# Minimization and Karnaugh Maps

VOLKAN KURSUN

*Some material from McGraw Hill*





## Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)





## Minimization and Karnaugh Maps

- We have used algebraic manipulation utilizing the axioms, theorems, and properties of Boolean algebra to find reduced-cost implementation of a function in either sum-of-products or product-of-sums form: can be quite tedious and success not guaranteed
- **Karnaugh map**: provides **a systematic way** of producing a **minimum-cost logic expression**





## Key to Karnaugh Maps

- Karnaugh maps apply **the combining property in a systematic way** for logic minimization:

$$14a:\ x \cdot y + x \cdot y' = x$$

- The combining property **replaces the sum of two minterms that differ in the value of only one variable with a single product term that does not include the different variable**: two minterms simplified to one product term

# Key to Karnaugh Maps

- Karnaugh maps apply **the combining property in a systematic way** for logic minimization:

$$14b: (x + y) . (x + y') = x$$

- The combining property **replaces the product of two maxterms that differ in the value of only one variable with a single sum term that does not include the different variable**: two maxterms simplified to one sum term





# Combining Property in Action: Example

- Consider the function with the following truth table:

$$f = x_1'x_2'x_3' + x_1'x_2x_3' + x_1x_2'x_3' + x_1x_2'x_3 + x_1x_2x_3' = \sum m(0,2,4,5,6)$$

| Row number | $x_1$ | $x_2$ | $x_3$ | $f$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 1 |
| 3 | 0 | 1 | 1 | 0 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 0 |

The combining property replaces sum of two minterms that differ in the value of only one variable with a single product term that does not include the different variable: sum of two minterms simplified to one product term

Example-1: $m_0$ OR $m_2$

Example-2: $m_4$ OR $m_6$





# Combining Property in Action: Example

- $m_0$ and $m_2$ differ only in the value of one variable $x_2$: **combine $m_0$ OR $m_2$ into one simplified product term without $x_2$**

$$f1 = x_1'x_2'x_3' + x_1'x_2x_3' = x_1'x_3'(x_2' + x_2) = x_1'x_3'$$

- $m_4$ and $m_6$ differ only in the value of one variable $x_2$: **combine $m_4$ OR $m_6$ into one simplified product term without $x_2$**

$$f2 = x_1x_2'x_3' + x_1x_2x_3' = x_1x_3'(x_2' + x_2) = x_1x_3'$$

- The two newly generated product terms **can be further simplified** by the combining property as

$$f3 = f1 + f2 = x_1'x_3' + x_1x_3' = (x_1' + x_1)x_3' = x_3'$$





# Combining Property in Action: Example

- The remaining minterm **$m_5$ can be combined with $m_4$**:

$$f4 = x_1x_2'x_3' + x_1x_2'x_3 = x_1x_2'$$

Note: using theorem 7b ($m_4 = m_4 + m_4$), minterm $m_4$ was used twice (replicated) and combined with minterms $m_0$, $m_2$, and $m_6$ to yield the simplifed term $x_3'$ and combined with $m_5$ to yield the simplified product term $x_1x_2'$

- All minterms m(0,2,4,5,6) in f are covered with these two simplified terms, thereby providing **the minimum cost-expression** for f:

$$f = f3 + f4 = x_3' + x_1x_2'$$

## Combining Property in Action: Example

- These optimization steps indicate that four minterms $m_0$, $m_2$, $m_4$, and $m_6$ can be replaced with a single term $x_3'$

| Row | $x_1$ | $x_2$ | $x_3$ | $f$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 2 | 0 | 1 | 0 | 1 |
| 4 | 1 | 0 | 0 | 1 |
| 6 | 1 | 1 | 0 | 1 |

m(0, 2, 4, 6) represent all possible minterms for which $x_3 = 0$. And f = 1 for these minterms. This implies that f = 1 when $x_3 = 0$, regardless of the values of $x_1$ and $x_2$

- Similarly, two minterms $m_4$ and $m_5$ can be replaced with a single product term $x_1x_2'$

| Row | $x_1$ | $x_2$ | $x_3$ | $f$ |
|---|---|---|---|---|
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 1 |

m(4, 5) represent all possible minterms for which $x_1 = 1$ and $x_2 = 0$. And f = 1 for these minterms. This implies that f = 1 when $x_1 = 1$ and $x_2 = 0$, regardless of the value of $x_3$





## Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)





## Karnaugh Map

- Karnaugh map provides **an easy (visual) way** to discover groups of minterms for which f = 1 and can be combined into simpler terms
- Karnaugh map is **an alternative to the truth table for representing a logic function**: the map consists of **cells (squares) that correspond to the minterms (rows) of the truth table**





## Two-Variable Karnaugh Map

- Four cells: columns are labeled by the value of $x_1$. Rows are labeled by the value of $x_2$.

| $x_2$ | $x_1$ | |
|---|---|---|
| 0 | 0 | $m_0$ |
| 0 | 1 | $m_1$ |
| 1 | 0 | $m_2$ |
| 1 | 1 | $m_3$ |

