Combinational Logic Design Principles

Switching algebra

Doru Todinca

Department of Computers and Information Technology
Politehnica University of Timisoara

Outline

Outline

Introduction. Definitions

- Logic circuits are classified as *combinational* or *sequential*
- "A *combinational circuit* is one whose outputs depend only on its current inputs"
- "The outputs of a *sequential circuit* depend not only on its current inputs, but also on past sequence of inputs, possible arbitrarily far back in time" [Wakerly]
- A combinational circuit should not contain *feedback loops*
- "A *feedback loop* is a signal path of a circuit that allows the output of a gate to propagate back to the input of the same gate"
- In general a feedback loop creates sequential behaviour
- "Combinational circuit *analysis*: we start with a logic diagram and proceed to a formal description of the function performed by the circuit such as a truth table or a logic expression"

Definitions

- *Synthesis* is the opposite process, where we start with a formal description and obtain a logic diagram
- *Logic design* starts with an informal description of the circuit (in words), from which we obtain first a formal description and at the end the logic diagram
- Hence, logic design includes synthesis
- Usually the most difficult and creative part is to obtain a formal description from the informal description
- Once we have the formal description we can use tools for synthesis, in order to obtain the logic diagram for a target technology (e.g. FPGA, ASIC, etc)
- Combinational circuits can have more than one output, but in this chapter we discuss only techniques that apply to combinational circuits with one output

Outline

Boolean algebra

- Boolean, or switching algebra, deals with two truth values: FALSE and TRUE, or 0 and 1, or LOW and HIGH (signal voltages)
- Created by George Boole in 1854
- Claude Shannon (1938): adapted the Boolean algebra to switching circuits (relays at that time):
  - A variable expresses the condition of a switching device: closed (1) or open (0)
- Positive logic convention: we associate logic 0 to LOW signal values and logic 1 to HIGH signal values
- Negative logic convention: we associate logic 0 to HIGH values and logic 1 to LOW values (seldom used)
- The choice of positive or negative convention does not affect the results of boolean algebra
- We will use this when we will talk about *duality*

Switching algebra

- A variable denotes the value of a signal: X, Y1, Input1, etc
- A *literal* is a logic variable or its complement: X, X', Y1', etc
- An *expression* combines
  - literals
  - logic operators: AND ($\cdot$), OR ($+$), complementation (')
  - and parenthesis
- Examples of logic expressions:
  - $(X' + Y1) \cdot W$
  - $((Y + Z1') \cdot CS\_L) \cdot RESET'$
- An equation has the form: *variable* = *expression*
- Examples:
  - $P = (A \cdot B + (C \cdot D))' + Z1$
  - $((Y + Z1') \cdot CS\_L) \cdot RESET' = Q0$

Outline

Axioms of switching algebra

Axioms, or *postulates*, are a minimal set of definitions and relations that we assume to be true, and from which we can derive all other relations and informations of a mathematical system

(A1)	$X = 0$ if $X \neq 1$	(A1')	$X = 1$ if $X \neq 0$
(A2)	If $X = 0$, then $X' = 1$	(A2')	If $X = 1$, then $X' = 0$
(A3)	$0 \cdot 0 = 0$	(A3')	$1 + 1 = 1$
(A4)	$1 \cdot 1 = 1$	(A4')	$0 + 0 = 0$
(A5)	$0 \cdot 1 = 1 \cdot 0 = 0$	(A5')	$1 + 0 = 0 + 1 = 1$

Table 1: Axioms of switching algebra

Axioms of switching algebra

- Axioms (A1) and (A1') formalize the fact that a boolean variable can take only two values: 0 and 1
- Axioms (A2) and (A2') describe the inverting operator (NOT, denoted also ')
  - The symbol for an inverter with input X and output Y and its algebraic notation $Y = X'$ are given in figure 1
  - X' is an expression, and $Y = X'$ is an equation
- Axioms (A3)-(A5) describe formally the AND operator, or *logical multiplication*, with the symbol *multiplication dot* $(\cdot)$
- Axioms (A3')-(A5') describe formally the OR operator, or *logical addition*, symbolized by a *plus sign* $(+)$
- *In a logical expression, multiplication has precedence over addition*
- The symbols for AND and OR gates and their algebraic equations are given in figure 2 (a) and (b)
- The five pairs of axioms (A1-A5) and (A1'-A5') completely characterize switching algebra

