1. The following Truth Table defines the function $H(A,B,C,D)$.

A	B	C	D	H(A,B,C,D)
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

a. (7 points) Express $H(A,B,C,D)$ in sum-of-minterm form. That is, write it in the form of $H(A,B,C,D) = \Sigma m(\ldots\ldots)$ where the appropriate minterm numbers are listed inside the parentheses.

$H(A,B,C,D) = \sum m(0,1,4,6,11,12,14)$

b. (7 points) Draw the Karnaugh Map (K-Map) for $H(A,B,C,D)$.

CD \ AB	00	01	11	10
00	1	1	1	
01	1			
11				1
10		1	1	

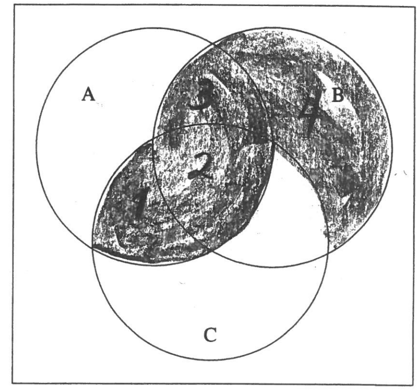
c. (3 points) If the function $H(A,B,C,D)$ was synthesized/implemented in *Canonical* Sum-of-Products (CSoP) form, what would be the cost *(as we have defined it in this class)* of that synthesis/implementation?

$Cost_{CSoP} = m(i+2)+1 = 7(4+2)+1 = 43$

d. (3 points) If the function $H(A,B,C,D)$ was implemented in *Canonical* Product-of-Sums (CPoS) form, what would be the cost *(as we have defined it in this class)* of that synthesis/implementation?

$Cost_{CPoS} = M(i+2)+1 = 9(4+2)+1 = 55$

2. (8 points) Consider the Venn Diagram shown below.



Write a *Sum-of-Products (SoP)* logic expression that this Venn Diagram represents. You *must* provide an explanation for how you arrived at your answer.

Regions 1 & 2 together is the intersection of A & C: AC

Regions 3 & 4 together is " " " B & $\bar{C}$: $B \cdot \bar{C}$

The shaded area is the union of those two areas, so the logic expression is $\boxed{AC + B \cdot \bar{C}}$

Several other SoP answers possible.

3. Consider the following 3-input K-Map for function $J(A, B, C)$.

C \ AB	00	01	11	10
0	0	0	0	1
1	1	0	0	1

a. (5 points) Find a minimum-cost Sum-of-Products (SoP) synthesis/implementation (just a logic expression) for $J(A, B, C)$ *AND* give the cost of that synthesis/implementation.

$J = A \cdot \overline{B} + \overline{B} \cdot C$

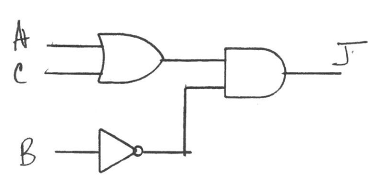
Cost $= 2 \cdot 3$ (2, 2-input ANDs) $+ 3$ (2-input OR) $= 9$

b. (5 points) Find a minimum-cost Product-of-Sums (PoS) synthesis/implementation (just a logic expression) for $J(A, B, C)$ *AND* give the cost of that synthesis/implementation.

$J = \overline{B} \cdot (A + C)$

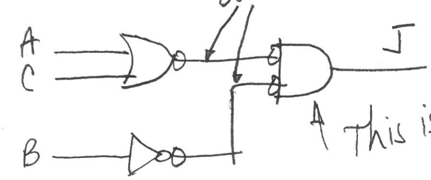
Cost $= 3$ (2-input OR) $+ 3$ (2-input AND) $= 6$.

c. (5 points) Using your logic template, draw the logic network/circuit for the Product-of-Sums (PoS) synthesis/implementation that you found in part (b), using only AND, OR, and NOT gates.



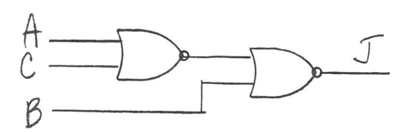
d. (8 points) Using your logic template, draw the logic network/circuit for the Product-of-Sums (PoS) synthesis/implementation that you found in part (b), *using only NOR gates.* Be sure to simplify your network/circuit to use the fewest gates and show *only* gates that are actually needed to produce $J(A, B, C)$. You may do preliminary work freehand (without using your logic template), but your final logic network/circuit must be drawn neatly using your logic template.

Double-invert



This is NOR by De Morgan

Double NOT cancels out



e. (5 points) If your final NOR-only logic network/circuit from the previous part was implemented with *CMOS* gates, how many total *transistors* would be required? You *must* give a brief explanation/calculation for your answer.

