Circuit Optimization

Logic Design

Standard Forms

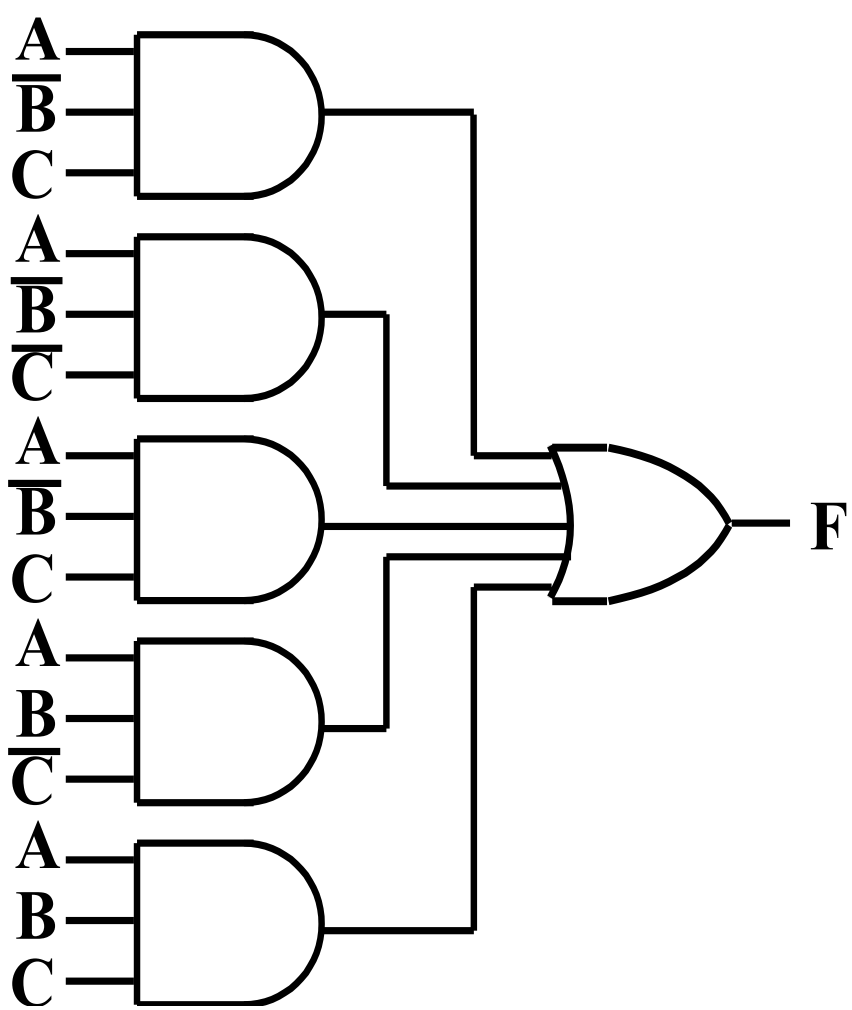
- **Standard Sum-of-Products (SOP)** form: equations are written as an OR of AND terms
- **Standard Product-of-Sums (POS)** form: equations are written as an AND of OR terms
- Examples:
  - SOP:ABC + A'B'C+B
  - POS: (A+B)(A+B'+C')C
- These "mixed" forms are neither SOP nor POS
  - (AB+C)(A+C)
  - ABC'+AC(A+B)

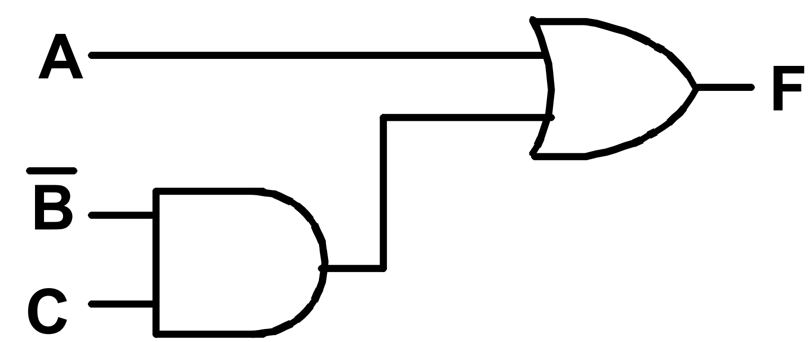
Standard Sum-of-Products (SOP)

- A Simplification Example:
  - $F = \Sigma_m(1,4,5,6,7)$
- Writing the minterm expression:
  - F =
- Simplifying:
  - F =
- Simplified F contains 3 literals compared to 15 in minterm F
  - Literal: variable or its complement

AND/OR Two-level Implementation of SOP Expression

- The two implementations for F are shown below – it is quite apparent which is simpler!





Observations

- Canonical Forms (Sum-of-minterms, Product-of-Maxterms), or other standard forms (SOP, POS) differ in complexity
  - Boolean algebra can be used to manipulate equations into simpler forms.
  - Simpler equations lead to simpler two-level implementations
- Questions:
  - How can we attain a **simplest** expression?

Karnaugh Maps (K-map)

- A K-map is a collection of squares
  - Each square represents a **minterm**
  - The collection of squares is a graphical representation of a Boolean function
  - **Adjacent squares** differ in the value of **<u>one variable</u>**
- The K-map can be viewed as
  - A reorganized version of the truth table

Two Variable Maps

- A 2-variable Karnaugh Map:
  - minterm m_0 and minterm m_1 are **adjacent**
    - differ in the value of the variable y

	y = 0	y = 1
x = 0	$m_0 = \bar{x}\,\bar{y}$	$m_1 = \bar{x}\,y$
x = 1	$m_2 = x\,\bar{y}$	$m_3 = x\,y$

  - interm m_0 and minterm m_2 differ in the x variable.
  - m_1 and m_3 differ in the x variable as well.
  - m_2 and m_3 differ in the value of the variable y

K-Map and Truth Tables

- The K-Map is just a different form of the truth table.
- Example – Two variable function:

Function Table

Input Values (x,y)	Function Value F(x,y)
0 0	1
0 1	0
1 0	0
1 1	1

K-Map

	y = 0	y = 1
x = 0	1	0
x = 1	0	1

Three Variable Maps

- A three-variable K-map:

	yz=00	yz=01	yz=11	yz=10
x=0	m_0	m_1	m_3	m_2
x=1	m_4	m_5	m_7	m_6

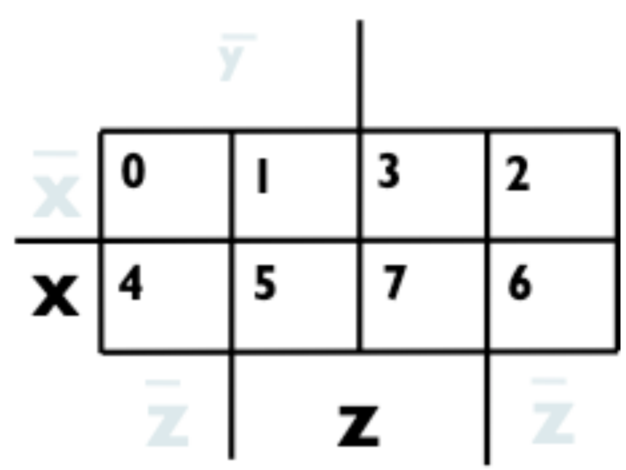
- Where each minterm corresponds to the product terms:

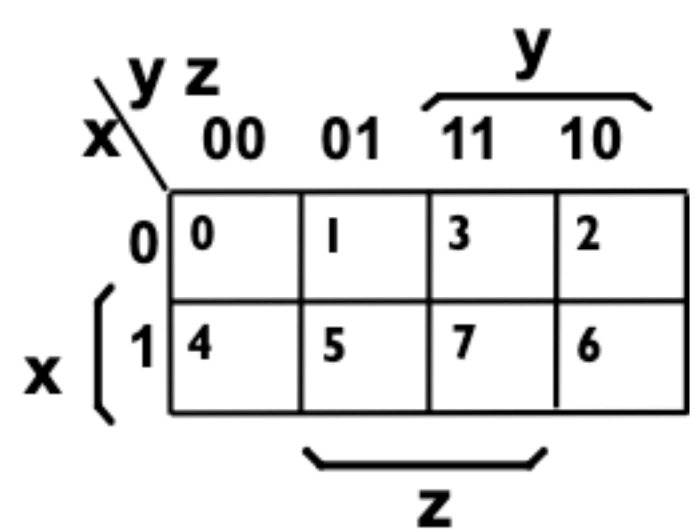
	yz=00	yz=01	yz=11	yz=10
x=0	$\overline{x}\,\overline{y}\,\overline{z}$	$\overline{x}\,\overline{y}\,z$	$\overline{x}\,y\,z$	$\overline{x}\,y\,\overline{z}$
x=1	$x\,\overline{y}\,\overline{z}$	$x\,\overline{y}\,z$	$x\,y\,z$	$x\,y\,\overline{z}$

Note that if the binary value for an index **differs in one bit position**, the minterms are adjacent on the K-Map

Alternative Map Labeling

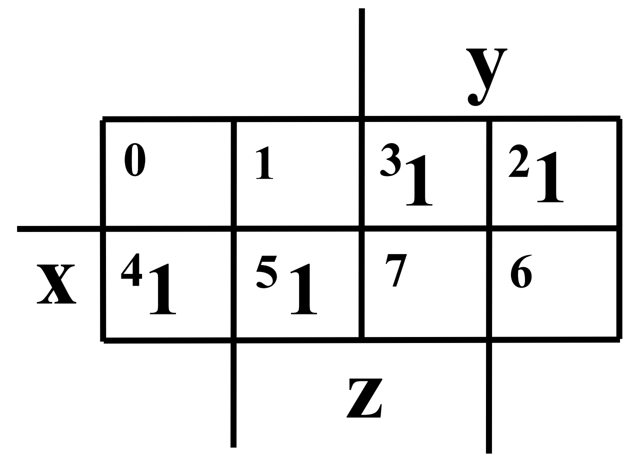
- Map use largely involves:
  - Entering values into the map, and Reading off product terms from the map.
- Alternate labelings are useful:



