

DIGITAL DESIGN AND COMPUTER ORGANIZATION

Logic Minimization, K-Maps - 2

Reetinder Sidhu

Department of Computer Science and Engineering



DIGITAL DESIGN AND COMPUTER ORGANIZATION

Logic Minimization, K-Maps - 2

Reetinder Sidhu

Department of Computer Science and Engineering



Course Outline



- Digital Design
 - Combinational logic design
 - ★ Logic Minimization, K-Maps 2
 - Sequential logic design
- Computer Organization
 - Architecture (microprocessor instruction set)
 - Microarchitecure (microprocessor operation)

Concepts covered

- K-Map Introduction
- K-Map Method
- K-Maps for Three Inputs

LOGIC MINIMIZATION, K-MAPS - 2 Why use K-Maps?



Why use K-Maps?



а	Ь	С	У	minterm	name
0	0	0	0	$\overline{a}\overline{b}\overline{c}$	m_0
0	0	1	0	$\overline{a}\overline{b}c$	m_1
0	1	0	0	āb c	m_2
0	1	1	1	ābc	m_3
1	0	0	1	$a\overline{b}\overline{c}$	m_4
1	0	1	0	$a\overline{b}c$	m_5
1	1	0	1	ab c	m_6
1	1	1	1	abc	m_7

SOP form:

$$v = \overline{abc} + \overline{abc} + ab\overline{c} + abc$$

• Minimized form:

$$y = bc + a\overline{c}$$

- Minimization requires:
 - Combining implicants on row 3 and 7
 - Combining implicants on row 4 and 6
 - ► Takes some effort to determine implicants to combine

Why use K-Maps?



а	Ь	С	У	minterm	name
0	0	0	0	$\overline{a}\overline{b}\overline{c}$	m_0
0	0	1	0	$\overline{a}\overline{b}c$	m_1
0	1	0	0	āb c	m_2
0	1	1	1	ābc	m_3
1	0	0	1	$a\overline{b}\overline{c}$	m_4
1	0	1	0	$a\overline{b}c$	m_5
1	1	0	1	ab c	m_6
1	1	1	1	abc	m_7

SOP form:

$$y = \overline{a}bc + a\overline{b}\overline{c} + ab\overline{c} + abc$$

• Minimized form:

$$y = bc + a\overline{c}$$

- Minimization requires:
 - Combining implicants on row 3 and 7
 - Combining implicants on row 4 and 6
 - Takes some effort to determine implicants to combine

- Minimizing: $v = bc + a\overline{c} + ab$
- Requires converting to: $bc + a\overline{c} + ab(c + \overline{c})$

- Minimization may require:
 - Unintuitve steps
 - Trial and error

Why use K-Maps?



а	Ь	С	y	minterm	name
0	0	0	0	$\overline{a}\overline{b}\overline{c}$	m_0
0	0	1	0	$\overline{a}\overline{b}c$	m_1
0	1	0	0	āb c	m_2
0	1	1	1	ābc	m_3
1	0	0	1	$a\overline{b}\overline{c}$	m_4
1	0	1	0	$a\overline{b}c$	m_5
1	1	0	1	ab c	m_6
1	1	1	1	abc	m_7

SOP form:

$$v = \overline{a}bc + a\overline{b}\overline{c} + ab\overline{c} + abc$$

Minimized form:

$$y = bc + a\overline{c}$$

- Minimization requires:
 - Combining implicants on row 3 and 7
 - Combining implicants on row 4 and 6
 - Takes some effort to determine implicants to combine

- Minimizing: $v = bc + a\overline{c} + ab$
- Requires converting to:
- $bc + a\overline{c} + ab(c + \overline{c})$

- Minimization may require:
 - Unintuitve steps
 - Trial and error

So minimization with Boolean identities is important to know since useful in some cases, but difficult to use in general

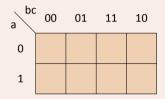
K-Map Structure



Kanaugh Map (K-Map)

- Maurice Karnaugh's refinement in 1953 of original idea from 1881
- Key idea: Minterms that differ in one literal must be adjacent
- Utilize brain's visual processing for efficient logic minimization

