Logic and Computer Design Fundamentals

Chapter 2 – Combinational Logic Circuits

Part 2 – Circuit Optimization

Charles Kime & Thomas Kaminski

Circuit Optimization

- Goal: To obtain the simplest implementation for a given function
- Optimization is a more formal approach to simplification that is performed using a specific procedure or algorithm
- Optimization requires a cost criterion to measure the simplicity of a circuit
- Two distinct cost criteria we will use:
  - Literal cost (L)
  - Gate input cost (G)
  - Gate input cost with NOTs (GN)

Literal Cost

- Literal – a variable or its complement
- Literal cost – the number of literal **appearances** in a Boolean expression corresponding to the logic circuit diagram
- Examples:
  - $F = BD + A\overline{B}C + A\overline{C}\,\overline{D}$ L = 8
  - $F = BD + A\overline{B}C + A\overline{B}\,\overline{D} + AB\overline{C}$ L =
  - $F = (A + B)(A + D)(B + C + \overline{D})(\overline{B} + \overline{C} + D)$ L =
  - Which solution is best?

Gate Input Cost

- **Gate input costs - the number of inputs to the gates in the implementation corresponding exactly to the given equation or equations.** **(G - inverters not counted, GN - inverters counted)**
- **For SOP and POS equations, it can be found from the equation(s) by finding the sum of:**
  - **all literal appearances**
  - **the number of terms excluding terms consisting only of a single literal,(G) and**
  - **optionally, the number of distinct complemented single literals (GN).**
- **Example:**
  - $F = BD + A\overline{B}C + A\overline{C}\,\overline{D}$ **G = 11, GN = 14**
  - $F = BD + A\overline{B}C + A\overline{B}\,\overline{D} + AB\overline{C}$ **G = , GN =**
  - $F = (A + \overline{B})(A + D)(B + C + \overline{D})(\overline{B} + \overline{C} + D)$ **G = , GN =**
  - **Which solution is best?**

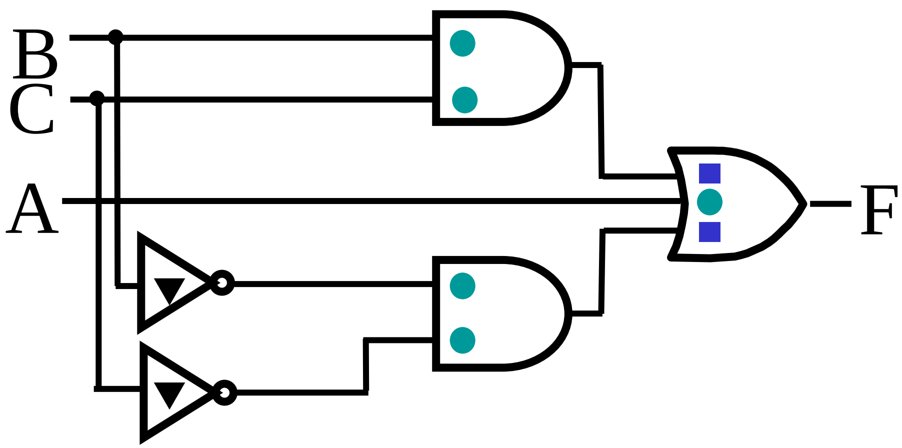
Cost Criteria (continued)