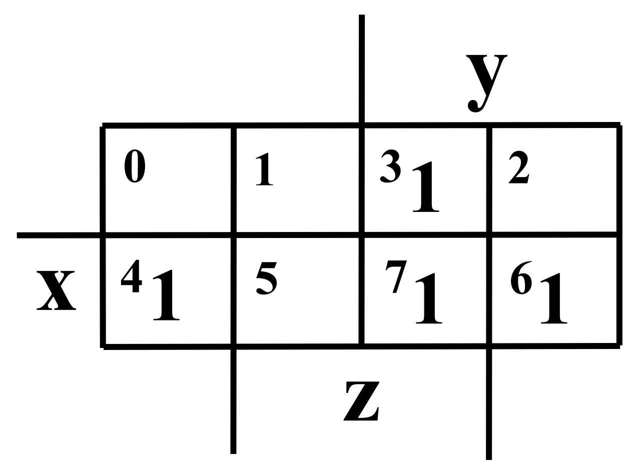


Example Functions

- By convention, we represent the minterms of F by a "1" in the map and leave the minterms of blank
- Example:
  - $F(x,y,z)=\Sigma_m(2,3,4,5)$



  - $G(a,b,c)=\Sigma_m(3,4,6,7)$

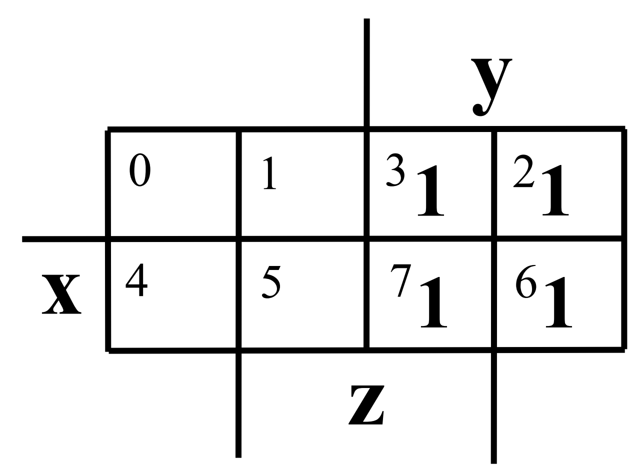


Combining Squares

- On a 3-variable K-Map:
  - One square represents a minterm with three variables
  - Two adjacent squares represent a product term with two variables
  - Four “adjacent” terms represent a product term with one variable
  - Eight “adjacent” terms is the function of all ones.
- Example:
  - $F(x,y,z)=\boldsymbol{\Sigma}_m(2,3,6,7)$
  - After minimization **F = y**

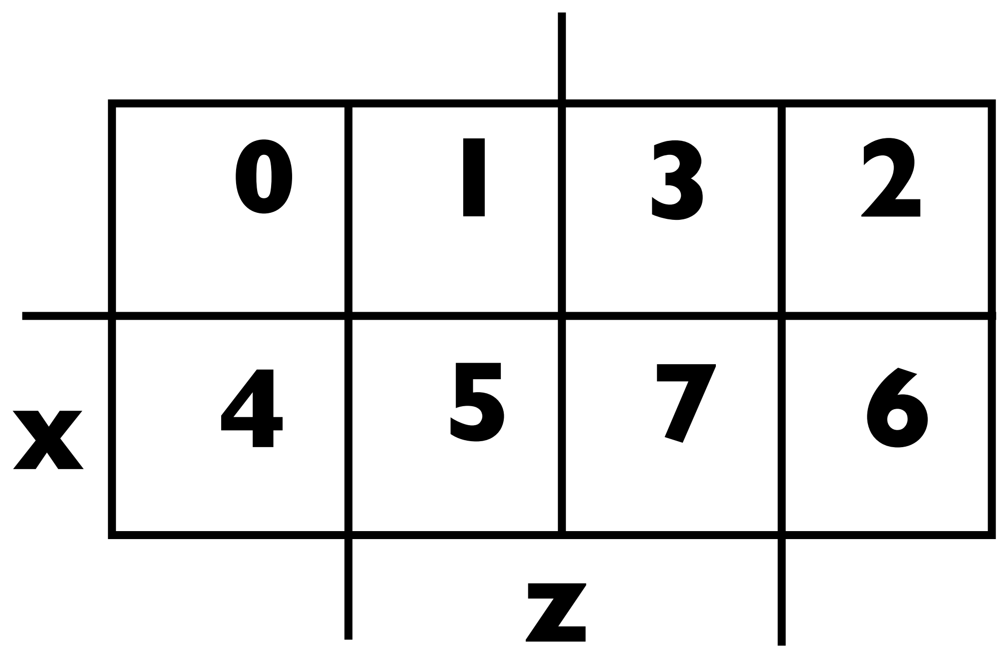
Combining Squares

- On a 3-variable K-Map:
  - One square represents a minterm with three variables
  - Two adjacent squares represent a product term with two variables
  - Four “adjacent” terms represent a product term with one variable
  - Eight “adjacent” terms is the function of all ones.
- Example:
  - $F(x,y,z)=\Sigma_m(2,3,6,7)$
  - After minimization **F = y**



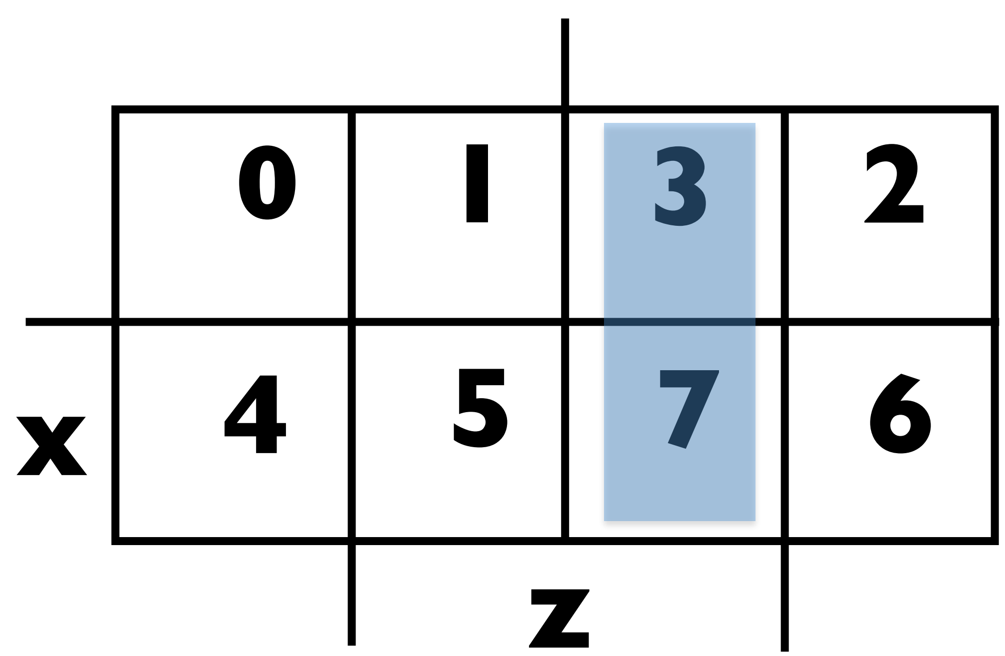
Three-Variable Maps

- Example Shapes of 2-cell Rectangles:
  - Read off the product terms for the rectangles shown



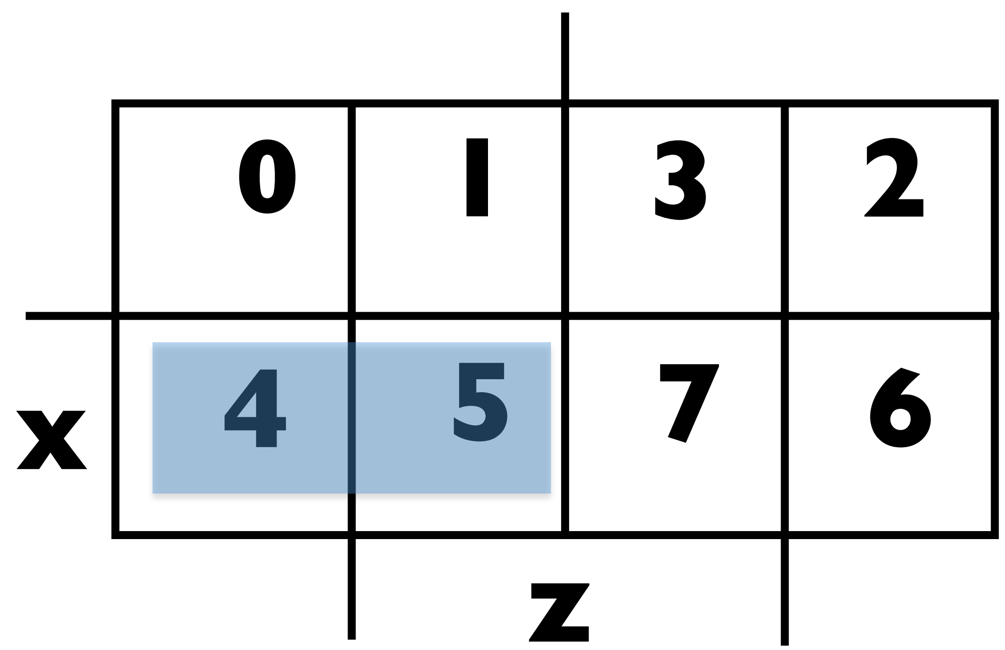
Three-Variable Maps

- Example Shapes of 2-cell Rectangles:
  - Read off the product terms for the rectangles shown



