

Lecture 3: Gate-level Minimisation – 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



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: Lecture 1 - Number Systems & Number-base Conversions




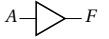
- ▶ In a **digital system**, input is given with the help of switches.
- ▶ **Number systems**: decimal, binary, octal, hexadecimal systems.
- ▶ **Number-based conversions**:
 - ▶ Base- r system to decimal: for $(a_n a_{n-1} \dots a_2 a_1 . b_1 b_2 \dots b_m)_r$,

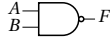


$$a_n \cdot r^{n-1} + a_{n-1} \cdot r^{n-2} + a_{n-2} \cdot r^{n-3} + \dots + a_2 \cdot r^1 + a_1 \cdot r^0 \\ + b_1 r^{-1} + b_2 \cdot r^{-2} + \dots + b_m \cdot r^{-m}$$

- ▶ Decimal to base- r system: **division** for integer part, **multiplication** for fraction
 - ▶ Conversion between binary and octal systems
- ▶ **Complements of numbers**: r 's complement & $r-1$'s complement
 - ▶ For subtraction
 - ▶ For representing signed binary numbers
- ▶ **Representation of data: code**

Recap: Lecture 2 - Boolean Algebra and Logic Gate I

- **Logic gates** (逻辑门) are electronic circuits that operates on one or more inputs signals to produce an output.
- Any desired circuit can be realised through **various combinations of logic gates**.
- 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	A	B	F
AND		$F = AB$	0	0	0
			0	1	0
			1	0	0
			1	1	1
OR		$F = A + B$	0	0	0
			0	1	1
			1	0	1
			1	1	1
NOT		$F = A'$	0	-	1
			1	-	0
Buffer		$F = A$	0	-	0
			1	-	1

Name	Graphic symbol	Algebraic function	A	B	F
NAND		$F = (AB)'$	0	0	1
			0	1	1
			1	0	1
			1	1	0
NOR		$F = (A + B)'$	0	0	1
			0	1	0
			1	0	0
			1	1	0
XOR		$F = AB' + A'B$ $= A \oplus B$	0	0	0
			0	1	1
			1	0	1
			1	1	0



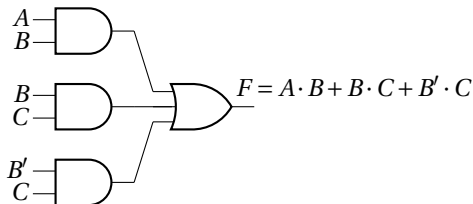
Recap: Lecture 2 - Boolean Algebra and Logic Gate II

- ▶ A **Boolean function** describes how to determine the binary output given binary variables as inputs and binary operators.
- ▶ Boolean functions can be expressed by
 1. **truth table** (真值表): unique for a boolean function;
 2. **algebraic expression** (逻辑表达式): can be more than one possible expressions, but equivalent;
 3. **logic diagram** (逻辑图): 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

$$F = A \cdot B + C$$

$$F = A \cdot B + B \cdot C + B' \cdot C$$





Recap: Lecture 2 - Boolean Algebra and Logic Gate III

- **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: Lecture 2 - Boolean Algebra and Logic Gate IV

► **Canonical forms** (规范表达式):

► Notions:

Product term (乘积项) Logical product of several variables.

Minterm (极小项) A product term is called a minterm when all variables are involved.

Sum term (和项) Logical sum of several variables.

