Lecture 3: Gate-level Minimisation I — The Map Method

CS207: Digital Logic

Jialin Liu

Department of Computer Science and Engineering (CSE)
Southern University of Science and Technology (SUSTech)

23 September 2022

Acknowledgement

These slides were prepared based on the slides by Dr. Jianqiao Yu and the ones by Prof. Georgios Theodoropoulos of the Department of CSE at the SUSTech, as well as the contents of the following book:

M. M. Mano and M. Ciletti, *Digital design: with an introduction to the Verilog HDL*. Pearson, 2013

Recap: Boolean Algebra and Logic Gate I

- Boolean algebra is used to find simpler and cheaper but **equivalent** realisations of a circuit with its postulates and theorems.
  - Example: $F = AB + BC + B'C = AB + C(B + B') = AB + C$

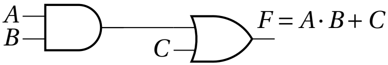
Postulate 2	(a)	$x + 0 = x$	(b)	$x \cdot 1 = x$
Postulate 5	(a)	$x + x' = 1$	(b)	$x \cdot x' = 0$
Theorem 1	(a)	$x + x = x$	(b)	$x \cdot x = x$
Theorem 2	(a)	$x + 1 = 1$	(b)	$x \cdot 0 = 0$
Theorem 3, involution		$(x')' = x$		
Postulate 3, commutative	(a)	$x + y = y + x$	(b)	$xy = yx$
Theorem 4, associative	(a)	$x + (y + z) = (x + y) + z$	(b)	$x(yz) = (xy)z$
Postulate 4, distributive	(a)	$x(y + z) = xy + xz$	(b)	$x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a)	$(x + y)' = x'y'$	(b)	$(xy)' = x' + y'$
Theorem 6, absorption	(a)	$x + xy = x$	(b)	$x(x + y) = x$

Figure: Screenshot of Table 2.1 in [1].

Recap: Boolean Algebra and Logic Gate II

- Boolean functions can be expressed by
  1. truth table ← unique for a boolean function;
  2. algebraic expression ← can be more than one possible expressions, but equivalent;
  3. logic diagram ← can be more than one possible diagrams, but equivalent

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



Recap: Boolean Algebra and Logic Gate III

- Canonical forms:
  - Notions:

    Product term Logical product of several variables.

    Minterm A product term is called a minterm when all variables are involved.

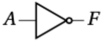
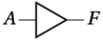
    Sum term Logical sum of several variables.

    Maxterm A sum term is called a maxterm when all variables are involved.

    Sum of products (SOP) Logical sum of two or more logical product terms.

    Product of sums (POS) Logical product of two or more logical sum terms.
  - Minterms are the complement of corresponding maxterms
  - Examples:
    - $F_1(A,B,C) = ABC + A'B$
    - $F_2(A,B,C) = A'BC' + AB'C' + AB'C + ABC' = \sum(2,4,5,6)$
    - $F_3(A,B,C) = (A+B+C)(A+B+C')(A+B'+C')(A'+B'+C') = \prod(0,1,3,7)$
    - $F_2(A,B,C) = F_3(A,B,C)$

- Logic operations/gates:
  - `NOT`, `OR`, `AND`, `NOT`, `NAND`, `NOR`, `XOR`, `NOT`, ...
  - `NAND` and `NOR` gates are called universal gates as any type of gates or logic functions can be implemented by these gates.

Name	Graphic symbol	Algebraic function
AND		$F = AB$
OR		$F = A + B$
NOT		$F = A'$
Buffer		$F = A$

Name	Graphic symbol	Algebraic function
NAND		$F = (AB)'$
NOR		$F = (A + B)'$
XOR		$F = AB' + A'B$ $= A \oplus B$

This Week: The Map Method for Gate-level Minimisation

- The complexity of digital logic gates to implement a Boolean function is directly related to the complexity of algebraic expression.
  - Exercise of last week: $F = ABC + AB'C + ABC' = A(C+B)$
    3 terms, 9 literals → 2 terms, 3 literals

- **Gate-level minimisation** is the design task of finding an optimal gate-level implementation of Boolean functions describing a digital circuit.
  - Difficult by hand for more than few inputs.
  - Typically by computer, need to understand the underlying principle.