(a) Truth table

| $x_2 \backslash x_1$ | 0 | 1 |
|---|---|---|
| 0 | $m_0$ | $m_1$ |
| 1 | $m_2$ | $m_3$ |

(b) Karnaugh map

# Two-Variable Karnaugh Map

- Karnaugh map **allows easy recognition of minterms that can be combined** using property 14a: **minterms in any two cells that are adjacent, either in the same row or the same column, can be combined if the function is true for those minterms**

| $x_2$ | $x_1$ | |
|---|---|---|
| 0 | 0 | $m_0$ |
| 0 | 1 | $m_1$ |
| 1 | 0 | $m_2$ |
| 1 | 1 | $m_3$ |

(a) Truth table

| $x_2 \backslash x_1$ | 0 | 1 |
|---|---|---|
| 0 | $m_0$ | $m_1$ |
| 1 | $m_2$ | $m_3$ |

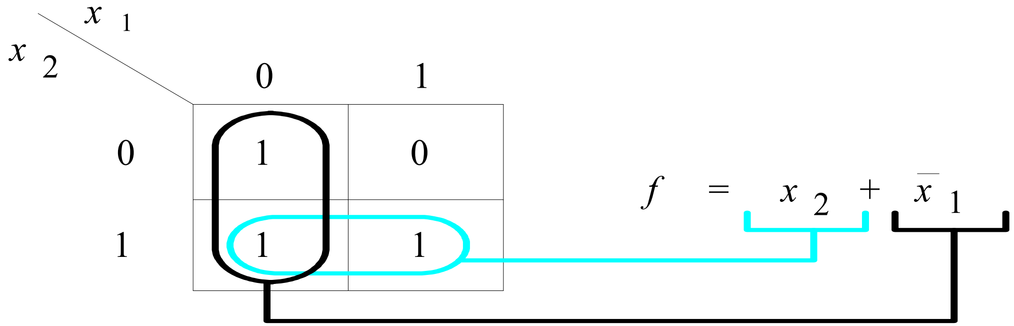
(b) Karnaugh map





# Two-Variable Map Example

- **Blue circle** at the bottom: **if $x_2 = 1$, f = 1 regardless of $x_1$**. The product term that represents these two cells is therefore $x_2$
- **Black circle**: **if $x_1 = 0$, f = 1 regardless of $x_2$**. The product term that represents these two cells is therefore $x_1'$

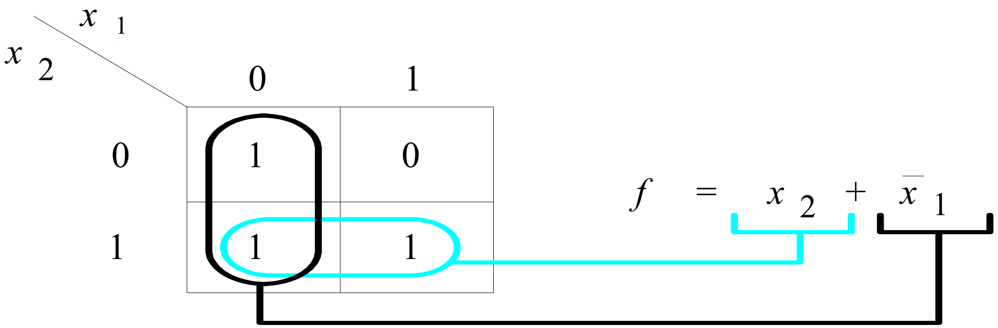







# Two-Variable Map Example

- Blue circle and black circle cover all the minterms for which the function is 1 (f = 1)
- Rule **to find the minimum-cost implementation**: find the **smallest number of product terms with the smallest number of variables that produce f = 1**








# Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)

# Three-Variable Karnaugh Map

- To ensure that minterms in adjacent cells can be combined into a single product term, the **adjacent cells must differ in the value of only one variable**
- Gray code: a sequence of codes where consecutive codes differ in one variable only

| $x_3$ | $x_2$ | $x_1$ | |
|---|---|---|---|
| 0 | 0 | 0 | $m_0$ |
| 0 | 0 | 1 | $m_1$ |
| 0 | 1 | 0 | $m_2$ |
| 0 | 1 | 1 | $m_3$ |
| 1 | 0 | 0 | $m_4$ |
| 1 | 0 | 1 | $m_5$ |
| 1 | 1 | 0 | $m_6$ |
| 1 | 1 | 1 | $m_7$ |

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

(b) Karnaugh map





# Three-Variable Karnaugh Map

- The first and fourth columns also differ only in one variable x2: these two columns are also considered to be adjacent (**assume the left and right edges of the map touch**)

| $x_3$ | $x_2$ | $x_1$ | |
|---|---|---|---|
| 0 | 0 | 0 | $m_0$ |
| 0 | 0 | 1 | $m_1$ |
| 0 | 1 | 0 | $m_2$ |
| 0 | 1 | 1 | $m_3$ |
| 1 | 0 | 0 | $m_4$ |
| 1 | 0 | 1 | $m_5$ |
| 1 | 1 | 0 | $m_6$ |
| 1 | 1 | 1 | $m_7$ |