- **Example 1:**
- **F = A + B C + $\bar{B}$ $\bar{C}$**

GN = G + 2 = 9

L = 5

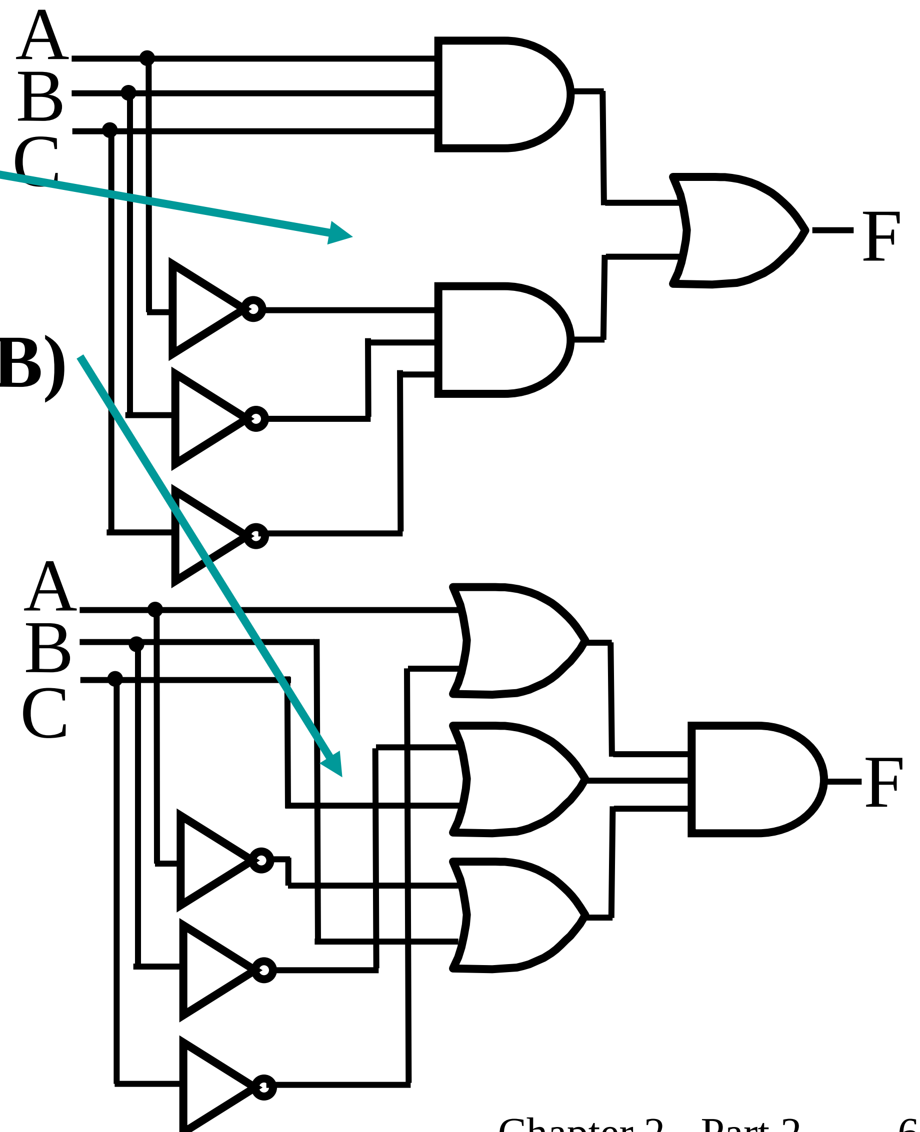
G = L + 2 = 7



- **L (literal count) counts the AND inputs and the single literal OR input.**
- **G (gate input count) adds the remaining OR gate inputs**
- **GN(gate input count with NOTs) adds the inverter inputs**

Cost Criteria (continued)

- **Example 2:**
- **$F = A\,B\,C + \bar{A}\,\bar{B}\,\bar{C}$**
- **L = 6 G = 8 GN = 11**
- **$F = (A + \bar{C})(\bar{B} + C)(\bar{A} + B)$**
- **L = 6 G = 9 GN = 12**
- **<u>Same</u> function and <u>same</u> literal cost**
- **But first circuit has <u>better</u> gate input count and <u>better</u> gate input count with NOTs**
- **Select it!**



Boolean Function Optimization

- Minimizing the gate input (or literal) cost of a (a set of) Boolean equation(s) reduces circuit cost.
- We choose gate input cost.
- Boolean Algebra and graphical techniques are tools to minimize cost criteria values.
- Some important questions:
  - When do we stop trying to reduce the cost?
  - Do we know when we have a minimum cost?
- Treat optimum or near-optimum cost functions for two-level (SOP and POS) circuits first.
- Introduce a graphical technique using Karnaugh maps (K-maps, for short)

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 one variable**
  - **Alternative algebraic expressions for the same function are derived by recognizing patterns of squares**
- **The K-map can be viewed as**
  - **A reorganized version of the truth table**
  - **A topologically-warped Venn diagram as used to visualize sets in algebra of sets**

Some Uses of K-Maps

- Provide a means for:
  - Finding optimum or near optimum
    - SOP and POS standard forms, and
    - two-level AND/OR and OR/AND circuit implementations

    for functions with small numbers of variables
  - Visualizing concepts related to manipulating Boolean expressions, and
  - Demonstrating concepts used by computer-aided design programs to simplify large circuits

Two Variable Maps

- A 2-variable Karnaugh Map:
  - Note that minterm m0 and minterm m1 are "adjacent" and differ in the value of the variable y
  - Similarly, minterm m0 and minterm m2 differ in the x variable.
  - Also, m1 and m3 differ in the x variable as well.
  - Finally, m2 and m3 differ in the value of the variable y

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

K-Map and Truth Tables

- The K-Map is just a different form of the truth table.
- Example – Two variable function:
  - We choose a,b,c and d from the set {0,1} to implement a particular function, F(x,y).

