CS211: Assignment 1

Machine Representations

Abstract: For this project you will emulate in software a set of bit representation and handling schemes that computers use in hardware.

Due Date: Code is due before 5pm, 7 February, 2007.

Submission: Email your source code to your TA as an attachment.

Input: Your code will take a single decimal number as input.
The number may be a float or an integer.
The number may be positive or negative.

Output: Your code will output the:
1. ASCII character representation of the input as big and little endian
2. 8-bit binary one’s complement representation of the input as big and little endian
3. 8-bit binary two’s complement representation of the input as big and little endian
4. 8-bit signed binary fixed-point representation of the input as big and little endian with the decimal fixed before the most significant bit and with a 3-bit signed exponent. The fixed-point number and the exponent should both use the leftmost bit to indicate sign.
   e.g.: the bit sequence 10001101 would mean:
   1 .0001 1 01
   sign number sign exponent
   The leftmost bit should be the most significant in all your output.

Additional Specifications:

1. You may write your code in C, C++ or Java, however it must compile and run correctly on either romulus or remus. Code that does not compile or run correctly on either machine will be regarded as non-functional.
2. You may not use print formatting to perform your translations. For example, the use of printf with formats that output automatically the correct binary representation is not allowed. All transformations should be mapping bits to bits. If you have questions about the methods you use in your code ask your TA before submitting it.
3. Be sure to label each translation that your code outputs (e.g. Big Endian One’s Complement: 10011010). Bit strings outputted without a label (e.g. 10011010) will be presumed to be incorrect.
4. Be sure to catch error conditions and to output them appropriately. If your code
can not represent the number exactly due to low precision or overflow be sure to output the closest your code can get as well as the error condition (e.g. Big Endian One's Complement: 00000000 - overflow).

5. Your work must be your own. No group work is permitted. Please be sure to familiarize yourself with Rutgers’ guidelines on academic integrity.

Additional Resources:

provides an excellent tutorial and description of number conversions and representations. Please refer to this page before emailing/asking questions.