Karnaugh maps

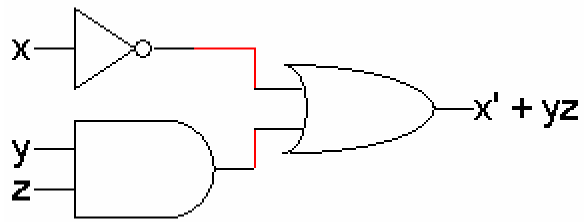
- So far this week we've used Boolean algebra to design hardware circuits.
  - The basic Boolean operators are AND, OR and NOT.
  - Primitive logic gates implement these operations in hardware.
  - Boolean algebra helps us simplify expressions and circuits.
- Today we present Karnaugh maps, an alternative simplification method that we'll use throughout the summer.

Minimal sums of products

- When used properly, Karnaugh maps can reduce expressions to a minimal sum of products, or MSP, form.
  - There are a minimal number of product terms.
  - Each product has a minimal number of literals.
- For example, both expressions below (from the last lecture) are sums of products, but only the right one is a *minimal* sum of products.

$$x'y' + xyz + x'y = x' + yz$$

- Minimal sum of products expressions lead to minimal two-level circuits.



- A minimal sum of products might not be "minimal" by other definitions! For example, the MSP $xy + xz$ can be reduced to $x(y + z)$, which has fewer literals and operators—but it is no longer a sum of products.

Organizing the minterms

- Recall that an *n*-variable function has up to 2^n minterms, one for each possible input combination.
- A function with inputs x, y and z includes up to eight minterms, as shown below.

x	y	z	Minterm	
0	0	0	x'y'z'	(m_0)
0	0	1	x'y'z	(m_1)
0	1	0	x'y z'	(m_2)
0	1	1	x'y z	(m_3)
1	0	0	x y'z'	(m_4)
1	0	1	x y'z	(m_5)
1	1	0	x y z'	(m_6)
1	1	1	x y z	(m_7)

- We'll rearrange these minterms into a Karnaugh map, or K-map.

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

- You can show either the actual minterms or just the minterm numbers.
- Notice the minterms are almost, but not quite, in numeric order.

Reducing two minterms

- In this layout, any two adjacent minterms contain at least one common literal. This is useful in simplifying the sum of those two minterms.
- For instance, the minterms x'y'z' and x'y'z both contain x' and y', and we can use Boolean algebra to show that their sum is x'y'.

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

$$
\begin{aligned}
x'y'z' + x'y'z &= x'y'(z' + z) \\
&= x'y' \bullet 1 \\
&= x'y'
\end{aligned}
$$

- You can also "wrap around" the sides of the K-map—minterms in the first and fourth columns are considered to be next to each other.

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

$$
\begin{aligned}
x y'z' + x y z' &= xz'(y' + y) \\
&= xz' \bullet 1 \\
&= xz'
\end{aligned}
$$

Reducing four minterms

- Similarly, rectangular groups of four minterms can be reduced as well. You can think of them as two adjacent groups of two minterms each.

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

- These four green minterms all have the literal y in common. Guess what happens when you simplify their sum?

$$
\begin{aligned}
x'yz + x'yz' + xyz + xyz' &= y(x'z + x'z' + xz + xz') \\
&= y(x'(z + z') + x(z + z')) \\
&= y(x' + x) \\
&= y
\end{aligned}
$$

Reducible groups

- Only rectangular groups of minterms, where the number of minterms is a power of two, can be reduced to a single product term.
  - Non-rectangular groups may not even contain a common literal.

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

  - Groups of other sizes cannot be simplified to just one product term.