Function Table

Input Values (x,y)	Function Value F(x,y)
0 0	a
0 1	b
1 0	c
1 1	d

K-Map

	y = 0	y = 1
x = 0	a	b
x = 1	c	d

K-Map Function Representation

- **Example: F(x,y) = x**

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

- **For function F(x,y), the two adjacent cells containing 1's can be combined using the Minimization Theorem:**

$$F(x, y) = x\,\overline{y} + x\,y = x$$

K-Map Function Representation

- **Example: $G(x,y) = x + y$**

G = x+y	y = 0	y = 1
x = 0	0	1
x = 1	1	1

- **For G(x,y), two pairs of adjacent cells containing 1's can be combined using the Minimization Theorem:**

$$G(x, y) = \left(x\,\bar{y} + x\,y\right) + \left(xy + \bar{x}\,y\right) = x + y$$

Duplicate xy

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

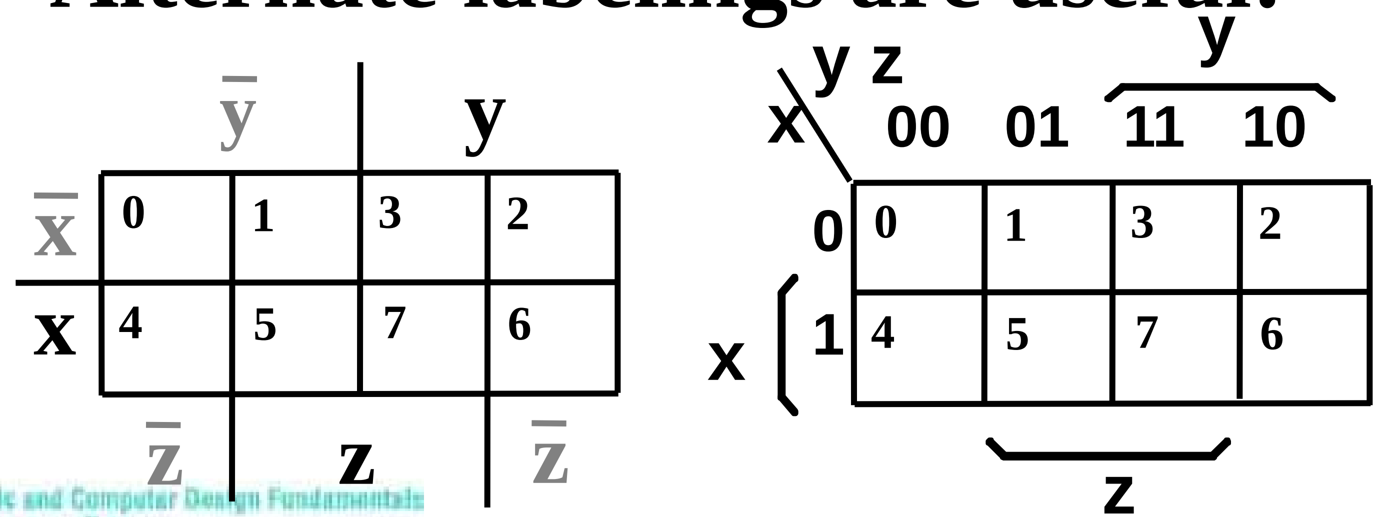
- **Where each minterm corresponds to the product terms:**

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

- **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 $\overline{F}$ blank
- Example:

  $F(x, y, z) = \Sigma_m(2,3,4,5)$

			y	y
	0	1	3 1	2 1
x	4 1	5 1	7	6
		z	z	

- Example:

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

			b	b
	0	1	3 1	2
a	4 1	5	7 1	6 1
		c	c	

- <u>Learn</u> the locations of the 8 indices based on the variable order shown (x, most significant and z, least significant) on the map boundaries

Combining Squares

- By combining squares, we reduce number of literals in a product term, reducing the literal cost, thereby reducing the other two cost criteria
- 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 (no variables) = 1.

