1. Consider the circuit shown below. Assume that all gates have a minimum propagation delay (for both rising and falling edges) of 1 ns, and a maximum propagation delay of 2 ns. Construct two timing diagrams showing how all the labeled signals change when input $A$ changes from 1 to 0, while inputs $B$ and $C$ are held at 1 and 0, respectively. In the first timing diagram, assume the minimum propagation time for all gates and in the other, assume the maximum propagation time for all gates. Draw a third timing diagram which combines the first two, using shading to show the time periods where signal values could be either 0 or 1.
2. Analyze the two circuits shown below. Specifically, create a truth table showing how the outputs depend on the inputs, and give simplified Boolean expressions for each of the outputs.

\[ X = AB + B' + C' = A + B' + C' \]
\[ Y = X' C = A' BC \]

\[ S = ((AB)' + B(AB)'')' = (A' + B' + A'B + BB')' = AB \]
\[ T = ((B(AB)')(BC)')' = ((A'B(B' + C'))') = A + B' + C \]

<table>
<thead>
<tr>
<th>ABC</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>010</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>110</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ABC</th>
<th>S</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>101</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>110</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
3. Show how to implement the function $F = AB + AB'C$ using a 3→8 decoder and two additional gates (use only 2 input gates).
4. Design circuits that implement the logic in the truth table shown below. Design two circuits, one which has the smallest possible worst-case propagation delay, and a second which minimizes the gate count. Use only inverters and 2-input AND gates and 2-input OR gates. Assume each inverter has a propagation delay of 1 ns and each AND and OR gate has a 2 ns propagation delay. When counting gates, treat an inverter as a “half-gate.” What are the gate counts and worst-case propagation delays for your circuits?

\[
\begin{array}{c|c}
ABCD & Z \\
0000 & 0 \\
0001 & 0 \\
0010 & 0 \\
0011 & 1 \\
0100 & 0 \\
0101 & 1 \\
0110 & 0 \\
0111 & 0 \\
1000 & 0 \\
1001 & 0 \\
1010 & 0 \\
1011 & 1 \\
1100 & 0 \\
1101 & 1 \\
1110 & 0 \\
1111 & 1 \\
\end{array}
\]

\[
Z = ACD + B'CD + B'CD' = ((A+B')C + BC'D)
\]

\[
\begin{array}{c|c|c|c|c|c}
AB & CD & Z \\
00 & 0 & 0 & 1 & 0 \\
00 & 1 & 0 & 0 & 0 \\
01 & 0 & 1 & 1 & 0 \\
01 & 1 & 0 & 1 & 0 \\
10 & 0 & 0 & 1 & 0 \\
10 & 1 & 0 & 1 & 0 \\
11 & 0 & 0 & 1 & 0 \\
11 & 1 & 0 & 1 & 0 \\
\end{array}
\]

The first circuit shown corresponds directly to the minimum sum-of-products form of the logic expression and the second to a factorization of the sum-of-products. The first has a gate count of 9 and a worst-case propagation delay of 8 ns, while the second has a gate count of 6 and a worst-case propagation delay.
propagation delay of 9 ns. However, if we cover the zeros in the K-map to obtain the product-of-sums form, we get a simpler expression, which results in the last circuit shown. This has a gate count of 6 and a worst-case propagation delay of 7 ns, so it is both smallest and fastest.

5. Show how to implement a circuit for each of the following expressions using a multiplexor (as on page 3-16 of the notes).

(a) \((A + B)C' + B'D\)

(b) \(A'C + (B'D + C)(A' + D)\)
6. The equations below are the basis for the ripple-carry adder circuit (see page 3-18 of the notes).

\[ S = X \oplus Y \oplus C_{in}, \quad C_{out} = XY + XC_{in} + YC_{in} = XY + (X + Y)C_{in} \]

There is a similar set of equations that can be used to create a ripple-carry subtractor circuit.

\[ D = X \oplus Y \oplus B_{in}, \quad B_{out} = X'Y + X'B_{in} + YB_{in} = X'Y + (X' + Y)B_{in} \]

Draw a logic diagram for a 4 bit subtractor, based on these equations.
7. An $n$ bit barrel shifter has two inputs, a data input $x$ with $n$ bits, a control input $i$ with $\log_2 n$ bits and produces an output $y$ with $n$ bits. The output $y$, contains the same bits as $x$, but rotated to the right by $i$ bit positions. If $x = x_{n-1} \ldots x_1 x_0$ then $y = x_i x_{i-1} \ldots x_0 x_{n-1} \ldots x_{i+1} x_i$. Design a circuit that implements a barrel shifter using $n$ multiplexors with $n$ inputs each and draw a 4 bit version of this circuit. How many simple gates does this design require? How many simple gates does the $n$ bit version require?

The 4 bit version is shown above. The enables are not really necessary. A 4 bit multiplexor without an enable can be implemented with a gate count of 12, so the shifter will have a gate count of 64. An $n$ bit multiplexor has a gate count of about $n + n \log_2 n$ so a barrel shifter constructed using $n$ bit multiplexors would have a gate count of about $n(n + n \log_2 n)$. 
Design an alternative barrel shifter circuit that uses multiple 2-input multiplexors. Draw the 4 bit version of this circuit. How many simple gates does this design require? How many simple gates does the \( n \) bit version of this circuit require?

This design uses 2 input multiplexors. These have a gate count of 3.5 gates each, giving an overall gate count of 28. An \( n \) bit version uses \( n \log_2 n \) multiplexors, giving it a gate count of 3.5(\( n \log_2 n \)).