Inf2C Computer Systems

Tutorial 1, Week 3

1. **Two's complement.** What decimal number does the two's complement binary number

   `1111 1111 1111 1111 1111 1111 1110 0101`

   represent?

2. **Number representation and addition.** Using 8 bits represent the numbers $+13$ and -4, using both 2's complement and sign-magnitude binary representation. Perform the common binary addition of the numbers, in both representations. What are the results? Are they correct? Based on the above results, comment on the advantages and disadvantages of two forms of representation.

3. **Floating point representation.**

   (a) Compute the equivalent normalised binary number and the corresponding IEEE 754 32-bit representation for the decimal value of 61.

   (b) What decimal number is represented by this single precision float?
   `1 10000001 00100000000000000000000`.

4. **Overflow.** Using only 8 bits, perform the arithmetic operation: $115 + 33$ using 2's complement. What is wrong with the result? Discuss what are the conditions that can cause overflow when adding or subtracting two 2's complement numbers. Without looking at the values of the operands, is there a way of detecting overflow while performing the addition? Does it work for both positive and negative overflow? (Hint: consider the carries at the 2 most significant bit positions.)

5. **Bit-masking and hexadecimal notation.** Bit masking is an operation used for extracting specific bits from a binary word.

   (a) Perform the following operation and give the result as a hex number: `0x5e` AND `0x30` (AND is the bitwise logical "and" operation, *i.e.*, result is 1 only if both bits are 1.)

   (b) How can you set the fifth bit (from the right hand side) of a binary word w?

   (c) How can you clear the third bit (from the right hand side) of a binary word w?

   (d) How can you flip (regardless of its value) the eighth bit (from the right hand side) of a binary word w?