Example: Combining Squares

- **Example: Let** $F = \Sigma m(2,3,6,7)$

			y	y
	0	1	3 **1**	2 **1**
x	4	5	7 **1**	6 **1**
		z	z	

- **Applying the Minimization Theorem three times:**

$$
\begin{aligned}
F(x,y,z) &= \bar{x}\,y\,z + x\,y\,z + \bar{x}\,y\,\bar{z} + x\,y\,\bar{z} \\
&= yz + y\bar{z} \\
&= y
\end{aligned}
$$

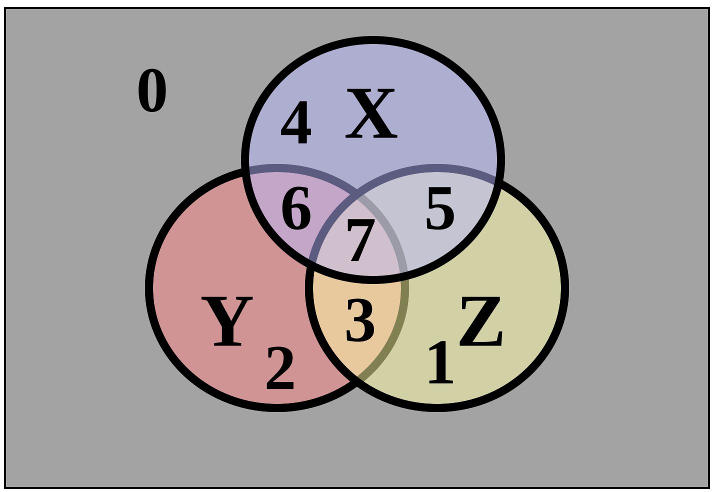
- **Thus the four terms that form a 2 × 2 square correspond to the term "y".**

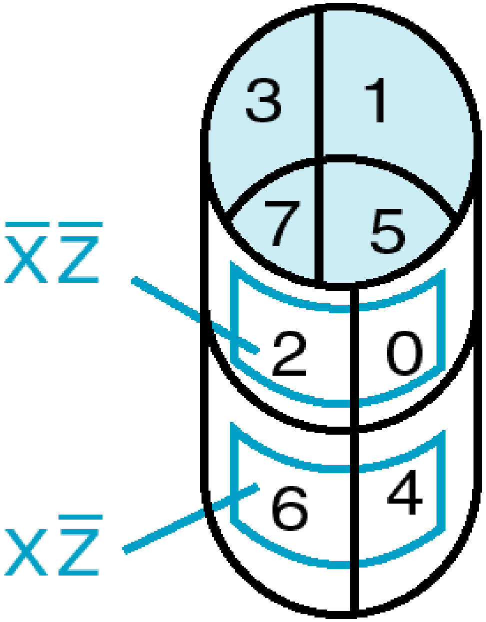
Three-Variable Maps

- **Reduced literal product terms for SOP standard forms correspond to <u>rectangles</u> on K-maps containing cell counts that are powers of 2.**
- **Rectangles of 2 cells represent 2 adjacent minterms; of 4 cells represent 4 minterms that form a "pairwise adjacent" ring.**
- **Rectangles can contain non-adjacent cells as illustrated by the "pairwise adjacent" ring above.**

