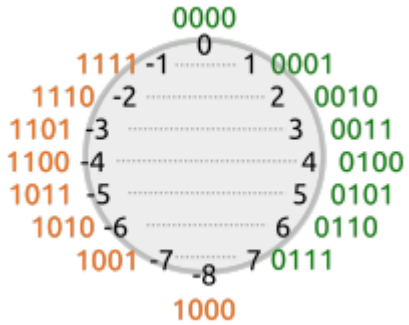
Considerar otro ejemplo:

$$7+4=1 \text{ (con acarreo 1)}$$

entonces

$$I(7) + I(4) = -3 + 4 = 1$$

Esta idea se puede llevar al sistema binario, para lo cual se ubican las cadenas en un círculo ordenadas lexicográficamente, y luego asignan números positivos en sentido horario y negativos en sentido anti-horario, como se muestra en la
figura ubicada debajo de este texto. De esta manera se consigue que algunas cadenas (las mas pequeñas en orden lexicográfico) queden asociadas a valores positivos y otras a valores negativos (las mayores en orden lexicográfico). Este sistema es el Complemento a 2, y se denota CA2(n).



En este sistema se utilizarán las cadenas que empiezan en 0 para los números positivos y las cadenas que comiencen en 1 para los negativos. Es importante notar que no se utilizará ese bit para signo como es el caso del sistema signomagnitud, pero si ocurre que el bit da información a la hora de interpretar.

Adicionalmente, se necesita definir la operación complemento() sobre una cadena. El complemento de una cadena *w* es otra cadena *w'* que se obtiene a partir de invertir los bits de *w* y sumarle 1 al resultado. Por ejemplo, el complemento a 2 de la cadena 0001 es 1111, y se denota:

$$complemento(0001) = 1111$$

ya que en primer lugar se invierten los bits obteniendo 1110 y luego se suma 1.

La operación complemento cumple con la siguiente propiedad:

$$w + w' = 0_{n-1}...0_0$$

es decir, que al sumar una cadena y su complemento se obtiene como resultado la cadena de n ceros. Por ejemplo, con 4 bits: 0001 + 1111 = 0000. Esto es interesante porque justifica la asociación entre cadenas y valores que se muestra en el gráfico ubicado más arriba.

4.1. Representación

A la hora de representar se tiene en cuenta si el n´umero es positivo (o cero), o si es negativo. Los números positivos se representan en binario sin signo y los números negativos se representan con el complemento a dos de la cadena que representa su valor absoluto.

Por ejemplo, en CA2(4), la representación del 3 es 0011, es decir, igual que en binario sin signo. Para representar -3 en CA2(4), primero se debe representar el valor 3 en binario sin signo (es decir 0011) y a continuación se toma el complemento a
2 de dicha cadena 0011 = 1100 + 1 = 1101. Entonces, la representación en el sistema Complemento a 2 de un número negativo se puede formalizar como:

$$complemento(R(|x|))$$

4.2. Interpretación

Al momento de interpretar se recorre el camino inverso a la representación, por lo tanto también se debe determinar si la cadena comienza con 0 (*bn*-1 = 0) o con 1 (*bn*-1 = 1). Si *bn*-1 = 0, entonces se trata de un valor positivo, y en ese caso simplemente se interpreta como en un sistema *binario sin signo (n)*. En caso contrario, si *bn*-1 = 1, se sabe que representa un valor negativo, en cuyo caso se aplica la operación *complemento()* a la cadena, luego se interpretar el resultado en *BSS(n)* y finalmente se le agrega el signo negativo.

Por ejemplo, la cadena 1111 comienza con 1, por lo que representa un número negativo.

1. Calculamos su complemento a dos: *complemento*(1111) = 0000+1 = 0001.
2. Interpretamos la cadena resultante 0001 en BSS, lo cual nos da 1
3. Ahora, como sabemos que se trataba de un negativo, agregamos el signo, siendo el resultado final -1.

