K-maps

a	b	c	F	G
0	0	0	0	0
0	0	1	0	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	0

$b'c$

bc'

write canonical sop for $F = a'bc' + a'bc =$
$a'b(c' + c)$
$a'b$

Factoring works because both product terms had
$a'b$ in common but the minterm trick
$a=0$ $b=1$ inputs only differ by $\underline{1}$ bit

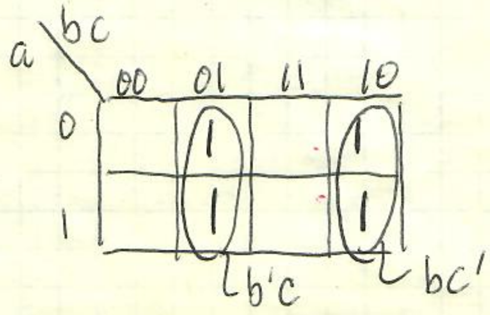
<u>Simplification trick</u> If <u>two minterms</u> have inputs which differ by 1 bit then they can be replaced with a product term formed from the variables which do not change.

Write simplified form for $G = b'c + bc'$

Let's rearrange our truth table to make finding inputs which differ by 1 bit. easier.

A Karnaugh map or K-map is a truth table arranged so that inputs which differ by 1-bit are physically adjacent.

Ex: G



$A\backslash BC$	00	01	11	10
0		1	1	
1		1	1	

Kmap for H

$H = C$ Grouping can be of size four. This can be explained by performing Boolean Algebra on the canonical SOP expression for H.

$$\begin{aligned} H(A,B,C) = \; & A'B'C + A'BC + AB'C + ABC = \\ & (B+B')A'C + (B'+B)AC = \\ & A'C + AC = \\ & (A'+A)C = \\ & C \end{aligned}$$

In general, grouping can be of size $2^i \times 2^j$ for integer i and j. It is a common mistake of students to make a grouping of size 3x2 – 3 is not a power of 2!

$A\backslash BC$	00	01	11	10
0	1	1		
1		1	1	

Kmap for I

$I = A'B' + AC$ One does not have to form every possible grouping. It is a common error for students to include the term $B'C$ in the expression for I. The expression $B'C'$ does not cause the function to output an incorrect value. Rather, this expression is not necessary for the circuit to function properly. Hence, including the expression $B'C'$ would make the circuit larger by one more AND gate than necessary.

$A\backslash BC$	00	01	11	10
0	1			1
1	1	1		

Kmap for J

$J = A'C' + AB'$ Grouping can be made over the edge of the Kmap using the doughnut (torus) property illustrated in Figure 3.1. Notice, the minterms in the grouping on the upper row differ in the B bit.

$A\backslash BC$	00	01	11	10
0	1	1		1
1		1	1	1

Kmap for K

$K = A'B' + AC + BC'$ **or** $K = A'C' + B'C + AC$ A Kmap may have more than one correct solution.

$A\backslash BC$	00	01	11	10
0	1	1	1	
1	1	1	1	

Kmap for L

$L = B' + C$ Avoid the temptation to make a grouping of size 3×2. Instead, make two groupings of size 2×2 which overlap in two cells. Yes, overlapping big groupings is legal, too.

$A\backslash BC$	00	01	11	10
0				
1				

Kmap for M

$M = 0$ This trivial function can be confusing when first encountered. Notice that regardless of the input, the output of the function is always 0. Hence, $M = 0$.

The process of "solving" a Kmap correctly leads to a minimal 2-level SOP expression (abbreviated SOP_{min}). SOP expressions are composed of two main levels of gates. A set of AND gates (level 1) leading into an OR gate (level 2). The NOT gates are ignored because they are both small and fast compared to AND and OR gates. The term "minimal" refers to the fact that the realization of the function requires the fewest possible gates among any 2-level SOP realizations.

To explore more fully how to solve Kmaps, consider the following truth table which lists seven functions $F \ldots M$ each having three inputs. The $\text{SOP}_{\min}$ expression is derived for each of these functions, along the way illustrating many of the properties needed to solve Kmaps.

A	B	C	F	G	H	I	J	K	L	M
0	0	0	0	0	0	1	1	1	1	0
0	0	1	0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	1	1	1	1	0	0	0	1	0
1	0	0	1	0	0	0	1	0	1	0
1	0	1	1	0	1	1	1	1	1	0
1	1	0	0	0	0	0	0	1	0	0
1	1	1	0	1	1	1	0	1	1	0

$F = AB' + A'BC$ The grouping of cell 4 and cell 5 yields the product term AB'. Cell 5 and cell 3 cannot be combined because they differ in two bits. What is to do be done with the single cell 3? Since this cell cannot be combined with another cell, it is just a minterm by itself. Sad perhaps, but perfectly legal. Hence, $F(A, B, C) = AB' + A'BC$.

$A\backslash BC$	00	01	11	10
0			1	
1	1	1		

Kmap for F

$G = A'B + BC$ The natural question arising from this Kmap is, "Can cell 3 be reused in two different groups?" The answer is "Yes," and the reason can be demonstrated by performing some Boolean Algebra on the canonical SOP expression for G.

$A\backslash BC$	00	01	11	10
0			1	1
1			1	

Kmap for G

$$\begin{aligned} G(A,B,C) = \quad & A'BC' + A'BC + ABC = \\ & A'BC' + A'BC + A'BC + ABC = \\ & A'B(C' + C) + BC(A' + A) = \\ & A'B(1) + BC(1) = \\ & A'B + BC \end{aligned}$$

In the second line, the minterm $A'BC$ was duplicated using Law 3 of Boolean Algebra. Note, this is the same minterm covered twice in the Kmap. Consequently, if a cell of the Kmap is covered by three different groupings then it would have to be replicated three times in the Boolean Algebra simplification.

The grouping of cell 2 and cell 3 yields the product term $A'B$. The grouping of cell 3 and cell 7 yield the product term BC. Consequently, $G(A, B, C) = A'B + BC$. It is now possible to relate what happens in the symbolic expression when $(A, B, C) = (0, 1, 1)$ to the method used to solve the Kmap.