Three-Variable Maps

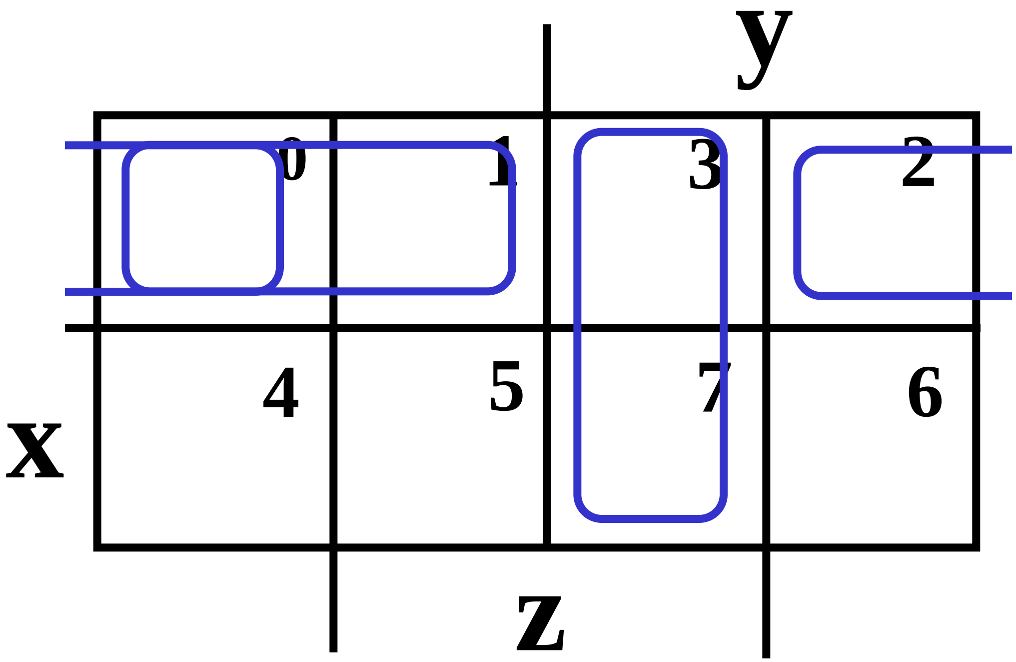
- **Topological warps of 3-variable K-maps that show *all* adjacencies:**
  - **Venn Diagram**
  - **Cylinder**





Three-Variable Maps

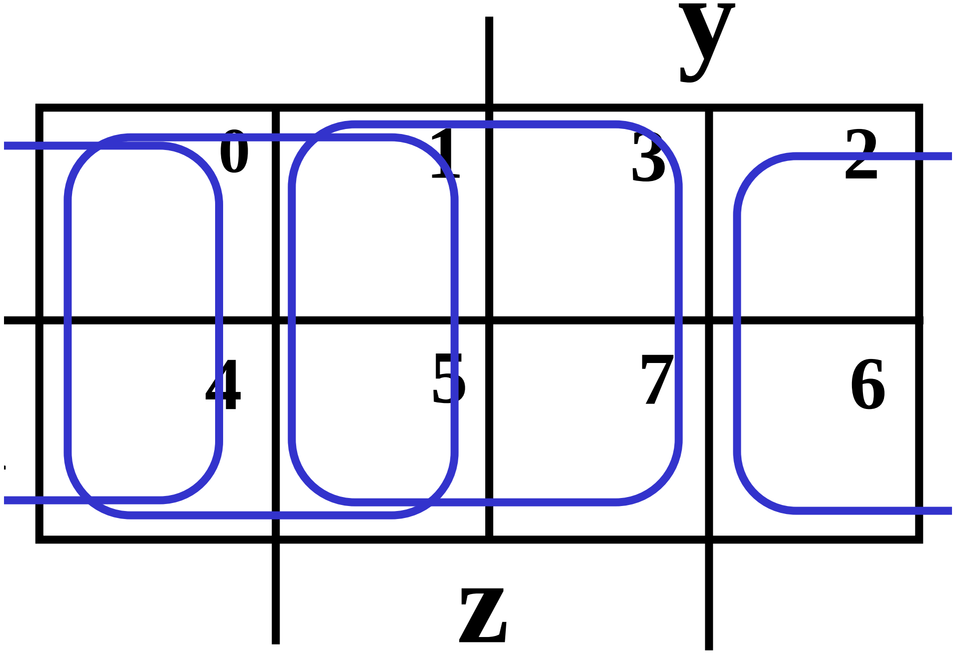
- Example Shapes of 2-cell Rectangles:



- Read off the product terms for the rectangles shown

Three-Variable Maps

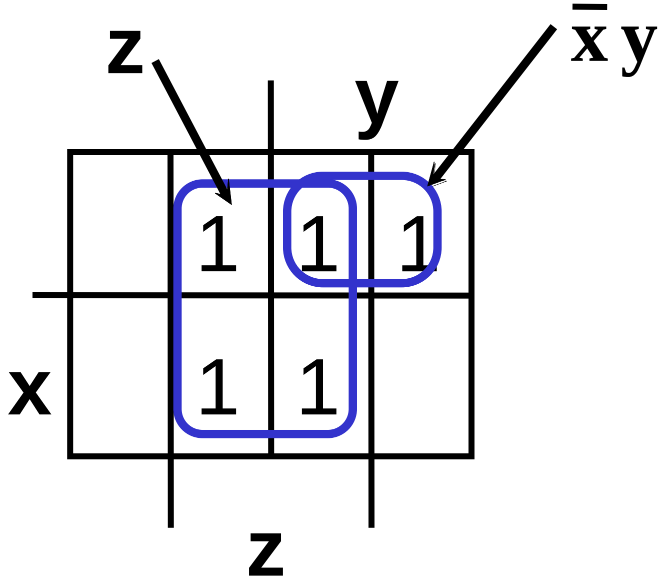
- **Example Shapes of 4-cell Rectangles:**



