1. (1 point) The output of a two input AND gate is 1 if _both_ of the inputs are 1.

2. (1 point) Definitions of logical operations may be listed in a compact form called _truth table_.

3. (1 point) A _literal_ is a single variable within a term, in complemented or uncomplemented form.

4. (1 point) A _register_ is a group of binary cells.

5. (1 point) For n bits, there are _$2^n$_ distinct combinations of 0s and 1s.

6. (10 points) Convert (2012₃) to decimal:

   $2012₃ = 2 \times 3^3 + 0 \times 3^2 + 1 \times 3^1 + 2 \times 3^0 = 2 \times 27 + 0 \times 9 + 1 \times 3 + 2 \times 1 = 54 + 0 + 3 + 2 = 59_{10}$

7. (5 points) Convert 1110001110001011110101001111111₂ to hexadecimal

   $1110001110001011110101001111111₂ = 1111 0001 1100 0101 1101 0011 1010 0111₂ = F1C5D3A7₁₆$

8. (5 points) We can perform logical operations on strings of bits by considering each pair of corresponding bits separately (called bitwise operation). Given two eight-bit strings $A = 10110001$ and $B = 10101100$, evaluate the eight bit result after a NOR operation.

   \[
   \begin{array}{c}
   A = 1011 0001 \\
   B = 1010 1100 \\
   A \text{ NOR } B = 0100 0010
   \end{array}
   \]

9. (20 points) Convert decimal +25 and +62 to binary, using the signed-2's-complement representation and enough digits to accommodate the numbers. Then perform the binary equivalent of (+25) + (-62). Convert the answer back to decimal and verify that it is correct.

   \[
   \begin{array}{c}
   2  62  \\
   2  31  0 \\
   2  15  1 \\
   2  7   1 \\
   2  3   1 \\
   2  1   1 \\
   0   1 \\
   \hline
   +25 = 001 1001 \\
   +62 = 011 1110 \\
   -62 = -64 + 2 \\
   -62 = 100 0010
   \end{array}
   \]

   \[
   \begin{array}{c}
   2  62  \\
   2  31  0 \\
   2  15  1 \\
   2  7   1 \\
   2  3   1 \\
   2  1   1 \\
   0   1 \\
   \hline
   +25 = 001 1001 \\
   25 - 62 = -37 \\
   +(-85)= 100 0010 \\
   101 1011 = 1 \times -2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\
   = -64 + 0 + 16 + 8 + 0 + 2 + 1 \\
   = -37 \checkmark
   \end{array}
   \]
10. (5 points) Convert \( F(A, B, C, D) = \Pi(0, 1, 5, 7) \) to the other canonical form.

\[
F(A, B, C, D) = \Sigma(2, 3, 4, 6, 8, 9, 10, 11, 12, 13, 14, 15)
\]

11. (10 points) Formulate a weighted binary code for the decimal digits, using weights 5321

<table>
<thead>
<tr>
<th>Digit</th>
<th>5</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>9</td>
</tr>
</tbody>
</table>

12. (10 points) Reduce \( A'C' + ABC + AC' \) to three literals.

\[
A'C' + ABC + AC' = A'C' + AC' + ABC = C'(A' + A) + ABC = C' + ABC
\]

13. (10 points) Find the complement of \( z'(x + (y + z)x') \)

\[
(z'(x + (y + z)x'))' = z + (x + (y + z)x')' = z + (x'(y + z)x')' = z + (x'(y'z' + x)) = z + x'y'z' + xx' = z + x'y'z' = (z + z')(z + x'y') = z + x'y'
\]

14. (20 points) Obtain the truth table of the following function, and express it in sum-of-minterms and product-of-maxterms form: \( F = (x' + y + z')(x + y')(x + z) = \Sigma(1, 4, 6, 7) = \Pi(0, 2, 3, 5) \)

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
<th>x' + y'</th>
<th>x + z</th>
<th>x' + y + z'</th>
<th>(x' + y + z')(x + y')(x + z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</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>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>