Gates: symbols and algebraic notations

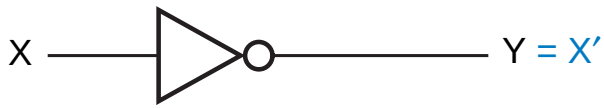


Figure 1: Inverter

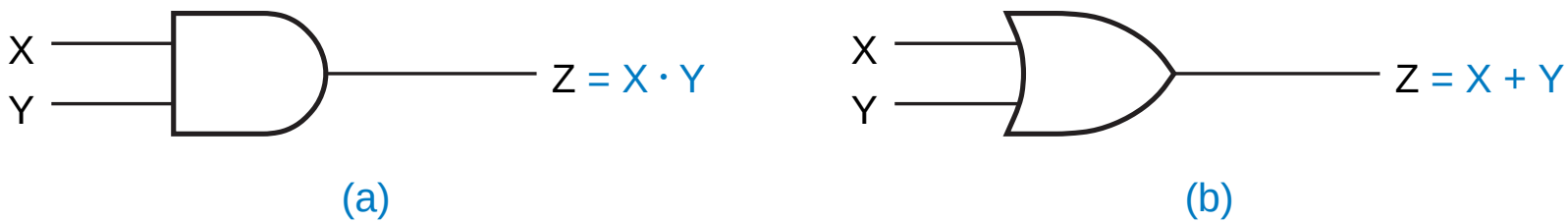
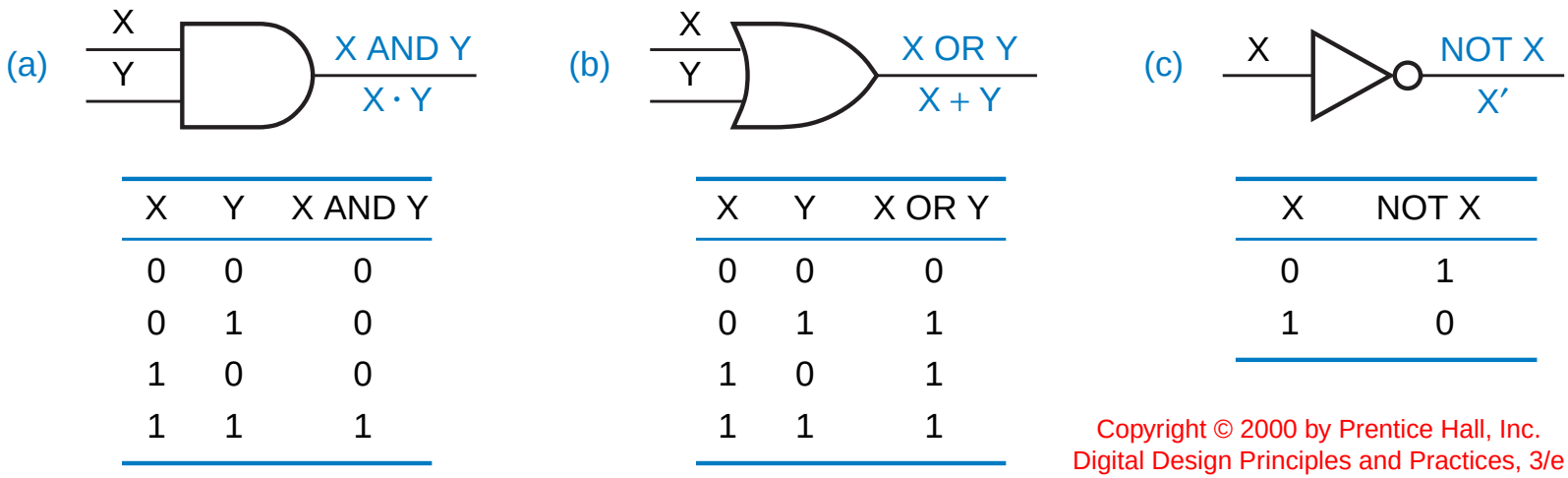


Figure 2: (a) AND gate and (b) OR gate

Digital gates and truth tables



X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

X	NOT X
0	1
1	0

Figure 3: Fundamental gates and truth tables: (a) for AND gate, (b) for OR gate, (c) for NOT gate

A gate's behaviour can be expressed using the truth table (see figure 3)
The truth tables for NOT, AND and OR gates are equivalent to the axioms (A2)-(A5) and (A2')-(A5')

Outline

Theorems of switching algebra

- Switching algebra theorems are statements that are always true and that can be obtained from axioms
- The theorems are very useful for simplifying algebraic expressions used for analysis and synthesis of combinational devices
- Most theorems can be proved by induction: either *perfect induction* or *finite induction*
  - Perfect induction means to prove that the theorem is true for all possible cases
  - Finite induction means to prove that the theorem is true for $n = 2$ (the *basis step*) and that, if the theorem is true for $n = i$, then it is true for $n = i + 1$ (*induction step*)

Single-Variable Theorems

Table 4-1
Switching-algebra theorems with one variable.

(T1)	$X + 0 = X$	(T1′)	$X \cdot 1 = X$	(Identities)
(T2)	$X + 1 = 1$	(T2′)	$X \cdot 0 = 0$	(Null elements)
(T3)	$X + X = X$	(T3′)	$X \cdot X = X$	(Idempotency)
(T4)	$(X')' = X$			(Involution)
(T5)	$X + X' = 1$	(T5′)	$X \cdot X' = 0$	(Complements)

Figure 4: Switching algebra theorems with one variable

Single-Variable Theorems

- Single variable theorems allow us to simplify algebraic expression:
  - For example, to replace $X + 0$ with X, $X + 1$ with 1, $X + X$ with X
  - Or, to replace $X \cdot 1$ with X, $X \cdot 0$ with 0, $X \cdot X$ with X
- All can be proved by perfect induction
- Proof of (T1). We can have two situations, because X can have only two values (according to A1 and A1'):
  1. If $X = 0$ (T1) becomes $0 + 0 = 0$, which is true, according to (A4')
  2. If $X = 1$ (T1) becomes $1 + 0 = 0$, true, according to (A5')
- All single variable theorems can be proved in a similar way (at the lab !)

Two- and Three-Variable Theorems

Table 4-2 Switching-algebra theorems with two or three variables.