Maxterm (极大项) 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:

► $F_1(A, B, C) = ABC + A'B$

► $F_2(A, B, C) = A'BC' + AB'C' + AB'C + ABC' = \sum(2, 4, 5, 6)$

► $F_3(A, B, C) = (A + B + C)(A + B + C')(A + B' + C')(A' + B' + C') = \prod(0, 1, 3, 7)$

► $F_2(A, B, C) = F_3(A, B, C)$



Quiz

1. Prove the following postulate using Truth Table.

Distributive Law: $A \cdot (B + C) = A \cdot B + A \cdot C$ and $A + B \cdot C = (A + B) \cdot (A + C)$

2. Simplify the following three-variable Boolean function algebraically: $F(A, B, C) = \sum(1, 2, 3, 5, 7)$.



Quiz

1. Prove the following postulate using Truth Table.

Distributive Law: $A \cdot (B + C) = A \cdot B + A \cdot C$ and $A + B \cdot C = (A + B) \cdot (A + C)$

Answer:

Decimal	A	B	C	$A \cdot (B + C)$	$A \cdot B + A \cdot C$	$A + B \cdot C$	$(A + B) \cdot (A + C)$
0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
2	0	1	0	0	0	0	0
3	0	1	1	0	0	1	1
4	1	0	0	0	0	1	1
5	1	0	1	1	1	1	1
6	1	1	0	1	1	1	1
7	1	1	1	1	1	1	1

According to the above truth table, for any input variables A , B and C , $A \cdot (B + C) = A \cdot B + A \cdot C$ holds, and $A + B \cdot C = (A + B) \cdot (A + C)$ holds.



Quiz

2. Simplify the following three-variable Boolean function algebraically: $F(A, B, C) = \sum(1, 2, 3, 5, 7)$.

Answer:

Decimal	A	B	C	F	Minterms
0	0	0	0	0	$A'B'C'$
1	0	0	1	1	$A'B'C$
2	0	1	0	1	$A'BC'$
3	0	1	1	1	$A'BC$
4	1	0	0	0	$AB'C'$
5	1	0	1	1	$AB'C$
6	1	1	0	0	ABC'
7	1	1	1	1	ABC

$$\begin{aligned}
 F(A, B, C) &= \sum(1, 2, 3, 5, 7) \\
 &= A'B'C + A'BC' + A'BC + AB'C + ABC \\
 &= A'B'C + A'BC' + A'BC + AB'C + ABC \\
 &= A'C(B' + B) + A'B(C' + C) + AC(B' + B) \\
 &= A'C + A'B + AC = (A' + A)C + A'B = C + A'B
 \end{aligned}$$



This Week and Next Week: 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 \rightarrow 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:
 - ▶ **Karnaugh** map: the map method \leftarrow This lecture (Week 3)
 - ▶ **NAND and NOR implementation** of logic diagrams \leftarrow Next lecture (Week 4)



Outline of This Lecture

Introduction to Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary

Karnaugh map

- ▶ **Karnaugh map** (卡诺图) or **K-map**: first proposed by *Veitch* and slightly improved by *Karnaugh*, provides a *simple, straightforward procedure* for the simplification of Boolean functions.
 - ▶ The **map** is a diagram consisting of **squares** or **cells**. For n variables on a Karnaugh map there are 2^n numbers of squares.
 - ▶ Each square represents one of the **minterms** of the function to be minimised.
- ▶ Any Boolean function can be expressed as a sum of minterms.
 - Any Boolean function can be recognised 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

Introduction to 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.



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**.
- ▶ It is sometimes possible to find two or more expressions that satisfy the minimisation criteria. In that case, either solution is satisfactory.



Outline of This Lecture

Introduction to 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

$\begin{array}{c} \diagup BC \\ A \end{array}$		00	01	11	10
		m_0	m_1	m_3	m_2
0					
1		m_4	m_5	m_7	m_6

¹Recap: 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: $\sum(1,5) = m_1 + m_5 = A'B'C + AB'C = (A' + A)B'C = B'C$

$A \backslash BC$		BC			
		00	01	11	10
A	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 $F(A, B, C) = A'BC + A'BC' + AB'C' + AB'C = \sum(3, 2, 4, 5)$.



Three-variable K-map

Example

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

$A \backslash BC$	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



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

Example of Combining Squares of The Leftmost and Rightmost Columns

- 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



Three-variable K-map

Example of Combining Squares of The Leftmost and Rightmost Columns

- 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: $AB'C' + ABC' = AC'$.
- $F = A'BC + AB'C' + ABC + ABC' = BC + AC'$.



