Lógica Computacional

Aritmética binaria

Aritmética binaria - Suma

Para sumar dos (o más) números en sistema binario seguimos el mismo procedimiento que para sistema decimal, teniendo en cuenta que:

$1_2 + 1_2 = 10_2 \rightarrow$ el resultado tiene un bit más que los sumandos.

```
Ej.    0101        1010        0110        0110
   +   1000    +   0010    +   1100    +   0111
       1101        1100       10010       1101
```

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Suma

- Si las operaciones se realizan con una cantidad fija de bits, hay que considerar que el resultado puede ser incorrecto:

```
   1
   0110
+  1100
  10010   ←
```

> Se necesitan 5 bits para expresar el resultado correcto.
> Si se trabaja con 4 bits, el resultado es 0010 => incorrecto
> Si se trabaja con una cantidad de bits fija, y carry = 1 indica que el resultado es incorrecto



Carry / Acarreo

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Resta

Para restar dos (o más) números en sistema binario seguimos el mismo procedimiento que para sistema decimal, teniendo en cuenta que:

$0_2 - 1_2 = 1_2$ => se tuvo que “prestar” (borrow) un 1_2 => $10_2 - 1_2$

```
Ej.    1001          0101
     -  0100        -  1000
        0101         11101
```

Borrow → si es 1, indica que el resultado no se puede expresar con la cantidad de bits usada (4 bits en este caso)

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Multiplicación

Ej. $1010_2 \times 101_2$

```
    1010
x    101
------
    1010
   0000
  1010
------
  110010
```

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> Para representar números negativos usamos el signo - antes del número.

y para positivos el + -39_{10} $-A4F_{16}$ $+1011_{2}$ -10110110_{2} $+9C8_{16}$

> Pero en una computadora, el signo + o - se debe representar con un 1 o un 0

> Hay varios métodos para representar números con signo:

- Signo + módulo
- Complemento a 1
- Complemento a 2

Usan una **cantidad de bits fija**

Diego Codevilla - CC-BY-SA 4.0

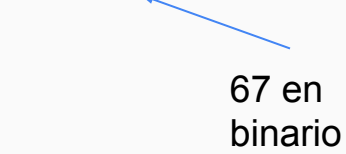
Aritmética binaria - Números con signo

> **Signo + Módulo**: Se usa un bit para representar el signo, y uno o más para representar el módulo del número.

> Si el bit de signo es 1 representa - y si es 0 representa +

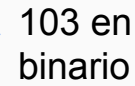
Ej. Si uso 8 bits, y quiero representar -67 en signo+módulo: 11000011





+103 en signo + módulo con 8 bits: 0110111





Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> Con **Signo + Módulo** se pueden representar números entre $-(2^{n-1}-1)$ y $2^{n-1}-1$ (n es la cantidad de bits)

> Ej. Con 4 bits se puede representar entre $-(2^{4-1}-1)$ y $2^{4-1}-1$ =>
entre $-(2^3-1)$ y 2^3-1=> entre -7 y +7

> Hay dos representaciones de “cero” : 0000 y 1000 (ej. con 4 bits)

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Complemento a 2:** Para obtener la representación de un número en CA2:

> Si es positivo, es igual que en signo+módulo.

> Si es negativo, se obtiene su representación de la siguiente forma:

1° → escribir el número como positivo (como signo + módulo)

2° → complementarlo (cambiar los ceros por unos y unos por ceros)

