8. Complemento a la base

El complemento a la base es lo que le falta a un número X_b, este complemento se encuentra restando cada cifra del número, comenzando por la izquierda, de la base menos uno, hasta llegar a la última cifra distinta de cero la que se resta de la base y se completa con ceros a la derecha. Por ejemplo, para encontrar el complemento a 10 de 3700 se tiene:

$$C_{10}\ (3700) = 6300$$

$$\begin{array}{ccccc} & 3 & 7 & 0 & 0 \quad + \\ & 6 & 3 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \end{array}$$

Ejemplo de encontrar el complemento a la base $b = 2$, para 1010_2:

$$\begin{array}{ccccc} & 1 & 0 & 1 & 0_2 \quad + \\ & 0 & 1 & 1 & 0_2 \\ \hline 1 & 0 & 0 & 0 & 0_2 \end{array}$$

Así mismo el complemento a la base menos uno es igual a C_{b-1}, este se obtiene restando cada cifra del número de la base menos uno. Por ejemplo, para encontrar el complemento a 9 de 3705 se tiene:

$$C_9\ (3705) = 6294$$

$$\begin{array}{cccc} 3 & 7 & 0 & 5 \quad + \\ 6 & 2 & 9 & 4 \\ \hline 9 & 9 & 9 & 9 \end{array}$$

Ejemplo de encontrar el complemento a la base menos uno de 2, o sea el complemento a 1 de 1010_2:

$$\begin{array}{cccc} 1 & 0 & 1 & 0_2 \quad + \\ 0 & 1 & 0 & 1_2 \\ \hline 1 & 1 & 1 & 1_2 \end{array}$$

Vale indicar que dado que en el sistema de numeración binario existen sólo los dígitos 0 y 1, el complemento de 0 es 1 y el complemento de 1 es 0. Así, el complemento de un número binario se obtiene complementando cada uno de los bits, sin considerar el signo, por lo que es suficiente cambiar todos los ceros por unos y los unos por ceros, sin cambiar el bit de signo que se ubica al extremo izquierdo. Ejemplo:

$$\begin{array}{cccccccccccl} 1 & & 1 & 1 & 0 & 1 & 1 & 1 & , & 0 & 1_2 & \text{Magnitud verdadera} \\ 1 & & 0 & 0 & 1 & 0 & 0 & 0 & , & 1 & 0_2 & \text{Complemento a 1} \end{array}$$

$$\begin{array}{ccccccccccl} 0 & & 1 & 0 & 1 & 0 & , & 1 & 0_2 & \text{Magnitud verdadera} \\ 0 & & 0 & 1 & 0 & 1 & , & 0 & 1_2 & \text{Complemento a 1} \end{array}$$

8.1. Generalización de complementos

Existen tres formas para representar cantidades, la magnitud verdadera de un número X, complemento a la base menos uno y complemento a la base, usando las fórmulas:

$$C_{b-1}(X) = C_b(X) - 1$$

$$C_b(X) = C_{b-1}(X) + 1$$

Por ejemplo, para encontrar el complemento a 10 de X=3705 se tiene $C_9(X)$ + 1:

$$C_9\ (3705) = 6294$$

$$\begin{array}{r l} 6294 & + \\ 1 & \\ \hline 6295 & \end{array}$$

Por lo tanto C_{10} (3705) = 6295

Por ejemplo, encontrar el complemento a 2 de 1010_2 esto es $C_1(1010_2)$+1:

$$C_1\ (1010_2) = 0101_2$$

$$\begin{array}{r l} 0101 & + \\ 1 & \\ \hline 0110 & \end{array}$$

$$C_2\ (1010_2) = 0110_2$$

Por lo tanto, el complemento a 2 se obtiene sumando 1 al bit menos significativo del complemento a 1, por ejemplo:

$$\begin{array}{cccccccccc|l} 1 & 0 & 0 & 1 & 0 & 0 & 0 & , & 1 & 0_2 & + \ \text{Complemento a 1} \\ & & & & & & & & & 1 & \\ \hline 1 & 0 & 0 & 1 & 0 & 0 & 0 & , & 1 & 1_2 & + \ \text{Complemento a 2} \end{array}$$

$$\begin{array}{cccccccc|l} 0 & 0 & 1 & 0 & 1 & , & 0 & 1_2 & + \ \text{Complemento a 1} \\ & & & & & & & 1 & \\ \hline 0 & 0 & 1 & 0 & 1 & , & 1 & 0_2 & + \ \text{Complemento a 2} \end{array}$$