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

The pattern behind the K-map

- The literal x occurs in the bottom four minterms, while the literal x' appears in the top four minterms.

x'	x'y'z'	x'y'z	x'y z	x'y z'
x	x y'z'	x y'z	x y z	x y z'

- The literal y shows up on the right side, and y' appears on the left.

y'		y	
x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'

- The literal z occurs in the middle four squares, while z' occurs in the first and fourth columns.

x'y'z'	x'y'z	x'y z	x'y z'
x y'z'	x y'z	x y z	x y z'
z'	z		z'

Map simplifications

- Knowing this pattern lets us find common literals, and simplify sums of minterms, without using any Boolean algebra at all!
- For example, look at the position of minterms x'y'z' and x'y'z.
  - They are both in the top half of the map, where x' appears.
  - They are also in the left half of the map, where y' appears.
  - This means x'y'z' + x'y'z = x'y', as we proved earlier.

			y	
	x'y'z'	x'y'z	x'y z	x'y z'
x	x y'z'	x y'z	x y z	x y z'
		z		

- Similarly, the four minterms x'yz, x'yz', xyz and xyz' all occur on the right half of the map, and they all contain the literal y.

			y	
	x'y'z'	x'y'z	x'y z	x'y z'
x	x y'z'	x y'z	x y z	x y z'
		z		

Multiple groups

- If our function has minterms that aren't all adjacent to each other in the K-map, then we'll have to form multiple groups.
- Consider the expression $x'y'z' + x'y'z + xyz + xyz'$.

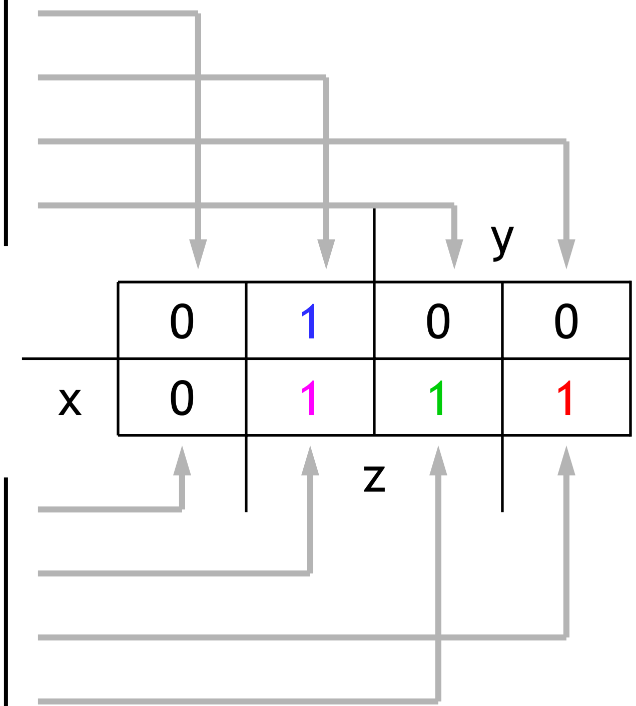
			y	
	$x'y'z'$	$x'y'z$	$x'yz$	$x'yz'$
x	$xy'z'$	$xy'z$	xyz	xyz'
		z	z	

- These minterms form two separate groups in the K-map. As a result, the simplified expression will contain *two* product terms, one for each group.
  - The sum $x'y'z' + x'y'z$ simplifies to $x'y'$, as we already saw.
  - Then we can also simplify $xyz + xyz'$ to xy.
- The result is that $x'y'z' + x'y'z + xyz + xyz' = x'y' + xy$.

Filling in the K-map

- Since our labels help us find the correct position of minterms in a K-map, writing the minterms themselves is redundant and repetitive.
- We usually just put a 1 in the K-map squares that correspond to the function minterms, and 0 in the other squares.
- For example, you can quickly fill in a K-map from a truth table by copying the function outputs to the proper squares of the map.

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



f(x,y,z) = x’y’z + xy’z + xyz’ + xyz

Four steps in K-map simplifications

1. Start with a sum of minterms or truth table. x’y’z’ + x’y’z + xyz + xyz’

2. Plot the minterms on a Karnaugh map.

			y	y
	1	1	0	0
x	0	0	1	1
		z	z	

3. Find rectangular groups of minterms whose sizes are powers of two. Be sure to include all the minterms in at least one group!