First column    Fourth column

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

(b) Karnaugh map





# Three-Variable Map Example

- For a minimum-cost implementation, cover the four 1s in the map as efficiently as possible

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |

$$f = x_3'x_2 + x_3x_1'$$





# Three-Variable Map Example-2

- Product terms can be formed by either a single cell, or combining two adjacent cells, or combining four adjacent cells, or combining all cells (f is permanently 1, f = 1)
- **Red circle**: **if $x_3 = 0$, then f = 1 regardless of $x_2$ and $x_1$ (for all possible values of $x_2$ and $x_1$, f = 1 if $x_3 = 0$).** Therefore, the product term **$x_3'$** represents these four cells

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |

$$f = x_3' + x_2x_1'$$

# Three-Variable Map Example-2

- Product terms can be formed by either a single cell, or combining two adjacent cells, or combining four adjacent cells, or combining all cells (f is permanently 1, f = 1)
- **Black circle**: **if $x_2$ = 1 AND $x_1$ = 0, then f = 1 regardless of $x_3$ (for all possible values of $x_3$, f = 1 if $x_2$ = 1 AND $x_1$ = 0)**. The product term $\mathbf{x_2x_1'}$ represents these two cells

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |

$$f = x_3' + x_2x_1'$$





# Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)





# Four-Variable Karnaugh Map

- The first and fourth columns **differ only in one variable x2**: these two columns are also considered to be adjacent **(assume the left and right edges of the map touch)**: **(m0 and m2), (m4 and m6), (m12 and m14), and (m8 and m10) are adjacent**

First column

Fourth column

| $x_4x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 01 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |
| 11 | $m_{12}$ | $m_{13}$ | $m_{15}$ | $m_{14}$ |
| 10 | $m_8$ | $m_9$ | $m_{11}$ | $m_{10}$ |

$x_2$ $x_3$ $x_4$ $x_1$





# Four-Variable Karnaugh Map

- The first and fourth rows **differ only in one variable x4**: these two rows are also considered to be adjacent **(assume the top and bottom edges of the map touch)**: **(m0 and m8), (m1 and m9), (m3 and m11), and (m2 and m10) are adjacent**

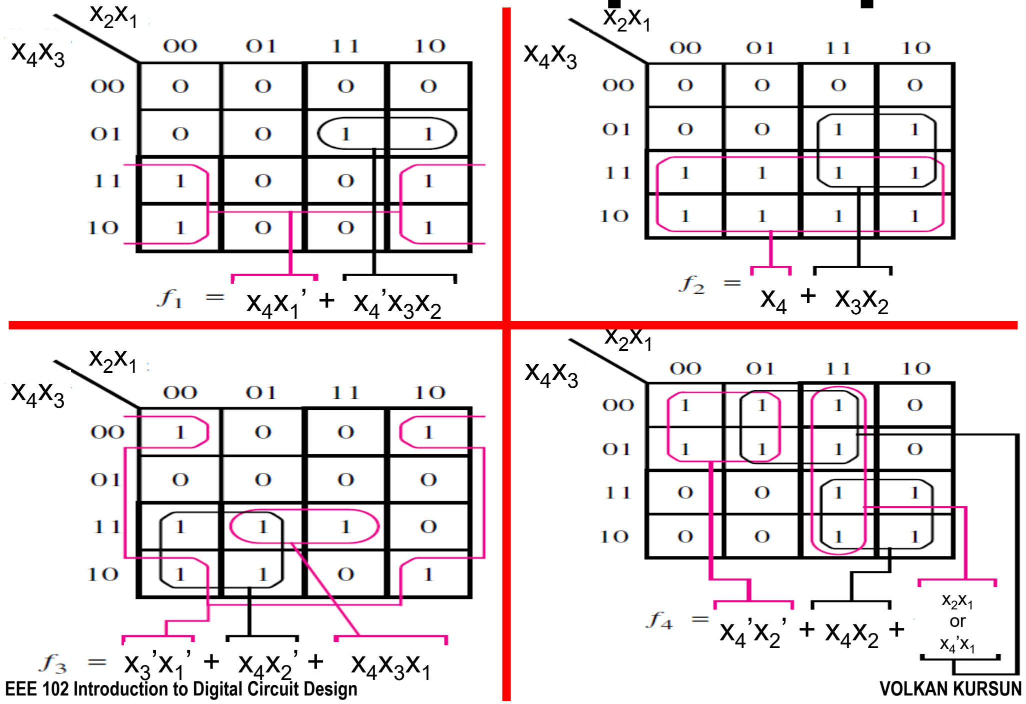
First row

Fourth row

| $x_4x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 01 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |
| 11 | $m_{12}$ | $m_{13}$ | $m_{15}$ | $m_{14}$ |
| 10 | $m_8$ | $m_9$ | $m_{11}$ | $m_{10}$ |

