SIMPLIFICATION AND MINIMIZATION OF BOOLEAN FUNCTIONS

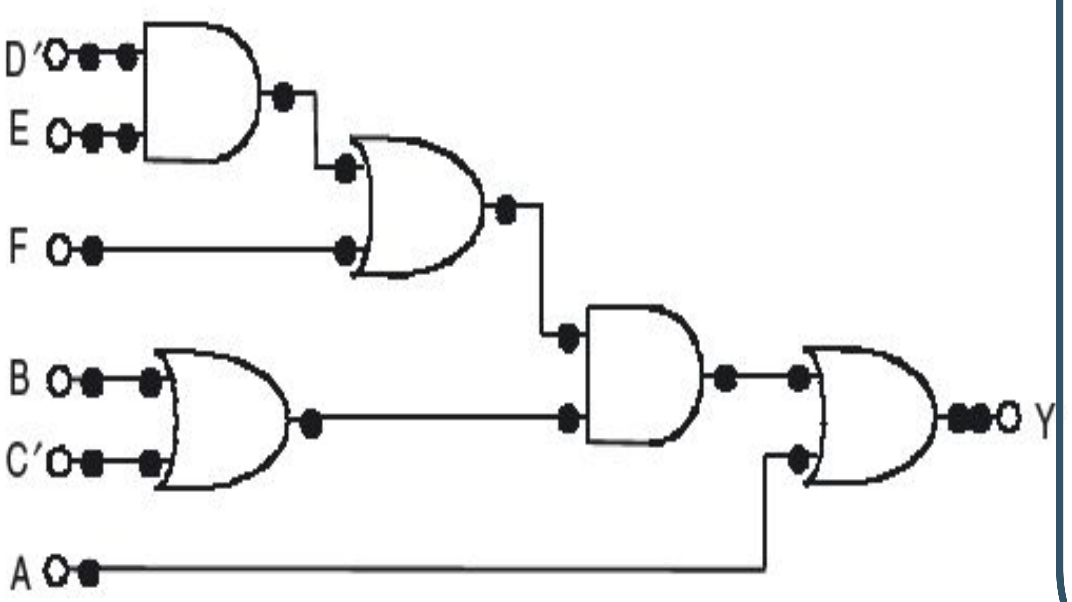
Prepared By:
Nuren Zabin Shuchi
Lecturer
Department of Electrical and Electronic Engineering
Shahjalal University of Science and Technology

Outline

- Necessity of Simplification and Minimization of Boolean Functions
- Definition of a K-Map
- Drawing a K-Map
- Minimizing Function using K-Map
- Don’t Care Conditions
- Numerical Problems

Necessity of Simplification and Minimization of Boolean Functions

- Reducing number of gates needed to implement a circuit
- Increasing efficiency and speed of circuit



The figure in left illustrates an arbitrary function realized with a multilevel gate implementation. If this function is further minimized:

1. The number of gate to implement the circuit is reduced; hence making the circuit economical.

2. The amount of time required for the signal to travel from one stage to the other is called the propagation delay. Propagation delay is decreased if the number of stages is reduced through further simplification of the function. The efficiency and operating speed is increased with the decreased propagation delay.

Definition of a K-Map

- A Karnaugh map (K-map) is a pictorial method used to minimize Boolean expressions without having to use Boolean algebra theorems and equation manipulations. A K-map can be thought of as a special version of a truth table.
- Using a K-map, expressions with two to four variables are easily minimized. Expressions with five to six variables are more difficult but achievable, and expressions with seven or more variables are extremely difficult to minimize using a K-map.
- The algebraic minimization procedure lacks specific rules to predict the succeeding step in the manipulative process. The Karnaugh map provides a systematic method for simplification and manipulation of a Boolean expression.

Drawing a K-Map

- A Karnaugh map (K-map) is a diagram made of squares or cells.
- If a function has n-variables, the corresponding K-Map will have 2^n squares or cells.
- Each square represents a minterm.
- K-map cells are arranged such that adjacent cells correspond to truth rows that differ in only one bit position.

Drawing a 2 variable K-Map

A	B	Minterms
0	0	m_0 =A’B’
0	1	m_1 = A’B
1	0	m_2 = AB’
1	1	m_3 = AB

Fig-1: Truth Table

	B′	B
A′	A′B′	A′B
A	AB′	AB

Fig-2: K-Map (1)

A \ B	0	1
0	A’B’	A’B
1	AB’	AB

Fig-3: K-Map (2)

A \ B	0	1
0	m_0	m_1
1	m_2	m_3

Fig-4: K-Map (3)

- Since a two-variable system can form four minterms, the map consists of four cells—one for each minterm.
- The adjacent cells differ in only one bit position.
- Figures 2,3 and 4 represent the same K-Map in different forms.
- If the positions of the variables are changed the corresponding cells dedicated for the minterms will also change.