(T6)	$X + Y = Y + X$	(T6′)	$X \cdot Y = Y \cdot X$	(Commutativity)
(T7)	$(X + Y) + Z = X + (Y + Z)$	(T7′)	$(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z)$	(Associativity)
(T8)	$X \cdot Y + X \cdot Z = X \cdot (Y + Z)$	(T8′)	$(X + Y) \cdot (X + Z) = X + Y \cdot Z$	(Distributivity)
(T9)	$X + X \cdot Y = X$	(T9′)	$X \cdot (X + Y) = X$	(Covering)
(T10)	$X \cdot Y + X \cdot Y' = X$	(T10′)	$(X + Y) \cdot (X + Y') = X$	(Combining)
(T11)	$X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$			(Consensus)
(T11′)	$(X + Y) \cdot (X' + Z) \cdot (Y + Z) = (X + Y) \cdot (X' + Z)$			

Figure 5: Switching algebra theorems with two and three variable

In all theorems it is possible to replace any variable with an arbitrary logic expression.

Commutativity and Associativity

- *Commutativity:*
  - T6 and T6' indicate that we can change the order of therms in a logical sum or a logical product
- *Associativity:*
  - Without associativity, an expression like $W + X + Y + Z$ or $W \cdot X \cdot Y \cdot Z$ is ambiguous
  - theorems T7 and T7' indicate that parenthesizetion is not relevant,
  - so that we can write $W + X + Y + Z$ instead of $(((W + X) + Y) + Z)$,
  - or $W \cdot X \cdot Y \cdot Z$ instead of, e.g., $(W \cdot X) \cdot (Y \cdot Z)$
- Associativity and commutativity tell us that:
  - We can extend the $\cdot$ and $+$ operators, that have been defined as *binary* operators, to any number of variables
  - That means that we can have AND and OR gates with 2, 3, 4, 8, ... inputs
  - We may connect the gates' inputs in any order

Two- and Three-Variable Theorems: Distributivity

- *Distributivity:*
  - Theorem T8 looks like distributivity of real and integer number multiplication over addition
  - It allows us to "multiply out" expressions in order to obtain a *sum-of-product* form:
  - $W \cdot (X + Y + Z) = W \cdot ((X + Y) + Z) =$
    $W \cdot (X + Y) + W \cdot Z = W \cdot X + W \cdot Y + W \cdot Z$
  - Theorem T8' does not hold for real (or integer) number addition and multiplication, but in Boolean algebra, logic addition is distributive over logic multiplication
  - It means that we can "add out" expressions to obtain the *product-of-sum* form:
  - $W + (X \cdot Y \cdot Z) = W + (X \cdot (Y \cdot Z)) =$
    $(W + X) \cdot (W + (Y \cdot Z)) = (W + X) \cdot (W + Y) \cdot (W + Z)$

2- and 3-Variable Theorems: Covering and Combining

- Both covering and combining theorems are used for minimization of logic expressions
- They can be proved by perfect induction, but also using other theorems, as follows:

$$\begin{aligned} X + X \cdot Y &= X \cdot 1 + X \cdot Y && \text{(according to T1')} \\ &= X \cdot (1 + Y) && \text{(according to T8)} \\ &= X \cdot 1 && \text{(according to T2)} \\ &= X && \text{(according to T1')} \end{aligned}$$

Table 2: Proof of covering theorem

$$\begin{aligned} X \cdot Y + X \cdot Y' &= X \cdot (Y + Y') && \text{(according to T8)} \\ &= X \cdot 1 && \text{(according to T5)} \\ &= X && \text{(according to T1')} \end{aligned}$$

Table 3: Proof of combining theorem

The consensus theorem

- T11 is called the consensus theorem
- The term $Y \cdot Z$ is called *the consensus of* $X \cdot Y$ *and* $X' \cdot Z$
- The theorem can be proved by perfect induction, or by the following steps:
  1. If $Y \cdot Z = 1$ then both Y and Z must be 1.
  2. It results that either $X \cdot Y = 1$ or $X' \cdot Z = 1$ (since either X or X' must be 1).
  3. That means that from the expression $X \cdot Y + X' \cdot Z + Y \cdot Z$ the term $Y \cdot Z$ can be eliminated
  4. Hence, $X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$
  5. If $Y \cdot Z = 0$, then the theorem is obviously true.
- Another proof:
  $X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z + Y \cdot Z \cdot 1 =$
  $X \cdot Y + X' \cdot Z + Y \cdot Z \cdot (X + X') = (X \cdot Y + X \cdot Y \cdot Z) + (X' \cdot Z +$
  $X' \cdot Z \cdot Y) = X \cdot Y \cdot (1 + Z) + X' \cdot Z \cdot (1 + Y) = X \cdot Y + X' \cdot Z$

n-Variable Theorems

Table 4-3 Switching-algebra theorems with *n* variables.

(T12)	$X + X + \cdots + X = X$	(Generalized idempotency)
(T12′)	$X \cdot X \cdot \ \cdots \ \cdot X = X$	
(T13)	$(X_1 \cdot X_2 \cdot \ \cdots \ \cdot X_n)' = X_1' + X_2' + \cdots + X_n'$	(DeMorgan's theorems)
(T13′)	$(X_1 + X_2 + \cdots + X_n)' = X_1' \cdot X_2' \cdot \ \cdots \ \cdot X_n'$	
(T14)	$[F(X_1, X_2, \ldots, X_n, +, \cdot)]' = F(X_1', X_2', \ldots, X_n', \cdot, +)$	(Generalized DeMorgan's theorem)
(T15)	$F(X_1, X_2, \ldots, X_n) = X_1 \cdot F(1, X_2, \ldots, X_n) + X_1' \cdot F(0, X_2, \ldots, X_n)$	(Shannon's expansion theorems)
(T15′)	$F(X_1, X_2, \ldots, X_n) = [X_1 + F(0, X_2, \ldots, X_n)] \cdot [X_1' + F(1, X_2, \ldots, X_n)]$	