			y	y
	1	1	0	0
x	0	0	1	1
		z	z	

4. Reduce each group to one product term. x’y’ + xy

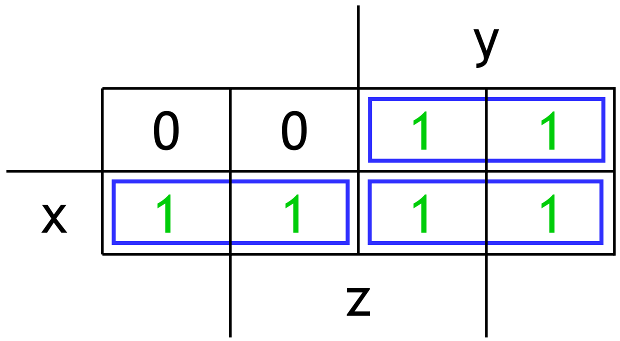
The tricky part

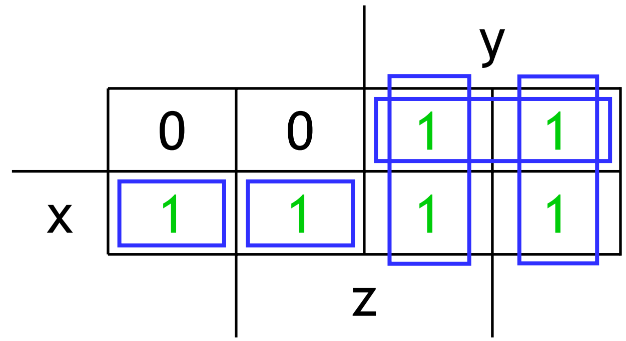
- The tricky part is finding the best groups of minterms.
  - Each group represents one product term, so *making as few groups as possible* will result in a minimal number of products.
  - *Making each group as large as possible* corresponds to combining more minterms, and will result in a minimal number of literals.
- Which groups would you form in the following example map?

			y	y
	0	0	1	1
x	1	1	1	1
		z	z	

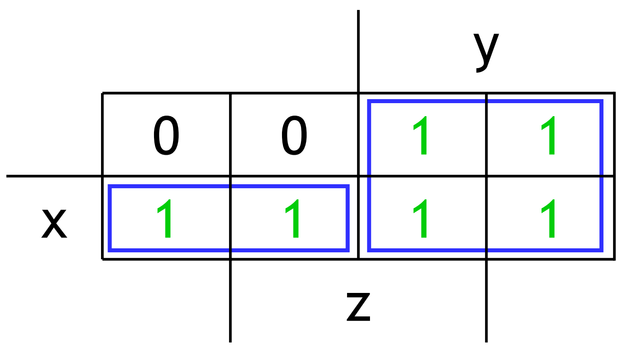
Minimizing the number of groups

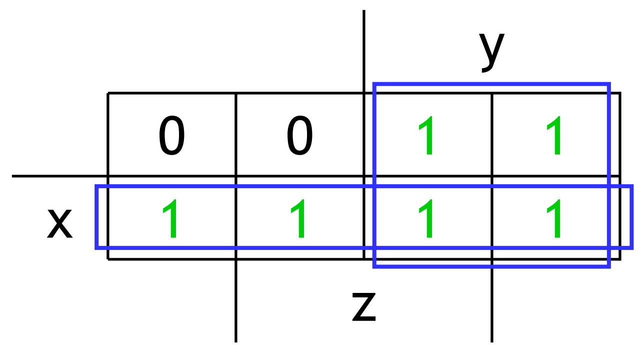
- The following two possibilities include too many groups, and would result in more product terms than necessary.



