

Acarreos en suma

Evidentemente $a_i + b_i \neq c_i + b_i$: como consecuencia de los acarreos, debidos a la aplicación de la regla según la cual b unidades de cualquier orden forman una unidad de orden superior.

En general, la suma de 2 dígitos en base b de cualquier orden se puede formalizar con la siguiente ecuación:

$$a_i + b_i = c_i \cdot b_i + s_i$$

Donde el dígito s_i representa la suma en el orden i -ésimo

$$s_i = (a_i + b_i) \bmod b$$

y el dígito c_i representa el posible acarreo al orden superior.

$$c_i = \left\lfloor \frac{a_i + b_i}{b} \right\rfloor = \frac{(a_i + b_i) - s_i}{b}$$

En general, dados 2 dígitos de base b su suma produce un dígito suma y otro acarreo tal que:

- o el dígito suma es el resultado de aplicar la función módulo b a la suma de los dígitos;
- o el dígito acarreo es el cociente entero resultante de dividir la suma entre la base b .

En binario, la suma de los 2 bits de cualquier orden se puede formalizar con la siguiente ecuación:

$$a_i + b_i = c_i \cdot 2 + s_i$$

La implementación de esta ecuación se conoce como **semisumador**.

Semisumador

La ecuación $a_i + b_i = c_i \cdot 2 + s_i$ responde a la implementación conocida como semisumador, es decir, un sumador de dos operandos de 1 bit como entrada (a_i y b_i) y un bit de suma s_i y otro de acarreo al peso siguiente c_i como salida. La tabla de verdad es la siguiente:

a_i	b_i	s_i	c_i	$s_i = a_i \oplus b_i$
0	0	0	0	
0	1	1	0	
1	0	1	0	
1	1	0	1	

a_i	b_i	$c_i = a_i \cdot b_i$
0	0	0
0	1	0
1	0	0
1	1	1

Observamos que, en binario:

- la función módulo de s_i se implementa con la función XOR; y
- el cociente entero de c_i se implementa con la función AND

Sumando c_i y s_i

El valor de c a partir de la ecuación $a_i + b_i = c_i \cdot 2 + s_i$ será:

$$c = \sum_{i=0}^{n-1} (a_i + b_i) \cdot 2^i = \sum_{i=0}^{n-1} (c_i \cdot 2 + s_i) \cdot 2^i = \sum_{i=0}^{n-1} (c_i \cdot 2^{i+1} + s_i \cdot 2^i) =$$

$$= \sum_{i=0}^{n-1} c_i \cdot 2^{i+1} + \sum_{i=0}^{n-1} s_i \cdot 2^i = \sum_{i=0}^{n-1} r_i \cdot 2^i$$

$$\text{PRIMER OPERADOR DE SUMA: } r = \sum_{i=0}^{n-1} r_i \cdot 2^{i+1} + \sum_{i=0}^{n-1} s_i \cdot 2^i$$

La ecuación tiene 2 sumandos: un sumatorio representando un vector de bits de suma s_i y un sumatorio representando un vector de bits de acarreo c_i .

Acarreo entrante C_{-1}

Si incluimos un acarreo entrante C_{-1} , que puede o no ser nulo, podemos simplificar la expresión anterior representando el operador de suma así:

$$r = C_{-1} \cdot 2^n + \sum_{j=0}^{n-1} C_{j-1} \cdot 2^j + \sum_{i=0}^{n-1} s_i \cdot 2^i$$

La ecuación tiene 3 sumandos: un sumatorio representando un vector de bits de suma s_i , un sumatorio representando un vector de bits de acarreo provenientes del peso anterior C_{j-1} (incluyendo el acarreo de entrada C_{-1}) y un acarreo de salida C_{n-1} .

Agrupando por posos en un único sumatorio nos queda otra expresión para el sumador:

$$r = C_{n-1} \cdot 2^n + \sum_{i=0}^{n-1} (c_{i-1} + s_i) \cdot 2^i$$

La suma de los bits s_i y c_{i-1} la podemos escribir así:

$$c_{i-1} + s_i = c_i^* \cdot 2 + s_i^*$$

Ahora bien, ¿qué es $(c_{i-1} + s_i)$?