B \ A	0	1
0	A’B’	AB’
1	A’B	AB

A \ B	0	1
0	m_0	m_2
1	m_1	m_3

Drawing a 3 variable K-Map

A	B	C	Minterms
0	0	0	m_0 =A'B'C'
0	0	1	m_1 = A'B'C
0	1	0	m_2 = A'BC'
0	1	1	m_3 = A'BC
1	0	0	m_4 = AB'C'
1	0	1	m_5 = AB'C
1	1	0	m_6 = ABC'
1	1	1	m_7 = ABC

Fig-1: Truth Table

1. Cells 2^3 =8 ,one for each minterm.
2. The adjacent cells differ in only one bit position.
3. If the positions of the variables are changed the corresponding cells dedicated for the minterms will also change (shown in fig-3).

	B'C'	B'C	BC	BC'
A'	A'B'C'	A'B'C	A'BC	A'BC'
A	AB'C'	AB'C	ABC	ABC'

C' C C'

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

B' B

Fig-2: K-Map

	A'B'	A'B	AB	AB'
C'	m_0	m_2	m_6	m_4
C	m_1	m_3	m_7	m_5

Fig-3 (a)

	A'	A
B'C'	m_0	m_4
B'C	m_1	m_5
BC	m_3	m_7
BC'	m_2	m_6

Fig-3 (b)

Drawing a 4 variable K-Map

A	B	C	D	Minterms
0	0	0	0	A’B’C’D’=m_0
0	0	0	1	A’B’C’D=m_1
0	0	1	0	A’B’CD’=m_2
0	0	1	1	A’B’CD=m_3
0	1	0	0	A’BC’D’=m_4
0	1	0	1	A’BC’D=m_5
0	1	1	0	A’BCD’=m_6
0	1	1	1	A’BCD=m_7
1	0	0	0	AB’C’D=m_8
1	0	0	1	AB’C’D=m_9
1	0	1	0	AB’CD’=m_{10}
1	0	1	1	AB’CD=m_{11}
1	1	0	0	ABC’D’=m_{12}
1	1	0	1	ABC’D=m_{13}
1	1	1	0	ABCD’=m_{14}
1	1	1	1	ABCD=m_{15}

Fig-1: Truth Table

C’ C

	C′D′	C′D	CD	CD′
A′B′	A′B′C′D′	A′B′C′D	A′B′CD	A′B′CD′
A′B	A′BC′D′	A′BC′D	A′BCD	A′BCD″
AB	ABC′D′	ABC′D	ABCD	ABCD′
AB′	AB′C′D′	AB′C′D	AB′CD	AB′CD′

A’ A B’ B B’

D’ D D’

Fig-2: K-Map

	00 C′D′	01 C′D	11 CD	10 CD′
00 A′B′	m_0	m_1	m_3	m_2
01 A′B	m_4	m_5	m_7	m_6
11 AB	m_{12}	m_{13}	m_{15}	m_{14}
10 AB′	m_8	m_9	m_{11}	m_{10}

1. Cells 2^4 =16 ,one for each minterm.
2. The adjacent cells differ in only one bit position.
3. If the positions of the variables are changed the corresponding cells dedicated for the minterms will also change (**Try yourself**).

Minimizing Function using K-Map

Step 1: Select K-map according to the number of variables of the given function.

$$F = A'BC + A'BC' + AB'C' + AB'C.$$

Step 2: Identify the minterms of the given problem and put 1’s in the respective cells and 0’s elsewhere.

	B’C’	B’C	BC	BC’
A’	**0**	**0**	**1**	**1**
A	**1**	**1**	**0**	**0**

Step 3: Form **vertical or horizonal** groups of the adjacent cells containing 1’s. The groups should contain as many cells containing 1 as possible. This will result in the fewest number of literals (variables) in the term that represents the group. A group can contain total number of cells only in power of two like 1, 2,4,8…..

	B’C’	B’C	BC	BC’
A’	**0**	**0**	**1**	**1**
A	**1**	**1**	**0**	**0**

Minimizing Function using K-Map (Cont.)

Step 3: From the groups made in find the product terms and sum them up for SOP form.

	B'C'	B'C	BC	BC'
A'	**0**	**0**	**1**	**1**
A	**1**	**1**	**0**	**0**

Notice, the red group has two minterms, AB'C' and AB'C. the reduced term of AB'C' + AB'C is AB', as AB'(C' + C)= AB'.

Therefore,
From **red** group we get product term—AB' (Since, the C variable changes form.)

Similarly,
From **Black** group we get product term—A'B

The simplified Boolean expression,

F=AB'+A'B

Guidelines for Grouping Adjacent Cells

Before proceeding to the minimization of 4-variable functions using a K-Map, Let us clarify some rules regarding the grouping of adjacent cells.

Guidelines for Grouping Adjacent Cells

- Always combine as many cells in a group as possible. This will result in the fewest number of literals in the term that represents the group.
- Make as few groupings as possible to cover all minterms. This will result in the fewest product terms.
- Always begin with the largest group, which means if you can find eight members group is better than two four groups and one four group is better than pair of two-group.
- Groups can **not include** any cell containing a **zero** .

