1. (15 points)
   (a) (5 points) NAND, NOR gates
   Simplify the following Boolean function and then implement the simplified result using
   2-level NAND-NAND realization
   \[ F(x, y, z) = (x + y + z)(\overline{x} + \overline{y} + z) \]
   Assume complements of the input Boolean variables are available.
   
   Answer:

   (b) (5 points) NOR gate implementation
   Convert the following logic schematic diagram into NOR-only realization. You may use
   only two-input NOR gates and inverters. Assume the complements of input Boolean
   variables are NOT available.
   
   Answer:
(c) (5 points) **Tabular Method**

Let: \( P(v,w,x,y,z) = \sum m(12,13,14,15,29,31) + \sum d(17,18). \)

Execute the Quine-McCloskey tabulation algorithm to find all the **Prime Implicants** of the function \( P(v,w,x,y,z), \) where the second summation is the *don’t care* minterms. The algorithm has been started for you below (minterms are listed in increasing 1’s count order). Complete the algorithm and **CIRCLE** the PRIME IMPlicants.

\[ \begin{array}{cccccc}
\hline
& & & & & \checkmark \\
\hline
\end{array} \]

Note that there are no three-cubes and that duplicate two-cubes have been eliminated.

2. (10 points) **Combinatorial circuit analysis**

The logic diagram of a combinational logic circuit is given below:

Express the corresponding Boolean function in the **product of Maxterm** format:

**Answer:**

3. (10 Points) **Code Conversion, Adders**

A binary multiplier produces partial products that must be added together. For adding partial products you will build a circuit that adds a column of six binary digits, producing a three bit sum, depending upon how many of the six bits are one. As shown below, the three bit sum \( S_2S_1S_0 = 000 \) for a six binary digit column 000000 (zero ones). A sum \( S_2S_1S_0 = 110 \) results from a six binary digits column 111111 (six ones). A sum \( S_2S_1S_0 = 011 \) results from a six binary digits column 101001 (three ones).
The circuit below starts the process using two Full Adders (FA) connected to ABC and DEF as shown. **Complete the wiring**, (no additional logic is needed) connecting the first level of Full Adders to the second level of Full Adders and Half Adders so that the sum $S_2S_1S_0$ is produced. **Label the sum bits $S_2$, $S_1$, and $S_0$.** (Hint: the first two half-adders produce two, 2-bit sums. Complete the sum by adding their outputs together.)

**Above is one solution. Alternate solutions swap the connections to the X and Y inputs of the HA below and swap any of X, Y, and Ci inputs of the FA below.**

4. (15 Points) **Decoder Implementations**

Two decoders constructed with NAND gates are connected to primary input variables A, B, C and D as shown in the diagram below. The function table for the decoder is shown below:

<table>
<thead>
<tr>
<th>$S_1$</th>
<th>$S_0$</th>
<th>$D_3$</th>
<th>$D_2$</th>
<th>$D_1$</th>
<th>$D_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
(a) (5 points) Write a Boolean equation in the box below for the signal shown as "M" on the
diagram in terms of the primary input signals A, B, C and D.

\[
M = \quad \text{[Box]} \quad \text{(To be filled in)}
\]

(b) (5 points) Using ONE gate (of the required input width) selected from the set of gates
{AND, NOT, OR, NAND, NOR}, draw the gate on the diagram above and connect it
to DECODER OUTPUTS (signals F,G,H,I,J,K,L,or M) to implement:
\[
Q(A,B,C,D) = (A+B+C+D)
\]

(c) (5 points) Using one or more gates (of the required input width) selected from the set of
gates {AND, NOT, OR, NAND, NOR}, draw a logic diagram on the diagram above
that uses DECODER OUTPUTS (signals F,G,H,I,J,K,L,or M) to implement
\[
R(A,B,C,D) = AB' + CD'
\]

5. (15 points) **Combinational circuit analysis**
Find the simplified Boolean equation in **sum of product** format of the output F of the circuit
below. Note that the decoder's outputs are **not** complemented.

Answer:

6. (15 points) **Multiplexer Logic Implementation**
Use the 8-to-1 multiplexer below to implement an exclusive or function for four bits. This is
also known as the "odd" function. The function \( \text{exor}(w,x,y,z) \) is to be:
\[\text{exor}(w, x, y, z) = w \oplus x \oplus y \oplus z\]

Draw the circuit in the space below. You may use only NOT, OR, and AND gates. (Hint: the variable "w" has been factored out).

7. (15 Points) Signed Arithmetic, Complements, Number Representation

Fill in the table below with 8-bit binary numbers in the given signed representation. The first row has been done for you. The decimal numbers (-45) and (-120) are a "negative forty-five" and a "negative one-hundred, twenty" respectively. Convert the decimal numbers to the proper representations, state them as an ADDITION PROBLEM in the representation, and give the correct result in each representation. If the calculation produces an overflow in any of the representations used, note it as "OVERFLOW" and omit the answer.

(Hint: 120 and 45 are 01111000 and 00101101 expressed as 8-bit unsigned binary numbers)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Sign-Magnitude</th>
<th>1's Complement</th>
<th>2's Complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 + 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(-120) + 45</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>120 + 45</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>120 + (-45)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8. (10 points) Binary subtraction using 2's complement, 1's complement, sign-magnitude
The following are two binary addition/subtractions performed in an unknown signed binary number representation. For each of them, determine their possible format (or formats) and whether there is an overflow occurred. Mark your answer in each blank box in the table below.

<table>
<thead>
<tr>
<th></th>
<th>Sign-magnitude</th>
<th>1’s Complement</th>
<th>2’s Complement</th>
<th>Overflow?</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>10110 − 01011 = 01011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(b)</td>
<td>11001 − 10010 = 10111</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(c)</td>
<td>01011 + 10111 = 00011</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

9. (10 Points) **Carry Look-ahead Adders**

Consider an adder with inputs $A = A_3 A_2 A_1 A_0$ and $B = B_3 B_2 B_1 B_0$, $c_0$ and outputs $S = S_3 S_2 S_1 S_0$ and $c_4$. Recall that for the stage $i$ in an adder, the propagate and generate functions are:

\[
P_{i} = A_{i} \oplus B_{i} \\
G_{i} = A_{i} \cdot B_{i}
\]

and the output sum $S(i)$ and carry $C(i+1)$ is defined as:

\[
S_{i} = P_{i} \oplus C_{i} \\
C_{i+1} = G_{i} + P_{i} C_{i}
\]

Let $A = 1110$ and $c_0 = 1$. Write a Boolean expression for the carry, $c_4$ as a function of $B_3$, $B_2$, $B_1$, and $B_0$.

**Answer:**

\[
c_4 =
\]
10. (10 Points) *Decimal Arithmetic*
   (a) (6 points) *BCD decimal digit adder*
   Below is a partial logic diagram of a BCD-based decimal digit adder. The missing part is the one that generates the carry-out to the next digit, and proper inputs to the lower 4-bit binary adder. Use AND, OR, NOT gates to implement this part.

   (b) (4 points) *9's complement and 10's complement*

   Find the 9's complement of the decimal number 73522: [Blank]

   Use 10's complement to perform the subtraction of two unsigned decimal numbers

   \[ 1995 - 2188 = \] [Blank]

   You must show intermediate steps. Answer alone will not receive any credit.