$x_2$ $x_3$ $x_4$ $x_1$

# Four-Variable Map Examples








# Outline

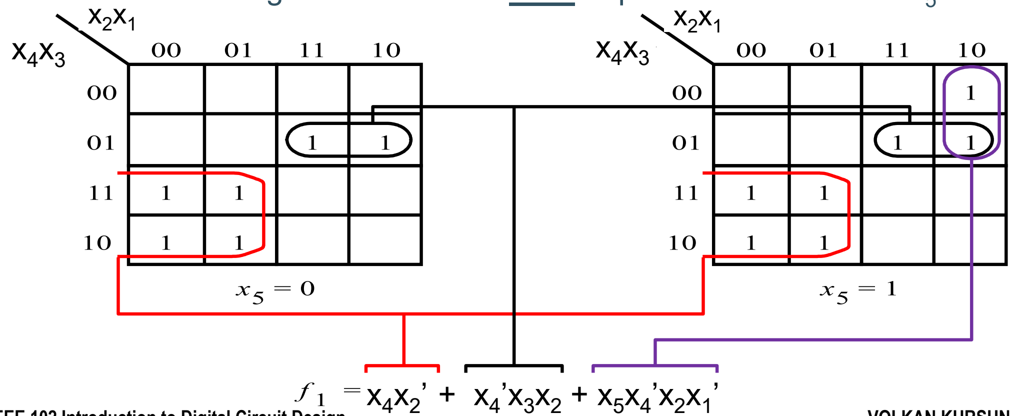
- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)





# Five-Variable Karnaugh Map

- **Two four-variable maps** can be used **to construct a five-variable Karnaugh map**: assume one map is directly above the other and the two maps are distinguished by the fifth variable $x_5$ ($x_5 = 0$ in one map and $x_5 = 1$ in the other overlapping map)
- In the following example, two groups of four 1s (two red circles) appear in the same place (they **overlap**) in both four-variable maps: the minimized logic function does **NOT** depend on the value of $x_5$

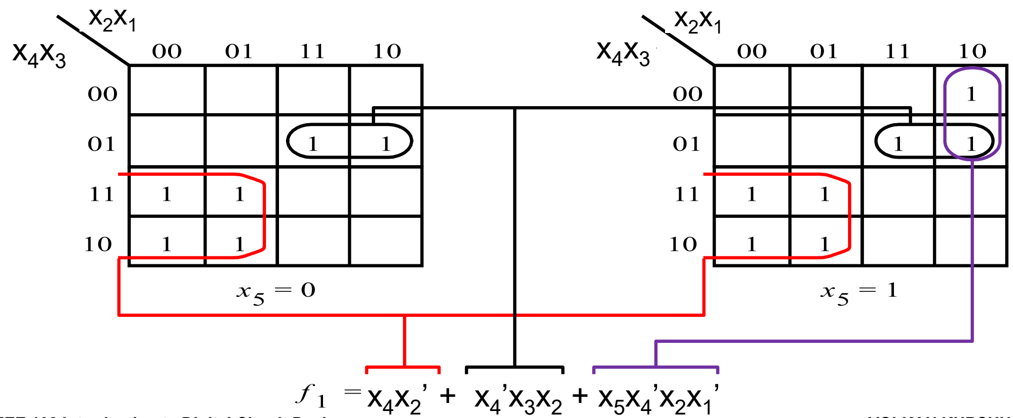







# Five-Variable Karnaugh Map

- The two groups of two 1s (**two black circles**) also **overlap** in the two four-variable maps: the minimized logic function does **NOT depend on the value of $x_5$**
- The group of two 1s (**purple circle**) appears **only in the second map where $x_5 = 1$**: $x_5$ can **NOT** be eliminated, forming a simplified term of $x_5x_4'x_2x_1'$



# Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)





# Intuition for Minimization

- Used intuition to group 1s in a Karnaugh map to obtain a minimum-cost implementation of a logic function
- The larger the group of 1s, the fewer the number of variables in the product term
- **Intuitive strategy**: **minimize the number and maximize the sizes of groups of 1s that cover all cases where the function has a value of 1**

**Goal-1: Minimize the number of groups** to minimize the number of terms in the representation of the function

**Goal-2: Maximize the sizes of groups to minimize the number of variables** in each term in the representation of the function

- Intuition works well for simple functions with small maps





# Strategy for Minimization

- For large functions with many variables, intuitive method and Karnaugh maps are not suitable: **a methodology for logic minimization** is used by the CAD tools
- In the following examples, we will continue to use the Karnaugh maps to illustrate these strategies used by the CAD tools
- Terminology for describing the minimization process:
- **Literal**: **A variable in a product term** is called a literal. A literal can be in either uncomplemented or complemented form.
- Examples: the product term $x_1'x_2'x_3$ has three literals.

The product term abc'de'f'g has 7 literals.





# Terminology for Minimization

- **Implicant**: **Any product term or single variable for which the function is 1** is called an implicant of the function
- The minterms for which the function is equal to 1 are implicants
- Any other product terms or single variables for which the function is 1 are also called implicants of the function
- Example: Identify the implicants for the following three-variable function

$$f(x_3, x_2, x_1) = \Sigma m(0, 1, 4, 5, 7)$$

# Finding the Implicants Example

- Example: The function has 11 implicants.