- Methods for gate-level minimisation:
  - The map method (Karnaugh map). ← This lecture.
  - NAND and NOR implementation (logic diagrams). ← Next lecture.

Outline of This Lecture

Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary

Karnaugh map

- The map method, first proposed by Veitch and slightly improved by Karnaugh, provides a *simple, straightforward procedure* for the simplification of Boolean functions. The method is called **Karnaugh map** (卡诺图) or **K-map** .
  - The map is a diagram consisting of **square**s or **cell**s. For n variables on a Karnaugh map there are 2^n numbers of squares.
  - Each square represents one of the **minterm**s of the function that is to be minimised.
  - Since any Boolean function can be expressed as a sum of minterms, it is possible to recognise a Boolean function graphically in the map from the area enclosed by those squares whose minterms appear in the function.

$A \backslash B$	0	1
0	m_0	m_1
1	m_2	m_3

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

$AB \backslash CD$	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}

Outline of This Lecture

Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary

Two-variable K-map

- A two-variable system can form 4 minterms.

Decimal	A	B	Minterm
0	0	0	$A'B'$
1	0	1	$A'B$
2	1	0	AB'
3	1	1	AB

$A \backslash B$	0	1
0	m_0	m_1
1	m_2	m_3

$A \backslash B$	0	1
0	$A'B'$	$A'B$
1	AB'	AB

- The two-variable Karnaugh map is a useful way to represent any of the 16 Boolean functions.

- Example:

$$
\begin{aligned}
F(A,B) &= A+B = A(B+B') + B(A+A') \\
&= AB + AB' + AB + A'B = AB + AB' + A'B \\
&= \sum(1,2,3)
\end{aligned}
$$

$A \backslash B$	0	1
0	0	1
1	1	1

So the squares corresponding to AB, AB', and $A'B$ are marked with 1.

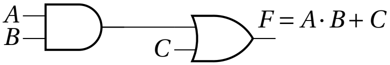
Two-variable K-map

Remarks

- The simplified expressions produced by the map are always in one of the two standard forms: **sum of products** or **product of sums**.

- The simplest algebraic expression is an algebraic expression with *a minimum number of terms and with the smallest possible number of literals in each term*.
  → A circuit diagram with a minimum number of gates and the minimum number of inputs to each gate.

- The simplest expression is not unique. In that case, either solution is satisfactory.



Outline of This Lecture

Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary

Three-variable K-map

- Since there are 8 minterms for 3 variables, the map consists of 8 cells or squares.
- Minterms are arranged, not according to the binary sequence, but according to the sequence similar to the gray code[1].
  - Between two adjacent rows or columns, only one single variable changes its logic value from `0` to `1` or from `1` to `0`.

Gray Code	Decimal
000	0
001	1
011	2
010	3
110	4
111	5
101	6
100	7

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

[1] Recall: Minimum change code. A number changes by only one bit as it proceeds from one number to the next.

Three-variable K-map

Properties of Adjacent Squares

- To understand the usefulness of the map for simplifying the Boolean functions, we must observe the basic properties of the adjacent squares.
  - Any two adjacent squares in the Karnaugh map differ by only one variable, which is complemented in one square and uncomplemented in one of the adjacent squares.
  - The sum of two minterms in adjacent squares can be simplified to a single `AND` term consisting of fewer literals.
    Example: $m_1 + m_5 = A'B'C + AB'C = (A' + A)B'C = B'C$

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

Three-variable K-map

Example

- Simplify the Boolean function $F(A,B,C) = A'BC + A'BC' + AB'C' + AB'C = \sum(3,2,4,5)$.

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

$A \backslash BC$	00	01	11	10
0	0	0	1	1
1	1	1	0	0

$A \backslash BC$	00	01	11	10
0	0	0	1	1
1	1	1	0	0

- The first row: $A'BC + A'BC' = A'B$.
- The second row: $AB'C' + AB'C = AB'$.
- $F = A'BC + A'BC' + AB'C' + AB'C = A'B + AB'$.

Three-variable K-map

Another Example

- Simplify the Boolean function $F = A'BC + AB'C' + ABC + ABC' = \sum(3,4,7,6)$.