Three-variable K-map

Example of Combining Squares of The Leftmost and Rightmost Columns

- ▶ 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** can be combined.



Three-variable K-map

Example of Combining 4 Adjacent Squares

- 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	



Three-variable K-map

Example of Combining 4 Adjacent Squares

- 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

- $F = C + A'B$



Exercise

- Simplify the Boolean function $F = \sum(0, 2, 4, 5, 6)$.

Exercise

- Simplify the Boolean function $F = \sum(0, 2, 4, 5, 6)$.

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

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

- Solution: $F = C' + AB'$.



Outline of This Lecture

Introduction to 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 can be constructed with 16 squares (i.e., 16 minterms).

$AB \backslash CD$					
		00	01	11	10
AB	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 are combined, the function will be reduced to 1.
- ▶ The squares of the **top and bottom rows** as well as **leftmost and rightmost columns** can be combined.



Four-variable K-map

Multiple Ways to Find The Simplest Expression

- ▶ Simplify the Boolean function $F = \sum(1, 5, 10, 11, 12, 13, 15)$



Four-variable K-map

Multiple Ways to Find The Simplest Expression

- Simplify the Boolean function $F = \sum(1, 5, 10, 11, 12, 13, 15)$

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



Four-variable K-map

Multiple Ways to Find The Simplest Expression

► 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



Four-variable K-map

Multiple Ways to Find The Simplest Expression

- ▶ 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.
- ▶ If pairs are formed in different ways, the simplified expression may be different.



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 \ 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



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.



Exercise

- ▶ Simplify the Boolean function $F = \sum(7, 9, 10, 11, 12, 13, 14, 15)$.

Exercise

- Simplify the Boolean function $F = \sum(7, 9, 10, 11, 12, 13, 14, 15)$.

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				
01			1	
11	1	1	1	1
10		1	1	1

- Solution: $F = AB + AC + AD + BCD$.



Exercise

- ▶ Simplify the expression $F(W, X, Y, Z) = \sum(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$.

Exercise

- Simplify the expression $F(W, X, Y, Z) = \sum(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$.

WX \ YZ	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}

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



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.

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

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

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

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.
- ▶ $F = W'XY' + W'YZ + WY'Z + WXY$.

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



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 be placed instead of 1's at the corresponding maxterm squares.
- ▶ $F' = Y' + XZ' \rightarrow F = Y(X' + Z)$.

WX \ YZ	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 $F(W, X, Y, Z) = \prod(0, 1, 4, 5, 6, 8, 9, 12, 13, 14)$.
 - ▶ Recap: Minterms are the complement of corresponding maxterms.
 - ▶ 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 \ 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.
 - ▶ A five-variable Karnaugh map contains 2^5 or 32 cells.
 - ▶ The number of cells or squares becomes excessively large and combining the adjacent squares becomes complex.



Outline of This Lecture

Introduction to 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

$$F(A, B, C, D) = \sum(1, 3, 7, 11, 15), \quad d = \sum(0, 2, 5).$$

- ▶ $F = A'B' + CD$.

AB \ CD	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

$$F(A, B, C, D) = \sum(1, 3, 7, 11, 15), \quad d = \sum(0, 2, 5).$$

- $F = A'D + CD.$

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



Outline of This Lecture

Introduction to Karnaugh map

Two-variable K-map

Three-variable K-map

Four-variable K-map

Don't Care Conditions

Summary



- ▶ Karnaugh map:
 - ▶ A straightforward way to simplify Boolean expressions.
 - ▶ 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.
 - ← For n variables on a Karnaugh map there are 2^n numbers of squares !

Next week: NAND and NOR implementation of Boolean functions.

Today's Lab



- ▶ Bitwise operation in Verilog
- ▶ Gates in RTL vs. LUT in FPGA
- ▶ Design a circuit to perform addition of two two-bit unsigned numbers



- ▶ 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