8.2. Representación de datos en la computadora

En la computadora los datos pueden ser **numéricos** o **alfanuméricos**, dentro de los datos numéricos pueden ser **enteros** o **reales** y dentro de los **enteros** se tienen a los **enteros con signo** y los **enteros sin signo** las operaciones con enteros sin signo suelen ser representadas utilizando números en base binaria ($b = 2$).

8.2.1. Enteros sin signo

Se dispone de n bits, para representar 2^n enteros sin signo dentro de un rango de:

$$0 \leq X \leq 2^n - 1$$

Por ejemplo, si se disponen de 4 bits, el rango de números será de $0 \leq X \leq 2^4 - 1 = 15$, en el que se pueden representar números como 0=0000, 2=0010, 14=1110, 15=1111 como enteros sin signo con 4 bits.

8.2.2. Enteros con signo

Los enteros con signo se pueden representar con **signo y magnitud**, **complemento a uno**, **complemento a dos** y **exceso**, por tanto, si se disponen de n bits, se usa el primer bit de la izquierda para representar el signo 0 para positivo y 1 para negativo.

1. Considerando el bit de signo, el valor absoluto del número se representa en base dos en los $(n-1)$ bits restantes, el rango de números será de:

   $$-(2^{n-1}-1) \leq X \leq (2^{n-1}-1)$$

   Por ejemplo, si se disponen de 4 bits, el rango de números que puede ser representados será de $-(2^{4-1}-1) \leq X \leq (2^{4-1}-1) = -7 \leq X \leq 7$, en el que se pueden representar números como 0=0000, -0=1000, 4=0100, -4=1100, 7=0111, -7=1111 como **enteros con signo y magnitud de 4 bits**.

2. Si se disponen de n bits y $X \geq 0$ se representa al número como binario, con un cero en el primer bit de la izquierda, y si $X \leq 0$ se representa al número como el **complemento a uno** de la representación del valor absoluto de X con el rango:

   $$-(2^{n-1}-1) \leq X \leq (2^{n-1}-1)$$

   Por ejemplo, si se disponen de 4 bits, el rango de números que pueden ser representados en complemento a uno es de $-(2^{4-1}-1) \leq X \leq (2^{4-1}-1) = -7 \leq X \leq 7$.

3. Si se disponen de n bits y $X \geq 0$ se representa al número como binario con un cero en el primer bit de la izquierda, y si $X \leq 0$ se representa al número como el **complemento a dos** de la representación del valor absoluto de X con el rango:

   $$-(2^{n-1}) \leq X \leq (2^{n-1}-1)$$

   Por ejemplo, si se disponen de 4 bits, el rango de números que pueden ser representados en en complemento a dos es de $-(2^{4-1}) \leq X \leq (2^{4-1}-1) = -8 \leq X \leq 7$.

4. Si se dispone de n bits, la representación del valor en **exceso** se representa el número X como la representación binaria de $(X+2^{n-1})$ con el rango de:

   $$-(2^{n-1}) \leq X \leq (2^{n-1}-1)$$

   Por ejemplo, si se disponen de 4 bits, el rango de números que pueden ser representados en exceso es de $-(2^{4-1}) \leq X \leq (2^{4-1}-1) = -8 \leq X \leq 7$, en el que se pueden representar números como $0 = 0 + 2^{4-1} = 0 + 8 = 1000$.

8.2.3. Números reales

Los números reales se pueden presentar en **punto fijo**, **notación científica decimal**, **notación científica binaria** y **notación científica binaria con mantisa normalizada**.

