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

Why use K-Maps?

Logic Minimization, K-Maps - 2

Why use K-Maps?

a	b	c	y	minterm	name
0	0	0	0	$\bar{a}\bar{b}\bar{c}$	m_0
0	0	1	0	$\bar{a}\bar{b}c$	m_1
0	1	0	0	$\bar{a}b\bar{c}$	m_2
0	1	1	1	$\bar{a}bc$	m_3
1	0	0	1	$a\bar{b}\bar{c}$	m_4
1	0	1	0	$a\bar{b}c$	m_5
1	1	0	1	$ab\bar{c}$	m_6
1	1	1	1	abc	m_7

- SOP form:
  $y = \bar{a}bc + a\bar{b}\bar{c} + ab\bar{c} + abc$
- Minimized form:
  $y = bc + a\bar{c}$
- Minimization requires:
  - Combining implicants on row 3 and 7
  - Combining implicants on row 4 and 6
  - Takes some effort to determine implicants to combine

Logic Minimization, K-Maps - 2

Why use K-Maps?

a	b	c	y	minterm	name
0	0	0	0	$\bar{a}\bar{b}\bar{c}$	m_0
0	0	1	0	$\bar{a}\bar{b}c$	m_1
0	1	0	0	$\bar{a}b\bar{c}$	m_2
0	1	1	1	$\bar{a}bc$	m_3
1	0	0	1	$a\bar{b}\bar{c}$	m_4
1	0	1	0	$a\bar{b}c$	m_5
1	1	0	1	$ab\bar{c}$	m_6
1	1	1	1	abc	m_7

- SOP form:
  $y = \bar{a}bc + a\bar{b}\bar{c} + ab\bar{c} + abc$
- Minimized form:
  $y = bc + a\bar{c}$
- 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:
  $y = bc + a\bar{c} + ab$
- Requires converting to:
  $bc + a\bar{c} + ab(c + \bar{c})$

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

Logic Minimization, K-Maps - 2

Why use K-Maps?

a	b	c	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	$\overline{a}b\overline{c}$	m_2
0	1	1	1	$\overline{a}bc$	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	$ab\overline{c}$	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:
  $y = bc + a\overline{c} + ab$
- 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

Logic Minimization, K-Maps - 2

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

a	b	c	y	minterm
0	0	0	0	$\bar{a}\bar{b}\bar{c}$
0	0	1	0	$\bar{a}\bar{b}c$
0	1	0	0	$\bar{a}b\bar{c}$
0	1	1	1	$\bar{a}bc$
1	0	0	1	$a\bar{b}\bar{c}$
1	0	1	0	$a\bar{b}c$
1	1	0	1	$ab\bar{c}$
1	1	1	1	abc

K-Map Structure (three input Boolean functon)

a \ bc	00	01	11	10
0				
1				

- 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 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

a	b	c	y	minterm
0	0	0	0	$\bar{a}\bar{b}\bar{c}$
0	0	1	0	$\bar{a}\bar{b}c$
0	1	0	0	$\bar{a}b\bar{c}$
0	1	1	1	$\bar{a}bc$
1	0	0	1	$a\bar{b}\bar{c}$
1	0	1	0	$a\bar{b}c$
1	1	0	1	$ab\bar{c}$
1	1	1	1	abc

K-Map Structure (three input Boolean functon)

a \ bc	00	01	11	10
0	$\bar{a}\bar{b}\bar{c}$	$\bar{a}\bar{b}c$	$\bar{a}bc$	$\bar{a}b\bar{c}$
1	$a\bar{b}\bar{c}$	$a\bar{b}c$	abc	$ab\bar{c}$

- 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 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

a	b	c	y	minterm
0	0	0	0	$\bar{a}\bar{b}\bar{c}$
0	0	1	0	$\bar{a}\bar{b}c$
0	1	0	0	$\bar{a}b\bar{c}$
0	1	1	1	$\bar{a}bc$
1	0	0	1	$a\bar{b}\bar{c}$
1	0	1	0	$a\bar{b}c$
1	1	0	1	$ab\bar{c}$
1	1	1	1	abc

K-Map Structure (three input Boolean functon)

a \ bc	00	01	11	10
0	$\bar{a}\bar{b}\bar{c}_0$	$\bar{a}\bar{b}c_1$	$\bar{a}bc_3$	$\bar{a}b\bar{c}_2$
1	$a\bar{b}\bar{c}_4$	$a\bar{b}c_5$	abc_7	$ab\bar{c}_6$

- 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

a	b	c	y	minterm
0	0	0	0	$\bar{a}\bar{b}\bar{c}$
0	0	1	0	$\bar{a}\bar{b}c$
0	1	0	0	$\bar{a}b\bar{c}$
0	1	1	1	$\bar{a}bc$
1	0	0	1	$a\bar{b}\bar{c}$
1	0	1	0	$a\bar{b}c$
1	1	0	1	$ab\bar{c}$
1	1	1	1	abc

K-Map Structure (three input Boolean functon)

a \ bc	00	01	11	10
0	0_{0}	0_{1}	1_{3}	0_{2}
1	1_{4}	0_{5}	1_{7}	1_{6}

- 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 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 $(1, 2, 4, \ldots)$
  - 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