Figure 6: Switching algebra theorems with n variable

- The theorems are true for an arbitrary number of variables, *n*.
- Most of them can be proved by finite induction.
- DeMorgan's theorems T13 and T13' are probably the most used theorems from switching algebra.

n-Variable Theorems: proof of T12

- Proof is by finite induction:
- Basis step: for $n = 2$ the theorem is true, according to T3.
- Induction step: suppose that for $n = i$ T12 is true, i.e. $\underbrace{X + X + \ldots + X}_{i} = X$
- We will prove T12 for $n = i + 1$:

$$
\begin{aligned}
\underbrace{X + X + \ldots + X}_{i+1} &= X + (\underbrace{X + \ldots + X}_{i}) \\
&= X + X && (n = i) \\
&= X && \text{(according to T3)}
\end{aligned}
$$

Table 4: Proof of generalized idempotency by finite induction

DeMorgan: T13

- Theorem T13 says that an n-input AND gate with negated output is equivalent with an n-input OR gate with negated inputs
- Figure 7 (a) and (b) illustrate this, for $n = 2$
- Figure 7 (c) and (d) show the gates symbols
- In 7 (d) the bubbles on the inputs signify that the gate's inputs are inverted
- Theorem T13' says that an n-input OR gate with inverted output is equivalent with an n-input AND gate with inverted inputs
- It is illustrated in figure 8 (a) and (b), while (c) and (d) contain the gates symbols

DeMorgan: T13

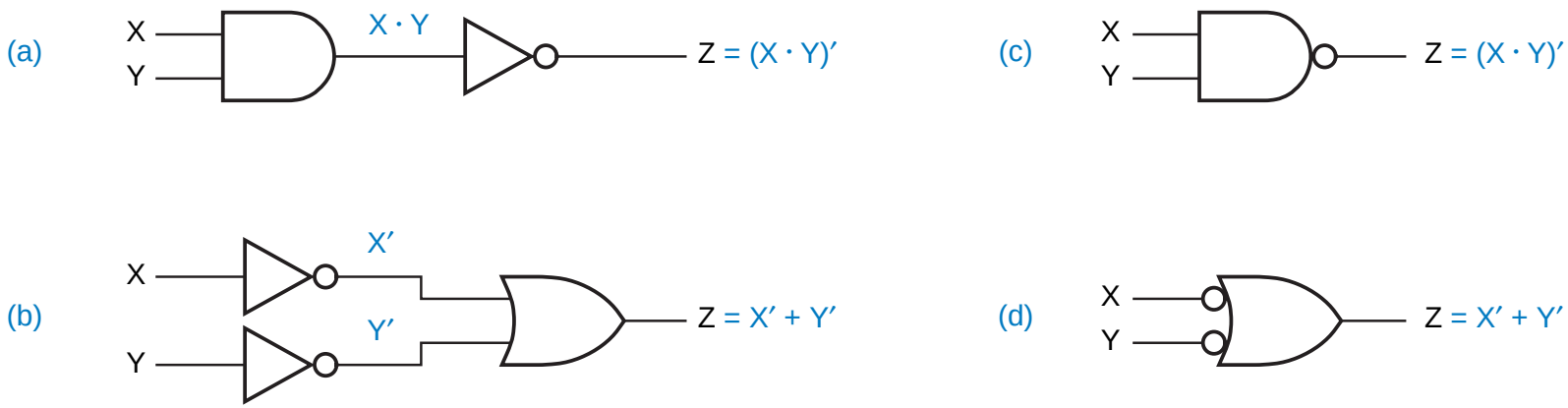


Figure 7: Equivalent circuits for NAND gate, according to DeMorgan's theorem T13

DeMorgan: T13'

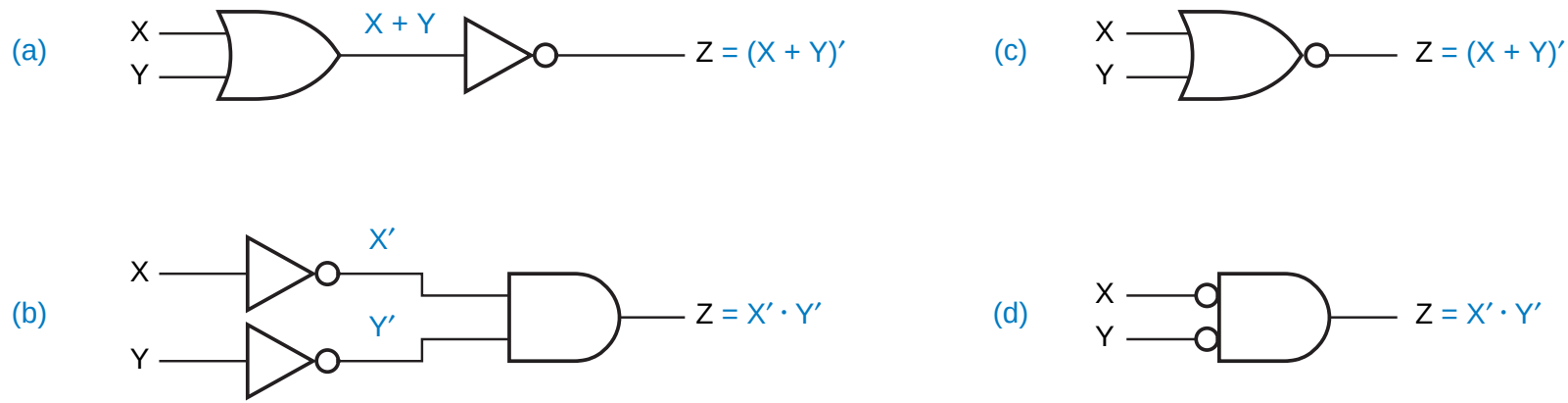


Figure 8: Equivalent circuits for NOR gate, according to DeMorgan's theorem T13'