1. El **Punto fijo** es el que trabaja con un número fijo de decimales, por ejemplo si se tienen 32 bits, se pueden asignar 1 bit para el signo, 21 bits para la parte entera y 10 bits para la parte decimal, así, para representar X=-38,320 se tiene que $- = 1$, $38 = 100110_2$ y $320 = 101000000_2$, por lo tanto, el -38.320 se representa por:

   $$1\ 000000000000000100110\ 0101000000_2$$

2. La **notación científica decimal**, en la cual se expresa el valor absoluto del número en notación científica decimal de la forma $f \times 10^K$, donde $f \in [0{,}1, 1[$ y $K \in Z$. Se usa el primer bit de la izquierda para representar el signo, un número fijo de bits para representar el **exponente en exceso** y un número fijo de bits para representar la **mantisa** como entero sin signo, por ejemplo, si se tienen 32 bits, y se asignan 1 bit para el signo, 7 bits para el exponente y 24 bits para la mantisa, se puede representar el número -3,25 como: $-3,25 = -0,325 \times 10^1$, además, $- = 1$, $325 = 101000101_2$ y el 1 se representa en exceso por 1000001_2, por lo tanto, el -3,25 se representa por:

$$1\ 1000001\ 000000000000000101000101_2$$

3. La **notación científica binaria** en la cual se usa el primer bit de la izquierda para representar el signo, se tiene un número fijo de bits para representar el **exponente en exceso** y un número fijo de bits para la **mantisa**. Por ejemplo, si se tienen 32 bits, 1 bit para el signo, 7 bits para el exponente y 24 bits para la mantisa, se puede representar el -419,8125 como: $-419,8125 = -110100011,1101_2 = -0,1101000111101_2 \times 2^9$, el 9 se presenta en exceso por 10010011_2, por lo tanto el -419,8125 se representa por:

$$1\ 1001001\ 110100011110100000000000_2$$

4. La **notación científica binaria con mantisa normalizada** en la cual se representa la mantisa sin el primer bit de la izquierda, ya que este bit es siempre igual a uno. Por ejemplo, si se tienen 32 bits, 1 bit para el signo, 7 bits para el exponente y 24 bits para la mantisa, se puede representar el -419,8125 como: $-419,8125 = -110100011,1101_2 = -0,1101000111101_2 \times 2^9$, el 9 se representa en exceso por 10010011_2, por lo tanto el -419,8125 se representa por:

$$1\ 1001001\ 101000111101000000000000_2$$

8.2.4. Caracteres alfanuméricos

Los caracteres alfanuméricos pueden ser dígitos, letras o caracteres especiales como signos de puntuación, agrupamiento, matemáticos, gráficos o de control. Tales caracteres son representados por cadenas binarias iguales al código asociado a cada caracter. Las tablas de códigos más usadas son la tabla ASCII y la tabla EBCDIC. La Tabla 4 muestra algunos caracteres de la tabla ASCII en código decimal y binario.

Caracter	Código Decimal	Código Binario
:	:	:
0	48	0011 0000
1	49	0011 0001
:	:	:
@	64	0100 0000
A	65	0100 0001
:	:	:
Z	90	0101 1010
a	97	0110 0001
z	122	0111 1010
:	:	:

Tabla 4: Tabla ASCII

8.3. Circuito Sumador

El circuito sumador tiene 3 entradas y 2 salidas y se implementa con compuertas lógicas como se muestra en la Figura 1, los resultados de este circuito según el valor de X_j Y_j y C_j se muestran en la Tabla 5, donde se indican todos los valores posibles según el valor que ingresa al circuito, cuyas salidas se muestran en C_{j+1} y S_j.