Guidelines for Grouping Adjacent Cells (Cont.)

- Groups may be horizontal or vertical, but not diagonal.

0	**1**
1	0

Wrong

0	**1**
1	1

Righ
t

- Groups must contain 1 (2^0), 2, 4, 8, or in general 2^n cells.
- Each group should be as large as possible.

0	**0**	**1**	**1**
1	**1**	**1**	**1**

Wrong

0	**0**	**1**	**1**
1	**1**	**1**	**1**

Righ
t

Guidelines for Grouping Adjacent Cells (Cont.)

- Each cell containing a one must be in at least one group..

0	1
1	0

Right

1	0	0	0
		1	1

- Groups may overlap each other.

0	0	1	1
1	1	1	1

Right

Guidelines for Grouping Adjacent Cells (Cont.)

- Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.

	B'C'	B'C	BC	BC'
A'	**1**	**0**	**1**	**1**
A	**1**	**0**	**1**	**1**

1	0	0	1
0	1	1	1
0	0	0	1
1	0	0	1

- There should be as few groups as possible, as long as this does not contradict any of the previous rules.

0	**0**	**1**	**1**
1	**1**	**1**	**1**

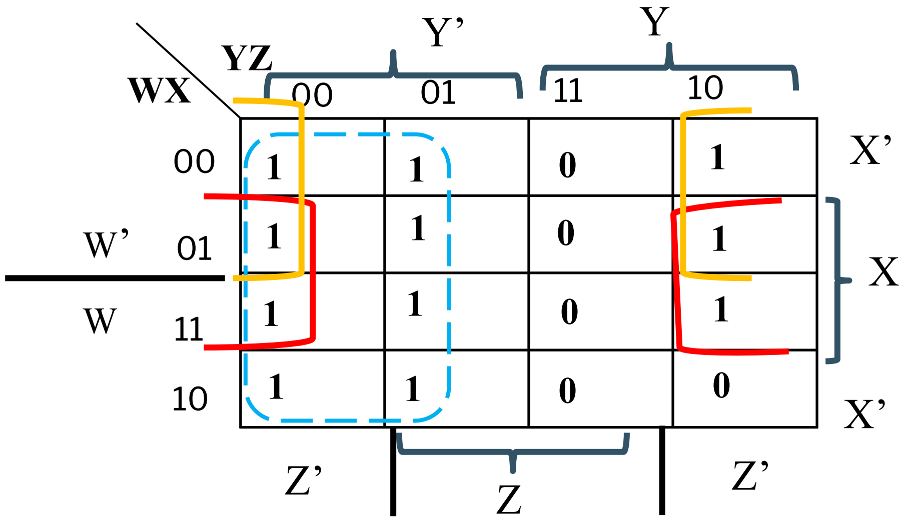
Wrong

0	**0**	**1**	**1**
1	**1**	**1**	**1**

Righ
t

Minimizing 4 Variable Functions using K-Map

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



F= Y’+ XZ’+W’Z’

	01 $C'D'$	11 $C'D$	10 CD	00 CD'
00 $A'B'$	m_0	m_1	m_3	m_2
01 $A'B$	m_4	m_5	m_7	m_6
11 AB	m_{12}	m_{13}	m_{15}	m_{14}
10 AB'	m_8	m_9	m_{11}	m_{10}

- When two adjacent squares are combined, it is called a pair and represents a term with three literals (variables).
- Four adjacent squares, when combined, are called a quad and its number of literals (variables) is two.
- If eight adjacent squares are combined, it is called an octet and represents a term with one literal (variable).
- If, in the case all sixteen squares can be combined, the function will be reduced to 1.

Don’t Care Conditions

- In certain digital systems, some input combinations never occur during the process of a normal operation because those input conditions are guaranteed never to occur. Such input combinations are called ***Don’t-Care Combinations or Conditions***.
- The function output can be either 1 or 0 for the Don’t-Care Combinations or Conditions.
- These input combinations can be plotted on the Karnaugh map for further simplification of the function.
- The don’t care combinations are represented by ***d*** **or** ***x*** **or Φ**.

Minimization of Function with Don't Care Conditions

$$F\,(A,\ B,\ C,\ D) = \Sigma\,(1,3,7,11,15) + \Phi(0,2,5).$$

- When a function with don't-care combinations is simplified to obtain minimal SOP expression, the value 1 can be assigned to the selected don't care combinations. This is done to form groups like pairs, quadoctet, etc., for further simplification. In each case, choice depends only on need to achieve simplification

	C′D′	C′D	CD	CD′
A′B′	X	1	1	X
A′B		X	1	
AB			1	
AB′			1	

$$F = A'B' + CD.$$

Numerical Problems

- Examples: 4.1- 4.15
- Exercises: 4.4, 4.7- 4.11

Reference: Chapter 4
Digital Principles and Logic Design
By: A. Saha & N. Manna

Any Questions?