DeMorgan

- Theorems T13 and T13' can be proved by finite induction
- The basis step, for $n = 2$, can be proved by perfect induction
- Theorems T13 and T13' can be used to prove T14, if we decompose the arbitrary function F in sum of products, and respectively product of sums and apply recursively T13 and T13'
- We will use another approach: first demonstrate T14 using *duality*, then T13 and T13' will result as particular cases of T14
- In T14, the complement $(F)'$ of a function F is defined as being the logic expression whose value is the opposite value of F, for every possible input combination.

Generalized DeMorgan

- Theorem T14 explains how to complement an n-variable logic expression:
  - by swapping $+$ and $\cdot$ and complementing all variables
- As a result, the uncomplemented variables will be complemented
- And the complemented variables will be uncomplemented (because of theorem T4, involution: $(X')' = X$)
- Example. Let's consider the following expression:
  - $F(W, X, Y, Z) = (W' + X) \cdot (X + Y) \cdot (W + (X' \cdot Z'))$
  - Applying theorem T14, we will obtain:
  - $[F(W, X, Y, Z)]' = ((W')' \cdot X') + (X' \cdot Y') + (W' \cdot ((X')' + (Z')'))$
  - Using theorem T4 we will have:
  - $[F(W, X, Y, Z)]' = (W \cdot X') + (X' \cdot Y') + (W' \cdot (X + Z))$

Outline

Duality

- All axioms of switching algebra are given in pairs
- The primed version of an axiom is obtained from the unprimed version by swapping 0 and 1 and, if present, $\cdot$ and $+$.
- If a theorem can be proved using certain axioms, then the primed theorem can be also proved (using the primed axioms)
- We can give the following *metatheorem*
- *Principle of duality*: Any theorem or identity in switching algebra remains true if 0 and 1 are swapped and $\cdot$ and $+$ are swapped too.
- A metatheorem is a theorem about theorems.
- We will formally define the *dual of a logic expression*

Duality

Definition

If $F(X_1, X_2, \ldots, X_n, \cdot, +, ')$ is a fully parenthesized logic expression involving the variables $X_1, X_2, \ldots, X_n$ and the operators $\cdot$, $+$, and $'$, then the dual of F, written F^D, is the same expression with $+$ and $\cdot$ swapped:

$$F^D(X_1, X_2, , \ldots X_n, \cdot, +, ') = F(X_1, X_2, \ldots X_n, +, \cdot, ')$$

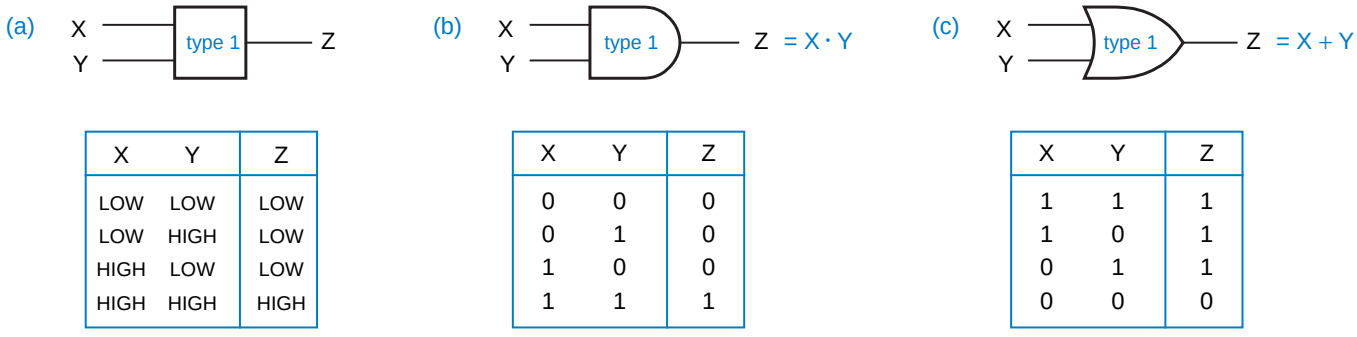
Using duality, theorem T14 can be expressed in the following way:

$$[F(X_1, X_2, , \ldots X_n)]' = F^D(X_1', X_2', \ldots X_n')$$

Duality: proof of theorem T14

- Figure 9 (a) shows the electrical function table of a gate that we call "type 1" gate
- In positive logic, i.e., if we associate 0 to LOW and 1 to HIGH, it is an AND gate (fig 9 (b))
- But in negative logic (LOW=1 and HIGH=0), "type 1" gate is an OR gate (fig 9 (c))
- In the same way we can consider the "type 2" gate from figure 10, which is an OR gate in positive logic and an AND gate in negative logic
- Obviously, similar tables can be given for gates with any number of inputs

