

# Aritmética binaria

06

## INTRODUCCION

El conocimiento sobre los cálculos binarios básicos es indispensable para el análisis y diseño de sistemas digitales. Las operaciones más simples que se trabajan aquí son la suma, la resta, la multiplicación, y la división binaria.

## SUMA

La suma de cada bit se realiza según el cuadro 6.1 y corresponde al incremento en 1 del código binario. Es decir, si tengo 0 y agrego 1, tengo 1. Si agrego 1 más, tengo 10. Y así sucesivamente.

SUMA BINARIA BÁSICA	
SUMA	RES.
0 + 0	0
0 + 1	1
1 + 0	1
1 + 1	10

Cuadro 6.1: Suma binaria básica.

Cuando se trata de sumar números de N bits, se requiere la propagación del acarreo bit a bit cómo se muestra en la ilustración 6.1.

$$\begin{array}{r} \phantom{+} \phantom{1} \phantom{0} \phantom{1} \phantom{1} \phantom{0} \phantom{1} \phantom{0} \phantom{0} \\ \phantom{+} \phantom{1} \phantom{0} \phantom{1} \phantom{1} \phantom{0} \phantom{1} \phantom{0} \phantom{0} \\ + \phantom{1} \phantom{0} \phantom{1} \phantom{1} \phantom{0} \phantom{1} \phantom{0} \phantom{0} \\ \hline \textcircled{1} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{1} \end{array}$$

Ilustración 6.1: Suma  $76+181=257$ . Se observa el desbordamiento en los 8 bits (256 máximos).

La propagación del acarreo del último de los bits, se lo llama carry out. En el caso de la ilustración 6.1, corresponde el marcado con un círculo rojo.

## RESTA

La resta tiene la dificultad de requerir del conocido **borrow** o “pedir prestado”. Su procesamiento del modo tradicional (como en la escuela primaria pero con número binarios) resulta complejo y suele trabajarse, por lo tanto, con complementos a la base o a la base-1, efectuando así, únicamente sumas. Por ejemplo, la resta  $65-32=33$ , puede hacerse con complemento a la base (100): Complementamos a 100 el número negativo  $32 \Rightarrow 68$ , y lo sumamos:  $65+68=133$ . Finalmente complementamos a 100 nuevamente, y resulta: 33, verificando la resta. Siendo redundante, se puede aclarar nuevamente, que a pesar de que este procedimiento parece ser claramente más complejo, para una computadora la suma y el complemento son operaciones rápidas a diferencia de la resta, con lo que sólo se usan complementos. A continuación se muestran las formas de complementar y posteriormente, en las *Ilustraciones 6.5 y 6.6*, se muestran ejemplos en los tres modos de resta.

### RESTA TRADICIONAL

$$\begin{array}{r} \phantom{-} \phantom{0} \phantom{0} \phantom{1} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{1} \\ \phantom{-} \phantom{0} \phantom{0} \phantom{1} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \\ \hline 0 \phantom{0} \phantom{0} \phantom{1} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{0} \phantom{1} \end{array}$$

Ilustración 6.2: Resta con borrow  $65-32=33$

### COMPLEMENTO A 2

La forma humana más eficiente para efectuar el complemento a 2 es invirtiendo todos los bits (pasar de 0 a 1 y al revés) de izquierda a derecha hasta llegar al último de los 1. De ahí en más dejar todo como está.

$$\begin{array}{l} 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ | \ 1 \ 0 \ \text{Número negativo} \\ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ | \ 1 \ 0 \ \text{Complemento a 2} \end{array}$$

Ilustración 6.3: Conversión de un número negativo a complemento a 2.

### COMPLEMENTO A 1

El complemento a la base menos 1 (o complemento a 1) se hace invirtiendo todos los bits del número.

$$\begin{array}{l} 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ \text{Número negativo} \\ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \ \text{Complemento a 1} \end{array}$$

Ilustración 6.4: Conversión de un número negativo a complemento a 1.

Notar que el complemento a 2 es el complemento a 1 + 1.