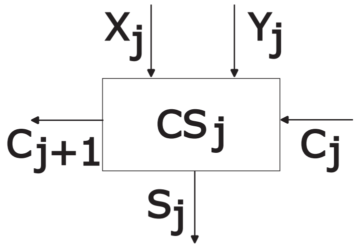

Figura 1: Circuito Sumador

X_j	Y_j	C_j	C_{j+1}	S_j
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Tabla 5: Tabla del Circuito Sumador

Cabe resaltar que en un circuito sumador binario real, el resultado de la suma de X_j y Y_j se espresa en S_j, si hubiera un acarreo, este se considera en el siguiente C, esto es en C_{j+1}.

Las operaciones que se realizan en computadora en general se llevan a cabo en el sistema de numeración binario, usando la operación básica de la suma, así la computadora solo realiza sumas, no realiza restas, ni multiplicaciones, ni divisiones, sin embargo, una multiplicación es una sucesión de sumas (por ejemplo, para multiplicar $X \times Y$ lo que realmente se hace es sumar Y veces la cantidad X) y una división es una sucesión de restas (por ejemplo, para dividir X/Y, a la cantidad X se le restan Z veces la cantidad Y, mientras Z sea mayor o igual a Y, siendo Z el cociente). Cuando las dos cantidades a sumar son positivas, se suman con sus magnitudes reales, pero si alguna de ellas es negativa, la cantidad negativa se complementa a 2 y después se suma a la otra cantidad, de forma que una resta se convierte en una suma.

Por ejemplo, utilizando un circuito sumador 4 bits, se pueden sumar los números $X = 10$ y $Y = 3$, representados como enteros sin signo, para tal, el $10 = 1010_2$ y $3 = 0011_2$, así, el circuito sumador correspondiente se muestra en la Figura 2.

Suponiendo tres variables enteras (X, Y y Z) que ocupan 4 bits de memoria cada una (fuera del bit de signo), se pueden dar cuatro casos, dependiendo de los números a sumar, a fin de obtener $Z = X + Y$, así, primero se convertiran los valores de ingreso al sistema binario y luego se realizarán las sumas. A continuación se resolverán ejemplos para todos los casos.

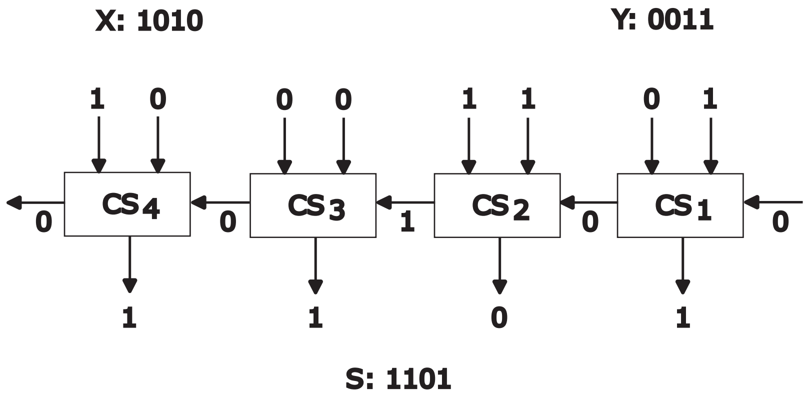

Figura 2: Ejemplo de Circuito Sumador, la cadena S: 1101_2 representa al 13.

1. Si $X = 12$ y $Y = 7$ se tiene:

$$\begin{array}{cccccccccc}
12 & + & = & 0 & & 1 & 1 & 0 & 0_2 & + \\
7 & & = & 0 & & 0 & 1 & 1 & 1_2 & \\
\hline
19 & & & 1 & & 0 & 0 & 1 & 1_2 &
\end{array}$$

El resultado obtenido es 1 0011_2 = -3, que es diferente al +19 esperado. En este caso se presentó un **desbordamiento** al querer guardar en la variable Z una cantidad mayor al rango aceptado para 4 bits. En programación, este error ocurre cuando se definen variables de cierto tipo y con cierta capacidad, y se quiere almacenar en ellas una cantidad que sobrepasa esa capacidad. En este caso se espera mostrar como se deben considerar los elementos que se presentan en el momento en que la computadora realiza una operación aritmética básica.

