Review Problem 9

Assuming all gates have the same delay (including inverters), complete the following timing diagram.
// Verilog code for AND-OR-INVERT gate

module AOI (F, A, B, C, D);
  output logic F;
  input logic A, B, C, D;
  logic        AB, CD, O;

  and a1(AB, A, B);
  and a2(CD, C, D);
  or  o1(O, AB, CD);
  not n1(F, O);
endmodule
Verilog Hierarchy

// Verilog code for 2-input multiplexer

module AOI (F, A, B, C, D);
    output logic F;
    input logic A, B, C, D;

    assign F = ~(A & B) | (C & D);
endmodule

module MUX2 (V, SEL, I, J);  // 2:1 multiplexer
    output logic V;
    input logic SEL, I, J;
    logic SELB, VB;

    not G1 (SELB, SEL);
    AOI G2 (.F(VB), .A(I), .B(SEL), .C(SELB), .D(J));
    not G3 (V, VB);
endmodule
module MUX2TEST; // No ports!
  logic SEL, I, J, V;

initial // Stimulus
begin
  SEL = 1; I = 0; J = 0;
  #10 I = 1;
  #10 SEL = 0;
  #10 J = 1;
end

MUX2 M (.V, .SEL, .I, .J);

initial // Response
$monitor($time, , SEL, I, J, , V);

if (SEL) $v = I
else $v = J

endmodule
int k;

initial begin
for (k = 0; k < 8; k++) begin
  for (i, j, s = 13 = k; #5
end
end
NAND and NOR Gates

- **NAND Gate: NOT(AND(A, B))**

  ![NAND gate diagram]

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X NAND Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- **NOR Gate: NOT(OR(A, B))**

  ![NOR gate diagram]

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X NOR Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Bubble Manipulation

- DeMorgan's Law

\[ \overline{A \cdot B} = \overline{A} + \overline{B} \]

\[ \overline{A + B} = \overline{A} \cdot \overline{B} \]

- Simplification: \[ AB + CD = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} = \overline{AB} \cdot \overline{CD} = \overline{A + B} \cdot \overline{C + D} = \overline{A^2 \cdot B^2 \cdot C^2 \cdot D^2} \]

4-input AND
Review Problem 10

- Write the Verilog for a 2-input gate that is TRUE when an odd number of inputs are true.

```verilog
module odd (F, A, B);
    output logic F;
    input logic A, B;
    assign F = (A & ~B) | (~A & B); // also ^ = xor operator
endmodule
```
NAND and NOR Gate Universality

- NAND and NOR gates are universal
  - can implement all the basic gates (AND, OR, NOT)
Converting Circuits to NAND/NOR Form

- Group gates into levels, insert double inversions on alternating levels

- Alternating AND/OR becomes all NAND or NOR
Converting to NAND/NOR Form (cont.)

- Some circuits may require internal inverters
XOR and XNOR Gates

- **XOR Gate:** \( Z = 1 \) if odd # of inputs are true

<table>
<thead>
<tr>
<th>( X )</th>
<th>( Y )</th>
<th>( Z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- **XNOR Gate:** \( Z = 1 \) if even # of inputs are true

<table>
<thead>
<tr>
<th>( X )</th>
<th>( Y )</th>
<th>( Z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Optimization via K-Maps to 2-level forms

- **Readings:** 2.11-2.12.2, 2.14
- **Sum of Products form:** the OR of several AND gates, inversions over only inputs
  - $F = \overline{X}+Y\overline{Z}+XYZ$
- **Circuit diagram & inversions:**
On Sets and Off Sets

- **On Set** is the set of input patterns where the function is **TRUE**

\[
\bar{X} \bar{Y} \bar{Z}, \bar{X} Y Z, \bar{X} \bar{Y} Z, \bar{X} Y \bar{Z}
\]

- **Off Set** is the set of input patterns where the function is **FALSE**

\[
\bar{X} \bar{Y} Z, X \bar{Y} Z, X \bar{Y} \bar{Z}, X Y \bar{Z}
\]

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>H</th>
<th>( X \bar{Y} Z )</th>
<th>( \bar{X} \bar{Y} Z )</th>
<th>( \bar{X} Y Z )</th>
<th>( \bar{X} \bar{Y} Z )</th>
<th>( \bar{X} Y \bar{Z} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Two-Level Simplification

**Key Tool: The Uniting Theorem —** \( A (\bar{B} + B) = A \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ F = A \bar{B} + A B = A (\bar{B} + B) = A \]

B's values change within the on-set rows

\( B \) is eliminated, \( A \) remains

A's values don't change within the on-set rows

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ G = \bar{A} \bar{B} + A \bar{B} = (\bar{A} + A) \bar{B} = \bar{B} \]

B's values stay the same within the on-set rows

\( A \) is eliminated, \( B \) remains

A's values change within the on-set rows

**Essence of Simplification:**
find two element subsets of the ON-set where only one variable changes its value. This single varying variable can be eliminated!
Karnaugh Maps

Karnaugh Map Method

K-map is an alternative method of representing the truth table that helps visualize adjacencies in up to 4 dimensions.

Beyond that, computer-based methods are needed.

2-variable K-map

3-variable K-map

4-variable K-map