Digital Logic

Chapter 2:
Introduction to logic circuits

Ru Han

2.6 Synthesis Using AND, OR, and NOT Gates

- Given a set of specifications, design a logic network to meet them. In this part of Chapter 2, we will explore systematic techniques for doing this.

- A simple example:
- Design a circuit: The function of the circuit is to continuously monitor the state of the switches.

If the state of the switches (x1,x2) are in states (0,0),(0,1),(1,1), the output are 1.

If the state of the switches is (1,0), the output should be 0.

Logic Design 1

- The truth table:

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

Figure 2.19. A function to be synthesized.

- Our aim: to derive a logic expression which are equal to the truth table in function.
- How ?
- Why ?

Logic Design 1

- The truth table:

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

Figure 2.19. A function to be synthesized.

- Two ways:

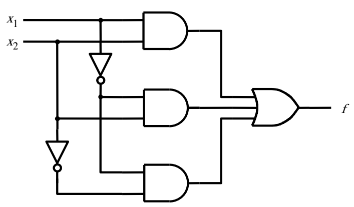
(1) From AND logic.

(2) From OR logic.

Logic Design 1

Design 1: ***From the view of AND logic***

$f\,(x1,x2)= x_1x_2+\bar{x}_1\bar{x}_2+\bar{x}_1x_2$



(a) Canonical sum-of-products

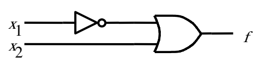
Figure 2.20. Two implementations of the function in Figure 2.19.

Logic Design 1

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

Design 2: ***From the view of OR logic***

$f\,(x1,x2)= x_2+\bar{x}_1$



(b) Minimal-cost realization

Figure 2.20. Two implementations of the function in Figure 2.19.

As the truth table is same ,so

$f\,(x1,x2)= x_1x_2+\bar{x}_1\bar{x}_2+\bar{x}_1x_2 = x_2+\bar{x}_1$

- Can you prove the identity?
- Can you get some clues to design the logic circuits based on the truth table?

A straightforward implementation of a truth table from AND logic

- 1: choose the rows in which the output is 1
- 2: generate the product term for each row.
  - What is the Product term?
  - Product term: Combine all input Variables by AND , each input variable just appears one time in either original form or complement form.
  - In the input valuation, If current input value is 1, then original form x_i is entered into Product term, else $\bar{x}_i$ is used.
- 3: OR all the Product terms to realize the desired function

7

Furthermore:

- There are many different networks to realize a given function.
- At least there is two ways, one based on the AND strategy and another based on the OR strategy.
- Boolean Algebraic manipulation can prove that they are equal in function.
- Boolean Algebraic manipulation can help us to find the simplified and lower-cost networks.

Synthesis:
The process whereby we begin with a description of the desired functional behavior and then generate a circuit that realizes this behavior is called *synthesis*.

8

Logic Design 2

Truth Table

x	y	z	*name*	f
0	0	0	M_0	
0	0	1	M_1	
0	1	0	M_2	
0	1	1	M_3	
1	0	0	M_4	
1	0	1	M_5	
1	1	0	M_6	
1	1	1	M_7	

Note: $f = 1$ if $x = 0, y = z = 1$
or $x = y = z = 1$

9

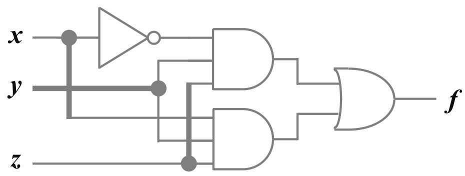
Logic Design 2

Design 1:

Let $M_3 = \bar{x} \bullet y \bullet z = 1$ only when $x = 0, y = z = 1$

Let $M_7 = x \bullet y \bullet z = 1$ only when $x = y = z = 1$

Then $f = M_3 + M_7$



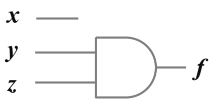
10

Logic Design 2

Design 2:

Further note: $f = 1$ if $y = z = 1$
independent of x value

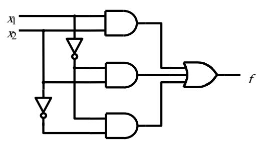
$\rightarrow$ $f = y \bullet z$



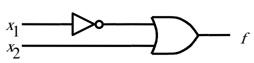
11

Logic Design

Design 1:

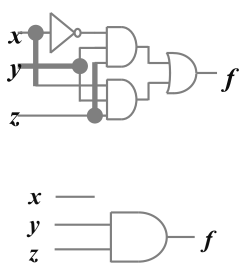


(a) Canonical sum-of-products



(b) Minimal-cost realization

Design 2:



12

More Example in Text Book

- See page 46,
- Example 2.7-2.9
- Hints:
  - Specification $\Rightarrow$ Truth Table
  - Synthesize by means of Sum of Product
  - Simplify the expression in a number of ways

Canonical Forms – Terminology

- **Literal** – Any Boolean variable or its complement.
  Example: A, x, $\bar{B}$, $\bar{x}y$, $A+B$
- **Sum Term** – Two or more literals joined by **OR** operators.
  Example: $A + B + C$, $A + \bar{B}C$
- **Product Term** – Two or more literals joined by **AND** operators.
  Example: ABC, $\bar{x}y$,
- **Sum-of-Products** – two or more product terms or literals joined by **OR** operators.
  Example: $A + \bar{B}C + \bar{B}D$, $A + \bar{B}(C + D)$
- **Product-of-Sums** – two or more sum terms or literals joined by **AND** operators.
  Example: $A(\bar{B} + C)(\bar{B} + D)$, $A(\bar{B} + CD)$

Canonical Forms – Terminology (Con'd)

- **Minterm** – a product term that contains each variable exactly once, in either complemented or uncomplemented form.
  Example: *f* is a Boolean function of the variables *A, B, C, D.*
  $AB\bar{C}D$, $AB\bar{C}\bar{D}$
- **Maxterm** – a sum term that contains each variable exactly once, in either complemented or uncomplemented form.
  Example: $A + B + \bar{C} + D$, $A + B + \bar{C} + \bar{D}$
- **Canonical Sum-of-Products** – A sum-of-products expression in which every product term is a minterm.
  Example: $f(A,B,C,D) = AB\bar{C}D + ABC\bar{D} + \bar{A}\bar{B}CD$
  $f(A,B,C,D) = AB\bar{C} + \bar{A}BD$
- **Canonical Product-of-sums** – A product-of-sums expression in which every sum term is a maxterm.
  Example: $f(A,B,C,D) = (A+B+C+\bar{D})\,(\bar{A}+B+C+D)\,(\bar{A}+\bar{B}+C+\bar{D})$
  $f(A,B,C,D) = (A+B+C)\,(\bar{A}+\bar{B}+D)$

Relation in Canonical Forms

Truth Tables:

- Suppose we have a truth table for a Boolean function of n variables. Let the rows be numbered 0 through 2^n-1.
- Each row corresponds to one minterm. Let m_k be the minterm for row k. *It is the only minterm that is true for row k.*

Row	A	B
2	1	0

$\Rightarrow$ $m_2 = A\bar{B}$

- Each row also corresponds to one maxterm. Let M_k be the maxterm for row k. *It is the only maxterm that is false for row k.*

Row	A	B
2	1	0

$\Rightarrow$ $M_2 = \bar{A} + B$

Relation in Canonical Forms

- **Example *f(x, y, z)***

k	*x*	*y*	*z*
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

More Properties of Minterm and Maxterm

- For n variable, there are (M = 2^n) maxterms or minterms and there are 2^M possible logic expressions.
- For given value of variables, the product of more than two different Minterms is 0
- For given value of variables, the sum of more than two different Maxterms is 1
- Sum of all minterms for n variables is 1
- Product of all maxterms for n variables is 0
- Not $m_i = M_i$

Canonical SOP Form

1) Write down the truth table for the logic function f.
2) For each row where $f = 1$, find the associated **minterm**.
3) The function f is the **sum of these minterms**.

Example: Prime number (smaller than 10) detector.

Definition: A prime number is an integer greater than 1 whose only divisors are 1 and itself.
Goal: Design a circuit to realize $f(x,y,z)$ where $f = 1$ if and only if xyz is a prime number.

