COMPUTER ARCHITECTURE
2019 - 2020

TD n°1 - Correction

Exercise 1:

1. Give the decimal value of the following integers, the base in which these integers are coded being specified.
   (a) 1011011 and 101010 in binary form (base 2)
   $1011011_2 = 91_{10}$, $101010_2 = 42_{10}$
   (b) A1BE and C4F3 in hexadecimal (base 16)
   $A1BE_{16} = 41\ 406_{10}$, $C4F3_{16} = 50\ 419_{10}$
   (c) 77210 and 31337 in octal (base 8)
   $77210_8 = 32\ 392_{10}$, $31337_8 = 13\ 023_{10}$

2. How many positive integers can be encoded in binary on one byte?

   One byte contains 8 bits, so $2^8 = 256$ integers can be encoded.

   How many bits are needed to represent 65,563 different integers in binary?

   With **b** bits, we can code 2^b different integers. To encode **n** integers, we need **m** bits such as :
   $2^{m-1} < n \leq 2^m$, i.e. $m-1 < \log_2 n \leq m$. So we have $m = [\log_2 n]$.
   For $n = 65\ 563$, we have $m = [\log_2 65\ 563] = 17$.

3. Let us take a computer whose memory words are composed of 32 bits. This computer has 4 MB of memory.
   Since an integer is encoded on a word, how many words can this computer memorize simultaneously?

   4 MB = 4×2^{20} bytes, a word is composed of 4 bytes.
   This computer can therefore store $(4 \times 2^{20}) / 4 = 2^{20} = 1\ 048\ 576$ words.

What is the largest integer (decimal) value that this computer can store, this value being represented by its pure binary coding? Give an order of magnitude of the number of digits in decimal code.

The memory contains 4 x 2^{20} bytes, i.e. 4 x 2^{20} x 8 = 33 554 432 bits. The largest integer value that this computer can store is therefore $2^{33\,554\,432} - 1$.
The number of decimal digits is:

$$\left\lceil \log_{10}\left(2^{32\times 2^{20}}-1\right)\right\rceil \simeq 2^{20}\log_{10}2^{32} \simeq 10^{6}\times 3{,}2\times \log_{10}2^{10} \simeq 10^{7}.$$

The exact number of decimal digits is 10 100 891.

4. Indicate the value encoded by the 16-bit word 1101 1001 0111 0101 depending on whether it represents an unsigned integer, or a signed integer.

In unsigned, the value is $1101\ 1001\ 0111\ 0101_2 = 55\ 669_{10}$.
In signed, the first bit (sign bit) is 1, so it is a negative number whose value is $-(0)101\ 1001\ 0111\ 0101_2 = -22\ 901_{10}$.

5. Code the information - 78_{10} in the following formats:
   (a) 8-bit signed value 11001110
   (b) Complement to 2 on 8 bits 10110010
   (c) IEEE 754 single precision
   1 10000101 00111000000000000000000 or $(C29C0000)_{16}$

Exercise 2:

We are trying to determine the cases of overflow in a signed addition. Overflow occurs when the result obtained (limited to the authorized size) is not correct.

Perform the following operations in 8-bit binary, in using the complement to 2 representation:

- 100 + 100, overflow
- (-1) + (-2),
- (-1) + 16,
- (-100) + (-100). overflow

In which cases is there an overflow?

Here are the calculations:

	100	:	0	1	1	0	0	1	0	0
+	100	:	0	1	1	0	0	1	0	0
		:	1	1	0	0	1	0	0	0 → −56

	−1	:	1	1	1	1	1	1	1	1
+	−2	:	1	1	1	1	1	1	1	0
		:	1	1	1	1	1	1	0	1 → −3

	−1	:	1	1	1	1	1	1	1	1
+	16	:	0	0	0	1	0	0	0	0
		:	0	0	0	0	1	1	1	1 → 15

	−100	:	1	0	0	1	1	1	0	0
+	−100	:	1	0	0	1	1	1	0	0
		:	0	0	1	1	1	0	0	0 → 56