El problema de **desbordamiento** se resuelve definiendo las variables con una capacidad mayor, por ejemplo, se puede suponer que las variables X, Y y Z son enteras, con capacidad de 8 bits sin contar el bit de signo, que sería lo adecuado para el rango en el que se quiere trabajar.

$$\begin{array}{cccccccccccccc}
12 & + & = & 0 & & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0_2 & + \\
7 & & = & 0 & & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1_2 & \\
\hline
19 & & & 0 & & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1_2 &
\end{array}$$

Ahora el resultado es 0 00010011_2, equivalente a +19, que es el resultado correcto, con lo que se evita el desbordamiento.

Vale indicar que el desbordamiento ocurre sólo cuando las dos cantidades a sumar tienen el mismo signo, ya que son los únicos casos en que el resultado puede requerir mayor espacio.

2. Si $X = -12$ y $Y = 7$ se tiene:

Usando Complemento a 2 para la cantidad negativa:

$$\begin{array}{cccccccc}
-12 & 1 & & 1 & 1 & 0 & 0_2 & & \text{Magnitud verdadera} \\
 & 1 & & 0 & 0 & 1 & 1_2 & + & \text{Complemento a 1} \\
 & & & & & & 1 & & \\
\hline
 & 1 & & 0 & 1 & 0 & 0_2 & & \text{Complemento a 2}
\end{array}$$

Efectuando la suma:

$$\begin{array}{c c c c c c c c}
-12 & 1 & 0 & 1 & 0 & 0_2 & + \\
7 & 1 & 0 & 1 & 1 & 1_2 & \\
\hline
-5 & 1 & 1 & 0 & 1 & 1_2 &
\end{array}$$

Convirtiendo este número a decimal se tiene que 1 1011_2, que es equivalente a -11 y no -5 que debería ser. En forma general se puede decir que si el resultado de la suma es negativo, se debe complementar a dos el resultado.

$$\begin{array}{c c c c c c c l}
 & 1 & 1 & 0 & 1 & 1_2 & & \text{Resultado negativo} \\
 & 1 & 0 & 1 & 0 & 0_2 & + & \text{Complemento a 1} \\
 & & & & & 1 & & \\
\cline{2-6}
-5 & 1 & 0 & 1 & 0 & 1_2 & & \text{Complemento a 2}
\end{array}$$

Así el resultado es el correcto, pues -5= 1 0101_2

3. Si $X = 12$ y $Y = -7$ se tiene:

   Usando Complemento a 2 para la cantidad negativa:

$$\begin{array}{c c c c c c c l}
-7 & 1 & 0 & 1 & 1 & 1_2 & & \text{Magnitud verdadera} \\
 & 1 & 1 & 0 & 0 & 0_2 & + & \text{Complemento a 1} \\
 & & & & & 1 & & \\
\cline{2-6}
 & 1 & 1 & 0 & 0 & 1_2 & & \text{Complemento a 2}
\end{array}$$

   Efectuando la suma:

$$\begin{array}{c c c c c c c c c}
12 & & 0 & 1 & 1 & 0 & 0_2 & + \\
-\ 7 & & 1 & 1 & 0 & 0 & 1_2 & \\
\hline
5 & 1 & 0 & 0 & 1 & 0 & 1_2 & \\
 & \text{Acarreo} & \text{Signo} & & & & &
\end{array}$$

   La suma da un resultado positivo, como indica el signo, con el resultado correcto, ya que sólo se complementan a dos los resultados negativos. El acarreo obtenido se debe despreciar en todos los casos.