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

$A \backslash BC$	00	01	11	10
0	0	0	1	0
1	1	0	1	1

$A \backslash BC$	00	01	11	10
0	0	0	1	0
1	1	0	1	1

- The third column: $A'BC + ABC = BC$.
- The second row: $AB'C' + ABC' = AC'$.
- $F = A'BC + AB'C' + ABC + ABC' = BC + AC'$.

→The squares of the leftmost and rightmost columns may be combined.

Three-variable K-map

Multiple Ways to Find The Simplest Expression

- Simplify the Boolean function $F = \sum(1,2,3,5,7)$.

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

$A \backslash BC$	00	01	11	10
0	0	1	1	1
1	0	1	1	0

$A \backslash BC$	00	01	11	10
0	0	1	1	1
1	0	1	1	0

- Way 1: $F = B'C + C + A'B = (B'+1)C + A'B = C + A'B$
- Way 2: $F = B'C + BC + A'B = (B'+B)C + A'B = C + A'B$

Outline of This Lecture

Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary

Four-variable K-map

- Similar to the method used for two-variable and three-variable Karnaugh maps, four-variable Karnaugh maps may be constructed with 16 squares consisting of 16 minterms.

$AB \backslash CD$	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}

Four-variable K-map

- Two, four, or eight adjacent squares can be combined to reduce the number of literals in a function.
  - When 2 adjacent squares are combined, it is called a **pair**: a term with 3 literals.
  - When 4 adjacent squares are combined, it is called a **quad**: a term with 2 literals.
  - When 8 adjacent squares are combined, it is called an **octet**: a term with 1 literal.
- In the case all 16 squares can be combined, the function will be reduced to `1`.
- The squares of the top and bottom rows as well as leftmost and rightmost columns may be combined.

Four-variable K-map

Example

- Simplify the Boolean function $F = \sum(1,5,10,11,12,13,15)$
  - $A'B'C'D + A'BC'D = A'C'D$
  - $ABC'D' + ABC'D = ABC'$
  - $ABCD + AB'CD = ACD$
  - $AB'CD + AB'CD' = AB'C$

$$F = \sum(1,5,10,11,12,13,15) = A'C'D + ABC' + ACD + AB'C.$$

AB \ CD	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}

AB \ CD	00	01	11	10
00		1		
01		1		
11	1	1	1	
10			1	1

AB \ CD	00	01	11	10
00		1		
01		1		
11	1	1	1	
10			1	1

- Attention! This reduced expression is not a unique one.

Simplified Expressions for the Previous Example

- Simplify the Boolean function $F = \sum(1,5,10,11,12,13,15)$
  - $F = A'C'D + ABC' + ACD + AB'C$
  - $F = A'C'D + ABC' + ABD + AB'C$

$AB \backslash CD$	00	01	11	10
00		1		
01		1		
11	1	1	1	
10			1	1

$AB \backslash CD$	00	01	11	10
00		1		
01		1		
11	1	1	1	
10			1	1

Plot Logical expressions on Four-variable Karnaugh Maps

Example

- Plot the logical expression $F(A,B,C,D) = ABCD + AB'C'D' + AB'C + AB$ on a four-variable Karnaugh map.

$$
\begin{aligned}
& F(A,B,C,D) \\
= & ABCD + AB'C'D' + AB'C + AB \\
= & ABCD + AB'C'D' + AB'C(D+D') + AB(C+C')(D+D') \\
= & \ldots \\
= & \sum(8,10,11,12,13,14,15) \\
= & AB + AC + AD'
\end{aligned}
$$

AB \ CD	00	01	11	10
00				
01				
11	1	1	1	1
10	1		1	1

Four-variable K-map

Another Example

- Simplify the expression $F(W,X,Y,Z) = W'X'Y' + X'YZ' + W'XYZ' + WX'Y'$.