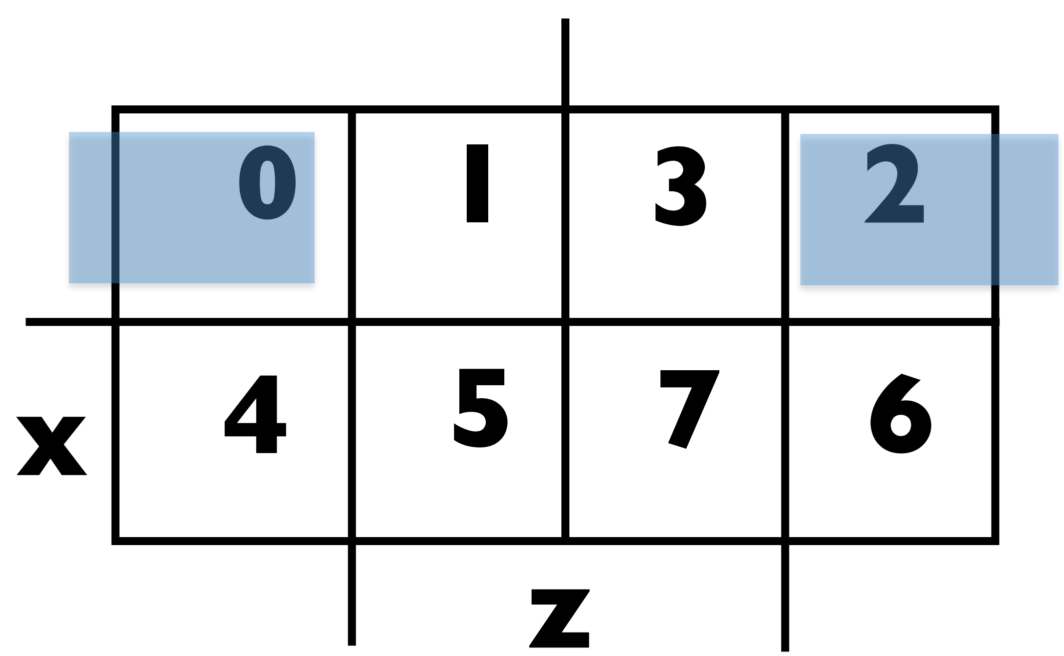
Three-Variable Maps

- Example Shapes of 2-cell Rectangles:
  - Read off the product terms for the rectangles shown



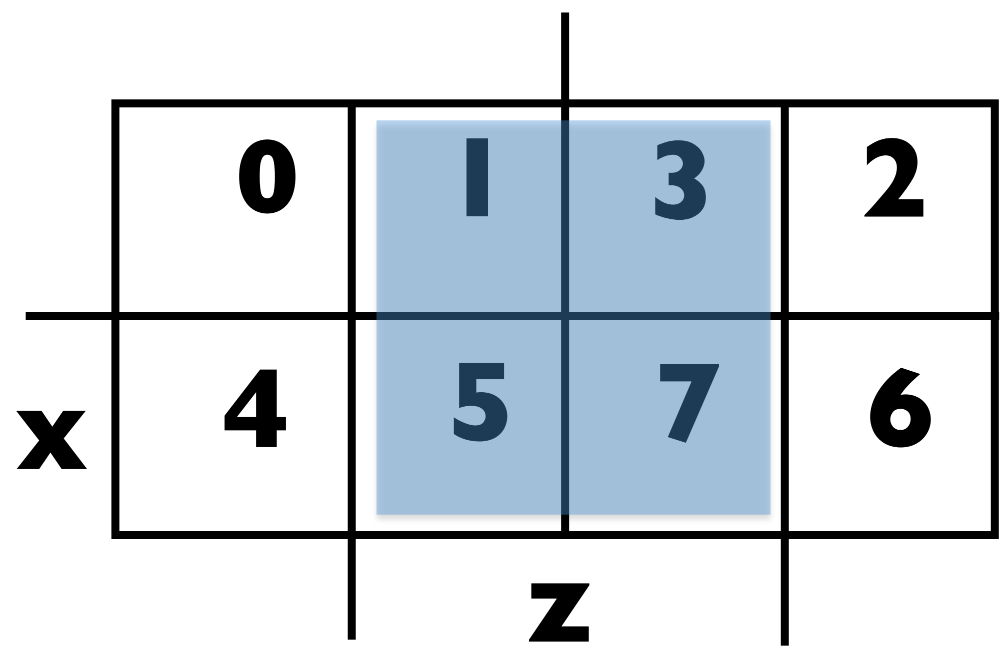
Three-Variable Maps

- Example Shapes of 2-cell Rectangles:
  - Read off the product terms for the rectangles shown



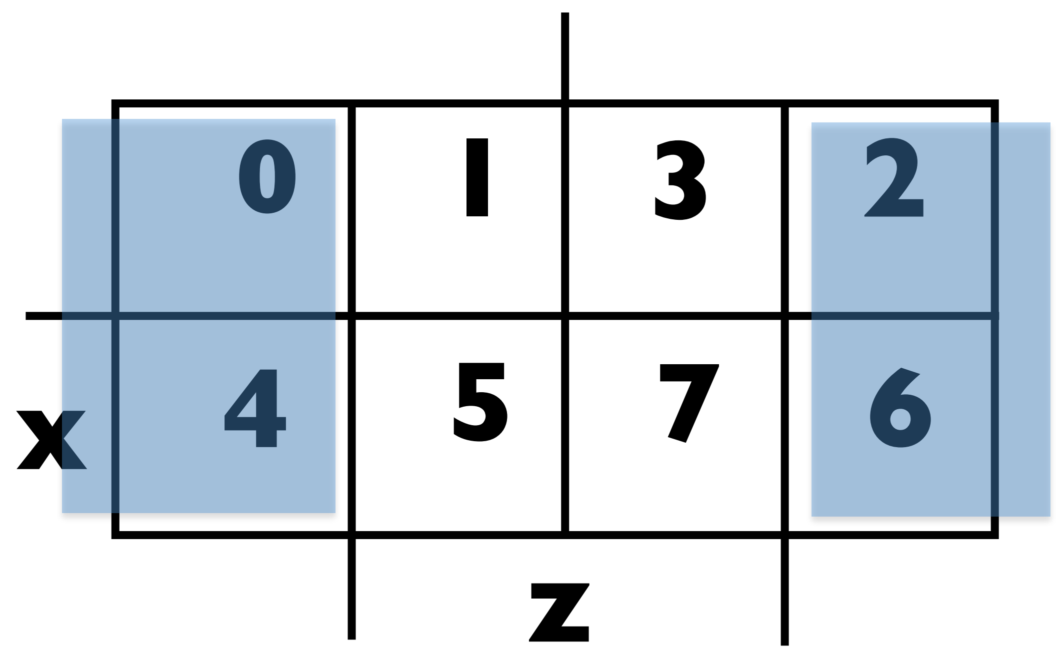
Three-Variable Maps

- Example Shapes of 4-cell Rectangles:
  - Read off the product terms for the rectangles shown



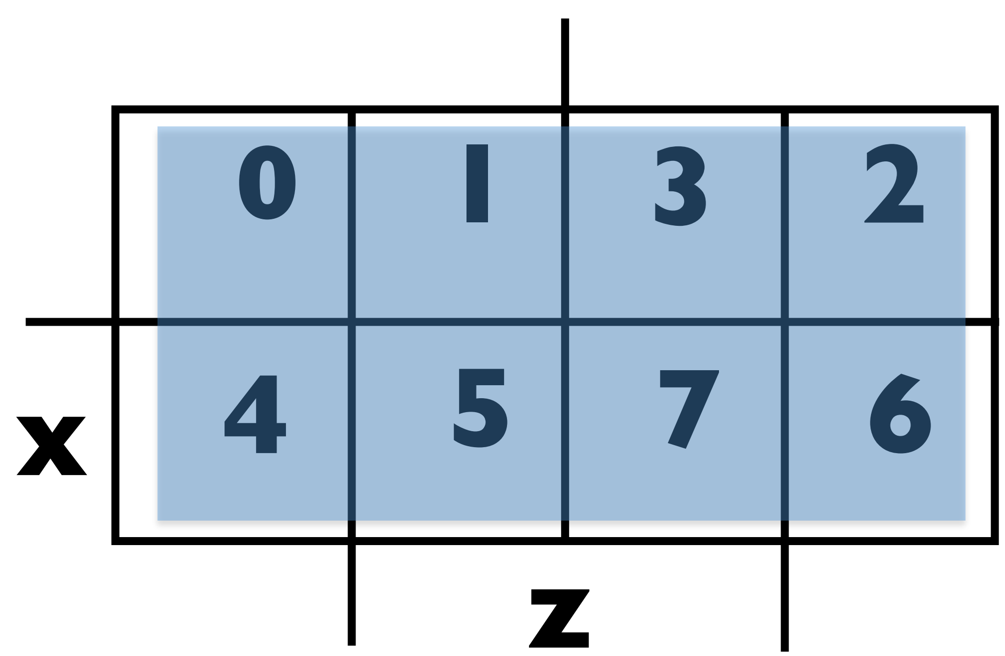
Three-Variable Maps

- Example Shapes of 4-cell Rectangles:
  - Read off the product terms for the rectangles shown