Recordemos que estamos sumando los dígitos de "a" y de "b". Para el peso i -ésimo tenemos:

$$c_{i-1} + b_{i-1} =$$

$$c_i + b_i = \boxed{c_i \cdot 2^{i+1}} + \boxed{\frac{c_{i-1} + s_i \cdot 2^{i+1}}{2}}$$

$$c_{i-1} \cdot 2^i + s_{i-1} \cdot 2^{i-1} + \boxed{(c_{i-1} + s_i) \bmod 2 \cdot 2^i}$$

La suma del bit s_i con el bit de acarreo precedente c_{i-1} genera un bit de suma s_i^* más un acarreo al siguiente poso que debe sumarse al provocado previamente por cómputo de s_i para dar el nuevo acarreo c_i^* .

Teniendo en cuenta todas las contribuciones nos quedará:

El dígito s_i^* representa la suma en el orden i -ésimo.

$$s_i^* = (c_{i-1} + s_i) \bmod 2 = (c_{i-1} + c_i^* + b_i) \bmod 2$$

Y el dígito c_i^* representa el posible acarreo al orden superior:

$$c_i^* = \frac{(c_{i-1} + s_i)}{2} + c_i = \frac{c_{i-1} + (a_i + b_i) \bmod 2}{2} + \frac{a_i + b_i}{2}$$

El acarreo c_i^* se produce en la suma $a_i + b_i = s_i$ o en la suma $C_{i-1} + s_i$ pero nunca en ambas.

Gráficamente podemos presentarlo así:

$$\begin{array}{cccc}
 & i+1 & i & i-1 \\
 \hline
 a_{i+1} + b_{i+1} + C_{i-2}^* = & & & \\
 a_i + b_i + C_{i-1}^* = & & & \\
 & C_i^* \cdot 2^{i+1} + & C_{i-1}^* \cdot 2^i + & C_{i-2}^* \cdot 2^{i-1} \\
 & S_{i+1} \cdot 2^{i+1} & S_i^* \cdot 2^i & S_{i-1}^* \cdot 2^{i-1} \\
 & & & r_i \cdot 2^i \\
 \hline
 a_i + b_i + C_{i-1}^* = C_i^* \cdot 2 + s_i^* & & &
 \end{array}$$

La implementación de esta ecuación se conoce como sumador completo.

Sumador completo

La ecuación $C_{i-1}^* + a_i + b_i = C_i^* \cdot 2 + s_i^*$ es la implementación conocida como sumador completo, es decir, un sumador de dos operandos de 1 bit (a_i y b_i) más un bit de acarreo (C_{i-1}^*) como entrada y un bit de suma s_i^* y otro acarreo al peso siguiente C_i^* como salida. La tabla de verdad es:

a_i	b_i	C_{i-1}^*	s_i^*	C_i^*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$s_i^* = a_i \oplus b_i \oplus C_{i-1}^*$$

$$C_i^* = a_i \cdot b_i + (a_i \oplus b_i) \cdot C_{i-1}^*$$

Expresiones para la suma

* Operador de suma basado en suma completa:

$$r = C_{n-1}^* \cdot 2^n + \sum_{i=0}^{n-1} (C_{i-1}^* + s_i^*) \cdot 2^i$$

* Operador de suma basado en semisuma:

$$r = \sum_{i=0}^{n-1} c_i \cdot 2^{i+1} + \sum_{i=0}^{n-1} s_i \cdot 2^i$$

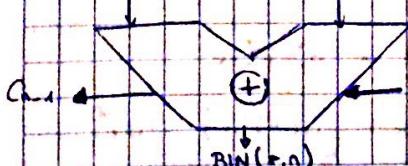
Sumador en BIN_n

BIN(a, n)

BIN(b, n)

• El acarreo de salida C_{n-1} permite determinar condiciones de desbordamiento, conectar operadores en cascada y corregir resultados en complemento restringido a la base.

• El acarreo de entrada C_0 permite conectar operadores en cascada, corregir resultados en complemento restringido a la base y efectuar restas.