Duality: proof of theorem T14

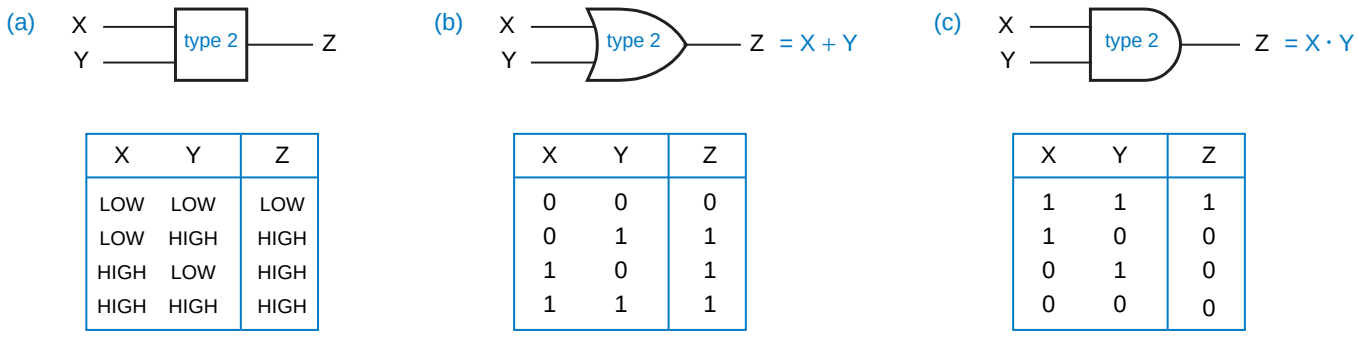


X	Y	Z
LOW	LOW	LOW
LOW	HIGH	LOW
HIGH	LOW	LOW
HIGH	HIGH	HIGH

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	Z
1	1	1
1	0	1
0	1	1
0	0	0

Figure 9: A "type 1" logic gate: (a) electrical functioning table; logic function table and symbol in positive logic (b) and negative logic (c)



X	Y	Z
LOW	LOW	LOW
LOW	HIGH	HIGH
HIGH	LOW	HIGH
HIGH	HIGH	HIGH

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

X	Y	Z
1	1	1
1	0	0
0	1	0
0	0	0

Figure 10: A "type 2" logic gate: (a) electrical functioning table; logic function table and symbol in positive logic (b) and negative logic (c)

Duality: proof of theorem T14

- Suppose we have an arbitrary logic expression $F(X_1, X_2, \ldots, X_n)$
- We can build a circuit for this logic expression under the positive logic convention, using inverters (NOT gates) for NOT operations, type 1 gates for AND operations and type 2 gates for OR operations
- We will obtain the circuit from figure 11
- Without changing the circuit, we change the logic convention and use negative logic
- The voltage levels do not change when we change the logic convention, which means that:
  - the "type 1" gates will be OR gates and "type 2" gates will be AND gates
  - each input will be replaced by its complement
  - inverters will remain unchanged
- We will obtain the circuit from figure 12

Duality: proof of theorem T14

- The function realized by the circuit from figure 12 is the dual of function F, realized by the circuit in positive logic (from figure 11)
- For each combination of input voltages, the circuit from figures 11 and 12 will produce the same voltage
- But, because we use different logic conventions, it means that the logic values of the circuit from the two figures will be always complemented.
- Which means that

$$F(X_1, X_2, , \ldots X_n) = [F^D(X_1', X_2', \ldots X_n')]'$$

- By complementing both sides, we obtain theorem T14.

Duality: proof of theorem T14

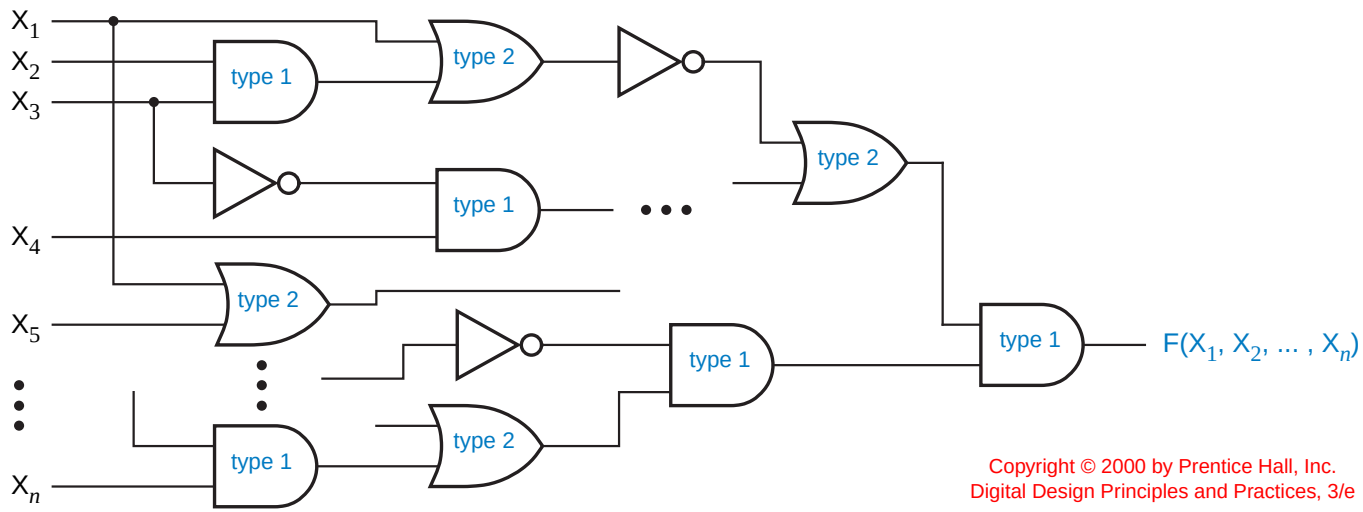


Figure 11: Circuit for a logic function using "type 1" and "type-2" logic gates in positive logic convention

