For all problems with gate diagrams, use the minimum number of gates possible. Clearly label your inputs and outputs in all diagrams, and mark the MSB and LSB of input and output where appropriate.

1) Each of these combinatorial circuits takes a three-bit binary number as input and produces a single binary output. Draw a gate diagram for each circuit.
   a. The output is 1 when the value of the input is less than three.
   b. The output is 1 when the value of the input is odd.
   c. The output is 1 when a majority of input bits are 1.

2) Implement your circuit from (1a) using only:
   a. NAND gates
   b. NOR gates

3) Each of these combinatorial circuits takes a three-bit binary number as input and produces a three-bit binary number as output. Draw a gate diagram for each circuit.
   a. When the input is less than four, the output is equal to two greater than the input. If the input is greater than or equal to four, the output is three less than the input.
   b. The output returns the input bits sorted from least to greatest. For example: input 000 generates output 000, 010 and 100 both generate 001, and 101 and 110 both generate 011.

4) Table 1.6 in the textbook (p. 24) shows the 4-bit gray codes. Draw a gate diagram implementing a combinatorial circuit for converting 4-bit binary numbers to the corresponding Gray code. You should be able to implement the circuit using XOR gates alone.

5) An incrementer is a combinatorial circuit which takes a binary number as input and outputs the number plus one. A decrementer outputs the number minus one. Using block diagrams,
   a. Construct a 4-bit binary incrementer using four half adders. It’s ok to assign constant inputs (0 or 1) to one or more of your adders.
   b. Construct a 4-bit binary decrementer using four full adders.
6) Most decoders have an enable line which suppresses the output when it is set to zero. The truth table for a 2-to-4 decoder with enable is given here:

<table>
<thead>
<tr>
<th>EN</th>
<th>I1</th>
<th>I0</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

a. Draw a gate diagram for the 2-to-4 decoder with enable.

b. Implement the circuit using only NAND gates.

c. Implement the circuit using only NOR gates.

d. Make a block diagram showing how to construct a 3-to-8 decoder using two 2-to-4 decoders with enable, along with a NOT gate.

e. Show how to construct a 4-to-16 decoder using five 2-to-4 decoders with enable. You shouldn’t need any extra gates.

7) Design a circuit to implement a seven-segment display by OR-ing together the outputs of a 4-to-16 decoder. Let the inputs to the circuit be a 4-bit BCD value. The segments should be all off when the input is an invalid decimal digit. Label the outputs of your circuit according to the diagram to the right.

8) Construct a 16×1 multiplexer using two 8×1 multiplexers and one 2×1 multiplexer.

9) Some arithmetic logic units implement a count leading zeros (CLZ) function that takes a non-zero 2^n-bit number as input, and outputs the number of consecutive 0’s from the MSB until the first 1.

a. Draw a truth table for a 4-bit CLZ circuit. You can use X’s in the input rows to cover multiple values.

b. Which of the combinatorial logic modules we have discussed in class would be best to implement the CLZ function? Draw a block diagram with any additional gates needed to implement CLZ.

c. Why is the output of the CLZ function undefined when the input is zero?