а	Ь	С	У	minterm
0	0	0	0	$\overline{a}\overline{b}\overline{c}$
0	0	1	0	$\overline{a}\overline{b}c$
0	1	0	0	āb c
0	1	1	1	ābc
1	0	0	1	$a\overline{b}\overline{c}$
1	0	1	0	$a\overline{b}c$
1	1	0	1	ab c
1	1	1	1	abc



- Each square corresponds to a row of the truth table
- Any two adjacent squares differ only in one literal
- Achieved using two rows and binary order 00,01,11,10
- Notion of "wrap-around": far left and right squares are adjacent

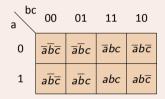
K-Map Structure



Kanaugh Map (K-Map)

- Maurice Karnaugh's refinement in 1953 of original idea from 1881
- Key idea: Minterms that differ in one literal must be adjacent
- Utilize brain's visual processing for efficient logic minimization

а	Ь	С	У	minterm
0	0	0	0	$\overline{a}\overline{b}\overline{c}$
0	0	1	0	$\overline{a}\overline{b}c$
0	1	0	0	āb c
0	1	1	1	ābc
1	0	0	1	$a\overline{b}\overline{c}$
1	0	1	0	$a\overline{b}c$
1	1	0	1	ab ¯ c
1	1	1	1	abc



- Each square corresponds to a row of the truth table
- Any two adjacent squares differ only in one literal
- Achieved using two rows and binary order 00,01,11,10
- Notion of "wrap-around": far left and right squares are adjacent

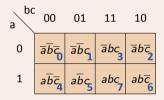
K-Map Structure



Kanaugh Map (K-Map)

- Maurice Karnaugh's refinement in 1953 of original idea from 1881
- Key idea: Minterms that differ in one literal must be adjacent
- Utilize brain's visual processing for efficient logic minimization

а	Ь	С	У	minterm
0	0	0	0	$\overline{a}\overline{b}\overline{c}$
0	0	1	0	$\overline{a}\overline{b}c$
0	1	0	0	āb c
0	1	1	1	ābc
1	0	0	1	$a\overline{b}\overline{c}$
1	0	1	0	$a\overline{b}c$
1	1	0	1	ab c
1	1	1	1	abc



- Each square corresponds to a row of the truth table
- Any two adjacent squares differ only in one literal
- Achieved using two rows and binary order 00,01,11,10
- Notion of "wrap-around": far left and right squares are adjacent

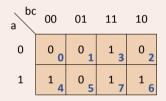
K-Map Structure



Kanaugh Map (K-Map)

- Maurice Karnaugh's refinement in 1953 of original idea from 1881
- Key idea: Minterms that differ in one literal must be adjacent
- Utilize brain's visual processing for efficient logic minimization

а	Ь	С	У	minterm
0	0	0	0	$\overline{a}\overline{b}\overline{c}$
0	0	1	0	$\overline{a}\overline{b}c$
0	1	0	0	āb c
0	1	1	1	ābc
1	0	0	1	$a\overline{b}\overline{c}$
1	0	1	0	$a\overline{b}c$
1	1	0	1	ab ¯ c
1	1	1	1	abc



- Each square corresponds to a row of the truth table
- Any two adjacent squares differ only in one literal
- Achieved using two rows and binary order 00,01,11,10
- Notion of "wrap-around": far left and right squares are adjacent

LOGIC MINIMIZATION, K-MAPS - 2 K-Map Method



K-Map Implicants

- Implicant
 - K-Map area composed of squares containing 1's
 - Area is square or rectangular (wraparound allowed)
 - No. of squares in area is a power of two (1, 2, 4, ...)
 - Each implicant corresponds to a product of literals
 - ★ Double the area, one less literal
- Prime implicant
 - Implicant having largest number of squares obeying above rules
- Essential prime implicant
 - Prime implicant containing a square not in any other prime implicant

K-Map Method

- Include all required prime implicants
 - Include all essential prime implicants
 - Include other prime implicants such that:
 - ★ Each square containing 1 is covered
 - ★ Boolean formula is minimal (may not be unique)
- Convert required implicants to Boolean formula
 - Each implicant is a product of literals
 - Include literals which do not change over its area