1) Sequential logic implementation

Consider the following incompletely-specified Moore machine ("X" indicates don’t cares).

\[
\begin{array}{c|c|c|c|c|c}
S_1S_0 & XY = 00 & 01 & 10 & 11 & F \\
\hline
A & B & C & X & B & 0 \\
B & A & B & X & C & X \\
C & C & A & X & C & 1 \\
\end{array}
\]

Suppose we use the following state encoding.

\[
\begin{array}{c|c}
\text{State} & S_1S_0 \\
\hline
A & 00 \\
B & 11 \\
C & 01 \\
\end{array}
\]

Derive the \textbf{minimum} two-level logic for each of the following (you should minimize each logic function separately, no need to worry about sharing logic between logic functions).

\[
F = \\
S_1^+ = \\
S_0^+ =
\]
2) Reverse engineer sequential circuit to finite state machine

![Circuit Diagram]

a) Fill out the following functions (OK to use $\oplus$ in functions).

\[
\begin{align*}
F &= \\
S_1^+ &= \\
S_0^+ &= 
\end{align*}
\]

b) Assume initial state is 00, fill in the following state table. If a state is not reachable, indicate the appropriate don’t cares (use “X” to indicate don’t cares).

<table>
<thead>
<tr>
<th>$S_1S_0$</th>
<th>$S_1^+S_0^+$</th>
<th>$AB = 00$</th>
<th>01</th>
<th>10</th>
<th>11</th>
<th>$F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


c) With initial state = 00, what states, if any, are not reachable?
Given the following inputs \((L, R, X_3X_2X_1X_0)\) and flip-flop values \((Q_3Q_2Q_1Q_0)\), what will be the new flip-flop values \((Q'_3Q'_2Q'_1Q'_0)\)?

<table>
<thead>
<tr>
<th>(LR)</th>
<th>(X_3X_2X_1X_0)</th>
<th>(Q_3Q_2Q_1Q_0)</th>
<th>(Q'_3Q'_2Q'_1Q'_0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0100</td>
<td>0011</td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>0100</td>
<td>0011</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0100</td>
<td>0011</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0100</td>
<td>0011</td>
<td></td>
</tr>
</tbody>
</table>
4) Timing analysis

Consider the following sequential logic diagram.

Assume the gate delay for 2-AND is 2 ns, assume the positive edge-trigged flip-flop delay is $T_{delay_{FF}} = 1$ ns, and assume the positive edge-trigged flip-flop setup time is also $T_{setup_{FF}} = 1$ ns.

Fill out the rest of the timing diagram for A, B, C, and D.
5) Clock period analysis

Consider the following sequential logic diagram where \(X_3, X_2, X_1, X_0, \) and \(Y_3, Y_2, Y_1, Y_0\) are \textbf{external inputs}, and \(F_3, F_2, F_1, F_0\) are \textbf{external outputs}.

Assume the gate delay for 2-XOR is 3 \textbf{ns}, assume the gate delay for INVERTER is 1 \textbf{ns}, assume the positive edge-triggered flip-flop delay is \(T_{\text{delay\text{}}FF} = 1 \text{ ns}\), and assume the positive edge-triggered flip-flop setup time is also \(T_{\text{setup\text{}}FF} = 1 \text{ ns}\).

What is the \textbf{minimum clock period} that would satisfy these delays? Please explain how you computed the clock period (e.g. give the equation).

Clock period =