Canonical SOP Form – Example

(1) Truth Table

k	x	y	z	f	m_k
0	0	0	0	*0*	$\overline{x}\,\overline{y}\,\overline{z}$
1	0	0	1	0	$\overline{x}\,\overline{y}\,z$
2	0	1	0	1	$\overline{x}\,y\,\overline{z}$
3	0	1	1	1	$\overline{x}\,y\,z$
4	1	0	0	0	$x\,\overline{y}\,\overline{z}$
5	1	0	1	1	$x\,\overline{y}\,z$
6	1	1	0	0	$x\,y\,\overline{z}$
7	1	1	1	1	$x\,y\,z$

(2) Minterms

$m_2 = \overline{x}\,y\,\overline{z};\quad m_3 = \overline{x}\,y\,z;\quad m_5 = x\,\overline{y}\,z;\quad m_7 = x\,y\,z$

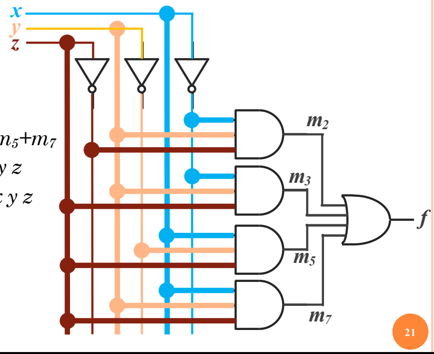
(3) Canonical SOP Expression

$f = m_2 + m_3 + m_5 + m_7 = \sum m_{(2,3,5,7)}$
$= \overline{x}\,y\,\overline{z} + \overline{x}\,y\,z + x\,\overline{y}\,z + x\,y\,z$

Canonical SOP Form – Example

(4) Logic Network

$f = m_2+m_3+m_5+m_7$
$= \overline{x}\,y\,\overline{z} + \overline{x}\,y\,z$
$+ x\,\overline{y}\,z + x\,y\,z$



Canonical POS Form

(a) Truth Table.
(b) For each row where $f=0$, find the associated **maxterms**.
(c) The function f is **the product of these maxterms**.

Example: Prime Number (smaller than 10) Detector

Canonical POS Form – Example

(1) Truth Table

k	x	y	z	f	m_k	M_k
0	0	0	0	*0*	$\overline{x}\,\overline{y}\,\overline{z}$	$x+y+z$
1	0	0	1	0	$\overline{x}\,\overline{y}\,z$	$x+y+\overline{z}$
2	0	1	0	1	$\overline{x}\,y\,\overline{z}$	$x+\overline{y}+z$
3	0	1	1	1	$\overline{x}\,y\,z$	$x+\overline{y}+\overline{z}$
4	1	0	0	0	$x\,\overline{y}\,\overline{z}$	$\overline{x}+y+z$
5	1	0	1	1	$x\,\overline{y}\,z$	$\overline{x}+y+\overline{z}$
6	1	1	0	0	$x\,y\,\overline{z}$	$\overline{x}+\overline{y}+z$
7	1	1	1	1	$x\,y\,z$	$\overline{x}+\overline{y}+\overline{z}$

(2) Maxterms

$M_0 = x + y + z;\ M_1 = x + y + \overline{z};\ M_4 = \overline{x} + y + z;\ M_6 = \overline{x} + \overline{y} + z$

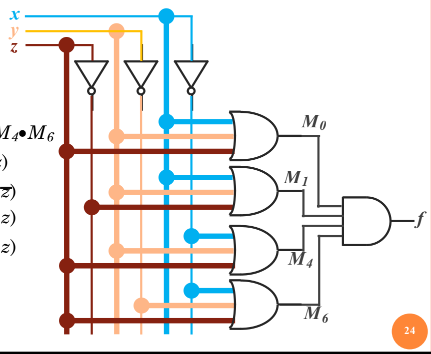
(3) Canonical POS Expression

$f = M_0 \bullet M_1 \bullet M_4 \bullet M_6 = \Pi\, M_{(0,1,4,6)}$
$= (x + y + z) \bullet (x + y + \overline{z}) \bullet (\overline{x} + y + z) \bullet (\overline{x} + \overline{y} + z)$

Canonical POS Form – Example

(4) Logic Network

$f = M_0 \bullet M_1 \bullet M_4 \bullet M_6$
$= (x + y + z)$
$\bullet\ (x + y + \overline{z})$
$\bullet\ (\overline{x} + y + z)$
$\bullet\ (\overline{x} + \overline{y} + z)$



Evaluating a Design

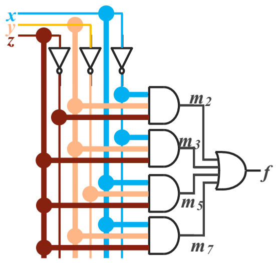
- Consider the **Prime Number Detector**…
  - Is this a good design?
  - Can we find a better one?
- Often the simplest design is best. For our purpose, we will try to minimize the following cost function:

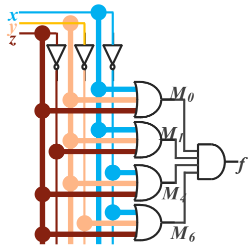
  Cost = Total number of Gates
  + total number of Gate Inputs

25

Evaluating a Design

Canonical SOP Network **Canonical POS Network**





26

Evaluating a Design

Canonical SOP Network	Canonical POS Network
3 NOT Gates	3 NOT Gates
4 3-input AND Gates	4 3-input OR Gates
1 4-input OR Gates	1 4-input AND Gates
8 Gates, 19 Inputs	8 Gates, 19 Inputs
Cost = 8 + 19 = 27	Cost = 8 + 19 = 27

Can we do better?

27

Minimal SOP Realization

- Apply **postulates + theorems** of Boolean algebra to simplify expression as much as possible.
- Must remain in SOP form.

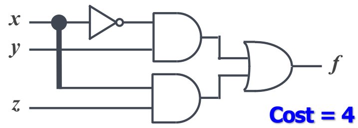
Example:

$f = \overline{x}\,y\,\overline{z} + \overline{x}\,y\,z + x\,\overline{y}\,z + x\,y\,z$

$= \overline{x}\,y\,(\overline{z} + z) + x\,z\,(\overline{y} + y)$ Distributive

$= \overline{x}\,y \bullet 1 + x\,z \bullet 1$ Postulate 5a

$= \overline{x}\,y + x\,z$ Identity for •



Cost ?

Cost = 4 + 7 = 11

28

Minimal POS Realization

- Simplify expression.
- Must remain in POS form.

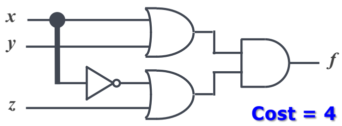
Example:

$f = (x+y+z)\,(x+y+\overline{z})\,(\overline{x}+y+z)\,(\overline{x}+\overline{y}+z)$

$= (x+y+z \bullet \overline{z})\,(\overline{x}+z+y \bullet \overline{y})$ Distributive

$= (x+y+0)\,(\overline{x}+z+0)$ Postulate 5a

$= (x+y)\,(\overline{x}+z)$ Identity for +



Cost = 4 + 7 = 11

29

Another Example for POS and SOP Realization

Row number	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Figure 2.23. A three-variable function.

SOP:
F = $\sum m(1,4,5,6)$

POS:
F = $\Pi\, M(0,2,3,7)$

Then manipulation to reduce the cost

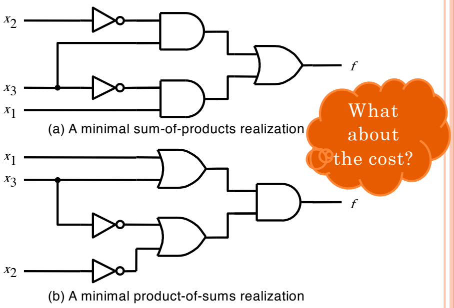


Figure 2.24. Two realizations of a function in Figure 2.23.

More example in Textbook

- Example 2.10 shows how to simplify the SOP
- Example 2.11 also shows how to simplify the SOP
- Example 2.12 shows how to simplify the POS
- List the Useful identities
  - $AB + A\overline{B} = A$
  - $(A+B)(A+\overline{B}) = A$

Relation of POS and SOP

- SOP is an AND-OR network
- POS is an OR-AND netwrok
- SOP and POS can be derived from each other
- For example:
  - Get the POS from the SOP
  - 1: Not F in the Truth Table
  - 2: Realize the Not F using SOP
  - 3: Not Not F to Get F and Simplify using the DeMorgan's theorem
    - Understand the identify : Not m_i = M_i

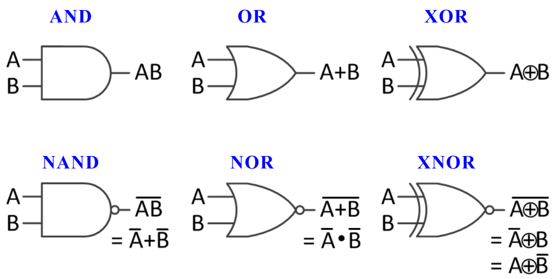
Conclusion

- SOP/AND-OR network and POS/OR-AND network
- The general method to synthesize the function
- The relationship between SOP and POS
- Can calculate the cost

Cost: Page: 89 Chapter 2.12.1
We will assume the input variables are available in both true and complemented forms at zero cost.