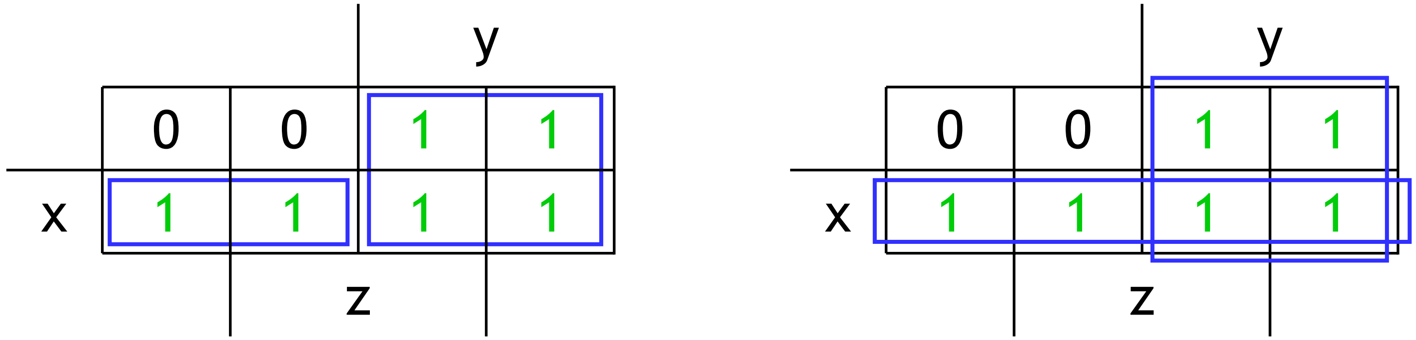


- We can put all six minterms into just two groups. Two ways of doing this are shown below.





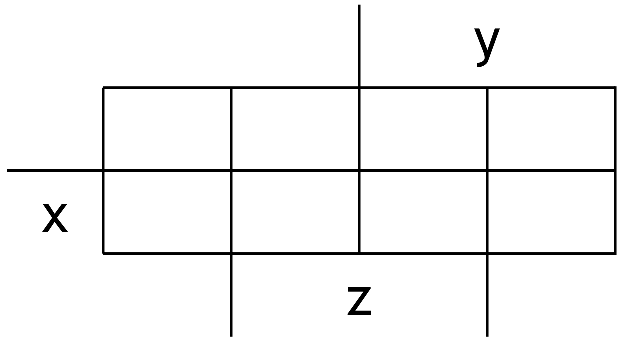
Maximizing the size of each group



- Since we want to make each group as large as possible, the solution on the right is *better* than the one on the left.
- Note that overlapping groups are acceptable, and sometimes necessary.
- Making poor choices of groups will produce a result that is still equivalent to the original expression, but it won't be minimal.
  - The maps on the left and right here yield xy' + y and x + y.
  - These are equivalent, but only x + y is a *minimal* sum of products.

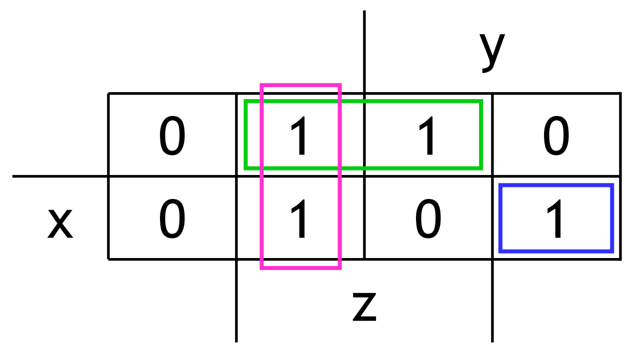
Practice K-map 1

- Simplify the sum of minterms $f(x,y,z) = m_1 + m_3 + m_5 + m_6$.



Solutions for practice K-map 1

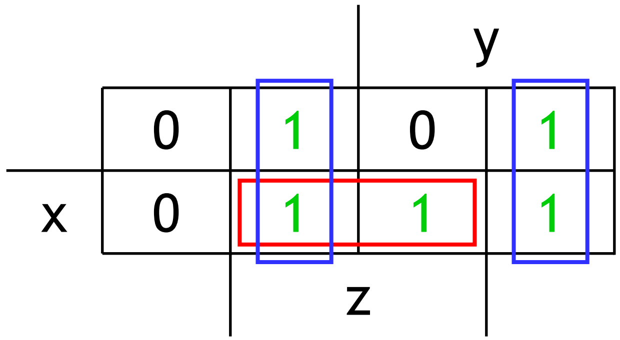
- Here is the K-map for $f(x,y,z) = m_1 + m_3 + m_5 + m_6$, with all groups shown.
  - The magenta and green groups overlap, which makes each of them as large as possible.
  - Minterm m_6 is in a group all by its lonesome.



