DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (II/2025)

Ayudantía 1

Ayudantes: Daniela Ríos (danielaarp@uc.cl), Alberto Maturana (alberto.maturana@uc.cl)

Pregunta 1: Representación de Números

(a) Convierta los siguientes números decimales a binario:

1. 93_{10}
2. 205_{10}

Solución:

Para pasar los números a binario usaremos la siguiente tabla de potencias de 2.

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
512	256	128	64	32	16	8	4	2	1	0.5	0.25	0.125	0.0625

1. 93_{10}:

$$93 - 2^6 = 93 - 64 = 29$$
$$29 - 2^4 = 29 - 16 = 13$$
$$13 - 2^3 = 13 - 8 = 5$$
$$5 - 2^2 = 5 - 4 = 1$$
$$1 - 2^0 = 1 - 1 = 0$$

Entonces,

$$93 = 2^6 + 2^4 + 2^3 + 2^2 + 2^0$$
$$93_{10} = 1011101_2$$

2. 205_{10}:

$$205 - 2^7 = 205 - 128 = 77$$
$$77 - 2^6 = 77 - 64 = 13$$
$$13 - 2^3 = 13 - 8 = 5$$
$$5 - 2^2 = 5 - 4 = 1$$
$$1 - 2^0 = 1 - 1 = 0$$

Entonces,

$$205 = 2^7 + 2^6 + 2^3 + 2^2 + 2^0$$
$$205_{10} = 11001101_2$$

(b) Reescriba los siguientes números según su representación complemento de 2. Después sume el segundo con el tercero.

1. -9_{10}
2. $-B4_{16}$
3. 97_{10}
4. -1_{10}
5. 0_n

Solución:

1. -9_{10}:

   Pasamos el número sin signo a binario,

   $$9_{10} = 1001_2$$

   Completamos el byte (8 bits) con 0's.

   $$1001_2 = 00001001_2$$

   Para complemento de 2 debemos invertir los bits y después sumar 1.

   $$00001001 \rightarrow 11110110 \rightarrow 11110111$$
   $$-9_{10} = 11110111_2 \quad \text{(complemento de 2)}$$

2. $-B4_{16}$:

   Pasamos el número sin signo a binario,

   $$B4_{16} = 10110100_2$$

Como el número ya usa un byte, vamos a tener que agregar otro byte entero.

$$10110100_2 = 0000\ 0000\ 1011\ 0100_2$$

Para complemento de 2 debemos invertir los bits y después sumar 1.

$$0000\ 0000\ 1011\ 0100 \rightarrow 1111\ 1111\ 0100\ 1011 \rightarrow 1111\ 1111\ 0100\ 1100$$

$$-B4_{16} = 1111\ 1111\ 0100\ 1100_2 \quad \text{(complemento de 2)}$$

3. 97_{10}:

   Pasamos el número sin signo a binario,

   $$97_{10} = 1100001_2$$

   Completamos el byte (8 bits) con 0's a la izquierda.

   $$1100001_2 = 01100001_2$$

   Para representar el número en complemento de 2 no debemos hacer nada porque es un número positivo y su MSB es 0.

   $$97_{10} = 01100001_2 \quad \text{(complemento de 2)}$$

4. -1_{10}:

   Pasamos el número sin signo a binario,

   $$1_{10} = 1_2$$

   Completamos el byte (8 bits) con 0's.

   $$1_2 = 00000001_2$$

   Para complemento de 2 debemos invertir los bits y después sumar 1.

   $$00000001 \rightarrow 11111110 \rightarrow 11111111$$

   $$-1_{10} = 11111111_2 \quad \text{(complemento de 2)}$$

5. 0_n:

   Pasamos el número sin signo a binario,

   $$0_n = 0_2$$

   Completamos el byte (8 bits) con 0's.

   $$0_2 = 00000000_2$$

   En complemento de 2 solo hay una representación.

   $$0_{10} = 00000000_2 \quad \text{(complemento de 2)}$$

6. $-B4_{16} + 97_{10}$: Ya sabemos que:

$$-B4_{16} = 1111\ 1111\ 0100\ 1100_2$$

$$97_{10} = 0110\ 0001_2$$

Sumamos (agregando ceros para igualar bits):

$$1111\ 1111\ 0100\ 1100_2 + 0000\ 0000\ 0110\ 0001_2$$

Resultado:

$$1111\ 1111\ 1010\ 1101_2$$

(c) ¿Qué ocurre si sumamos 52_{10} con 19_{10} usando solo 7 bits y representándolos en complemento 2?

Solución:

$$52_{10} = 0110100_2$$

$$19_{10} = 0010011_2$$

$$0110100_2 + 0010011_2 = 1000111_2$$

$$1000111_2 = -57_{10}$$

El resultado real debería ser 71_{10}, pero no cabe en 7 bits con complemento de 2, por lo que ocurre **overflow**.