- Textbook Reading: Chapter 2.6, Pages:43-54
- Assignment: 2.10, 2.11, 2.12,
- 2.20, 2.22, 2.23

2.7 NAND and NOR Logic Networks



DeMorgan's Theorem in Terms of Logic Gate

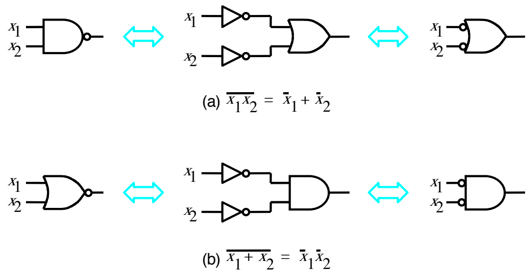
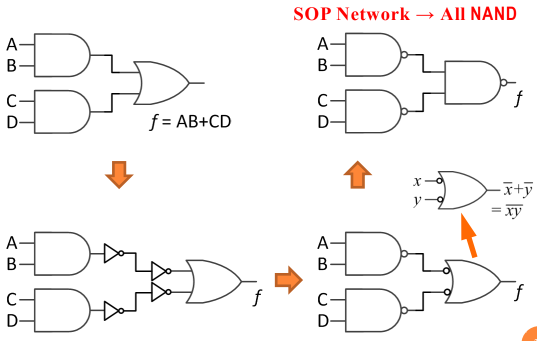


Figure 2.26. DeMorgan's theorem in terms of logic gates.

SOP Network



37

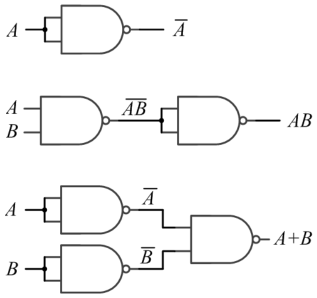
NAND + NOR are Universal

NAND

(i) $\overline{A} = \overline{A \cdot A}$

(ii) $A \cdot B = (\overline{\overline{A \cdot B}})$

(iii) $A + B = \overline{\overline{A} \cdot \overline{B}}$



38

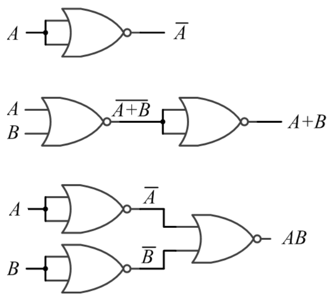
NAND + NOR are Universal

NOR

(i) $\overline{A+A} = \overline{A}$

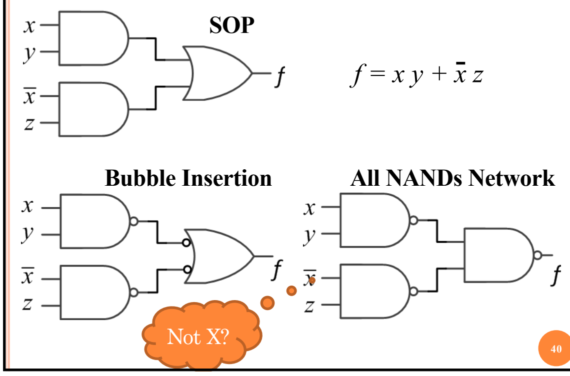
(ii) $(\overline{\overline{A+B}}) = A+B$