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_2'$ $x_3x_1$

Three-variable function

$f(x_3, x_2, x_1) = \Sigma m(0, 1, 4, 5, 7)$

- 5 of these 11 implicants are the minterms for which the function is 1: $x_3'x_2'x_1'$, $x_3'x_2'x_1$, $x_3x_2'x_1'$, $x_3x_2'x_1$, $x_3x_2x_1$.
- Remaining 5 of these 11 implicants correspond to all possible pairs of minterms that can be combined: $x_3'x_2'$ (combine $m_0$ and $m_1$), $x_2'x_1'$(combine $m_0$ and $m_4$), $x_2'x_1$ (combine $m_1$ and $m_5$), $x_3x_2'$ (combine $m_4$ and $m_5$), $x_3x_1$ (combine $m_5$ and $m_7$)
- The 11th implicant covers the group of 4 minterms: $x_2'$





# Terminology for Minimization

- **Prime Implicant**: An implicant that **cannot be combined into another implicant with fewer literals** is called a prime implicant
- The **literals in a prime implicant cannot be removed (or further simplified)** as they represent **the largest groups of 1s that can be circled** in the Karnaugh map

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_2'$ $x_3x_1$

Three-variable function

$f(x_3, x_2, x_1) = \Sigma m(0, 1, 4, 5, 7)$

$x_2'$ and $x_3x_1$ are the two prime implicants in this example: it is not possible to remove any literal in these two implicants





# Terminology for Minimization

- **Cover**: A collection of implicants that account for all valuations for which a given function is equal to 1 is called a cover of that function
- **A cover** is **one particular implementation** of a function: **a number of different covers may exist for a function**

Cover examples:

- A set of all minterms for which f = 1 is a cover (canonical sum of products)
- A set of all prime implicants is a cover





# Cover Examples

- The following three are all different covers of the same function

Three-variable function

$f(x_3, x_2, x_1) = \Sigma m(0, 1, 4, 5, 7)$

| $x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_2'$ $x_3x_1$

While all these covers are correct representations of this function, the **cover consisting of the prime implicants leads to the lowest-cost implementation** $f = x_2' + x_3x_1$

1) A cover consisting of the minterms for which f = 1:

$f = x_3'x_2'x_1' + x_3'x_2'x_1 + x_3x_2'x_1' + x_3x_2'x_1 + x_3x_2x_1$

2) Another cover: $f = x_3'x_2' + x_3x_2' + x_3x_1$

3) Another cover with the prime implicants: $f = x_2' + x_3x_1$

# Terminology for Minimization

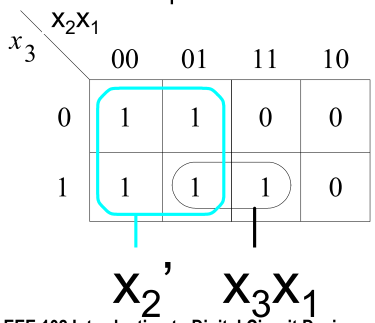
- **Lowest cost implementation is achieved with a cover consisting of prime implicants**
- Some prime implicants may be included in the minimum cost implementation while others may not (there are options)
- How to determine the optimum subset of prime implicants that cover a function with the minimum cost?
- **Essential Prime Implicant**: If a prime implicant includes a minterm (for which f = 1) that is **NOT** included in any other prime implicant, it is called an essential prime implicant





# Essential Prime Implicants

- **Essential prime implicants must be included in the minimum cost cover**
- Example: In the following example, both prime implicants are essential. $x_3x_1$ is the only prime implicant that covers the minterm $m_7$ and $x_2'$ is the only prime implicant that covers the minterms $m_0$, $m_1$, and $m_4$.




Three-variable function
$f(x_3, x_2, x_1) = \Sigma m(0, 1, 4, 5, 7)$

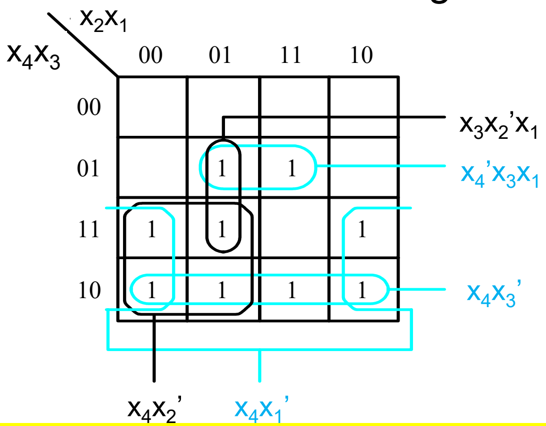
**The minimum cost cover consists of the essential prime implicants: $f = x_2' + x_3x_1$**





# Essential Prime Implicants Example-2

- Few examples with choices on which prime implicants to include in the final cover
- Consider the following 4-variable function




The essential prime implicants (highlighted with blue circles) must be included in the minimum-cost cover

