SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Representation of Integers

Prof. Daniele Gorla

Representing Integers

Differently from naturals, we have to represent the sign
There exists (at least) 3 possible representtions:

- with **modulo and sign**
- In **one's omplement**
- In **two's complement**.

The first two methods make operations harder to do (we need preliminary checks on the sign and absolute values of the operands)

By contrast, the third method allows for an immediate procedure for the sum and a very easy way to perform the difference (sum of the opposite), provided to ignore a possible overflow when working with negative numbers).

2

Two's Complement Representation

The sequence of digits $c_{n-1}\ldots c_1c_0$ in the base complement notation (base b) is the integer N given by the following expression:

$$-c_{n-1}b^{n-1} + \sum_{i=0}^{n-2} c_i b^i, c_i \in \{0,\ldots,b-1\}$$

OBS.: in the base's complement representation it is *fundamental* to know the exact length of the codeword
Ex.: 1101 as a 2-compl number in 4 bits is $-2^3+2^2+1 = -3$
as a 2-compl number in 5 bits is $2^3+2^2+1 = 13$

OBS: the most signifying digit is a sign indicator:

- If it is 0, the number is non-negative (we only sum non-negative quantities);
- otherwise, the number is negative $\left(b^{n-1} > \sum_{i=0}^{n-2} c_i b^i\right)$

3

Example

We work in 2-compl with 8 bits

Consider 11111101

By using the previous formula, we have:

$$\begin{aligned} 11111101 &= -2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^0 \\ &= -128+64+32+16+8+4+1 \\ &= -128+125 = -3 \end{aligned}$$

4

Representability Interval in 2-Compl

SAPIENZA Università di Roma Dipartimento di Informatica

The biggest number has its first bit at 0 and all the other ones at 1
The smallest number has its first bit at 1 and all the other ones at 0

8 bits (2-compl)

- $01111111 = 2^7 - 1 = +127$
- $10000000 = -2^7 = -128$

In general, with n bits (2-compl)

$0\underbrace{111\ldots1}_{n-1} = 2^{n-1} - 1$ $\qquad$ $1\underbrace{000\ldots0}_{n-1} = -2^{n-1}$

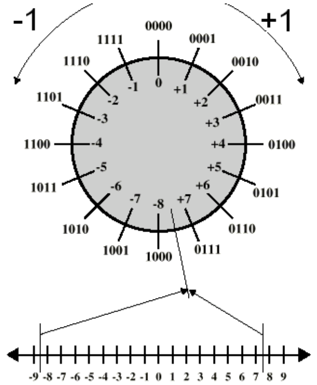
OBS.: the representability interval is NOT symmetric, in the sense that 10…0 has no opposite

→ usually, 10...0 is discarded and the interval is

$\{-2^{n-1}+1,\ldots, 2^{n-1}-1\}$

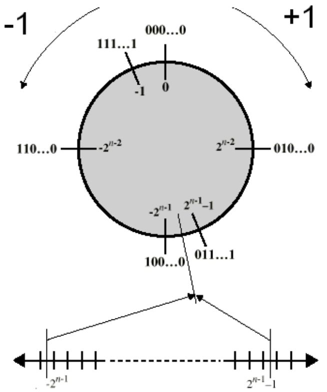
Geometric disposition of the 4 bits 2-Complement

SAPIENZA Università di Roma Dipartimento di Informatica



Geometric disposition of the *n* bits 2-complement

SAPIENZA Università di Roma Dipartimento di Informatica



How to find the opposite

SAPIENZA Università di Roma Dipartimento di Informatica

Given N in 2-compl, its opposite can be found by

- **Perfoming a bitwise complement of *N***
- **Summing 1** to the obtained number

Examples (with a 4 bits 2-complement):

the opposite of 1101 (= $-2^3+2^2+1 = -3$) is
0010+1 = 0011 (=$2^1+1=3$)

the opposite of 0101 (= $2^2+1 = 5$) is
1010+1 = 1011 (=$-2^3+2^1+1=-5$)

Proof

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Let $N = c_{n-1}c_{n-2}\ldots c_1c_0$ and $N' = \overline{c_{n-1}}\,\overline{c_{n-2}}\ldots\overline{c_1}\,\overline{c_0}+1$

We now show that N and N' are opposite, i.e. $N + N' = 0$

$$N+N' = \left(-c_{n-1}b^{n-1}+\sum_{i=0}^{n-2}c_ib^i\right)+\left(-\overline{c_{n-1}}b^{n-1}+\sum_{i=0}^{n-2}\overline{c_i}b^i+1\right)$$

$$= -\left(c_{n-1}+\overline{c_{n-1}}\right)b^{n-1}+\sum_{i=0}^{n-2}\left(c_i+\overline{c_i}\right)b^i+1$$

$$= -b^{n-1}+\sum_{i=0}^{n-2}b^i+1 = -b^{n-1}+\left(b^{n-1}-1\right)+1 = 0$$

9

Converting integers from base 10

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Turn an integer N from base 10 to base 2 in 2-compl with n bits

If $N \geq 0$, use the method of iterated divisions

- If less than n bits are needed, then the number can be represented and we shall add 0's at the beginning, up-to n bits
- Otherwise the number cannot be represented in the chosen format

If $N < 0$, use the method of iterated divisions to $-N$

- If less than n bits are needed, the number can be represented
  - Add 0's at the beginning, up-to n bits
  - Return the opposite of the obtained number
- Otherwise the number cannot be represented in the chosen format

10

Examples

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Format: 2-compl with 4 bits
Represent:

9: 9 is coded as 1001, that requires 4 bits $\rightarrow$ NOT REPR.

5: 5 is coded as 101, hence in 4 bits 2-compl it is 0101

-3: 3 is coded as 11, that in 4 bits 2-compl is 0011. Its opposite is 1100+1=1101

-9: 9 is coded as 1001, that requires 4 bits $\rightarrow$ NOT REPR.

OBS.: -8 can be represented as 1000, ma its opposite would be 0111+1=1000. Hence, typically -8 is considered as NOT representable in this format.

11

Artithmetics in 2-Complement

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

The sum is the same as for the naturals
(the only difference are the overflow conditions)

The subtraction *m-s* is done as the sum between *m* and the opposite of *s* (i.e., *m-s* = *m*+(-*s*))

Multiplication and division are done accordingly

12

Sum in 2-complement

Let us work in 2-compl with 8 bits and do

6+8 6-8 -6+8 -6-8

6 is represented as 00000110

-6 is represented as 11111001+1=11111010

8 is represented as 00001000

-8 is represented as 11110111+1=11111000

There is a final carry!!

6+8	6-8	-6+8	-6-8
00000110+	00000110+	11111010+	11111010+
00001000=	11111000=	00001000=	11111000=
00001110	11111110	00000010	11110010
(14_{10})	(-2_{10})	(2_{10})	(-14_{10})

But the result is representable

Overflow conditions in 2-compl

Hence, overflow in 2-compl is not the carry after summing the MSBs (as it was for naturals)

Let us consider 2-compl with 4 bits and calculate

- 7+2: 0111+0010=1001 (i.e., -7)
- -7-2: 1001+1110=0111 (i.e., 7)

→ ***Condition: operands with the same sign and result with the other sign***

(OBS.: the sign is given by the MBS)

Furthermore, if we forbid the sequence 1000 (since its opposite is not representable), then every operation that yields this result is an overflow.