Duality: proof of theorem T14

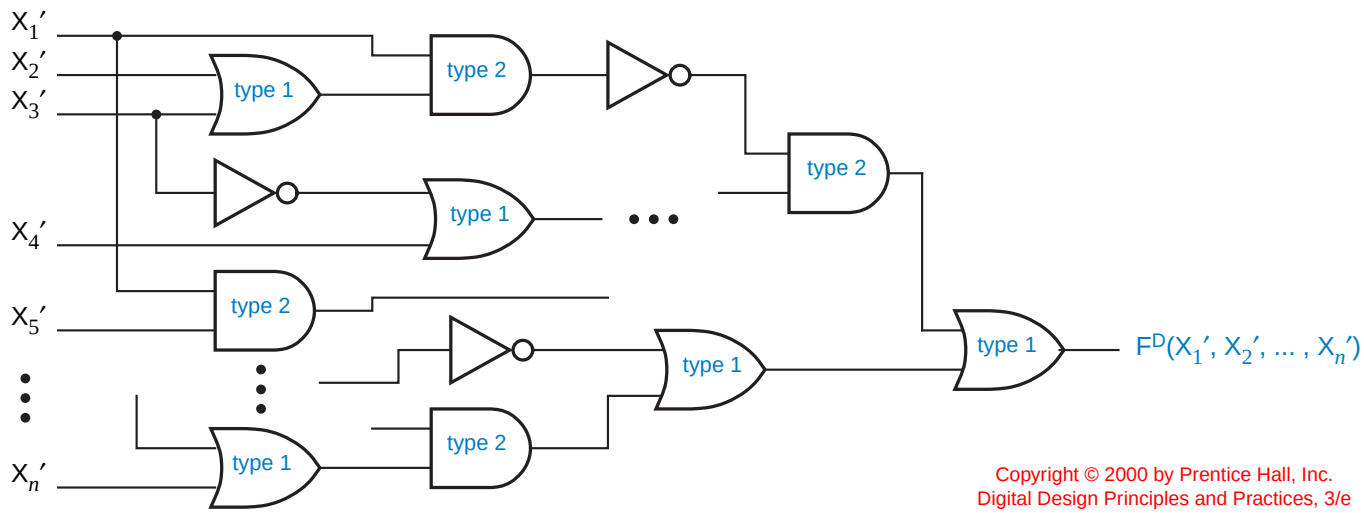


Figure 12: The circuit from figure 11 under negative logic convention

Outline

Truth table

- The *truth table* is the most basic representation of a logic function of a combinational circuit
- A truth table lists the output of a circuit for all its input combinations
- The order in which we list the input combinations is traditionally the ascending binary counting order
- A truth table for a combinational circuit with n inputs has 2^n rows.
- Figure 13 presents the general structure of a truth table for a circuit with 3 inputs (a three-variable truth table)
- With 3 variable a truth table has $2^3 = 8$ rows
- Figure 14 presents a truth table for a particular logic 3-input logic function F.

General truth table

Table 4-4
General truth table structure for a 3-variable logic function, F(X, Y, Z).

Row	X	Y	Z	F
0	0	0	0	F(0,0,0)
1	0	0	1	F(0,0,1)
2	0	1	0	F(0,1,0)
3	0	1	1	F(0,1,1)
4	1	0	0	F(1,0,0)
5	1	0	1	F(1,0,1)
6	1	1	0	F(1,1,0)
7	1	1	1	F(1,1,1)

Figure 13: General truth table for a 3-variable logic function

Truth table for a particular logic function

Row	X	Y	Z	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Table 4-5
Truth table for a particular 3-variable logic function, F(X, Y, Z).

Figure 14: The truth table for a particular 3-variable logic function

Standard representations of logic functions: Definitions

- A *literal* is a variable or the complement of a variable.
  - Examples: X, X', Y, Y', RESET.
- A *product term* is a single literal or a logic product of two or more literals.
  - Examples: Z', $W \cdot X \cdot Y$, $W \cdot X' \cdot Y' \cdot Z$
- A *sum-of-products* expression is a logical sum of product terms
  - Examples: $W \cdot X \cdot Y + W \cdot X' \cdot Y' + W' \cdot X \cdot Y$
- A *sum term* is a single literal or a logical sum of two or more literals.
  - Examples: $W + X + Y$, $W' + X' + Y'$, $W + X + Y' + Z'$
- A *product-of-sum* expression is a logical product of sum terms.
  - Examples: $(W + X + Y) \cdot (W' + X' + Y') \cdot (W' + X + Y)$, $Z' \cdot (W + X + Y' + Z')$

Standard representations of logic functions: Definitions

- A *normal term* is a product or sum term in which no variable appears more than once.
  - A non-normal term can always be simplified to a constant or a normal term using one of the theorems T3, T3'(idempotency), T5 or T5' (complements).
  - Examples of non-normal terms: $W \cdot X \cdot X \cdot Z$, $W \cdot X' \cdot Y' \cdot W'$, $W + W + X + Y$, $W + W' + Y + Z$
  - Examples of normal terms: $W \cdot X' \cdot Y' \cdot Z$, $W \cdot X \cdot Y' \cdot Z$, $W + X + Y + Z'$, $W' + X' + Y + Z$
- An n-variable *minterm* is a normal product term of n literals.
  - There are 2^n such product terms.
  - Examples of 3-variable minterms: $W \cdot X \cdot Y$, $W' \cdot X \cdot Y'$, $W' \cdot X' \cdot Y'$
  - Examples of 4-variable minterms: $W \cdot X \cdot Y \cdot Z$, $W' \cdot X' \cdot Y' \cdot Z$, $W' \cdot X' \cdot Y' \cdot Z'$
- An n-variable *maxterm* is a normal sum term of n literals.
  - Examples of 4-variable maxterms: $W + X + Y + Z$, $W' + X' + Y' + Z$, $W' + X' + Y' + Z'$