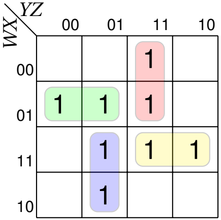
$$
\begin{aligned}
& F(W,X,Y,Z) \\
= {} & W'X'Y'(Z+Z') + X'YZ'(W+W') \\
& + W'XYZ' + WX'Y'(Z+Z') \\
= {} & W'X'Y'Z + W'X'Y'Z' + WX'YZ' \\
& + W'X'YZ' + W'XYZ' + WX'Y'Z \\
& + WX'Y'Z' \\
= {} & \sum(0,1,2,6,8,9,10) \\
= {} & X'Y' + X'Z' + W'YZ'
\end{aligned}
$$

WX \ YZ	00	01	11	10
00	1	1		1
01				1
11				
10	1	1		1

Four-variable K-map

Quad is not Always Good

- Simplify the expression $F(W,X,Y,Z) = \sum(3,4,5,7,9,13,14,15)$.
  - It may be noted that one quad can also be formed, but it is redundant as the squares contained by the quad are already covered by the pairs which are essential.
- $F = W'XY' + W'YZ + WY'Z + WXY$.



Four-variable K-map

Extended to Maxterms I

- Simplify the expression $F(W,X,Y,Z) = \prod(0,1,4,5,6,8,9,12,13,14)$.
  - The above expression is given in respect to the maxterms.
  - 0's are to placed instead of 1's at the corresponding maxterm squares.
- $F' = Y' + XZ' \rightarrow F = Y(X' + Z)$.

$WX \backslash YZ$	00	01	11	10
00	0	0	1	1
01	0	0	1	0
11	0	0	1	0
10	0	0	1	1

Four-variable K-map

Extended to Maxterms II

- Simplify the expression $F(W,X,Y,Z) = \prod(0,1,4,5,6,8,9,12,13,14)$.
  - The other way to achieve the minimised expression is to consider the 1's of the Karnaugh map.
- $F = YZ + X'Y = Y(X' + Z)$.

$WX \backslash YZ$	00	01	11	10
00	0	0	1	1
01	0	0	1	0
11	0	0	1	0
10	0	0	1	1

Five-variable K-map

- Karnaugh maps with more than four variables are not simple to use.
  - The number of cells or squares becomes excessively large and combining the adjacent squares becomes complex.
  - A five-variable Karnaugh map contains 2^5 or 32 cells :(

Prime Implicants

- A **prime implicant** is a product term obtained by combining the maximum possible number of adjacent squares in the map.
- The prime implicants of a function can be obtained from the map by combining all possible maximum numbers of squares.
  - If a minterm in a square is covered by only one prime implicant, that prime implicant is said to be **essential**.
- Gate-level minimisation:
  - Determine all essential prime implicants.
  - Find other prime implicants that cover remaining minterms.
  - Logical sum all prime implicants.

Outline of This Lecture

- Karnaugh map
- Two-variable K-map
- Three-variable K-map
- Four-variable K-map
- Don't Care Conditions
- Summary

Don't Care Conditions

- In practice, Boolean function is not specified for certain combinations of input variables.
  - Input combinations never occur during the process of a normal operation.
  - Those input conditions are guaranteed never to occur.
- Such input combinations are called **don't-care conditions**.
- These input combinations can be plotted on the Karnaugh map for further simplification.
  - The don't care conditions are represented by d or **X** in a K-map.
  - They can be either `1` or `0` upon needed.

Don't Care Conditions

Example

- Simplify the expression
  $F(A,B,C,D) = \sum(1,3,7,11,15),\ d = \sum(0,2,5)$.
- $F = A'B' + CD$.

$AB \backslash CD$	00	01	11	10
00	X	1	1	X
01		X	1	
11			1	
10			1	

Don't Care Conditions

Another Example

- Simplify the expression
  $F(A,B,C,D) = \sum(1,3,7,11,15),\ d = \sum(0,2,5)$.
- $F = A'D + CD$.

$AB \backslash CD$	00	01	11	10
00	X	1	1	X
01		X	1	
11			1	
10			1	

Outline of This Lecture

Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary

Summary

- Karnaugh map:
  - A diagram consisting of squares or cells, where each square or cell represents one of the minterms of the function that is to be minimised.
  - A simple, straightforward procedure for simplifying Boolean functions.
  - However, Karnaugh maps with more than four variables are not simple to use.
    $\leftarrow$ For n variables on a Karnaugh map there are 2^n numbers of squares !

Next week: `NAND` and `NOR` implementation of Boolean functions.

Essential Reading

- Essential reading for this lecture: pages 73-90 of the textbook.
- Essential reading for next lecture: pages 91-118 of the textbook.

[1] M. M. Mano and M. Ciletti, *Digital design: with an introduction to the Verilog HDL*. Pearson, 2013