3° → sumarle 1

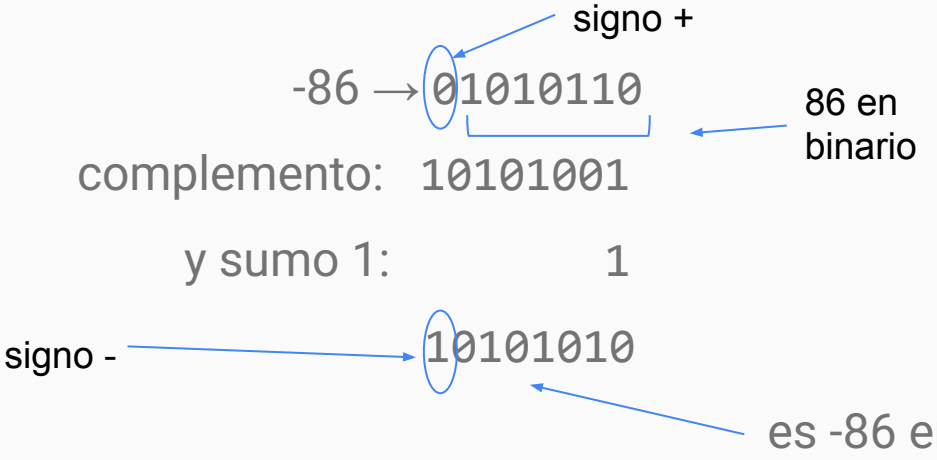
Comple-mento a 1

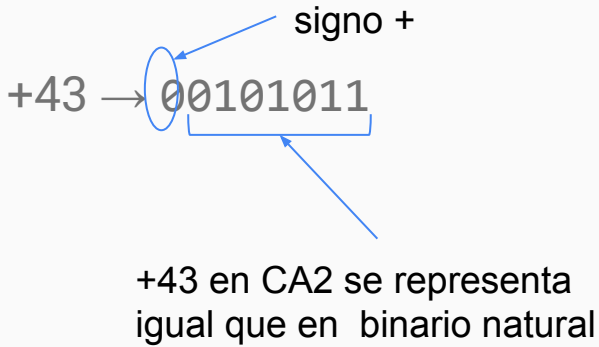
Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Complemento a 2:**

Ej.: Obtener la representación en CA2 de 8 bits de los siguientes números:



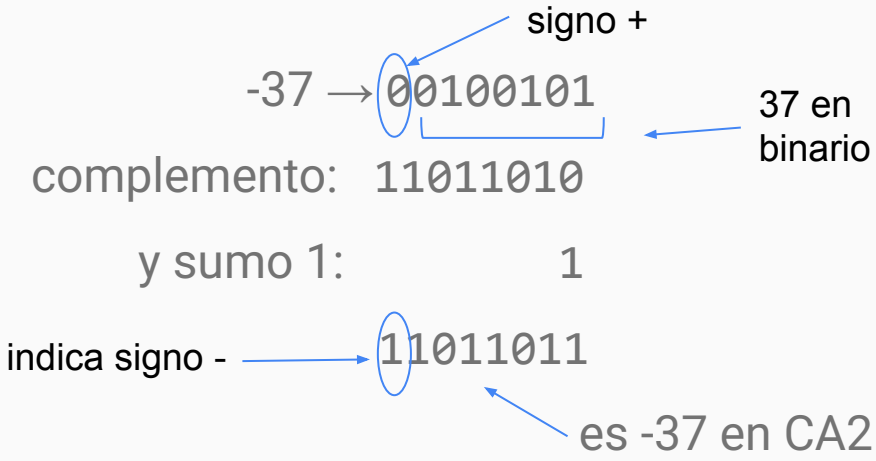


Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Complemento a 2:**

Ej.: Obtener la representación en CA2 de 8 bits de los siguientes números:



-6 → 00000110

complemento: 11111001

y sumo 1: 1

11111010

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> Con **Complemento a 2** se pueden representar números entre $-(2^{n-1})$ y $2^{n-1} -1$ (n es la cantidad de bits)

> Ej. Con 4 bits se puede representar entre $-(2^{4-1})$ y $2^{4-1} - 1$ =>
entre $-(2^3 - 1)$ y $2^3 - 1$=> entre -8 y +7

> Con este sistema hay UNA representación de “cero”. (ej. con 4 bits: 0000)

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Suma y Resta en Complemento a 2:**

- La suma se realiza como se mostró en este documento (Carry no se considera)
- La resta A - B se puede expresar también como A + (-B)
- Para sumar o restar números en CA2, se suma.
- Si se necesita restar, se suma el número negativo expresado en CA2.

ej. $5_{10} - 3_{10} = 0101_2 - 0011_2 = 0101_2 + (-0011_2) = 0101_2 + 1101_2$

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Suma y Resta en Complemento a 2:**

ej. $5_{10} - 3_{10} = 0101_2 - 0011_2 = 0101_2 + (-0011_2) = 0101_2 + 1101_2$

Carry = 1 PERO…
en CA2 NO se toma
como indicador de
resultado incorrecto

$$\begin{array}{r} 0101_2 \\ 1101_2 \\ \hline \cancel{1}0010_2 \end{array}$$

El resultado: 0010_2 ($+010_2$)

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Suma y Resta en Complemento a 2:**

ej. $-87_{10} -20_{10} = -01010111_2 + (-00010100_2) =$

Es un número negativo,

expresado en CA2...



```
 10101001
 11101100
---------
1 10010101
```

¿A qué numero corresponde en binario natural?

¿Qué número es en decimal?