(iii) $\overline{\overline{A}+\overline{B}} = A \cdot B$



39

Convert SOP network to ALL NANDs Network



40

Convert SOP network to ALL NANDs network

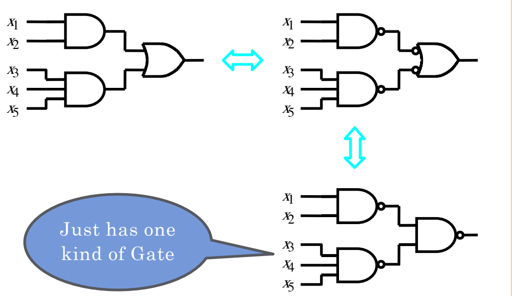
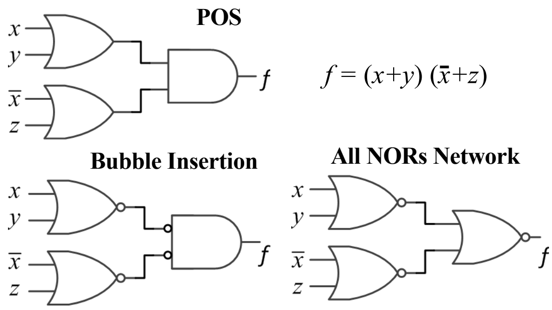


Figure 2.27. Using NAND gates to implement a sum-of-products.

Convert POS network to ALL NORs network



42

Convert POS network to ALL NORs network

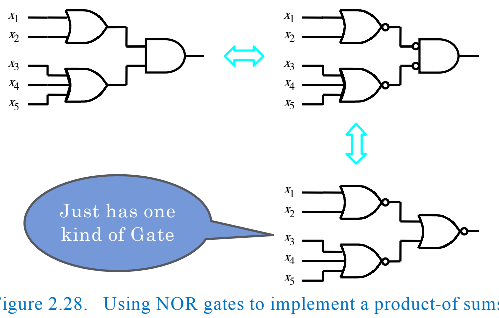


Figure 2.28. Using NOR gates to implement a product-of sums.

Example in the Textbook

- Example 2.13
- Implement the function F(x1,x2,x3) = ∑ m(2,3,4,6,7) using NOR gates only.
- Ans: $F = (x_1 + x_2)(x_2 + \bar{x}_3)$

- Example 2.14
- Implement the above function using the NAND gates only
- Ans: $F = x_2 + x_1\bar{x}_3$

44

Logic Circuit using NAND and NOR

$F = (x_1 + x_2)(x_2 + \bar{x}_3)$ $F = x_2 + x_1\bar{x}_3$

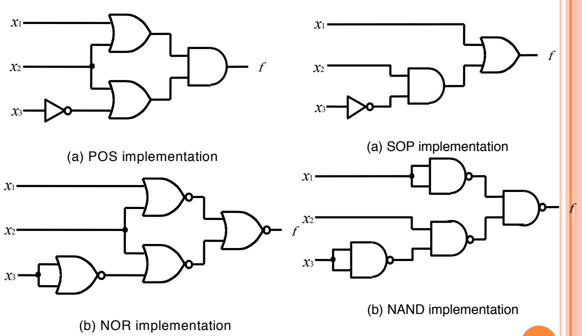


Figure 2.29 NOR-gate realization Figure 2.30. NAND-gate realization

Two Questions?

- Why do we need the NAND or NOR Network?
  - Because simple transistor circuit
  - Show the transistor model

- Can we directly derive the logic expression of NOR/NAND network
  - Yes F= NOT(NOT F)
  - Example
  - $F = AB + CD = \overline{\overline{AB} \cdot \overline{CD}}$

46

Conclusion

- NAND Logic networks
- NOR Logic networks
- Can solve simple problem to Circuit

- Textbook Reading: Chapter 2.7, Pages:54-59
- Assignment: 2.28, 2.31, 2.35

47

2.8 Design Examples

- (1) Three-way light control

Assume that a large room has three doors and that a switch near each door controls a light in the room. It has to be possible to turn the light on or off by changing the state of any one of the switches.

48

THREE-WAY LIGHT CONTROL

- A. Truth table

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Figure 2.31. Truth table for a three-way light control.

THREE-WAY LIGHT CONTROL

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- B. The canonical function
- sum-of-products form

$$f = m_1 + m_2 + m_4 + m_7$$
$$= \bar{x}_1\bar{x}_2x_3 + \bar{x}_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3 + x_1x_2x_3$$

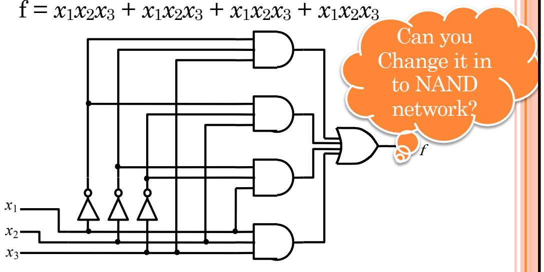
- product-of-sums form

$$f = M_0 \cdot M_3 \cdot M_5 \cdot M_6$$
$$= (x_1 + x_2 + x_3)(x_1 + \bar{x}_2 + \bar{x}_3)(\bar{x}_1 + x_2 + \bar{x}_3)(\bar{x}_1 + \bar{x}_2 + x_3)$$