Entonces, la interpretación en Complemento a 2 de un número negativo se puede formalizar como:

-IBSS(complemento(x))

4.3. Rango

Para calcular el rango, vamos a obtener las cadenas que nos dan el máximo y el mínimo. Para el primer caso, sabemos que necesitamos una cadena positiva y en particular a la que nos de el número mas grande. Dado que las cadenas positivas se comportan como binario sin signo, si trabajamos con complemento a 2 de n bits, la cadena que nos da al m´aximo es 011...11 (un 0 seguido de todos unos). Dicha cadena, si la interpretamos nos da:

$$2^0 + ... + 2^{n-2}$$

$$2^{n-1} - 1$$

Para el caso negativo, como al momento de interpretarlo vamos a aplicar la operación de complementar, lo que queremos es buscar al número cuyo complemento sea lo mas grande posible. De todas las cadenas negativas, esto lo vamos a obtener con 100...00 (un 1 seguido de todos ceros). Si complementamos

100...00

011...11 + 1

100...00

Que luego al interpretar (ahora usando binario sin signo, porque ya complementamos) nos da:

$$2^{n-1}$$

Recordamos que era negativa, por lo que el resultado es:

$$-2^{n-1}$$

Finalmente, el rango es:

$$[-2^{n-1}; 2^{n-1} - 1]$$

Notar que en este caso, la cantidad de números representables si es de 2n, ya que no hay doble representación de ningún número.

4.4. Artimética

La aritmética en CA2, por definición de complemento a la base, cumple con la propiedad de ser mecánicamente idéntica a la aritmética del sistema BSS. Es decir que tanto la suma como la resta se resuelven con los mismos circuitos de
suma (Full adder y restador). Suponer la siguiente operación de suma:

$$\begin{array}{r} 001 \\ +100 \\ \hline 101 \end{array}$$

Para comprobar si es válido en el contexto de un sistema Complemento a 2, se debe cumplir la siguiente propiedad:

$$I_{ca2}(001) + I_{ca2}(100) = I_{ca2}(101)$$

Por un lado se tiene que los operandos valen:

$$I_{ca2}(001) = I_{bss}(001) = 1$$

$$I_{ca2}(100) = -1 \times I_{bss}(011 + 1) = -1 \times I_{bss}(100) = -1 \times 4 = -4$$

También se sabe que el resultado vale:

$$I_{ca2}(101) = -1 \times I_{bss}(010 + 1) = -1 \times I_{bss}(011) = -1 \times 3 = -3$$

Por lo tanto:

$$\begin{array}{r} I_{ca2}(001) = 1 \\ +I_{ca2}(100) = -4 \\ \hline I_{ca2}(101) = -3 \end{array}$$

4.5. Recapitulando Ca1 y Ca2

El **complemento de 1** de un número binario es otro número binario pero cambiando los ceros por unos y los unos por ceros, es decir, transformar el bit 0 a 1 y el 1 bit a 0.

Ejemplos:

Supongamos los siguientes números de 4 bits:

- Complemento a 1 de 7 (0111) es 8 (1000)
- Complemento a 1 de 12 (1100) es 3 (0011)

El **complemento a 2** se calcula sumando una unidad al valor decimal que resulta al calcular el **complemento a 1**.

Ejemplos:

- El complemento a 2 de 7 (0111) es 9 (1001) (8+1)
- El complemento a 2 de 12 (1100) es 4 (0100) (3+1)

Fíjate que para calcular el **complemento a 2** de un número, lo primero que debes realizar es calcular el **complemento a 1** y luego 1.

La principal diferencia entre el **complemento a 1** y el **complemento a 2** es que el **complemento a 1** tiene dos representaciones para el cero:

- +0 = 00000000 (cero positivo)
- -0 = 11111111 (cero negativo)

Mientras que en el **complemento a 2**, sólo hay una representación para el valor cero 00000000 (+0) porque si añadimos 1 a 11111111 (-1), obtenemos 00000000 (+0) que es lo mismo que cero positivo. Esta es la principal razón por lo que se utiliza **el complemento a 2** en vez del **complemento a 1**.

Muchos de los primeros ordenadores, incluyendo el CDC 6600, el LINC, el PDP-1 y el UNIVAC 1107, usaron la notación de complemento a uno. Los sucesores del CDC 6600 continuaron usando el complemento a uno hasta finales de la década de los 80, y los descendientes de UNIVAC 1107 (la serie UNIVAC 1100/2200) todavía lo hacen, pero la mayoría de los ordenadores modernos usan el complemento a dos.

El **complemento a dos** se utiliza principalmente en las operaciones matemáticas con números binarios. En particular, la resta de números binarios se facilita enormemente utilizando el **complemento a dos**: la resta de dos números binarios puede obtenerse sumando al minuendo el complemento a dos del sustraendo. Se utiliza porque la unidad aritmético-lógica no resta números binarios, suma binarios negativos, por eso esta conversión al negativo.