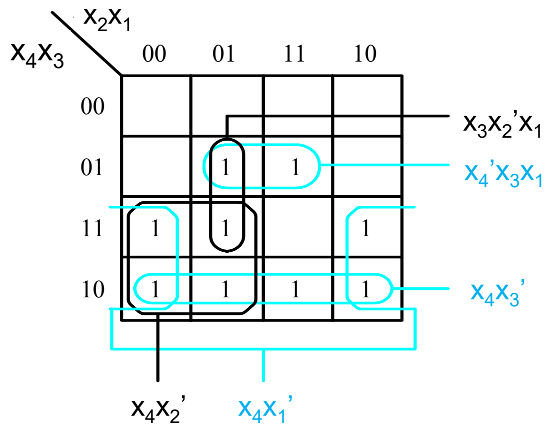
There are 5 prime implicants: $x_4x_2'$, $x_4x_1'$, $x_4x_3'$, $x_4'x_3x_1$, $x_3x_2'x_1$
3 of these 5 prime implicants are essential (highlighted with blue circles): $x_4x_1'$ (because of $m_{14}$), $x_4x_3'$ (because of $m_{11}$), and $x_4'x_3x_1$ (because of $m_7$)





# Essential Prime Implicants Example-2

- In this example, the essential prime implicants cover all minterms where the f = 1, except $m_{13}$.
- $m_{13}$ is covered by two prime implicants: $x_4x_2'$ and $x_3x_2'x_1$. $x_4x_2'$ is preferred since it has fewer literals.




The minimum cost implementation is:
**$f = x_4x_2' + x_4x_1' + x_4x_3' + x_4'x_3x_1$**

# Process of Finding a Minimum Cost Circuit

1) **Step-1**: **Identify all the prime implicants** for the given function

2) **Step-2**: **Identify** the set of **essential prime implicants**

3) **Step-3**: **Check if the set of essential prime implicants covers all the minterms for which f = 1**, then the minimum cost implementation is composed of only the essential prime implicants. Alternatively, **if some minterms** for which f = 1 **are not covered by the essential prime implicants**, **determine the nonessential prime implicants that should be included to form a complete minimum-cost cover**

- The choice of the nonessential prime implicants in step-3 may not be obvious: for large functions there may be many possibilities with similar number of literals and heuristics (arbitrary selection of a smaller subset of options and trial and error) may be used by the CAD tools





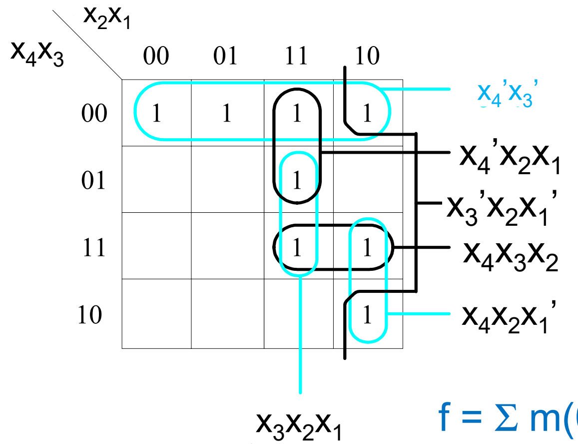
# Minimum Cost Implementation Example-3

- A possible implementation is:

$f = x_4'x_3' + x_4'x_2x_1 + x_4x_3x_2 + x_3'x_2x_1'$

- A second and **better** implementation is:

$f = x_4'x_3' + x_3x_2x_1 + x_4x_2x_1'$




There are 6 prime implicants: $x_4'x_3'$, $x_4'x_2x_1$, $x_3'x_2x_1'$, $x_4x_3x_2$, $x_4x_2x_1'$, $x_3x_2x_1$
Only 1 of these 6 prime implicants is essential: $x_4'x_3'$ (because of $m_0$ and $m_1$)

$f = \Sigma\, m(0, 1, 2, 3, 7, 10, 14, 15)$





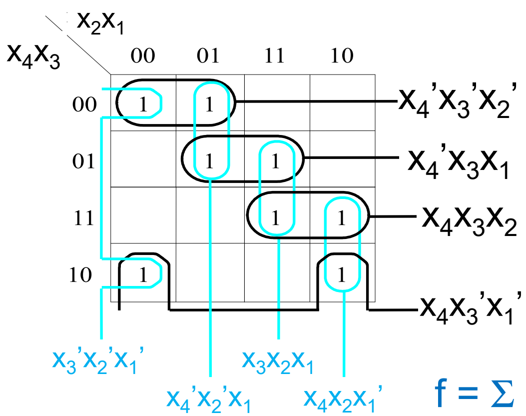
# Minimum Cost Implementation Example-4

- A possible implementation (includes the **prime implicants in black circles**) is:

$f = x_4'x_3'x_2' + x_4'x_3x_1 + x_4x_3x_2 + x_4x_3'x_1'$

- Another implementation which includes the **prime implicants in blue circles** is:

$f = x_3'x_2'x_1' + x_4'x_2'x_1 + x_3x_2x_1 + x_4x_2x_1'$




For some functions, there may be **no essential prime implicants**.
There are 8 prime implicants **none** of which are **essential**:
$x_4'x_3'x_2'$, $x_4'x_3x_1$, $x_4x_3x_2$, $x_4x_3'x_1'$, $x_4x_2x_1'$, $x_3x_2x_1$, $x_4'x_2'x_1$, $x_3'x_2'x_1'$

$f = \Sigma\, m(0, 1, 5, 7, 8, 10, 14, 15)$





# Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)

# Minimization of POS Forms

- To find the minimum-cost product-of-sums implementations, combine the **maxterms** for which f = 0 into groups that are as large as possible: the larger the group is (the more maxterms the group covers) the smaller the number of literals in the sum term and the cost of implementation
- Example:

| $x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_3 + x_2'$

$x_2' + x_1$

**$f(x_3, x_2, x_1) = \Pi M(2, 3, 6)$**
There are 3 maxterms that must be covered: $M_2$, $M_3$, and $M_6$
Two sum terms shown in the figure cover these three maxterms

**$f = (x_3 + x_2') \cdot (x_2' + x_1)$**





# Minimized POS and SOP Comparison

- Example:

| $x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_3 + x_2'$

$x_2' + x_1$

**$f(x_3, x_2, x_1) = \Pi M(2, 3, 6)$**
There are 3 maxterms that must be covered: $M_2$, $M_3$, and $M_6$
Two sum terms shown in the figure cover these three maxterms

**$f = (x_3 + x_2') \cdot (x_2' + x_1)$**
**Cost (POS) = 2 + 2*2 + 1 = 7**

- Compare the minimum-cost POS and the minimum-cost SOP implementations

$f(x_3, x_2, x_1) = \Sigma m(0, 1, 4, 5, 7)$

| $x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_2'$ $\quad x_3x_1$

**The minimum cost cover consists of the essential prime implicants**: **$f = x_2' + x_3.x_1$**
**Cost (SOP) = 1 + 2 + 2 = 5**





# Minimization of POS: Example-2

- Example: **f =** $\Pi$ M(0, 1, 2, 3, 4, 6, 15)

| $x_4x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 1 | 1 | 0 |
| 11 | 1 | 1 | 0 | 1 |
| 10 | 1 | 1 | 1 | 1 |

$x_4+x_3$

$x_4+x_1$

$x_4'+x_3'+x_2'+x_1'$

7 maxterms for which f = 0 can be covered as follows: **$f = (x_4 + x_3) \cdot (x_4 + x_1) \cdot (x_4' + x_3' + x_2' + x_1')$**

**Cost (POS) = 1*4 + 2*2 + 4 + 3 = 15**

- Compare the minimum-cost POS to the minimum-cost SOP:

| $x_4x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | | |
| 01 | | 1 | 1 | |
| 11 | 1 | 1 | | 1 |
| 10 | 1 | 1 | 1 | 1 |

$x_3x_2'x_1$

$x_4'x_3x_1$

$x_4x_3'$

$x_4x_2'$ $\quad x_4x_1'$

The minimum cost implementation is:
**$f = x_4.x_2' + x_4.x_1' + x_4.x_3' + x_4'.x_3.x_1$**

**Cost (SOP) = 1*4 + 3*2 + 3 + 4 = 17**





# Minimization of POS Forms

- **Alternative way** to find the minimum-cost POS implementation: **first find a minimum-cost SOP implementation for f' (complement of f) and then use DeMorgan's theorem** to find the minimum-cost POS implementation
- Example:

| $x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

$x_3 + x_2'$

$x_2' + x_1$

f Karnaugh map

| $x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |

f' Karnaugh map

$f' = \Sigma m(2, 3, 6)$
The simplest SOP implementation of f' is:
**$f' = (x_3'.x_2) + (x_2.x_1')$**
Using DeMorgan's theorem:
**$f = (f')'$**
**$= (x_3 + x_2') \cdot (x_2' + x_1)$**

**Same as the minimum POS implementation found before**

# Minimization of POS Forms

- Alternative way Example-2:

| $x_4x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 1 | 1 | 0 |
| 11 | 1 | 1 | 0 | 1 |
| 10 | 1 | 1 | 1 | 1 |

$x_4+x_3$

$x_4+x_1$

$x_4'+x_3'+x_2'+x_1'$

f Karnaugh map

| $x_4x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 0 | 0 | 1 |
| 11 | 0 | 0 | 1 | 0 |
| 10 | 0 | 0 | 0 | 0 |

f' Karnaugh map

$f' = \Sigma m(0, 1, 2, 3, 4, 6, 15)$

The simplest SOP implementation of f' is:

**$f' = x_4'.x_3' + x_4'.x_1' + x_4.x_3.x_2.x_1$**

Using DeMorgan's theorem:

**$f = (f')'$**

**$= (x_4 + x_3) . (x_4 + x_1) . (x_4' + x_3' + x_2' + x_1')$**

**Same as the minimum POS implementation found before**





# Outline

- Minimization
- Two-Variable Karnaugh Map
- Three-Variable Karnaugh Map
- Four-Variable Karnaugh Map
- Five-Variable Karnaugh Map
- Simplification Methodology
- POS Minimization
- Don't Cares (Incomplete Specs)