- **Read off the product terms for the rectangles shown**

Three Variable Maps

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



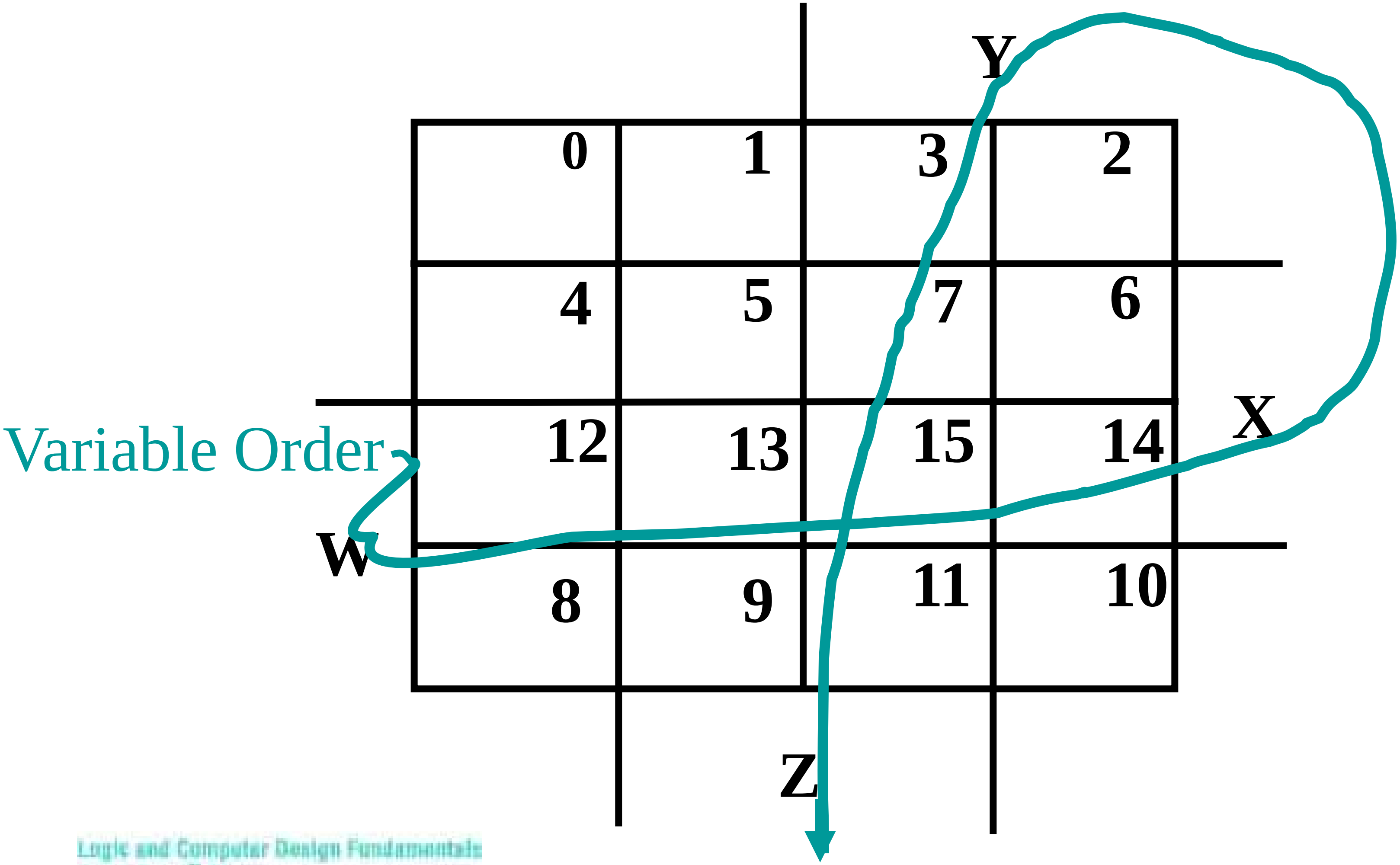
$$F(x, y, z) = z + \bar{x}\,y$$

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:**

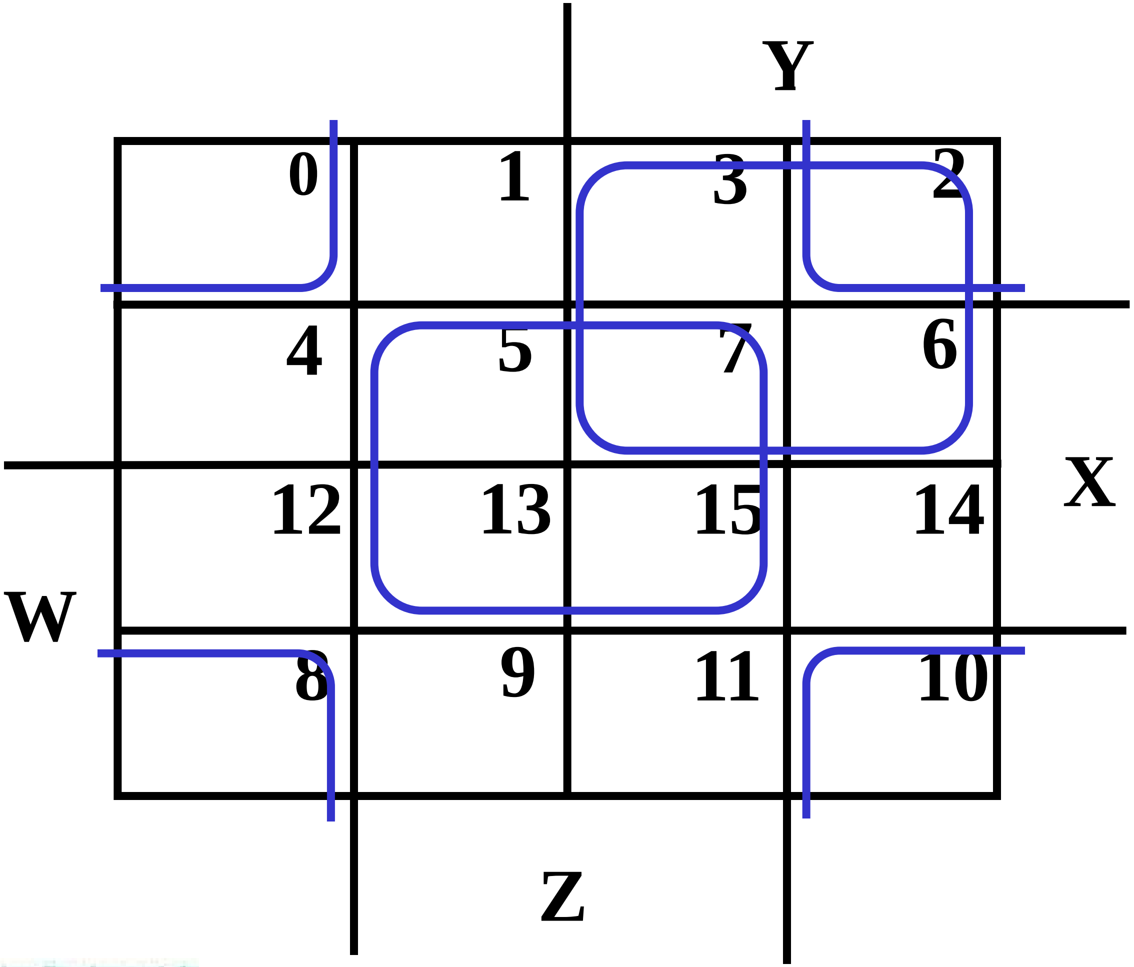


Four Variable Terms

- **Four variable maps can have rectangles corresponding to:**
  - **A single 1 = 4 variables, (i.e. Minterm)**
  - **Two 1s = 3 variables,**
  - **Four 1s = 2 variables**
  - **Eight 1s = 1 variable,**
  - **Sixteen 1s = zero variables (i.e. Constant "1")**