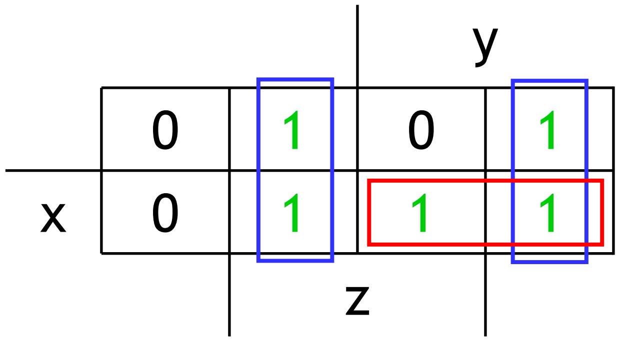
- The final MSP here is $x'z + y'z + xyz'$.

Multiple solutions are possible

- Sometimes there are multiple possible correct answers.



y'z + yz' + xz



y'z + yz' + xy

- Both maps here contain the fewest and largest possible groups.
- The resulting expressions are *both* minimal sums of products—they have the same number of product terms and the same number of literals.

Don’t care conditions

- There are times when we don’t care what a function outputs—some input combinations might never occur, or some outputs may have no effect.
- We can express these situations with don’t care conditions, denoted with X in truth table rows.
- An expression for this function has two parts.
  - One part includes the function’s minterms.
  - Another describes the don’t care conditions.

$$f(x,y,z) = m_3,\ d(x,y,z) = m_2 + m_4 + m_5$$

- Circuits *always* output 0 or 1; there is no value called “X”. Instead, the Xs just indicate cases where both 0 or 1 would be acceptable outputs.

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	0
0	1	0	X
0	1	1	1
1	0	0	X
1	0	1	X
1	1	0	0
1	1	1	0

Don’t care simplifications

- In a K-map we can treat each don’t care as 0 or 1. Different selections can produce different results.

			y	y
	0	0	1	X
x	X	X	0	0
		z	z	

- In this example we can use the don’t care conditions to our advantage.
  - It’s best to treat the bottom two Xs as 0s. If either of them were 1, we’d end up with an extra, unnecessary term.
  - On the other hand, interpreting the top X as 1 results in a larger group containing m_3.
- The resulting MSP is x’y.

Four-variable Karnaugh maps

- We can do four-variable Karnaugh maps too!
- A four-variable function f(w,x,y,z) has sixteen possible minterms. They can be arranged so that adjacent minterms have common literals.
  - You can wrap around the sides *and* the top and bottom.
  - Again the minterms are almost, but not quite, in numeric order.

			y	y	
	w’x’y’z’	w’x’y’z	w’x’y z	w’x’y z’	
	w’x y’z’	w’x y’z	w’x y z	w’x y z’	x
w	w x y’z’	w x y’z	w x y z	w x y z’	x
w	w x’y’z’	w x’y’z	w x’y z	w x’y z’	
		z	z		

			y	y	
	m_0	m_1	m_3	m_2	
	m_4	m_5	m_7	m_6	x
w	m_{12}	m_{13}	m_{15}	m_{14}	x
w	m_8	m_9	m_{11}	m_{10}	
		z	z		