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> Para obtener el valor absoluto (sin signo) de un número negativo expresado en CA2 se realiza el mismo procedimiento visto : se complementa y se suma 1.

Volviendo al resultado del ej. anterior: ~~1~~10010101

El carry se descarta → 10010101

Se complementa → 01101010

Se suma 1 → 1

01101011 → -01101011 es el resultado de -87 - 20

En decimal... (64 + 32 + 8 + 2 + 1) = -107

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Suma y Resta en Complemento a 2:**

ej. $-6_{10} + 2_{10} = -0110_2 + 0010_2 = (-0110_2) + 0010_2 = 1010_2 + 0010_2$

El resultado es un número negativo expresado en CA2

$$\begin{array}{r} 1010 \\ 0010 \\ \hline 1100 \end{array}$$

$$\begin{array}{r} 1100 \\ \cdots\cdots\cdots \\ 0011 \\ \quad\;\; 1 \\ \hline -0100 \end{array}$$

En binario natural es...

Diego Codevilla - CC-BY-SA 4.0

Aritmética binaria - Números con signo

> **Suma y Resta en Complemento a 2:**

6 + 4 =>

```
 0110
 0100
------
 1010
```

¡La suma de dos + es - !

-5 - 7 =>

```
 1011
 1001
------
10100
```

¡La suma de dos - es +!

- Cuando el resultado de sumar dos - es +, o el resultado de sumar dos + es - ...
  - Esta condición se llama **Overflow** (**V**) Indica que no podemos expresar un número en CA2 con la cantidad de bits con la que estamos trabajando.
    - En CA2 no se toma en cuenta Carry o Borrow, solo Overflow

Diego Codevilla - CC-BY-SA 4.0

Ejercitación

Encontrar la representación en CA2 de los siguientes números decimales con 4 bits y 8 bits. Si alguno no se puede representar con 8 bits indicar cuántos bits son necesarios como mínimo y realizar el ejercicio con esa cantidad de bits.

a. +15
b. -78
c. -67
d. +5
e. -4
f. -193
g. +222
h. -101
i. +1973
j. -2023

Diego Codevilla - CC-BY-SA 4.0

Ejercitación - Soluciones

	4 bits	8 bits	
+15	X	00001111	
-78	X	10110010	
-67	X	10111101	
+5	0101	00000101	
-4	1100	11111100	
-193	X	X	100111111 (9 bits) = $13F_{16}$
+222	X	X	011011110 (9 bits) = $0DE_{16}$
+1973	X	X	011110110101 (12 bits) = $7B5_{16}$
-2023	X	X	100000011001 (12 bits) = 819_{16}

Diego Codevilla - CC-BY-SA 4.0

Ejercitación

Hacer las siguientes operaciones en CA2 (8 bits), e indicar si hay Overflow.

a. 110101 + 11001
b. 101110 + 100101
c. 11011101 + 1100011
d. 1110010 + 1101101
e. $A5_{16} + 78_{16}$
f. $56_{10} + 100_{10}$
g. $121_{10} + 12_{10}$

h. 10100010 - 01010101
i. 10010010 - 11001100
j. $A5_{16} \cdot C_{16}$

- si no se especifica la base, se asume binario.
- si los números no están en binario, convertirlos y obtener el resultado en binario, y luego expresarlo en la base original. Ej. en el (e) pasar a binario, sumar, y volver a convertir el resultado a hexadecimal.

Diego Codevilla - CC-BY-SA 4.0

Ejercitación - Soluciones

a. 1001110
b. 1010011
c. 101000000
d. 11011111
e. $11D_{16}$
f. 156_{10}
g. 133_{10}
h. 1001101
i. 111000110
j. $7BC_{16}$

- si no se especifica la base, se asume binario.
- c / e / i / j -> no se pueden representar correctamente con 8 bits

Diego Codevilla - CC-BY-SA 4.0

Ejercitación

Hallar cuál es el mínimo y el máximo que se puede representar en binario sin signo, en signo y módulo y en complemento a 2 con las siguientes cantidades de bits:

a. 4
b. 8
c. 16
d. 32

Diego Codevilla - CC-BY-SA 4.0

Ejercitación - Soluciones

bits	binario sin signo		signo+módulo		CA2	
	mín	máx	mín	máx	mín	máx
4	0	+15	-7	+7	-8	+7
8	0	+255	-127	+127	-128	+127
16	0	+65535	-32767	+32767	-32768	+32767
32	0	+4294967295	-2147483647	+2147483647	-2147483648	+2147483647

Diego Codevilla - CC-BY-SA 4.0