Minterms, maxterms and truth tables

- In an n-variable minterm (or maxterm), each variable appears exactly once, in either uncomplemented or complemented form
- There is a close correspondence between the truth table and minterms and maxterms.
- A minterm can be defined as a product term that is 1 in exactly one row of the truth table
- A maxterm can be defined as a sum term that is 0 in exactly one row of the truth table
- In figure 15 we can see this correspondence for a 3-variable truth table.

Minterms and maxterms

Table 4-6
Minterms and maxterms for a 3-variable logic function, F(X, Y, Z).

Row	X	Y	Z	F	*Minterm*	*Maxterm*
0	0	0	0	F(0,0,0)	$X' \cdot Y' \cdot Z'$	$X + Y + Z$
1	0	0	1	F(0,0,1)	$X' \cdot Y' \cdot Z$	$X + Y + Z'$
2	0	1	0	F(0,1,0)	$X' \cdot Y \cdot Z'$	$X + Y' + Z$
3	0	1	1	F(0,1,1)	$X' \cdot Y \cdot Z$	$X + Y' + Z'$
4	1	0	0	F(1,0,0)	$X \cdot Y' \cdot Z'$	$X' + Y + Z$
5	1	0	1	F(1,0,1)	$X \cdot Y' \cdot Z$	$X' + Y + Z'$
6	1	1	0	F(1,1,0)	$X \cdot Y \cdot Z'$	$X' + Y' + Z$
7	1	1	1	F(1,1,1)	$X \cdot Y \cdot Z$	$X' + Y' + Z'$

Figure 15: Minterms and maxterms for a 3-variable logic function F(X,Y,Z)

Minterm and maxterm number

- An n-variable minterm can be represented by an n-bit integer, named *the minterm number*
- The name *minterm* i will denote the minterm corresponding to row i of the truth table
- In minterm i, a particular variable appears complemented if the corresponding bit in the binary representation of i is 0, and uncomplemented, if the corresponding bit in the binary representation of number i is 1.
- Example:
  - the minterm 5 corresponds to row 5 in table 4-5 from figure 15
  - the binary representation of number 5 on 3 bits is 101
  - The corresponding minterm will be $X \cdot Y' \cdot Z$
- In *maxterm* i a variable is complemented if the corresponding bit in the binary representation of number i is 1 and uncomplemented if that corresponding bit is 0
  - Example: maxterm 5 is $X' + Y + Z'$

Canonical sum, minterm list

- The *canonical sum* of a logic function is the sum of the minterms corresponding to truth-table rows (input combinations) for which the function is 1
- Example: for the table 4-5 (from figure 14) the canonical sum is: $F = \sum_{X,Y,Z}(0,3,4,6,7) =$
  $X' \cdot Y' \cdot Z' + X' \cdot Y \cdot Z + X \cdot Y' \cdot Z' + X \cdot Y \cdot Z' + X \cdot Y \cdot Z$
- The notation $\sum_{X,Y,Z}(0,3,4,6,7)$ is a *minterm list*, meaning "the sum of minterms 0, 3, 4, 6 and 7 with variables X, Y, and Z "
- The minterm list is called also the *on-set* of the logic function because each minterm "turns on" the output for exactly one input combination.
- Any logic function can be written as a canonical sum.

Canonical product, maxterm list

- The canonical product of a logic function is a product of the maxterms corresponding to input combinations for which the function produces a 0 output.
- Example: the canonical product for the logic function in Table 4-5 from figure 15 is:
  $F = \prod_{X,Y,Z}(1,2,5) = (X+Y+Z')\cdot(X+Y'+Z)\cdot(X'+Y+Z')$
- The notation $\prod_{X,Y,Z}(1,2,5)$ is a *maxterm list* and means "the product of maxterms 1, 2, and 5 with variables X, Y, and Z"
- The maxterm list is known also as the *off-set* for the logic function because each maxterm "turns off" the function for exactly one input combination.
- Any logical function can be written as a canonical product.

Minterm list and maxterm list

- For a function of n variables, the possible minterm and maxterm numbers are in the set $\{0, 1, \ldots, 2^n - 1\}$
- A minterm list or a maxterm list contain a subset of these numbers
- The minterm list and the maxterm list are complemented
- Hence, in order to switch from minterm list to maxterm list (or vice-versa), take the set complemented.
- Examples:
  - $\sum_{A,B,C}(0, 1, 2, 3) = \prod_{A,B,C}(4, 5, 6, 7)$
  - $\sum_{X,Y}(1) = \prod_{X,Y}(0, 2, 3)$
  - $\sum_{W,X,Y,Z}(0, 1, 2, 3, 5, 7, 11, 13) = \prod_{W,X,Y,Z}(4, 6, 8, 9, 10, 12, 14, 15)$

Possible representations for a combinational logic function

1. A truth table
2. An algebraic sum of minterms, the canonical sum
3. A minterm list using the $\sum$ notation
4. An algebraic product of maxterms, the canonical product
5. A maxterm list using the $\prod$ notation.

All representations are equivalent. Given one of them, we can obtain all others.