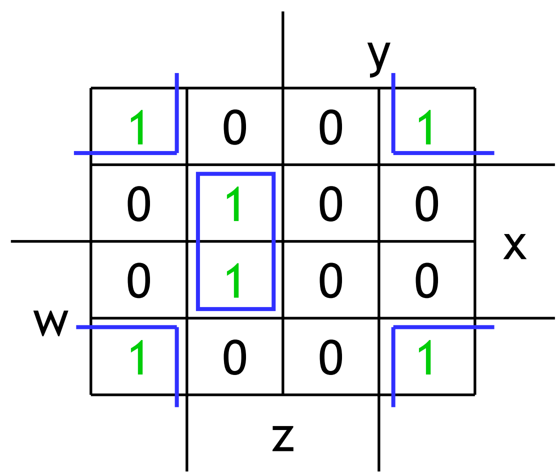
Four-variable example

- Let's say we want to simplify $m_0 + m_2 + m_5 + m_8 + m_{10} + m_{13}$

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

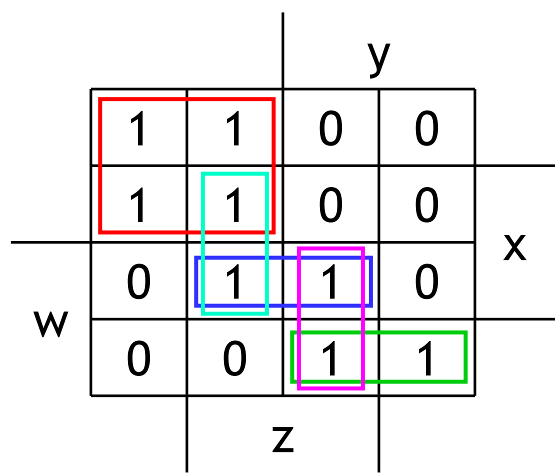
1	0	0	1
0	1	0	0
0	1	0	0
1	0	0	1

- The following groups result in the minimal sum of products $x'z' + xy'z$.



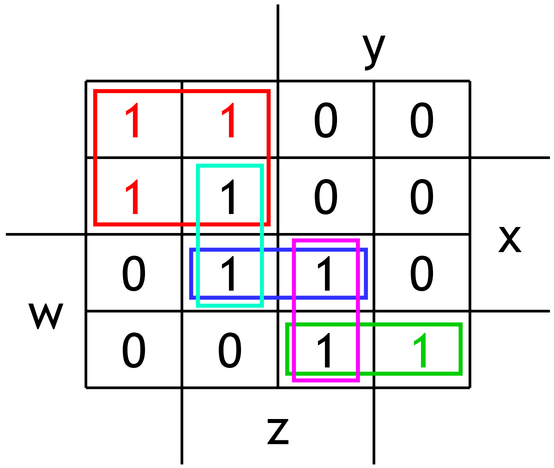
Prime implicants

- Finding the best groups is even more difficult in larger K-maps.
- One good approach to deriving an MSP is to first find the largest possible groupings of minterms.
  - These groups correspond to prime implicant terms.
  - The final MSP will contain a subset of the prime implicants.
- Here is an example K-map with prime implicants marked.



Essential prime implicants

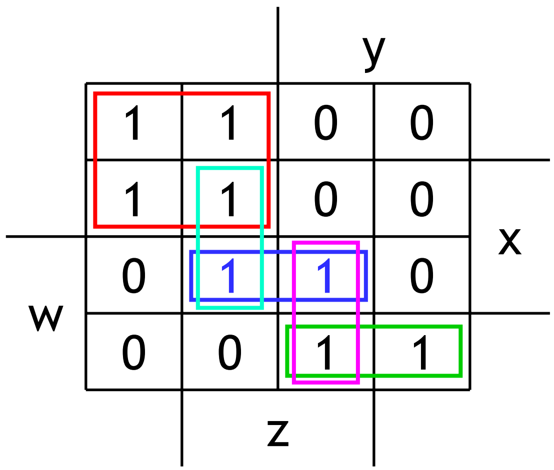
- If any minterm belongs to only one group, then that group represents an essential prime implicant.
- Essential prime implicants *must* appear in the final MSP, which has to include all of the original minterms.



- This example has two essential prime implicants.
  - The red group (w'y) is essential, since m_0, m_1 and m_4 are not in any other group.
  - The green group (wx'y) is essential because of m_{10}.

Covering the other minterms

- Finally, pick as few other prime implicants as necessary to ensure that all of the original minterms are included.



- After choosing the red and green rectangles in our example, there are just two minterms remaining, m_{13} and m_{15}.
  - They are both included in the blue prime implicant, wxz.
  - The resulting MSP is w'y' + wxz + wx'y.
- The magenta and sky blue groups are not needed, since their minterms are already included by the other three prime implicants.