Pregunta 2: Representaciones Numéricas [I1 - 2025-1]

(a) (1 pto.) Suponga que se suman dos números de 8 bits con signo, A y B, ambos negativos. Como resultado, se genera un acarreo de salida igual a 1. ¿Implica esto necesariamente la ocurrencia de un overflow? Justifique su respuesta.

Solución: No. El *overflow* ocurre cuando el resultado de una operación entre dos números está fuera del rango de su representación. Al usar una representación con signo, el acarreo de salida igual a 1 no implica que exista *overflow*. Esto se puede evidenciar con un contraejemplo en una representación de números enteros con 4 bits:

$$-12 + -22 = 1111 + 1110 = (1)1101 = -32$$

Si bien existe un acarreo de salida, no hay *overflow*. Un acarreo de salida solo evidencia *overflow* en una representación **sin signo**.
Desglose puntaje: Se otorgan 0.5 ptos. por responder correctamente; y 0.5 ptos. por justificación.

(b) (2 ptos.) Considere una función $C_B(x)$ definida para un número x en base B, tal que $x + C_B(x) = 0$, donde la suma se realiza dígito a dígito en base B y sobre una cantidad fija de cifras. Esta función representa el **complemento en base** B de x. Sabiendo que $C_2(x)$ corresponde al complemento a dos de x (estudiado en clases), defina paso a paso el procedimiento para calcular el complemento de 4 de un número x, es decir, $C_4(x)$. Además, considere que la representación del cero debe ser única, formada exclusivamente por ceros en todos sus dígitos, evitando representaciones alternativas (como ocurre con el complemento a uno). Justifique cómo su procedimiento garantiza que se cumple $x + C_4(x) = 0$ en base 4, utilizando una cantidad fija de dígitos.

Solución: El procedimiento, en este caso, no es más que la generalización del complemento de 2:

1. Para cada dígito xi de un número x = xN−1...x0 en base B de N dígitos, definir $x'_i = (B{-}1{-}x_i)_B$
2. Construir el complemento en base B de x como $C_B(x) = x'_{N-1}...x'_0 + 1$

Este funciona porque, al sumar x_0 con x_0 se tendrá que:

$$x_0 + x'_0 = (B - 1)_B$$

Luego, al sumar la unidad extra definida en el paso 2:

$$x_0 + x'_0 + 1 = (B - 1 + 1)_B = (B)_B = 10_B$$

Entonces, la suma resultará en un valor igual a B que, al no tener un dígito asociado en su propia base, será igual a 10_B. Esto **equivale a una suma igual a cero con acarreo de salida igual a 1**, por lo que $(x + C_B(x))_B = 0$. Finalmente, esto derivará en el siguiente resultado para el resto de los dígitos de $x + C_B(x)$:

$$(x{+}C_B(x))_i = \left(x_i + x'_i + \text{carry}_{i-1}\right) = \left(x_i + x'_i + 1\right) = (B{-}1{+}1)_B = B_B = 10_B = 0, \quad i > 0.$$

Así, se evidencia que $x + C_B(x)$ será igual a una secuencia de N dígitos iguales a cero con un acarreo de salida final igual a 1, que es el resultado esperado. Para computar $C_4(x)$, simplemente se reemplaza B por 4 en el procedimiento planteado. Un ejemplo a continuación para computar el complemento en base 4 de 123_4:

$$C_4(123_4) = 210_4 + 1 = 211_4$$

$$\begin{array}{rcccc} carry \rightarrow & & 1 & 1 & 0 \\ & & 1 & 2 & 3_4 \\ + & & 2 & 1 & 1_4 \\ \hline 1 & & 0 & 0 & 0_4 \end{array}$$

Otra opción válida es definir $C_B(x)$ de N dígitos como:

$$C_B(x) = (B_B)^N - x$$

Como la representación de B en su propia base es igual a 10_B, se tiene:

$$C_B(x) = 10_B^N - x$$

Podemos evaluar su correctitud reordenando la igualdad de la siguiente forma:

$$C_B(x) + x = 10_B^N$$

10_B^N, independiente de la base, tendrá un total de $N+1$ dígitos, por lo que al truncar el resultado en los N dígitos menos significativos, se descartará naturalmente el dígito igual a 1 que interpretamos como el acarreo de salida. Si aplicamos esta definición sobre el mismo ejemplo anterior, vemos que se llega al mismo resultado:

$$C_4(123_4) = 1000_4 - 123_4 = 211_4$$

Desglose puntaje: Se otorga **1 pto**. por definir un procedimiento correcto; y **1 pto**. por justificación. No es necesario proponer un método para $C_B(x)$, pero sí para $C_4(x)$.

1. Feedback ayudantía

Escanee el QR para entregar feedback sobre la ayudantía.