$\begin{array}{r} 0111 \\ - 0011 \\ \hline 0100 \end{array}$ <p>(a)</p>	$\begin{array}{r} 0111 \\ + 1101 \\ \hline 10100 \end{array}$ <p>(b)</p>	$\begin{array}{r} 0111 \\ + 1100 \\ \hline 10011 \\ + 1 \\ \hline 10100 \end{array}$ <p>(c)</p>
---	--	---

Ilustración 6.5: 7-3=4 (a) Resta tradicional; (b) Resta con complemento a 2; (c) Resta con complemento a 1.

$\begin{array}{r} 01000001 \\ - 00100000 \\ \hline 00100001 \end{array}$ <p>(a)</p>	$\begin{array}{r} 01000001 \\ + 11100000 \\ \hline 10010001 \end{array}$ <p>(b)</p>
$\begin{array}{r} 01000001 \\ + 11011111 \\ \hline 10010000 \\ + 1 \\ \hline 10010001 \end{array}$ <p>(c)</p>	

Ilustración 6.6: 65-32=33 (a) Resta tradicional -con Borrow- (b) Resta con complemento a 2; (c) Resta con complemento a 1.

## MULTIPLICACIÓN

La particularidad de la multiplicación binaria es que para un producto de 1 palabra de N bits por otra de igual cantidad de bits, el resultado se expresa en 2\*N bits. Esto es fundamental, dado que un microprocesador requerirá de dos bytes de memoria para el producto de 1byte por 1byte.

Los pasos a seguir son sencillos. En la *ilustración 6.7* se muestra el producto 200\*248.

$\begin{array}{r} 200 \\ \times 248 \\ \hline 49600 \end{array}$	$\begin{array}{r} 11001000 \\ \times 11111000 \\ \hline 11001000 \\ 110010000 \\ 1100100000 \\ 11001000000 \\ \hline 1100100011100000 \end{array}$ <p style="text-align: center;">Byte Superior                      Byte Inferior</p>
--	--

Ilustración 6.7 Multiplicación binaria. Se destaca el resultado en el doble de bits que los valores de entrada.

## MULTIPLICACIÓN POR POTENCIAS DE 2

Una forma sencilla y muy usada en programación para realizar productos, es el desplazamiento a izquierda. Cuando el multiplicador es 2 el resultado del producto es el multiplicando desplazado a la izquierda un bit. Por ejemplo, 100 (4) por 10 (2), da como resultado 1000 (8).

Si el multiplicador es una potencia de dos, sea 4, 8, 16..., se deberán desplazar los bits del multiplicando 2, 3, 4... veces, respectivamente.

## DIVISIÓN

La división es un procedimiento sencillo. Se debe tener cuidado al expresar el resultado, dado que al igual que la multiplicación, su salida está dada por dos palabras: El cociente y el resto. En la *ilustración 6.8* se muestra la división  $17/4=4,25$ . En microprocesadores, se emplea el resultado del cociente y el resto por separado.

$$\begin{array}{r} 10001 \quad \overline{)100} \\ \underline{001} \quad 100 \\ 1 \phantom{00} \\ \hline \end{array}$$

(a)

$$\begin{array}{r} 10001 \quad \overline{)100} \\ \underline{001} \quad 100,01 \\ 100 \\ \underline{0} \\ \hline \end{array}$$

(b)

Ilustración 6.8: División binaria. (a) División con Cociente y Resto como resultados. (b) División completa.

## DIVISIÓN POR POTENCIAS DE 2

La técnica es similar a la multiplicación por potencias de dos, pero el desplazamiento es a derecha. Por ejemplo, 111 (7) dividido 10 (2), da como resultado 11 (3). Notar que en el desplazamiento se pierde un bit 'encendido' que representa el resto: 0,1 binario (0,5 decimal).

## BANDERAS O "FLAGS".

- 1.Desbordamiento u "Overflow": Cuando en una suma con signo, dos números positivos arrojan un resultado negativo, o dos números negativos arrojan un resultado positivo. Se especifica en hojas de datos y bibliografía con la letra "V".
- 2.Acarreo de salida o "Carry out": Cuando existe acarreo en el último bit de la suma. Se especifica como "Co".
- 3.Cero o "Zero": Se enciende cuando el resultado de una operación es nulo. Se especifica con la letra "Z".
- 4.Bit de signo o "Sign bit": Vale 1 si el resultado es negativo. Se especifica con la letra "N".