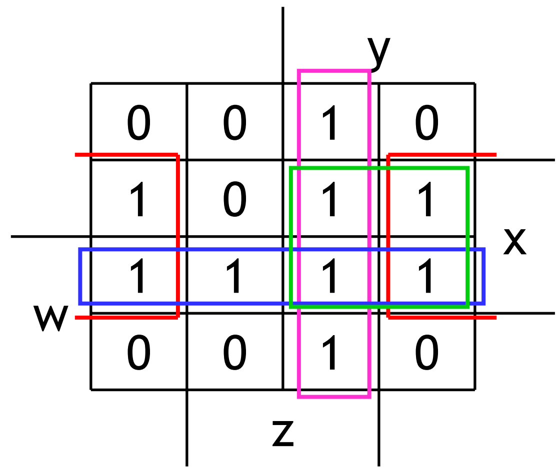
Practice K-map 2

- Simplify the following K-map.

			y		
	0	0	1	0	
	1	0	1	1	x
w	1	1	1	1	
	0	0	1	0	
		z			

Solutions for practice K-map 2

- Simplify the following K-map.

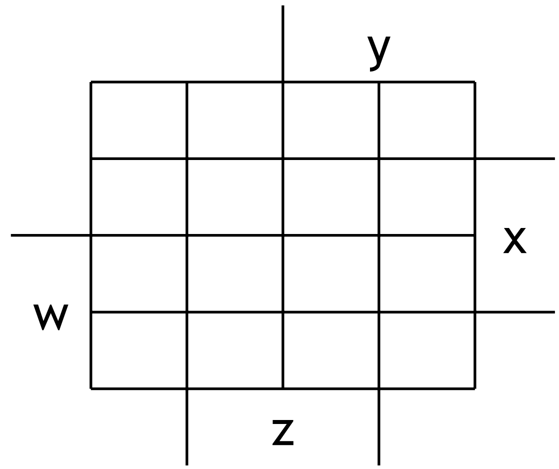


- All prime implicants are circled.
- The essential prime implicants are xz', wx and yz.
- The MSP is xz' + wx + yz. (Including the group xy would be redundant.)

Practice K-map 3

- Find a minimal sum of products for the following.

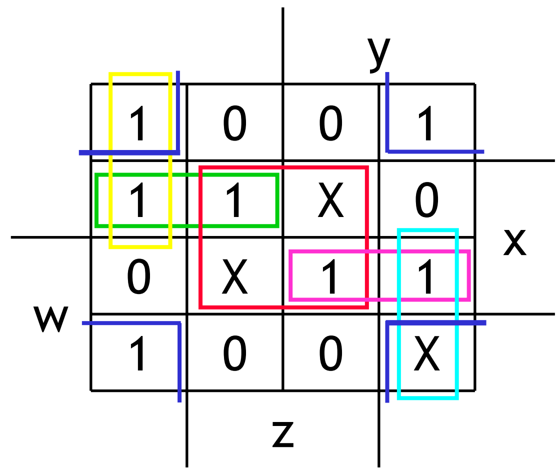
$$f(w,x,y,z) = \Sigma m(0,2,4,5,8,14,15),\ d(w,x,y,z) = \Sigma m(7,10,13)$$



Solutions for practice K-map 3

- Find a minimal sum of products for the following.

$$f(w,x,y,z) = \Sigma m(0,2,4,5,8,14,15),\ d(w,x,y,z) = \Sigma m(7,10,13)$$



- All prime implicants are circled. We can treat Xs as 1s if we want, so the red group includes two Xs, and the light blue group includes one X.
- The *only* essential prime implicant is x'z'. The red group is not essential because the two minterms in it also appear in other groups.
- The MSP is x'z' + wxy + w'xy'. It turns out the red group is redundant; we can cover all of the minterms in the map without it.

Summary

- Karnaugh maps can simplify functions to a minimal sum of products form.
  - This leads to a minimal two-level circuit implementation.
  - It's easy to handle don't care conditions.
- K-maps are really only practical for smaller functions, with four variables or less. But that's good enough for CS231!
- You should keep several things in mind.
  - Remember the correct position of minterms in the K-map.
  - Your groups can wrap around all sides of a Karnaugh map.
  - Make as few groups as possible, but make each of them as large as possible. Groups can overlap if that makes them bigger.
  - There may be more than one valid MSP for a given function.