4. Si $X = -12$ y $Y = -7$ se tiene:

   En este caso, como ambas cantidades a sumar son negativas, se debe obtener el complemento a dos de ambas cantidades antes de realizar la suma, también se debe considerar que la suma produce un desbordamiento, por lo tanto se debe trabajar con el número de bits correctos, complementando a dos ambas cantidades y considerando el desbordamiento.

$$\begin{array}{c c c c c c c c c c c l}
-12 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0_2 & & \text{Magnitud verdadera} \\
 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1_2 & + & \text{Complemento a 1} \\
 & & & & & & & & & 1 & & \\
\cline{2-10}
 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0_2 & & \text{Complemento a 2}
\end{array}$$

$$\begin{array}{c c c c c c c c c c c l}
-7 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1_2 & & \text{Magnitud verdadera} \\
 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0_2 & + & \text{Complemento a 1} \\
 & & & & & & & & & 1 & & \\
\cline{2-10}
 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1_2 & & \text{Complemento a 2}
\end{array}$$

Efectuando la suma:

$$\begin{array}{rccccccccccccc}
\text{-12} & + & = & & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0_2 & + \\
\text{- 7} & & = & & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1_2 & \\
\hline
\text{-19} & & & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1_2 &
\end{array}$$

El resultado no es el esperado (-19), además se sabe que si el resultado es negativo, se debe complementar a dos, despreciando el acarreo se tiene:

$$\begin{array}{rcccccccccccl}
 & & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1_2 & & \text{Resultado negativo} \\
 & & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0_2 & + & \text{Complemento a 1} \\
 & & & & & & & & & & 1 & & \\
\hline
\text{-19} & = & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1_2 & & \text{Complemento a 2}
\end{array}$$

Así el resultado es el correcto, pues -19= 1 00010011_2.

Finalmente, si se trabajan con números reales, que tengan parte fraccionaria, se debe cuidar que se deba tener el número de bits suficiente, para ello se debe considerar el rango de forma adecuada, además, para encontrar el complemento a uno, se debe sumar el uno en el bit menos significativo independientemente si pertenece a la parte entera o fraccionaria, esto es el que esté más a la derecha, y sólo se complementan a dos las cantidades negativas y los resultados negativos de las sumas.

Ejercicios

Usando Complemento a dos, realice las siguientes operaciones según lo requerido:

1. 568+13.
2. Considerando que el signo ocupa 1 bit, la parte entera del número ocupa 8 bits y que la parte fraccionaria ocupa 4 bits. Realice la suma en complemento a 2. Agregue bytes en caso de ser necesario para evitar el desbordamiento.
   a) 54,23 + 28,56
   b) -54,23 + 28,56
   c) 54,.23 - 28,56
   d) -54,23 - 28,56

9. Aplicaciones

El sistema numérico binario el lenguaje máquina, es el lenguaje natural de la computadora, pues con él se lleva a cabo las operaciones aritméticas, procesamiento de información, control de periférico, conexión y comunicación con otras computadoras. Para mejorar su entendimiento existe, por ejemplo, el código ASCII, que consiste en una tabla de equivalencias entre el sistema binario y los caracteres que se usan para representar palabras a través de una cadena de ocho bits, de esta forma, es relativamente fácil que la computadora traduzca una frase escrita en español, inglés o cualquier otro idioma a sistema binario. Otros sistemas numéricos importantes en el campo de Ingeniería de Sistemas, además del binario, son el octal y hexadecimal.

Por ejemplo, el retiro de una cantidad de dinero en un cajero automático.

10. Trabajos referidos al tema

1. Desarrolle los ejercicios propuestos en clase.
2. Realice un programa en C, C#, C++ o Java usando sólo librerías básicas de:
   a) Conversor de base, considerando de la base 2 a la base 16.
   b) Suma usando Complemento a 2.
3. Escriba con sus propias palabras un documento tipo monografía de mínimo 1 página y máximo 3 páginas (no es necesario que tenga carátula), indicando las referencias (páginas web, libros, etc.) que usó para realizar su trabajo de:
   a) La historia de los sistemas numéricos, o de algún sistema numérico en específico.
   b) Aplicaciones de sistemas de numeración.

Referencias

[1] Jiménez, J. A. (2005) Matemáticas para la computación, Primera Edición, Alfaomega Grupo Editor, S.A. de C.V., México.

[2] Pavletich, S. (2013) Estructuras discretas, Pontificia Universidad Católica del Perú.

[3] Vicarrioli, F. M. (2014) Matemática para la informática I, UNED.