HW2 Walkthrough: Digital Logic Problems

Below is a step-by-step walkthrough for each problem in HW2—covering 2's complement conversions, SOP (sum of products) derivations from truth tables, thermometer-to-binary encoding, and finally K-Map simplifications.

1 2's Complement Conversions

You have five rows of data to fill out; each row provides a partial specification (decimal, binary, or hex). You must fill in all four columns:

1. **Base-10** (decimal interpretation, signed for negatives)
2. **Positive Binary** (the unsigned/magnitude binary in 8 bits)
3. **Hexadecimal** (8-bit form, typically prefixed as `0x??`)
4. **2's Complement** (the signed 8-bit representation)

Below is one consistent way to fill in each row.

Important: 2's complement for a *negative* decimal n in 8 bits is computed by:

1. Converting $|n|$ to binary (8 bits).
2. Inverting (flipping) all bits.
3. Adding 1.

For a *positive* decimal, the "2's complement" column is simply the same as the "positive binary" (since its sign bit = 0).

Row 1

Given: Base-10 = -12

Base-10	Positive Binary	Hex	2's Complement
-12	00001100	0x0C	11110100

Explanation:

- $|-12| = 12 \rightarrow$ `00001100` in binary (8 bits).
- Invert `00001100` to get `11110011`, then add 1 to obtain `11110100`.

Row 2

Given: Positive Binary = `00010101` (leading 0 → positive number)

Base-10	Positive Binary	Hex	2's Complement
21	00010101	0x15	00010101

Explanation:

- `00010101` in decimal is $16 + 4 + 1 = 21$.
- Hex for `00010101` is `0x15`.
- Since it is positive, its 2's complement is identical to the positive binary.

Row 3

Given: Hex = `0x84`
Interpreted in 8-bit 2's complement as a *negative* number because the most significant bit (1) is set.

Base-10	Positive Binary	Hex	2's Complement
-124	01111100	0x84	10000100

Explanation:

- `0x84` = `10000100` in binary.
- To find its decimal value as signed: take the 2's complement of `10000100` by inverting to get `01111011` and adding 1 to obtain `01111100` (decimal 124). Hence, the value is -124.
- The "positive binary" (magnitude only) is `01111100`.
- The 2's complement form is the same `10000100` as given.

Row 4

Given: `11110001` (presumably the 2's complement form)

Base-10	Positive Binary	Hex	2's Complement
-15	00001111	0x0F	11110001

Explanation:

- Interpreting `11110001` as signed 8 bits:
  - Invert `11110001` to get `00001110` (14 in decimal) and add 1 to obtain `00001111` (15 in decimal).
- Thus, it represents -15.

Row 5

Given: `00001001` (leading 0 $\rightarrow$ positive number)

Base-10	Positive Binary	Hex	2's Complement
9	00001001	0x09	00001001

Explanation:

- `00001001` is decimal 9.
- Its hexadecimal form is `0x09`.
- Being positive, the 2's complement remains the same.

2 2's Complement Practice Tools

If you need additional practice, many online calculators exist (e.g., Exploring Binary's converter). Just verify the calculator's convention to ensure it computes *negative* two's complement in the same way.

3 Sum-of-Products from Truth Tables

(3a) Two-Input Table

A	B	Out
0	0	0
0	1	1
1	0	0
1	1	1

Observe that Out $= 1$ whenever $B = 1$.

SOP:

$$\text{Out} = B$$

(Alternatively, $\overline{A}B + AB = B$.)

(3b) Three-Input Table

A	B	C	Out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

The output is 1 for the following minterms:

- $A = 0,\ B = 1,\ C = 1 \to \overline{A}BC$
- $A = 1,\ B = 1,\ C = 0 \to AB\overline{C}$
- $A = 1,\ B = 1,\ C = 1 \to ABC$

Sum-of-Products:

$$\text{Out} = \overline{A}BC \ + \ AB\overline{C} \ + \ ABC$$

Sometimes factored:

$$\text{Out} = \overline{A}BC \ + \ AB(\overline{C} + C) = \overline{A}BC \ + \ AB.$$

(3c/d) 3-Bit Thermometer Code $\to$ Binary

Thermometer Code (3 bits) counts:

$$\texttt{000} \to \texttt{001} \to \texttt{011} \to \texttt{111}$$

Interpreting these as 0, 1, 2, 3 in decimal (then presumably wrapping around):

- $\texttt{000}_2 \to 0$ decimal $\to$ output $\texttt{00}_2$
- $\texttt{001}_2 \to 1$ decimal $\to$ output $\texttt{01}_2$
- $\texttt{011}_2 \to 2$ decimal $\to$ output $\texttt{10}_2$
- $\texttt{111}_2 \to 3$ decimal $\to$ output $\texttt{11}_2$

Any other 3-bit combination is a "don't care" (X) for this particular problem. You would build a truth table with inputs T_2, T_1, T_0 (the thermometer bits) and outputs X_1, X_0 (the 2-bit binary). Rows that are not `000`, `001`, `011`, or `111` receive an output of X.

Example Partial Table:

T_2	T_1	T_0	$(X_1\, X_0)$
0	0	0	00
0	0	1	01
0	1	0	X (don't care)
0	1	1	10
1	0	0	X (don't care)
1	0	1	X (don't care)
1	1	0	X (don't care)
1	1	1	11

Then, one K-Map is drawn for X_1 and another for X_0. Because of the "don't cares" in unused rows, the minimal expressions often simplify nicely.

4 K-Map Minimization

K-Maps are used on each of the truth tables given:

1. **(4a)**: Use the same 3-input table from Section 3(b) but now reduce it via K-Maps (you will get the same minimal form, e.g., $B + AC$ or similar).
2. **(4b)**: For the 4-input table with variables A, B, C, D:
   - Fill a 16-cell K-Map (with rows often labeled by AB and columns by CD, or vice versa).
   - Identify the largest prime implicants (circles) to obtain the minimal Sum-of-Products.
3. **(4c)**: The thermometer-code logic from Section 3(c) can also be minimized using K-Maps. Two K-Maps (one for each output bit) are created. Include the don't cares (X) in the map for unused input rows, which usually leads to a simpler final expression.

Summary of Key Points

- **2's Complement Table**: Ensure each final 8-bit pattern is correct; negative numbers should show their magnitude in the Positive Binary column and their actual 2's complement bit pattern in the last column.
- **SOP (Sum of Products)**: List the minterms where the output is 1. Factor if needed, or if the expression simplifies nicely (e.g., $\overline{A}BC + AB$).
- **Thermometer Code**: Only 4 valid states exist; all other states are considered don't care (X). This leads to compact minimal expressions when K-Maps are applied to each output bit.
- **K-Map Minimization**: Group as many adjacent 1-cells (and/or don't cares) as possible to find the minimal SOP form.

This guide should help you through all parts of the assignment:

1. Completing the 2's complement table,
2. Verifying your logic expressions in standard SOP form,
3. Converting thermometer code to binary,
4. Using K-Maps to derive minimal expressions.

Good luck!