- En operaciones de suma el acarreo de entrada es nulo $C_0 = 0$.
- Si no hay acarreo de salida $C_{n-1} = 0$ significa que el resultado está en el rango de representación mientras que $C_{n-1} = 1$ significa que se ha producido desbordamiento.
- En definitiva, podemos definir el sumador binario así:

* Sumador binario } Sean $a, b \in \text{rango } (\text{BIN}_n)$ y $r = a + b$
 $\text{BIN}(a,n) + \text{BIN}(b,n) = \text{BIN}(r,n)$ des = C_{n-1}

Definición de la resta en BIN_n

Sean $a, b \in \text{rango } (\text{BIN}_n)$ y $r = a - b$

$$\text{BIN}(a,n) - \text{BIN}(b,n) = \begin{cases} \text{BIN}(r,n) \text{ des} = '0' \\ \text{si } a > b \Rightarrow r \in \text{rango } (\text{BIN}_n) \\ \text{BIN}((r+2^n), n) \text{ des} = '1' \\ \text{si } a < b \Rightarrow r \notin \text{rango } (\text{BIN}_n) \end{cases}$$

- La definición impone que la resta de la representación de b de a de a debe producir la representación de la resta $r = a - b$ si pertenece al rango o la representación de un valor congruente módulo 2^n en caso contrario.
- La resta en el conjunto origen (representaciones) debe ser consistente en el conjunto destino (rango de valores).

◦ ACARREOS EN RESTA

En la operación de resta también aparecen los acarreos. El tratamiento de los acarreos viene a ser paralelo al de los acarreos en los sumadores. Tendremos una ecuación que modela un semirestador:

$$a_i - b_i = p_i \cdot (-2) + d_i$$

Y también otra que modela un restador completo:

$$d_i - p_{i-1} = p_i^* \cdot (-2) + d_i^*$$

Restadores lógicos

La implementación conocida como restador completo responde a la ecuación:

$$a_i - b_i - p_{i-1}^* = p_i^* \cdot (-2) + d_i^*$$

a_i	b_i	d_i	p_i
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$d_i = a_i \oplus b_i$$

$$p_i = \overline{a_i} \cdot b_i$$

La implementación conocida como restador completo responde a la ecuación:

$$a_i - b_i - p_{i-1}^* = p_i^* \cdot (-2) + d_i^*$$

a_i	b_i	p_{i-1}^*	d_i^*	p_i^*
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$d_i^* = a_i \oplus b_i \oplus p_{i-1}^*$$

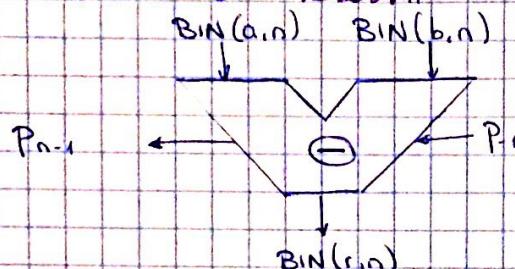
$$p_i^* = \overline{a_i} \cdot b_i + (\overline{a_i} \oplus b_i) \cdot p_{i-1}^*$$

• Expresiones para la resta

* Operador de resta basado en resta completa $\rightarrow r = p_{n-1} \cdot (-2^n) + \sum_{i=0}^{n-1} (d_i \cdot -c_{i+1}) \cdot 2^i$

* Operador resta basado en semiresta $\rightarrow r = \sum_{i=0}^{n-1} p_i \cdot (-2^{i+1}) + \sum_{i=0}^{n-1} d_i \cdot 2^i$

• Restador en BZ_N



- RESTADOR BINARIO

Sean $a, b \in \text{rango } (BIN_n)$ y $r = a - b$

$$BIN(a,n) - BIN(b,n) = BIN(r,n) \text{ des } = p_{n-1}$$

• Podemos diseñar restadores basados en los complementos a la base o restringido a la base.

• De esta forma no necesitaremos un operador de suma y otro de resta sino un único sumador.

• El uso de un sumador tanto en operaciones de suma como de resta hace más baratas las implementaciones y simplifica los diseños.

$$\begin{aligned} \text{En C2: } & BIN(a,n) - BIN(b,n) = BIN(a,n) + BIN(2^n, n) - BIN(b,n) = \\ & = BIN(a,n) + BIN(2^n - b, n) \end{aligned}$$

$$\begin{aligned} \text{En C1: } & BIN(a,n) - BIN(b,n) = BIN(a,n) + BIN(2^n - 1, n) - BIN(b,n) = \\ & = BIN(a,n) + BIN(2^n - 1 - b, n). \end{aligned}$$

• RESTADOR A PARTIR DE COMPLEMENTO