THREE-WAY LIGHT CONTROL

- C. SOP realization

$$f = \bar{x}_1\bar{x}_2x_3 + \bar{x}_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3 + x_1x_2x_3$$



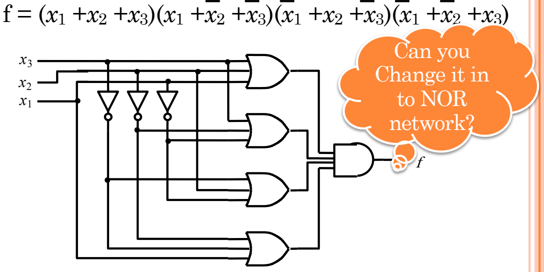
(a) Sum-of-products realization

Figure 2.32. Implementation of the function in Figure 2.31.

THREE-WAY LIGHT CONTROL

- C. POS realization

$$f = (x_1 + x_2 + x_3)(x_1 + \bar{x}_2 + \bar{x}_3)(\bar{x}_1 + x_2 + \bar{x}_3)(\bar{x}_1 + \bar{x}_2 + x_3)$$



(b) Product-of-sums realization

Figure 2.32. Implementation of the function in Figure 2.31.

MULTIPLEXER CIRCUIT

- (2) Multiplexer circuit

Suppose that there are two sources of data, provided as input signals $x1$ and $x2$.

We want to design a circuit that produces an output that has the same value as either $x1$ or $x2$, dependent on the value of a selection control signal s.

Therefore, the circuit should have three inputs: $x1$, $x2$, and s.

Assume that the output of the circuit will be the same as the value of input $x1$ if s=0, and it will be the same as $x2$ if s=1.

MULTIPLEXER CIRCUIT

- A. Truth table

s	x_1	x_2	$f(s, x_1, x_2)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

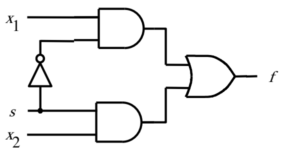
(a) Truth table

- B. The canonical function

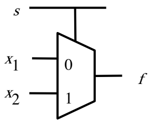
$$f = \bar{s}x_1 + sx_2$$

Multiplexer circuit

- C. Circuit realization: $f = \bar{s}x_1 + sx_2$



(b) Circuit



(c) Graphical symbol

s	$f(s, x_1, x_2)$
0	x_1
1	x_2

(d) More compact truth-table representation

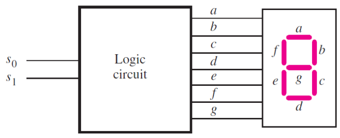
55

Extension of Multiplexer

- 4-to-1
- 8-to-1

56

Number Display



(a) Logic circuit and 7-segment display

	s_1	s_0	a	b	c	d	e	f	g
0	0	0	1	1	1	1	1	1	0
1	0	1	0	1	1	0	0	0	0
2	1	0	1	1	0	1	1	0	1

(b) Truth table

$a = d = e = \bar{s}_0$
b=1
$c = \bar{s}_1$
$f = \bar{s}_1 \bar{s}_0$
$g = s_1 \bar{s}_0$

Figure 2.34. Display of numbers.

57

- Problems: For the timing diagram in Figure p2.3, synthesize the function $f(x1, x2, x3)$ in the simplest sum-of-products form.

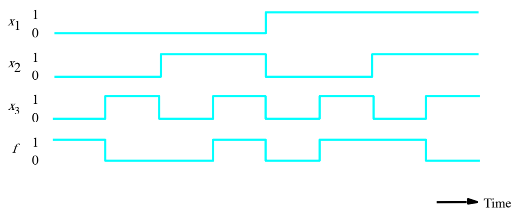


Figure p2.3. A timing diagram representing a logic function.

58

Conclusion

- Can solve simple problem to Circuit

- Textbook Reading: Chapter 2.8, Pages:59-64
- Assignment: 2.51, 2.52, 2.54, 2.56

59