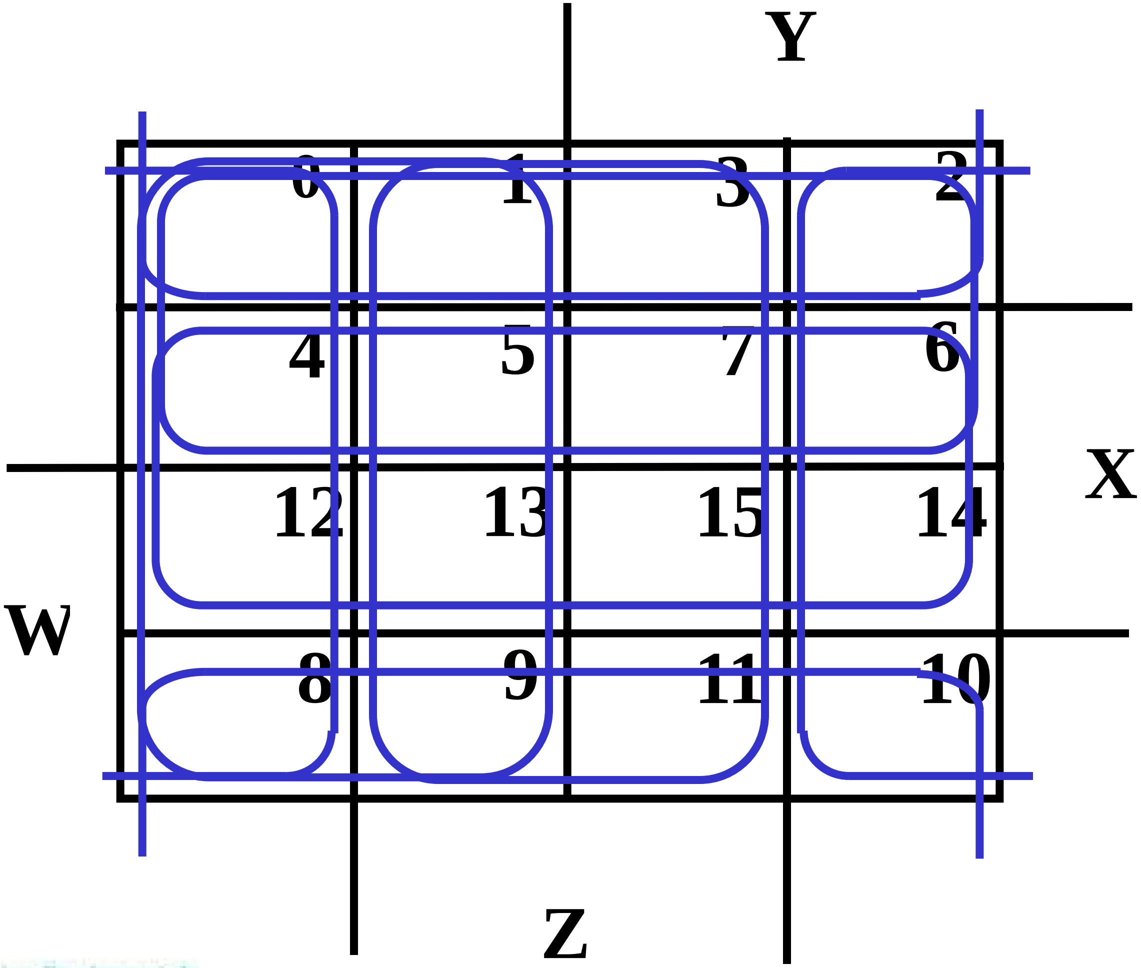
Four-Variable Maps

- **Example Shapes of Rectangles:**



Four-Variable Maps

- **Example Shapes of Rectangles:**



Four-Variable Map Simplification

- **$F(W, X, Y, Z) = \Sigma_m(0, 2,4,5,6,7,8,10,13,15)$**

Four-Variable Map Simplification

- $F(W, X, Y, Z) = \Sigma_m(3,4,5,7,9,13,14,15)$