These questions should be done by hand calculations. You can use some types of hand-held or computer calculators to check your answers. You can also check your answers using WolframAlpha on the web (http://www.wolframalpha.com/) by entering, for example, 101110_2 *as* `101110_2` *(using the underscore character), and WolframAlpha will respond with decimal, hexadecimal and other equivalents. You can also do modular arithmetic and conversions between bases (e.g. "*`convert octal 35 to hexadecimal`*") etc. there.*

1. Write the following numbers in base 10.

(a) 101110_2

(b) $12\mathrm{C}_{16}$

(c) 127_8

2. Compute the following directly (without any change of base):

(a) $1011_2 + 1010_2$

(b) $1011_2 \times 1100_2$

(c) $A2_{16} + B3_{16}$

(d) $1A_{16} \times 23_{16}$

3. Write the following in base 2.

(a) 151_{10}

(b) BAD_{16}

(c) 747_8

4. Recall that to represent negative integers in computer words a special convention is used. For 8-bit words, any negative integer x in the range $-2^7 = -128 \leq x \leq -1$, can be represented. To represent say $x = -25$:

1) write $|x|$ in 8-bit binary (pad with leading zeroes if necessary): 00011001;
2) toggle all the bits ($0 \to 1$, $1 \to 0$): 11100110;
3) add 1 to the resulting binary number: 11100111.

Provided x is in the stated range, the representation will automatically have 1 as the left-most bit. So if the left-most bit of an 8-bit word is 0 the word is interpreted as a binary representation of an integer x in the range $0 \leq x \leq 127 = 2^7 - 1$.

Using this representation:

(a) Write -123_{10} as an 8-bit word.

(b) Write -128_{10} as an 8-bit word.

(c) Evaluate `11001100`
(i.e. express it in ordinary decimal notation)

(d) Evaluate `01111000`
(i.e. express it in ordinary decimal notation)

5. Write the following numbers (from $\mathbb{Q}$, in base 10) as 8-bit words, with sign bit s followed by 5 bits of mantissa ("5 bit precision") m followed 2 bits of (non-negative) exponent n. For example -6.5 would be written `10110110`.

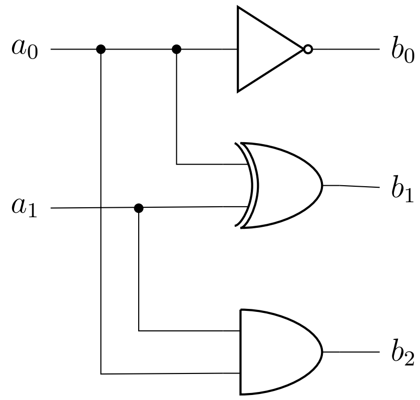
(a) 5.25

(b) -2

(c) 5.3 (requires approximating)

(d) -15.5

6. From the top down the circuit at right has a NOT gate, an XOR gate and an AND gate. The input is the integer x represented in binary as $(a_1a_0)_2$ and the output is the integer $y = f(x)$ represented in binary by $(b_2b_1b_0)_2$. Find y for each $x \in \{0, 1, 2, 3\}$ and hence determine the purpose of the circuit by finding the rule for the function $f : \{0, \ldots, 3\} \to \{0, \ldots, 7\}$.



7. Carry out the following binary arithmetic operations. Do not change base.

(a) $\begin{array}{r} 00001101 \\ +00000110 \\ \hline \end{array}$ (b) $\begin{array}{r} 00001101 \\ -00000110 \\ \hline \end{array}$ (c) $\begin{array}{r} 00001101 \\ \times 00000110 \\ \hline \end{array}$

8. Calculate the following.

(a) 128 div 5 (b) 128 mod 5 (c) 12365489 mod 2 (d) -20 mod 3

9. Write an algorithm that takes $n \in \mathbb{N}$ as input, and returns "true" as an output if n is prime, and "false" otherwise. Make use of the mod operation.

10. Determine the truth or falsity of the following statements. Give reasons.

(a) $765 \equiv 654 \pmod{37}$ (b) $765 \equiv -654 \pmod{37}$ (c) $-765 \equiv -654 \pmod{37}$

11. Calculate each of the following, using techniques of modular arithmetic.

(a) $13579 \times 24680 \bmod 9$

(b) $13579 \times 24680 \bmod 11$

(c) $13579^8 \bmod 13$

(d) $13579^{24680} \bmod 11$. (Use Fermat's little theorem.)