# Incompletely Specified Functions

- In some digital systems, certain input combinations can never occur: these input combinations that can never occur are called don't care conditions
- A circuit that has don't care conditions is incompletely specified
- Don't cares can be used to advantage in design: since these valuations will never occur, the **designer may assume the function is either 0 or 1 for these input combinations, whichever is more useful to find a minimum-cost implementation (to form the largest groups with the maximum number of 1s or 0s)**
- **Maximize group sizes by appropriate choice of don't cares**





# Don't Care Example: SOP Implementation

- Consider the following logic function where minterms $m_3$, $m_7$, $m_{11}$, and $m_{15}$ are don't cares:

$f(x_4, x_3, x_2, x_1) = \Sigma m(1,5,8,9,10) + D(3,7,11,15)$

| $x_4x_3 \backslash x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | d | 0 |
| 01 | 0 | 1 | d | 0 |
| 11 | 0 | 0 | d | 0 |
| 10 | 1 | 1 | d | 1 |

$x_4'x_1$

$x_4x_3'$

(a) SOP implementation

To form the largest possible groups of 1s for the lowest-cost prime implicants, **assume that the don't cares $D_3$, $D_7$, and $D_{11}$ are 1s while $D_{15}$ is 0**.

With these assumptions, there are only two prime implicants that provide a complete cover of f. The simplest SOP implementation of f is:

**$f = x_4'.x_1 + x_4.x_3'$**

# Don’t Care Example: POS Implementation

- Consider the following logic function where minterms $m_3$, $m_7$, $m_{11}$, and $m_{15}$ are don’t cares:

$f(x_4, x_3, x_2, x_1) = \Sigma\, m(1,5,8,9,10) + D(3,7,11,15)$

| $x_4x_3$ \ $x_2x_1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | d | 0 |
| 01 | 0 | 1 | d | 0 |
| 11 | 0 | 0 | d | 0 |
| 10 | 1 | 1 | d | 1 |

$x_4+x_1$

$x_4'+x_3'$

(b) POS implementation

To form the largest possible groups of 0s for the lowest-cost implementation, **assume that the don’t cares $D_3$, $D_7$, and $D_{11}$ are 1s while $D_{15}$ is 0.** With these assumptions, the simplest POS implementation is:

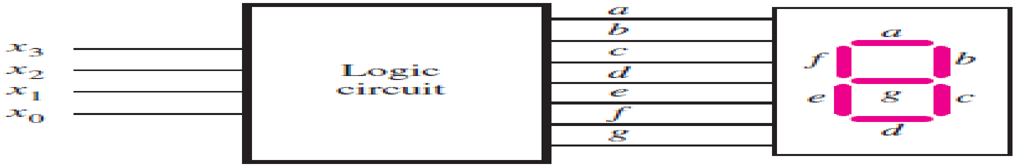
$$f = (x_4 + x_1) \,.\, (x_4' + x_3')$$





# Don’t Care Example-2

- Design a logic circuit that displays the decimal value of a four-bit number on a seven-segment display




(a) Logic circuit and 7-segment display

| $x_3$ | $x_2$ | $x_1$ | $x_0$ | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

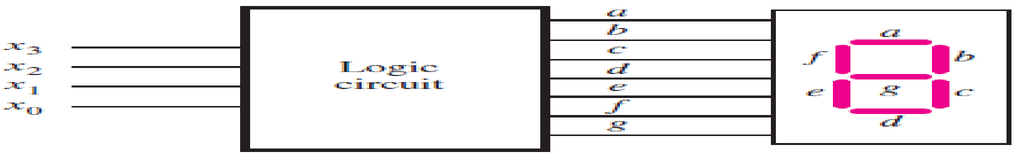
(b) Truth table





# Don’t Care Example-2

- Since the circuit can display only decimal digits (0 to 9), $x_3x_2x_1x_0$ = 0b1010 to 0b1111 are unused and can be treated as don’t care conditions in the design




(a) Logic circuit and 7-segment display

| $x_3$ | $x_2$ | $x_1$ | $x_0$ | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

(b) Truth table





# Don’t Care Example-2

- Since the circuit can display only decimal digits (0 to 9), $x_3x_2x_1x_0$ = 0b1010 to 0b1111 are unused and can be treated as don’t care conditions in the design

| $x_3$ | $x_2$ | $x_1$ | $x_0$ | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

| $x_3x_2$ \ $x_1x_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 1 | 1 |
| 01 | 0 | 1 | 1 | 1 |
| 11 | d | d | d | d |
| 10 | 1 | 1 | d | d |

$a = x_3 + x_1 + x_2.x_0 + x_2'.x_0'$

| $x_3x_2$ \ $x_1x_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 0 | 1 |
| 01 | 0 | 0 | 0 | 1 |
| 11 | d | d | d | d |
| 10 | 1 | 0 | d | d |

$e = x_1.x_0' + x_2'.x_0'$