1.) Produce the minimum Sum of Products equation of the following K-Map. Each of the inputs in your product terms must be in alphabetical order (i.e. AB, not BA).

\[
F = \overline{B} \overline{C} \overline{D} \lor \overline{C} \overline{D} \lor \overline{A} \overline{D}
\]

- or -

\[
\overline{B} \overline{C} \overline{D} + \overline{C} \overline{D} \lor \overline{B} \overline{D}
\]
2.) In lecture we discussed how non-inverting gates (AND, OR, etc.) don’t exist, and we can only build circuits out of inverting gates (NAND, NOR, Invert). However, we never discussed how to build an XOR.

Build a 2-input XOR gate (A xor B) out of only inverting gates. Your circuit should be as efficient as possible.

\[ A \oplus B = A \overline{B} + \overline{A} B \]
3.) For the following circuit, fill out the timing diagram for all gate outputs (C, D, F). The vertical lines in the timing diagram are 1ns apart, and each gate has a delay of 1ns. Also, two copies of the same diagram are given, in case you make a mistake. Make it clear which version you want graded.

Duplicate (if you make a mistake above):
4.) Draw the **state diagram** for the following circuit. Your state diagram should be *as simple as possible*.

An entrance gate at a secure facility requires a user to present a cardkey, and only allows one person to then go through the gate after each cardkey. At startup, the door should be locked.

Your circuit has a C (for CARDKEY) input, that will unlock a locked gate. It has an E (for ENTERED) input, that is true when someone has entered through the unlocked gate. And it has one output L (for LOCKED), that is true when the gate should be locked.

We employ only lazy employees, so C and E will never be true at the same time.

*Your circuit output should change as soon as possible – i.e. react immediately to the current inputs.*
5.) For the following state diagram, implement the circuit. You can use premade DFFs, and any basic gates. Your circuit should be as simple as possible.

The state encoding is Hop = 0, Skip = 1

<table>
<thead>
<tr>
<th>PS</th>
<th>IN</th>
<th>OUT</th>
<th>NS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Hop</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Skip</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ \text{out} = \overline{PS} \]

\[ NS = PS + \overline{IN} \]

\[ \overline{NS} = PS \]

\[ \overline{IN} \]

\[ \overline{DQ} \]

\[ \overline{CLK} \]

\[ \text{out} \]