2-bit binary numbers represent the values 0..3 with the following encoding:

<table>
<thead>
<tr>
<th>Encoding [1:0]</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

Develop the simplest Boolean equation possible that, when given two 2-bit binary numbers \(X[1:0]\) and \(Y[1:0]\), will output whether the value of \(X\) is greater than the value of \(Y\).

\[
F = X_0 \overline{Y_1} \overline{Y_0} + X_1 \overline{Y_1} + X_1 X_0 \overline{Y_0}
\]
Assume that we have already implemented the solution to the previous problem as a module "isGreater(F, X, Y)" in Verilog, and cannot change that module at all. Using that module, write the simplest possible Verilog module that will compute whether A[1:0] is greater than or equal to B[1:0].

// A>B is inverse of B>A

module isGreaterEqual (F, A, B);

output F;
input [1:0] A, B;
wire F;

isGreater ig (.F(Fb), .X(B), .Y(A));
assign F = ~Fb;

endmodule

Note: Could do F computation with an explicitly instantiated NOT gate as well.
The OnesCount of a multi-bit signal is the total number of true signals there are in that multi-bit signal, where the output count is given in the 2-bit binary encoding listed above. Thus, the OnesCount of “000” is “00” (representing the value zero), and the OnesCount of “101” is “10” (representing the value two). Design the simplest Boolean equation possible that will return the OnesCount of a 3-bit input Z[2:0].

\[
\begin{array}{c|c|c|c|c}
A & B & C & \text{OnesCount} \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c}
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 \\
\end{array}
\]

\[
A \oplus B \oplus C = \text{OnesCount} = A B + B C + A C
\]
We have a circuit that hooks a positive-edge triggered DFF (from class) and a negative-edge triggered DFF to the same D input. The negative-edge triggered DFF responds only to the falling (negative) edge of the clock, the instant when the clock goes from true to false. Fill in the following timing diagram for this case. Assume that all gate delays are very tiny compared to the length of the clock period.