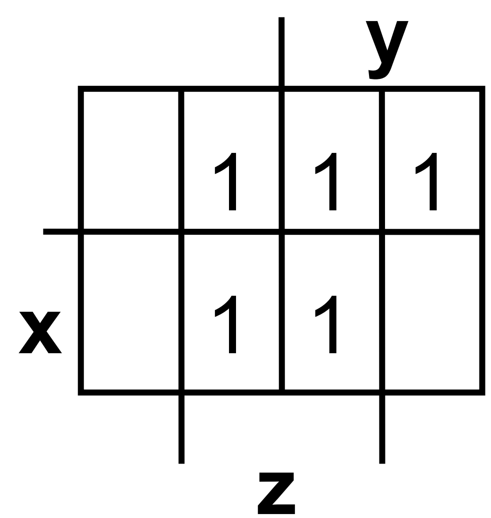
Three-Variable Maps

- Example Shapes of 8-cell Rectangles:
  - Read off the product terms for the rectangles shown



Three Variable Maps

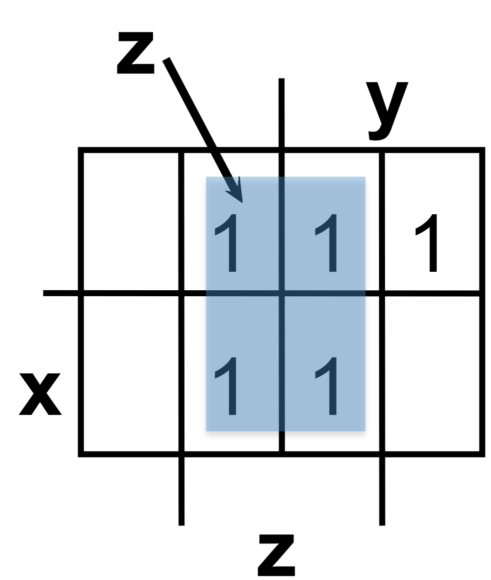
- K-Maps can be used to simplify Boolean functions by systematic methods.
  - Terms are selected to cover the “1s”in the map.
- Simplify $F(x,y,z)=\Sigma_m(1,2,3,5,7)$



=

Three Variable Maps

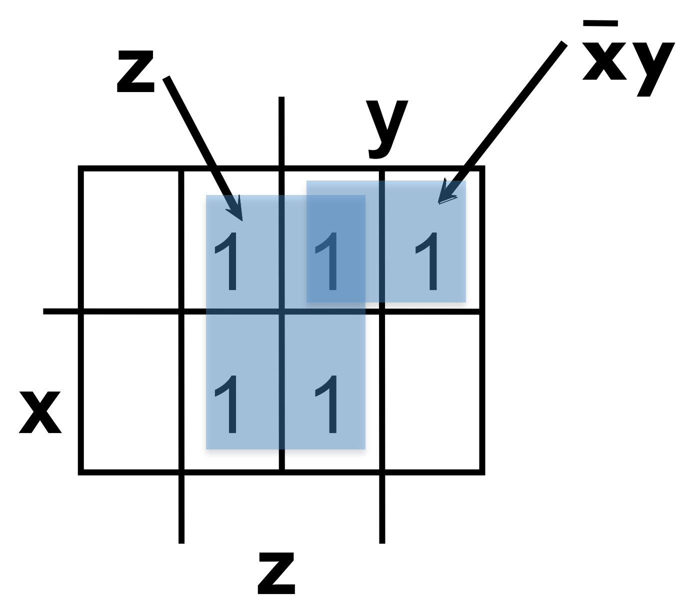
- K-Maps can be used to simplify Boolean functions by systematic methods.
  - Terms are selected to cover the “1s”in the map.
- Simplify $F(x,y,z)=\Sigma_m(1,2,3,5,7)$



$= z$

Three Variable Maps

- K-Maps can be used to simplify Boolean functions by systematic methods.
  - Terms are selected to cover the “1s”in the map.
- Simplify $F(x,y,z)=\Sigma_m(1,2,3,5,7)$



$$= z+\bar{x}y$$

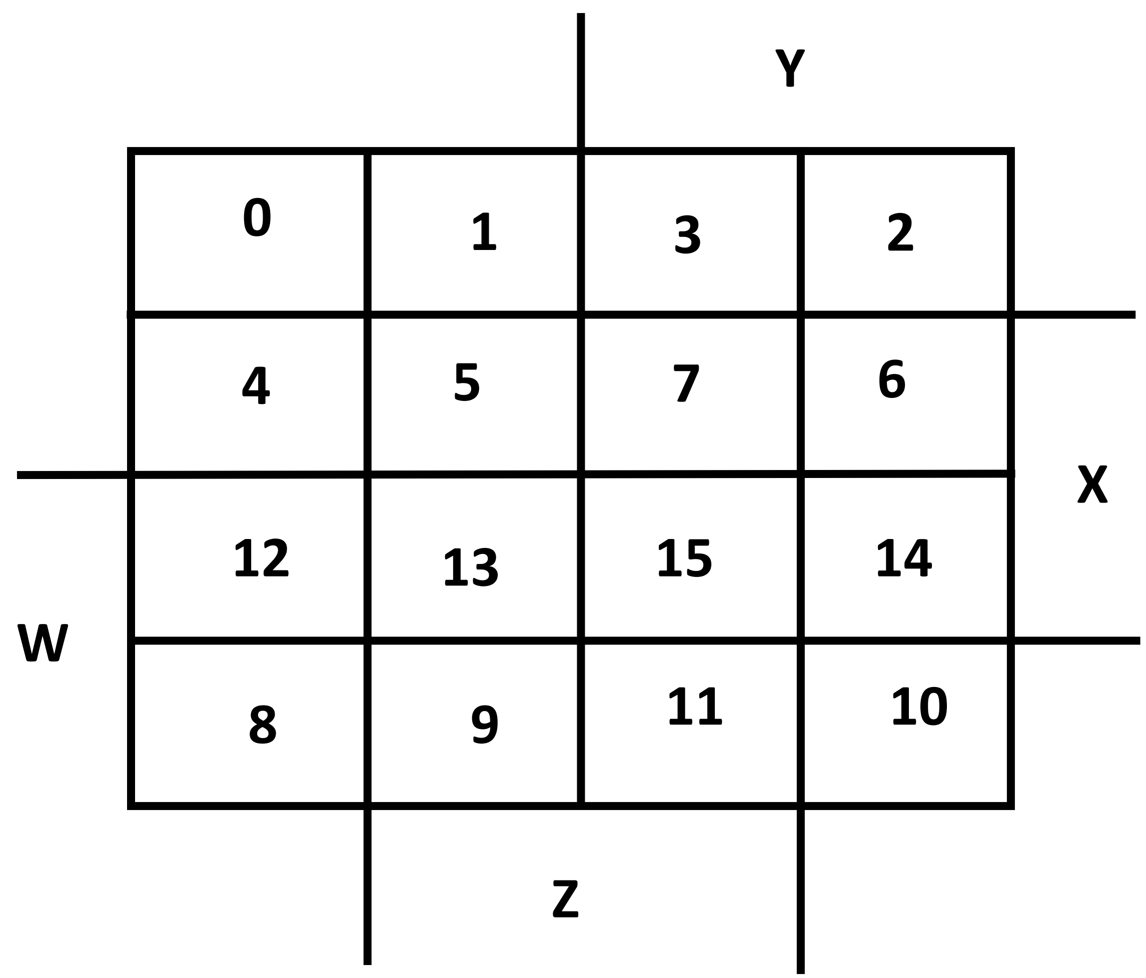
Three-Variable Map Simplification

- Use a K-map to find an optimum SOP equation for

$$F(X, Y, Z) = \Sigma_m(0,1,2,4,6,7)$$

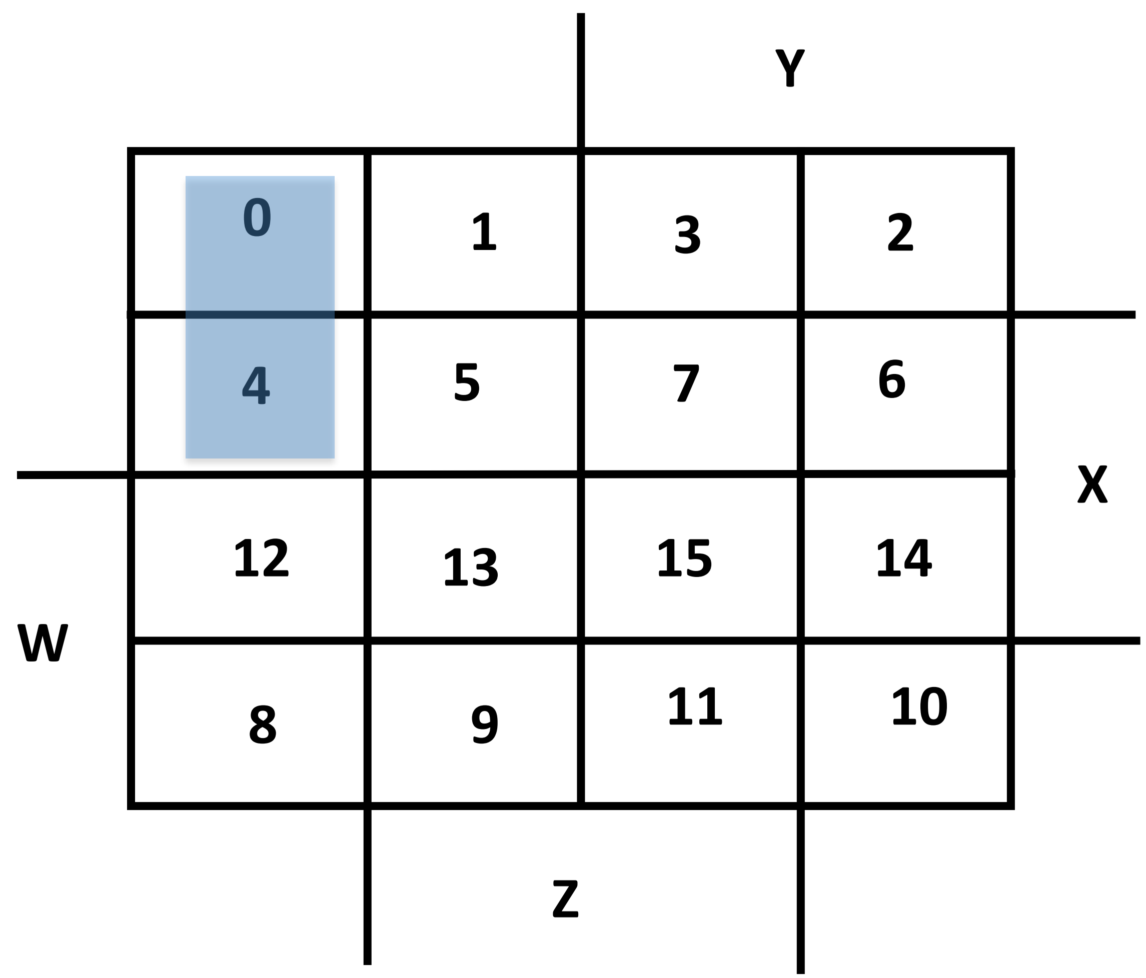
Four Variable Maps

- Map and location of minterms:



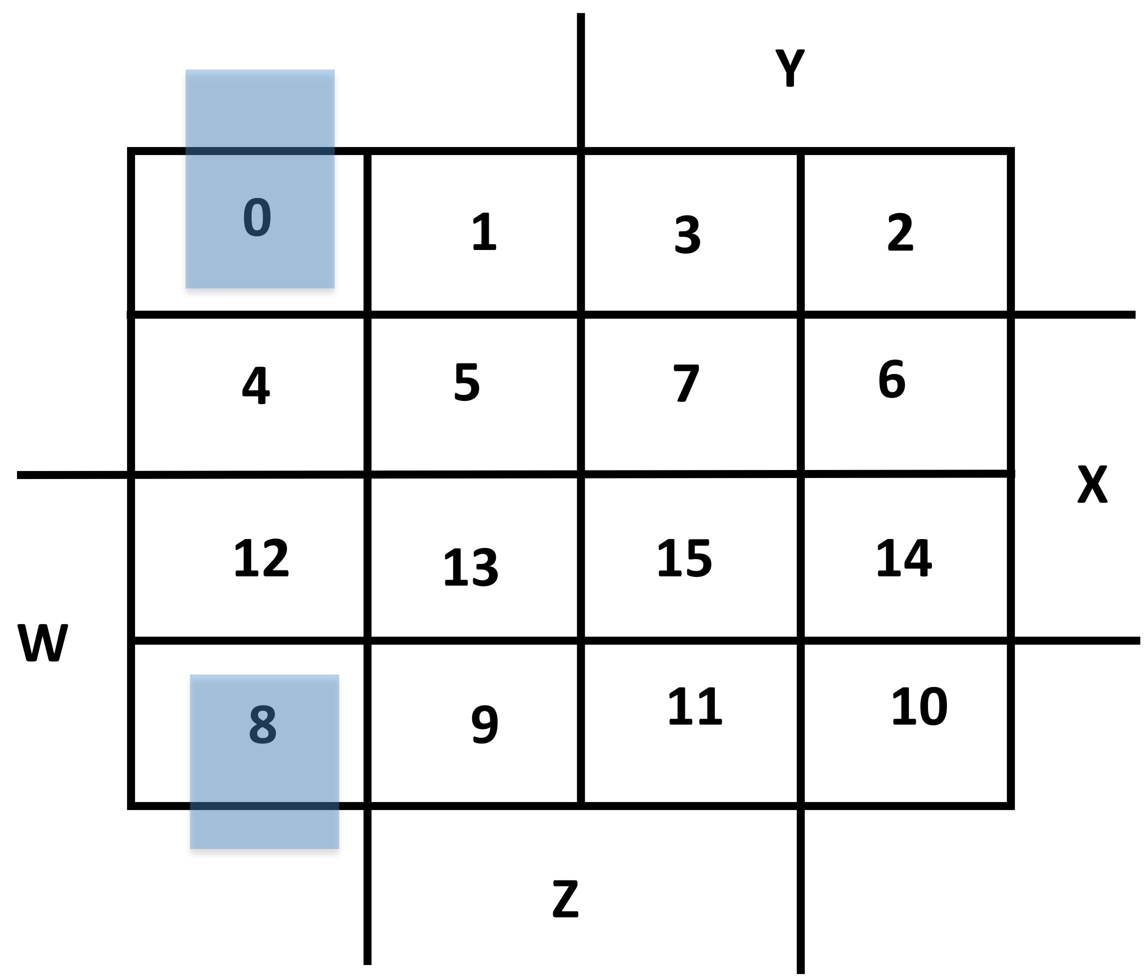
Four Variable Maps

- Map and location of minterms:



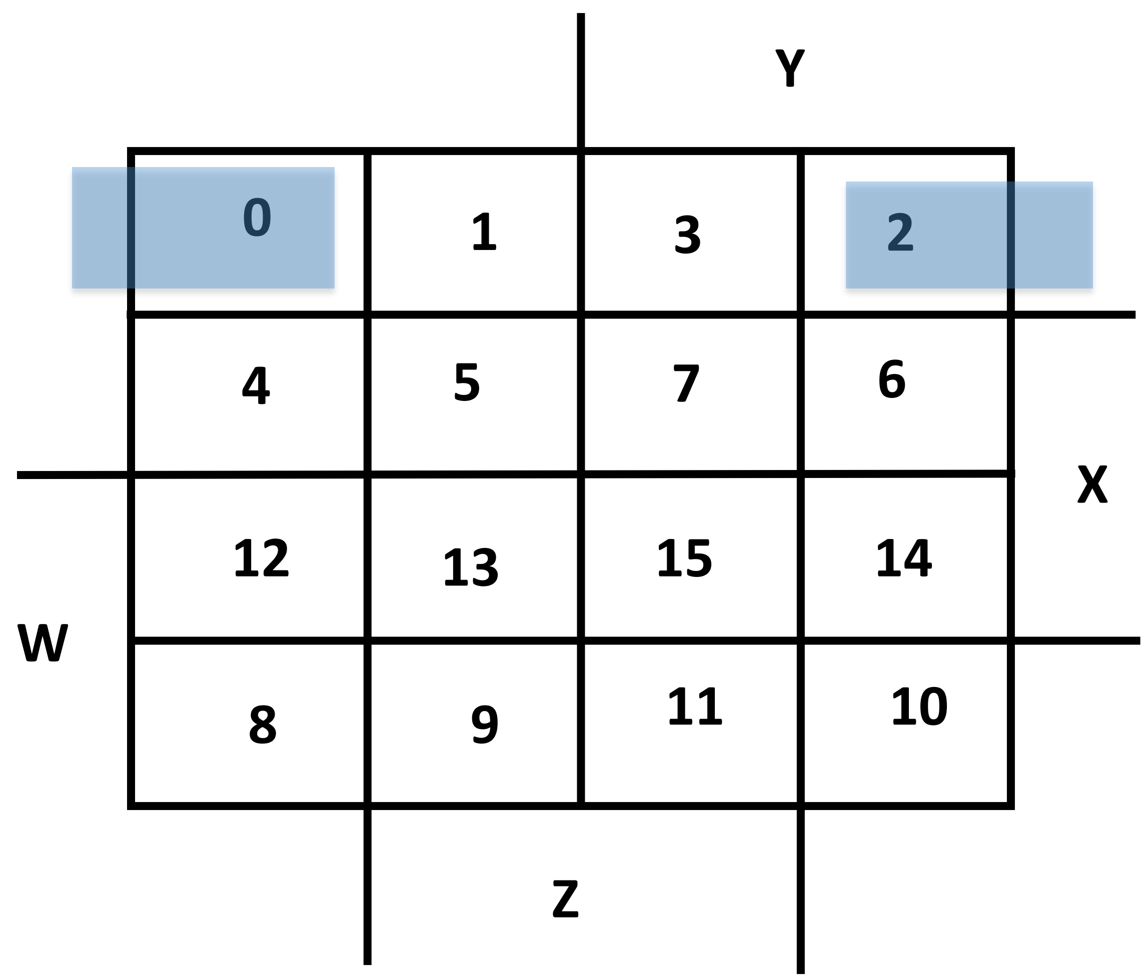
Four Variable Maps

- Map and location of minterms:



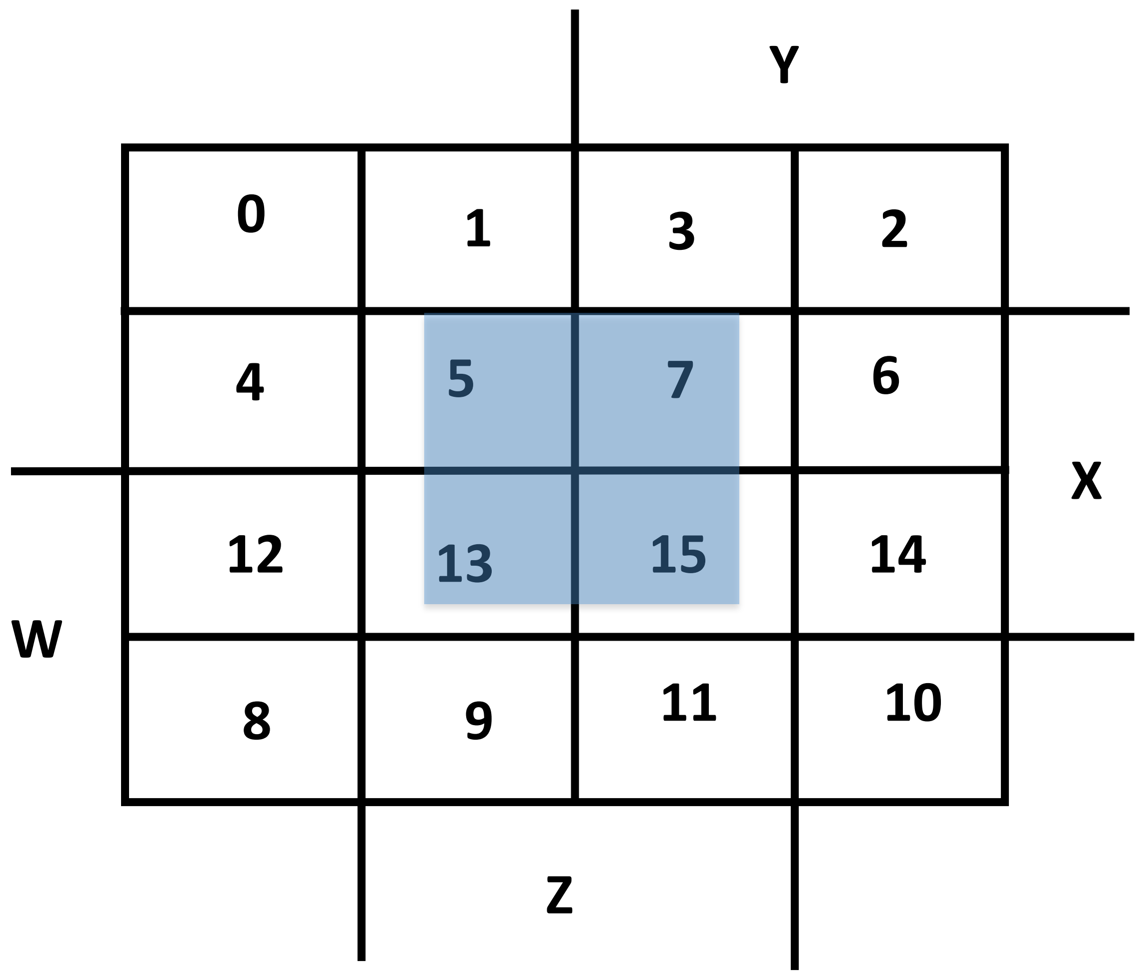
Four Variable Maps

- Map and location of minterms:



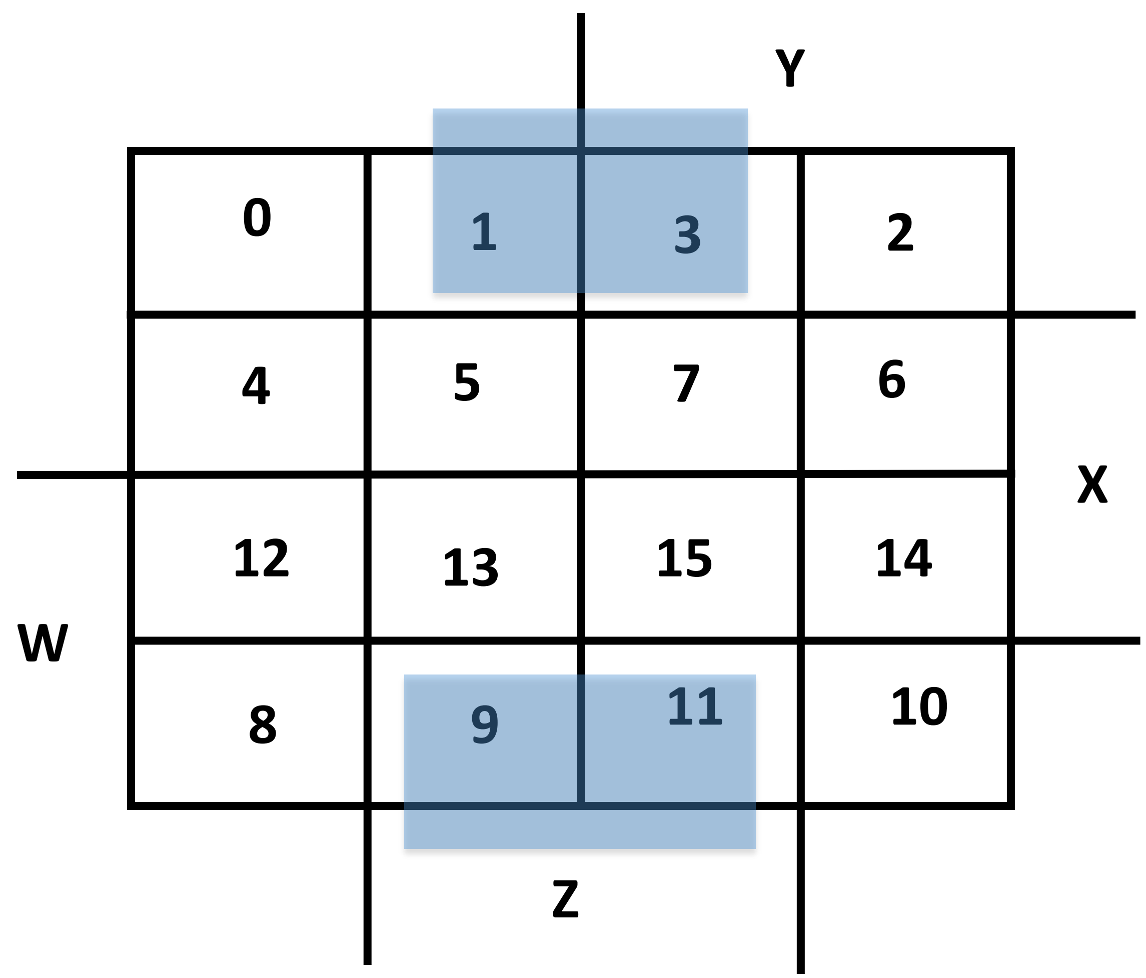
Four Variable Maps

- Map and location of minterms:



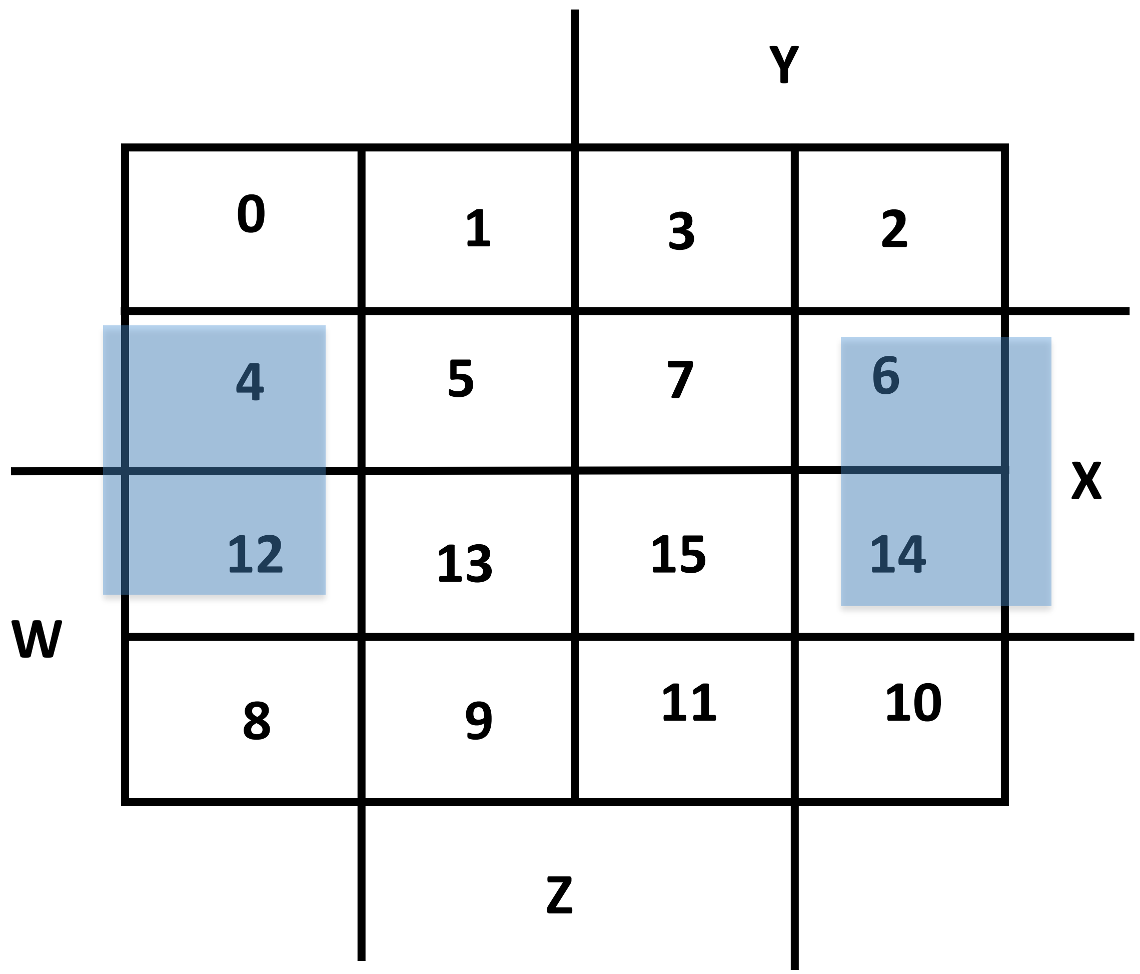
Four Variable Maps

- Map and location of minterms:



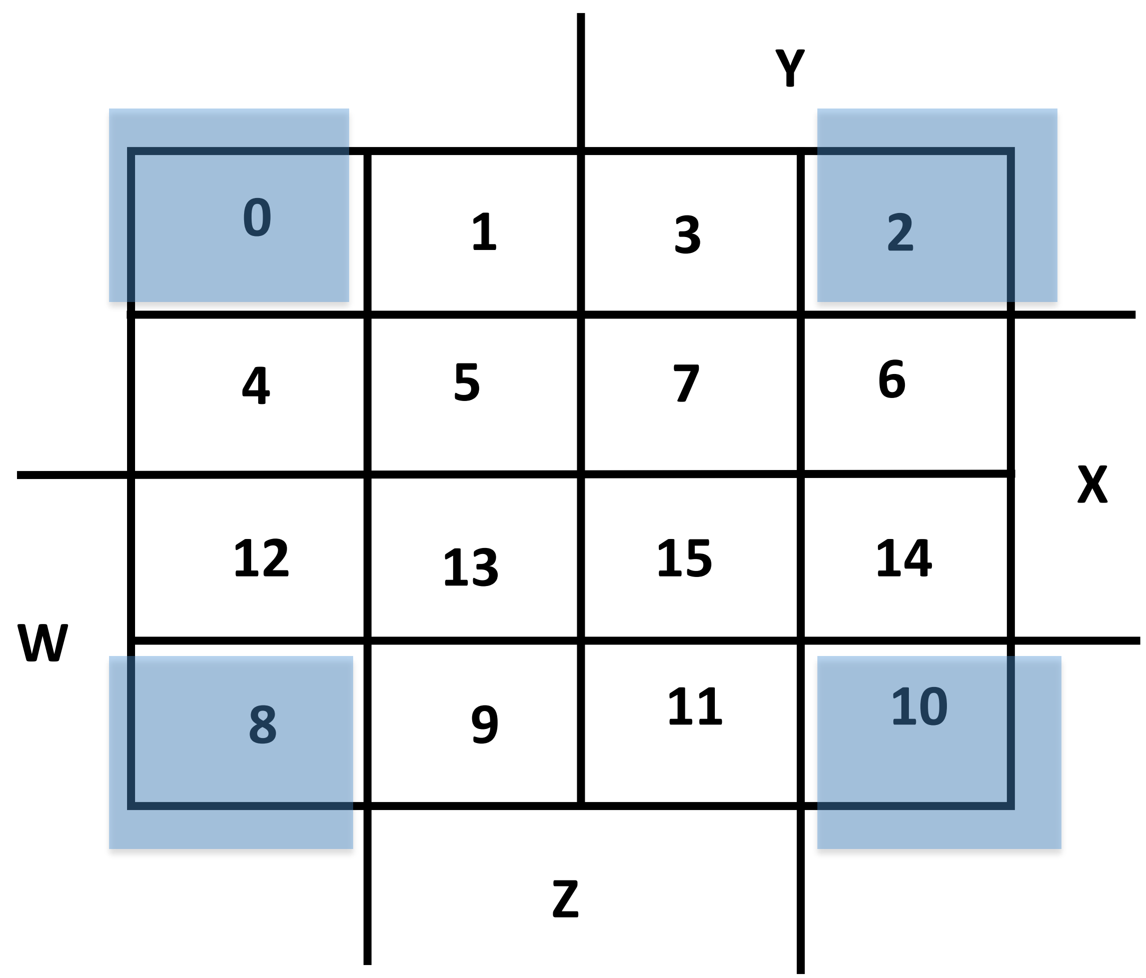
Four Variable Maps

- Map and location of minterms:



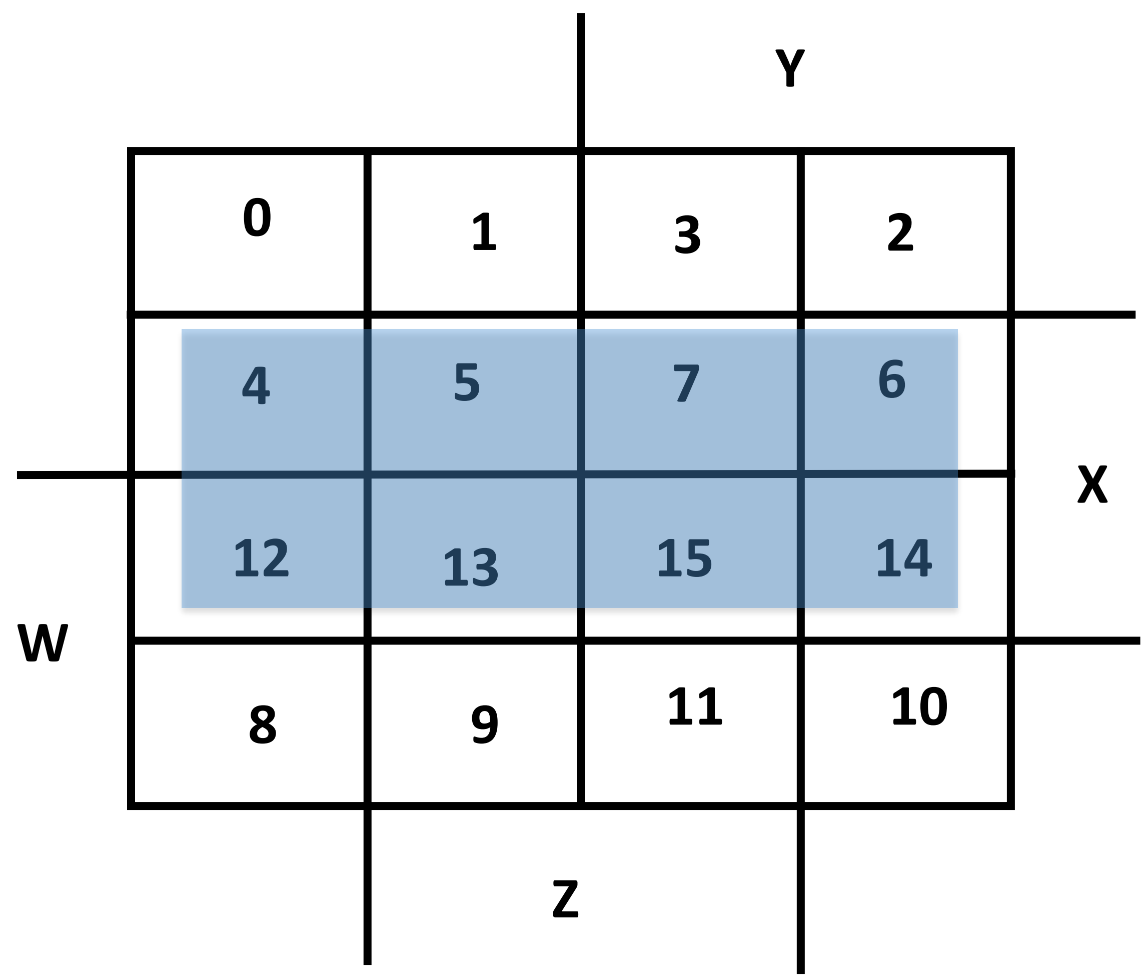
Four Variable Maps

- Map and location of minterms:



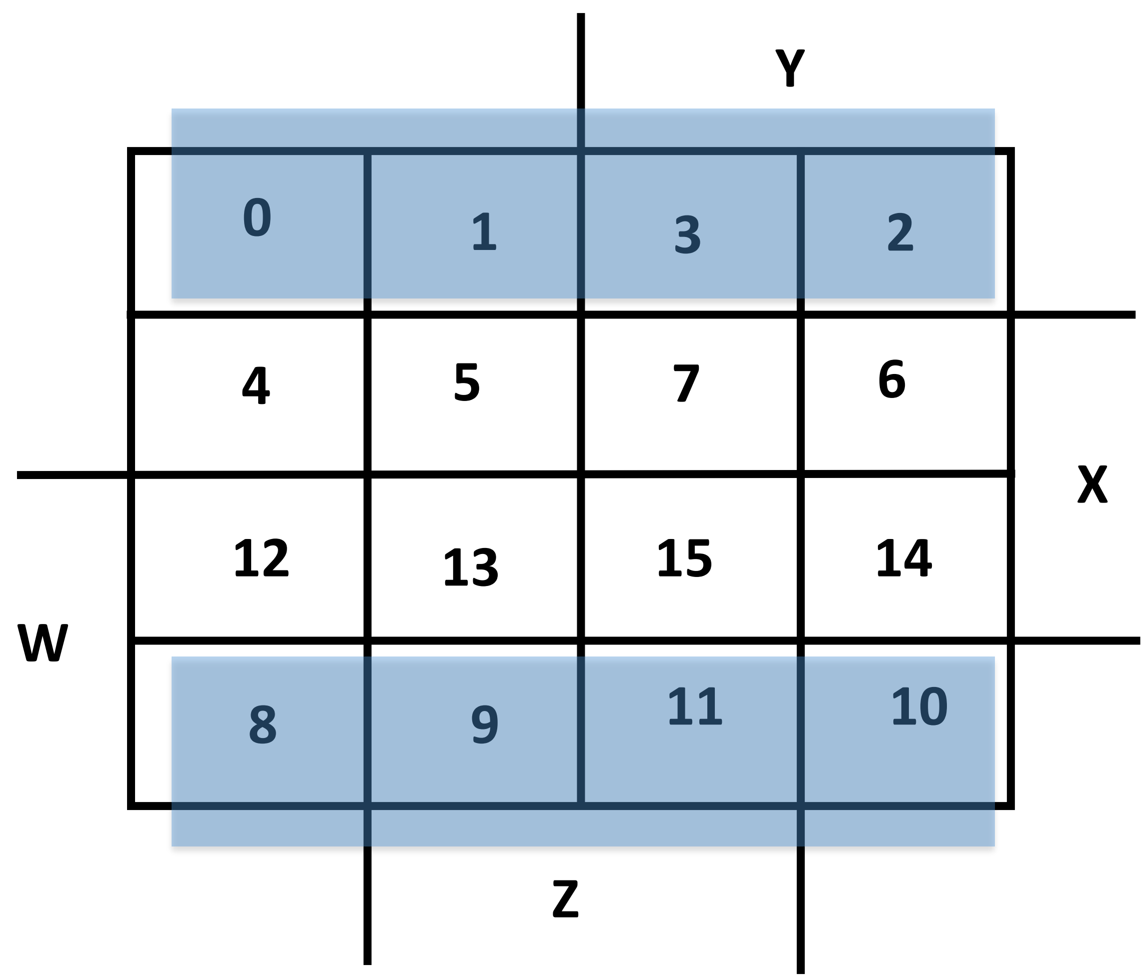
Four Variable Maps

- Map and location of minterms:



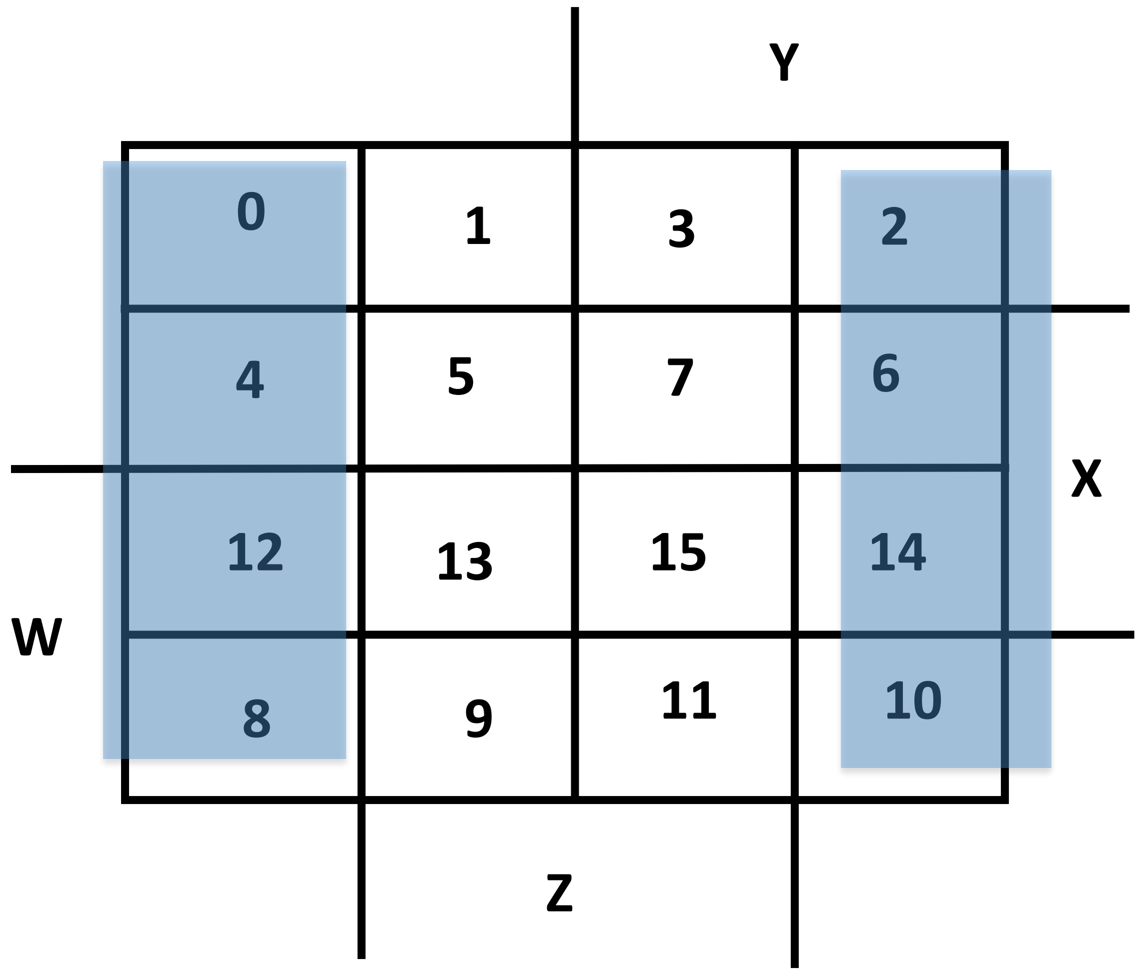
Four Variable Maps

- Map and location of minterms:



Four Variable Maps

- Map and location of minterms:



Four-Variable Map Simplification

- $F(w,x,y,z)=\boldsymbol{\Sigma}_m(0,2,4,5,6,7,8,10,13,15)$

Four-Variable Map Simplification

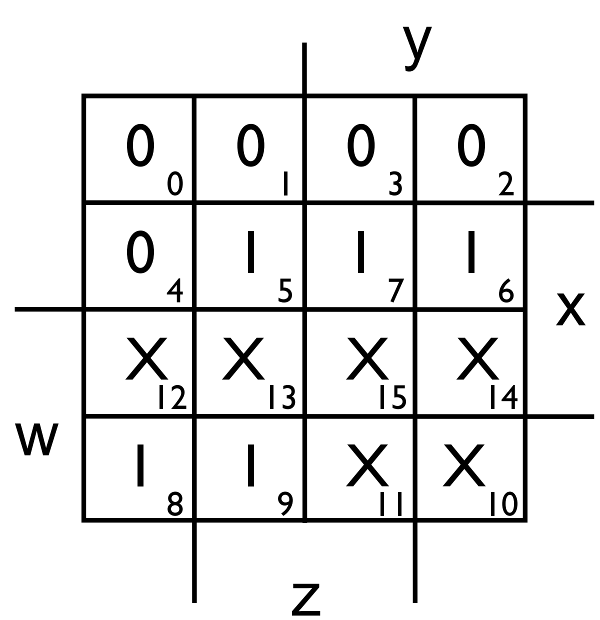
- $F(w,x,y,z)=\Sigma_m(3,4,5,7,9,13,14,15)$

Don't Cares in K-Maps

- Sometimes a function table or map contains entries for which it is known:
  - the input values for the minterm will never occur, or
  - The output value for the minterm is not used
- In these cases, the output value need not be defined
  - Instead, the output value is defined as a "**don't care**"
- By placing "don't cares" (an "x" entry) in the function table or map, the cost of the logic circuit may be lowered.

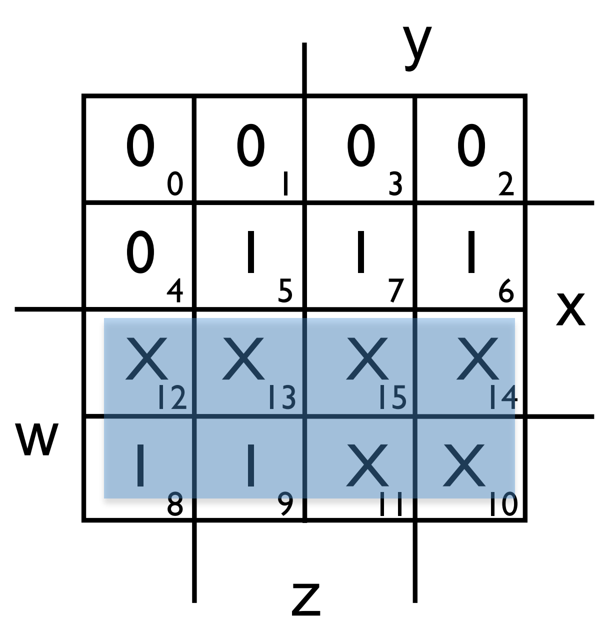
Example: BCD “5 or More”

- The map below gives a function F(w,x,y,z) which is defined as "5 or more" over BCD inputs.
  - With the don't cares used for the 6 non-BCD combinations:



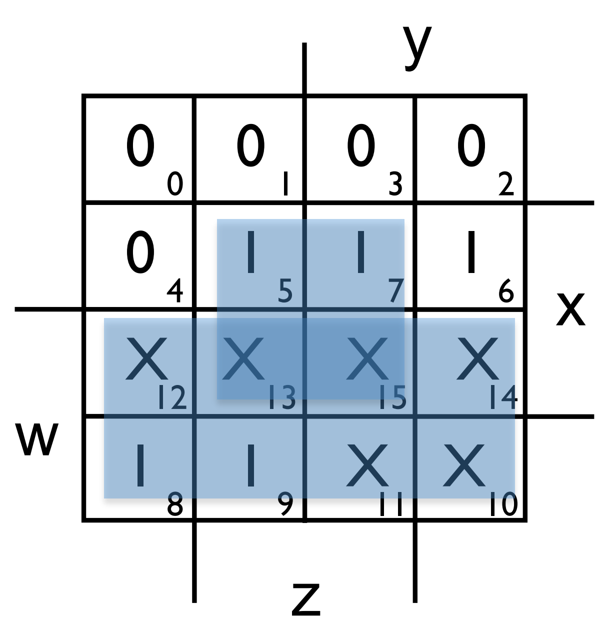
Example: BCD “5 or More”

- The map below gives a function F(w,x,y,z) which is defined as "5 or more" over BCD inputs.
  - With the don't cares used for the 6 non-BCD combinations:



Example: BCD "5 or More"

- The map below gives a function F(w,x,y,z) which is defined as "5 or more" over BCD inputs.
  - With the don't cares used for the 6 non-BCD combinations:



Example: BCD “5 or More”

- The map below gives a function F(w,x,y,z) which is defined as "5 or more" over BCD inputs.
  - With the don't cares used for the 6 non-BCD combinations: