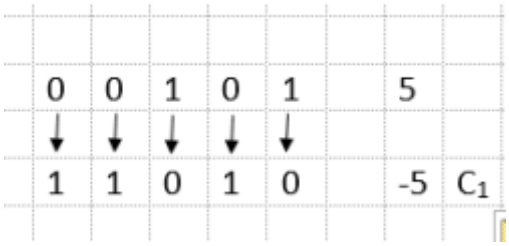
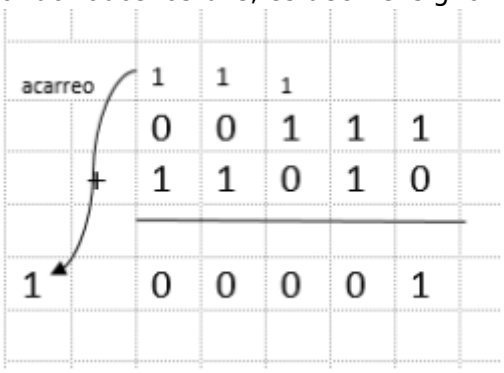
Operaciones aritméticas: Complemento a 1

Suma C1

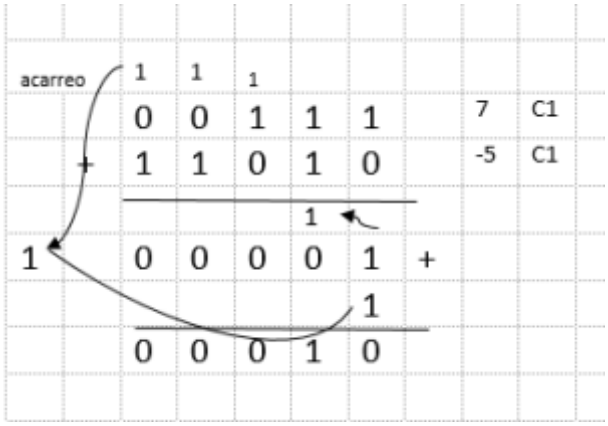
La suma es simple, la hacemos como binario puro. En el siguiente ejemplo debemos sumar en **C_1** 5 bits, por ejemplo 7-5_{10}. 7 es positivo asi que en C_1 queda exactamente igual que en SM, **00111**. Ahora bien, en una computadora no se resuelve una resta sino que se cambia el segundo valor a negativo para poder hacer una suma. Entonces queda 7+(-5) que es equivalente y tiene que dar el mismo resultado. Pasemos entonces -5 a C_1



Y hacemos la suma (iojo!) Estamos trabajando en 5 bits, el resultado debe dar en 5 bits. Vamos a resolver la suma tomando todos los bits, es decir el signo también se debe sumar.



Vemos que se va propagando un acarreo hasta el ultimo bit, o mejor dicho el más significativo. Si el carry se propaga más allá del tamaño de nuestra representación se produce una condición llamada “end-around carry” ¿Ok, entonces qué pasa con el acarreo? En C_1 **si el bit mas significativo produce acarreo**, este vuelve a entrar sumando al bit menos significativo:



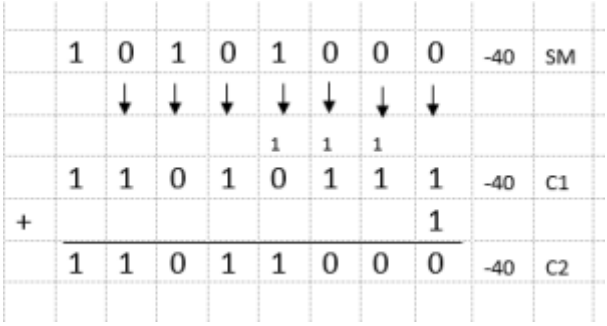
El resultado es `00010`, el bit 4 tiene el valor 0, quiere decir que es positivo, entonces resolvimos la suma en C1, pero al ser positivo, la representación es la misma para SM, y podemos decir directamente cuanto vale:

00010 C_1 —> $2_{(10)}$

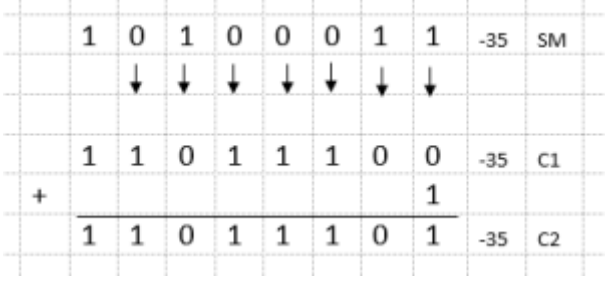
Suma C2

La operación suma en complemento a 2 es muy similar a complemento a 1, pero veamos algunos detalles a tener en cuenta con ejemplos. Hagamos la operación **-40-35**$_{(10)}$ en C_2, en este caso ambos números son negativos y el resultado debe quedar negativo

Paso 0) Pasamos ambos números a C_2



Sino recordas como pasar a complemento a 2 consulta acá



Paso 1) Bien, ahora que tenemos ambos números en C_2, hacemos la suma. Al igual que en C1 el bit de signo también debe sumarse.

		1		1	1					
		1	1	0	1	1	0	0	0	-40	C2
1	+									
		1	1	0	1	1	1	0	1	-35	C2
		1	0	1	1	0	1	0	1	-75	C2

Y que sucede con el acarreo, en C_2 cuando el bit más significativo produce acarreo a diferencia de C_1, este se descarta. Notemos que el resultado obtenido es un valor negativo. No podemos darnos cuenta a a primera vista cual fue el resultado obtenido (el valor).

¿Cómo podemos corroborar que el resultado está bien? **¡Aplicando C_2 nuevamente!**

	1	0	1	1	0	1	0	1	-75	C2
	↓	↓	↓	↓	↓	↓	↓			
	1	1	0	0	1	0	1	0		C1
+								1		
	1	1	0	0	1	0	1	1	-75	sm

Overflow o desbordamiento: Se produce un desbordamiento de enteros cuando una operación aritmética intenta crear un valor numérico que está fuera del rango que puede representarse con un número dado de dígitos, ya sea mayor que el máximo o menor que el mínimo valor representable.

Reglas para detectar el desbordamiento en la suma del complemento a dos:

- Si la suma de dos números positivos produce un resultado negativo, overflow.
- Si la suma de dos números negativos arroja un resultado positivo, overflow.
- De lo contrario, la suma no se ha desbordado.

`Otra forma de detectarlo:` Se puede detectar si hay desbordamiento si el acarreo que recibe el bit de signo es distinto que el acarreo que produce.

```
¿Qué pasa si sumamos 127+1  en C2 8 bits?
01111111 + 00000001 = 127 + 1 ?
```

Volver

— *Mariano Vargas* 🤣

---

(150)