Problem 1:
a) Convert the following truth table to a Boolean expression and simplify it. An X means we don’t care about the value of that output (it can be either 0 or 1).

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
</tbody>
</table>

The trick is to set the output of the last 2 rows to 1. Then we get:

!ABC + A!B!C + AB!C + ABC

So we can group terms 1 and 4 and terms 2 and 3:

=BC(!A + A) + A!C(!B + B)
=BC + A!C

This solution uses 4 gates (2 AND, 1 OR, 1 NOT).

b) Draw the transition state diagram from a FSM that reads a binary string bit-by-bit and outputs whether the total number of 1s seen since the beginning is divisible by 3.

State = (number of 1s seen) % 3

Note: a valid FSM needs an arrow pointing to the start state.
c) For the circuit below, assume that the setup time is 15ns, hold time is 30ns, and the AND gate delay is 10ns. If the clock rate is 10 MHz and x updates 25ns after the rising edge of the clock, what are the minimum and maximum values for the clk-to-Q delay to ensure proper functionality?

Min: 20 ns
Max: 75 ns

If the clk-to-Q delay is too fast, the input to the register will change before the hold time is finished. Thus, the minimum clk-to-Q delay is \( t_{hold} - t_{AND} = 30 - 10 = 20 \)ns.

On the other hand, we must make sure the critical path is no longer than the clock period, which is 100 ns (= 1/(10 MHz)). In other words, \( t_{setup} + t_{AND} + t_{clk-to-Q} \leq 100 \)ns, or \( t_{clk-to-Q} \leq 100ns - t_{setup} - t_{AND} \). Solving yields \( t_{clk-to-Q} \leq 75 \) ns.

d) Build a 3-bit majority circuit (outputs 1 if 2 or more bits are 1 otherwise outputs 0).

e) Reduce the Boolean expression \( AB + ABC + ABCD + ABCDE + ABCDEF \) and draw the logic gate.

f) Reduce the Boolean expression \(!((A + !B) * (!C + D + E) + (A + B) * (!C)) \). How many gates needed?

\( A'B'C'D'E + A'C + B'C = B'C^*(!A'*D'E + 1) + A'C = B'C + A'C = (IC)'(B + A) \)

3 gates (1 not, 1 and, 1 or)
Problem 2:
Consider the 4-bit adder shown to the right.
It takes:
- a carry in (cin)
- two four-bit inputs:
  a with bits $a_0$, $a_1$, $a_2$, $a_3$
  b with bits $b_0$, $b_1$, $b_2$, $b_3$
Outputs:
- a carry out (cout)
- one four-bit output:
  s with bits $s_0$, $s_1$, $s_2$, $s_3$

Assume each adder has a delay of 10ns, and any registers have a clk-to-q, hold time, and setup time of 5ns. Assume the inputs are driven by registers, and outputs are registers as well.

Assume each adder has a delay of 10ns, and any registers have a clk-to-q, hold time, and setup time of 5ns. Assume the inputs are driven by registers, and outputs are registers as well.

1. Write Boolean formulas for $s_0$ and $c_1$ in terms of the inputs $c_{in}$, $a_{0}$, and $b_{0}$. You may use XOR as an operator in the Boolean formulas. Each formula should use as few operators as possible.
   
   $s_0 = a_0 \ XOR \ b_0 \ XOR \ c_{in}$
   $c_1 = c_{in} \ XOR \ (a_0 \ XOR \ b_0) + a_0 \ XOR \ b_0$

2. What is the critical path delay of the circuit? Please include proper units in your answer.
   50ns = clk-to-q + 4 adders + setup time

3. What is the maximum clock frequency at which the circuit will function correctly? Please include proper units in your answer.
   20 MHz = 1 / 50 ns

4. What is the maximum hold time the output registers could have at which the circuit would still function correctly?
   15 ns
   The result for $s_0$ arrives in 15ns, so if it was greater, the hold time would be violated by the second set of inputs
Problem 3:

Using only NAND gates build AND, NOT, and XOR

[Diagram of NAND gate configurations for AND, NOT, OR, XOR functions]