1) Here’s a couple of shortcuts for converting numbers from binary to decimal.

It’s really fast to convert the number 01100011 to decimal because it’s equal to $3 = (11)_2$ plus $3\times32 = (1100000)_2$ - remember, multiplying by two is equivalent to shifting a number one bit to the left and tacking on a zero at the end.

Another way to convert numbers to decimal is by mentally grouping the digits into 4-bit nibbles, and multiplying by powers of 16. This is roughly the same as converting binary to hexadecimal, and subsequently to decimal. For instance, $(01011001)_2 = 16\times5 + 9$, because the binary number 0101 (decimal 5) is shifted four bits to the left (multiply by 16), and added to the binary number 1001 (decimal 9).

Finally, if an $n$-bit binary number has just one or two zeros in it, it might be faster to work by starting with $2^{n-1} - 1$, and subtracting off the zero elements. For example $(11101100)_2 = 127 - 8 = 119$.

Use your newfound powers of awesomeness to convert these binary numbers to decimal. For each number, indicate which shortcut you used.

a. 01110111
b. 01010101
c. 01110001

2) Here’s a couple of shortcuts when working with negative 2’s complement numbers.

We know that negating a binary number means flipping all the bits, and then adding one. So if we see a negative 2’s complement number (i.e. the MSB is equal to 1), then we can quickly convert it to decimal by adding up all the powers of two corresponding to the zeros, and then adding one. For example, in eight-bit 2’s complement:

$$(11101100)_2 \Rightarrow -(2^4 + 2^1 + 2^0 + 1) = -(16 + 2 + 1 + 1) = -20$$

Another way to quickly convert 2’s complement numbers to decimal is to remember the number wheel and its periodic nature. We can get the same result by converting an $n$-bit 2’s complement number to binary as if it were just a regular binary number, and then subtracting $2^n$. For example:

$$(10000000)_2 \Rightarrow 128 - 256 = -128$$
$$(11111111)_2 \Rightarrow 255 - 256 = -1$$
$$(11011110)_2 \Rightarrow 222 - 256 = -34$$
Note that these shortcuts only apply to negative 2’s complement numbers. For positive numbers, you can just convert normally.

Use all of the shortcuts above to convert these numbers in eight-bit 2’s complement to decimal. Indicate which shortcuts you use.

a. 10010010
b. 10111110
c. 10000011
d. 11100111

3) Convert these decimal numbers to their eight-bit 2’s complement representations, and then add them together. If the computation results in overflow, indicate so in your answer. Convert the eight-bit result back to decimal, even if overflow occurs.

a. 58 + (-22)
b. 64 + 83
c. (-118) + (-6)
d. (-98) + (-53)
e. 12 + 81

4) For the function

\[ F(w, x, y, z) = \Sigma(0, 1, 4, 5, 7, 10, 11, 13, 14, 15) \]

a. Simplify the function using a K-map to form a sum of products.

b. Implement the SOP using NAND gates alone. Draw the full gate diagram, making sure to implement all the inverters with NAND gates.

5) For the function

\[ F(a, b, c, d) = \Sigma(3, 6, 7, 9, 12, 13) \]

a. Simplify the function using a K-map to form a product of sums.

b. Implement the POS using NOR gates alone. Draw the full gate diagram, making sure to implement all the inverters with NOR gates.
6) For the function

\[ F(A, B, C, D) = \sum\{0, 1, 2, 3, 4, 6, 8, 9, 10, 11, 12, 14\} \]

a. Simplify the function using a K-map to form a product of sums or a sum of products.

b. Would it be more efficient to implement this function entirely with NAND gates, or entirely with NOR gates? Justify your answer by indicating how many of each type of gate is required to implement the function.

7) Let’s look at long division in binary.

a. Perform the decimal long division to compute 1/99 out to six places past the decimal point.

b. Write out a few sentences explaining the steps of the decimal long division you performed above.

c. Now, following the same steps (but with a reduced set of digits), compute the binary long division \((1/3)_{10} = (1/11)_2\) out to six places. 
*Hint:* if you get stuck on subtraction, remember that in binary, \(100 - 011 = 001\).

d. Convert the truncated answer you got back into a decimal fraction by adding up the negative powers of two corresponding to each digit.

e. Compute the error of the binary long division by subtracting the fraction above from 1/3. How far off was the answer?

8) A rational number is any number that can be expressed as \(p/q\) for some integers \(p\) and \(q\). Any such number is in canonical form if \(p\) and \(q\) share no common divisors. Note that some rational numbers such as \(3/4\) have a terminating expression in binary (there are no infinitely repeating digits), while others such as \(1/3\) do not terminate.

a. Given a rational number \(p/q\) in canonical form, what must be true of the integers \(p\) and/or \(q\) in order to have a terminating representation in binary?

b. By the way, what’s the expression of \(3/4\) in binary? What about \(3/2\)? What about \(3\)? How does this reinforce your answer above?
9) On a recent trip to a lab in the technologically backwards nation of Lower Slobbovia, Bob the digital circuit designer was shocked to find out that the lab didn't stock traditional AND and OR gates. Instead, all he found were NOT gates, along with a new gate he had never seen before: the SCHMAND gate.

Rifling through the lab's scant documentation, Bob was able to ascertain the mathematical notion used to denote SCHMAND, as well as a truth table for the gate and the symbol used in gate diagrams:

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>x ⊻ y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Notice that in general, $x \bowtie y \neq y \bowtie x$.

a. Does the lab have sufficient gates to implement any possible Boolean function as a circuit? If so, show that the set \{ NOT, SCHMAND \} is logically complete by implementing AND and OR gates using only NOT and SCHMAND gates.

b. Bob needs help implementing the function

$$F(a,b,c) = a'b' + bc'$$

Draw a gate diagram for the function using only NOT and SCHMAND gates, using the fewest possible gates.