GEK1536, Computation and Machine, Tutorial 9

(For week 12 starting 27 March 06)

1. Computers mostly use 32-bit two's complement to represent signed integers. For simplicity, we use 8-bit two's complement numbers in this exercise instead (it actually exists as a "char" in the programming language C). Work out the 8-bit two's complement representation of the following integers:
   (a) 1, (b) –2, (c) 123, (d) –27, (e) 226.
   What will happen to the last value (e)?

2. Consider arithmetic of the binary numbers worked out in problem 1. Calculate in binary bits the following:
   (a) 1 + (–2), (b) 123 – 27, (c) 123 + 123
   (d) (–2) × (–27), (e) 123 ÷ 27 (give quotient and remainder).
   What will happen to (c)? Is two's complement representation convenient for multiplication and division?

3. The floating-point numbers (numbers with decimal point) in computers are presented according to the IEEE 754 standard. In single precision, it uses 1 bit for sign (the 31-th bit), 8 bits for the biased exponent ($E=e+127$), and 23 bits for the fractional part xxxx… of the binary number 1.xxxx…. $\times 2^e$ (the leading 1 is assumed, but not represented explicitly in the bit pattern). What values do the following bit patterns represent? These are given in hexadecimal (base-16) notation.
   (a) "0 0 0 0 0 0 0 0"
   (b) "3 F 8 0 0 0 0 0",
   (c) "4 1 2 4 0 0 0 0",
   (d) "3 E AA AAAB".

Home Work (*hand in the following week tutorial*)

4. (**Homework**) What do the following 32-bit patterns represent, if they are interpreted as (a) unsigned integer, (b) signed integer, (c) IEEE floating point number? [That is, each pattern can have three possible interpretations]

   (I) 0000 0000 1000 0000 0000 0000 0000 0000
   (II) 1111 1111 1111 1111 1111 1111 1111 1110
   (III) 0000 0000 0000 0000 0000 0000 0000 0000

5. (**Homework**) Give the bit patterns for the following floating point numbers in 32-bit single precision IEEE format:
   (a) 1.0, (b) 128.0, (c) 0.30.