NMOS NOR: 2 transistors $\Rightarrow 2 + 2 = 4 = T_{NMOS}$.

$T_{CMOS} = 2 \cdot T_{NMOS} = 2 \cdot 4 = \boxed{8 \text{ transistors}}$

4. Consider the following Karnaugh Map (K-Map) for function $L(A, B, C, D)$, which we will be synthesizing/implementing with a Sum-of-Products (SoP) form. Each d in the K-Map indicates a *don't - care* output.

CD \ AB	00	01	11	10
00	1	0	0	d
01	1	1	1	0
11	1	0	1	d
10	d	d	d	d

a. (7 points) Identify *all* of the Prime Implicants (PIs) for $L(A, B, C, D)$, *writing a logic expression for each one.*

$\bar{A}\cdot\bar{B}$, AC, $\bar{B}\cdot C$, $\bar{B}\cdot\bar{D}$, $\bar{A}\cdot\bar{C}\cdot D$, $B\cdot\bar{C}\cdot D$, ABD

b. (4 points) Identify *all* of the *Essential* Prime Implicants (EPIs) for $L(A, B, C, D)$, *writing a logic expression for each one.*

<u>None</u> is essential: Each 1 included in at least 2 PIs.

c. (9 points) Find a minimum-cost (as we have defined cost in this class) SoP synthesis/implementation for $L(A, B, C, D)$. Your answer should be a logic expression for L, *not* a logic network/circuit. You do not need to provide the cost of your solution.

1) <u>Choose</u> $\bar{A}\cdot\bar{B} \rightarrow$ contributes 3 $L=1$ minterms to cover.

2) <u>Choose</u> $AC \rightarrow$ 2-literal term that contributes 1 more.

3) <u>Choose</u> $B\cdot\bar{C}\cdot D \rightarrow$ contributes the final 2 $L=1$ minterms.

$$\boxed{L = \bar{A}\cdot\bar{B} + A\cdot C + B\cdot\bar{C}\cdot D}$$

5. (12 points) Prove the following Boolean equality in variables A, B, and C using Boolean algebraic manipulation. You may *not* use a Truth Table, K-map, or Venn diagram to prove the equality. You *must* justify each step of your manipulation with the Boolean Algebra property number or numbers used to obtain that step.

$$\bar{A}\cdot B\cdot\bar{C} + B\cdot C + A\cdot B + A\cdot\bar{B}\cdot C = B + A\cdot C$$

LHS: $\bar{A}\cdot B\cdot\bar{C} + BC + AB + A\cdot\bar{B}\cdot C$

12a: $B\cdot(\bar{A}\cdot\bar{C} + C + A) + A\cdot\bar{B}\cdot C$

15b, 10b, 11b: $B\cdot(\overline{(A+C)} + (A+C)) + A\cdot\bar{B}\cdot C$

8b: $B\cdot(1) + A\cdot\bar{B}\cdot C$

6a, 10a: $B + \bar{B}\cdot AC$

16a: $\underline{B + AC}$ $x = B$ $y = AC$.

RHS Done.

Other proofs possible.

6. You are to begin the design of a digital logic circuit with the following characteristics. The circuit has two binary data inputs A and B, plus as many binary control inputs as needed. The binary output G is determined by the control inputs as follows:

For the first combination of control input values: G should be A AND B. You are free to specify the specific combination of control input values (you will do this below).

For the second combination of control input values: G should be A OR B.

For the third combination of control input values: G should be A XOR B, where XOR stands for eXclusive OR.

a. (4 points) Determine the minimum number of control inputs that are required, then *define* each one (that is, give it a name). Finally, *assign* values to the control inputs for the first, second, and third combinations as indicated above.

1 binary control input can only provide 2 conditions/combinations, so need 2 control inputs. Call them C_1 & C_0

1st combination: $C_1=0$ $C_0=0$: $G=A\cdot B$
2nd " : $C_1=0$ $C_0=1$: $G=A+B$
3rd " : $C_1=1$ $C_0=0$: $G=A\oplus B$

b. (8 points) Draw a complete Truth Table for G, showing all possible combinations of all inputs. List the control inputs first in the top row of the Truth Table, followed by data inputs A and B, followed by the output G. If needed, use d for the output for input combinations where the output does not matter.

C_1	C_0	A	B	G	
0	0	0	0	0	
0	0	0	1	0	$A\cdot B$
0	0	1	0	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	$A+B$
0	1	1	0	1	
0	1	1	1	1	
1	0	0	0	0	
1	0	0	1	1	$A\oplus B$
1	0	1	0	1	
1	0	1	1	0	
1	1	0	0	d	
1	1	0	1	d	Not Used.
1	1	1	0	d	
1	1	1	1	d	