Seventh Edition
Fundamentals of Logic Design

Charles H. Roth, Jr.   Larry L. Kinney

UNIT 4

Applications of Boolean Algebra Minterm and Maxterm Expansions

CENGAGE Learning

This chapter includes:

4.1   Conversion of English Sentences to Boolean Equations

4.2   Combinational Logic Design
        Using a Truth Table

4.3   Minterm and Maxterm Expansions

4.4   General Minterm and Maxterm Expansions

4.5   Incompletely Specified Functions

4.6   Examples of Truth Table Construction

4.7   Design of Binary Adders and Subtracters

Learning Objectives

1. Given a word description of the desired behavior of a logic circuit, write the output of the circuit as a function of the input variables. Specify this function as an algebraic expression or by means of a truth table, as is appropriate.

2. Given a truth table, write the function (or its complement) as both a minterm expansion (standard sum of products) and a maxterm expansion (standard product of sums). Be able to use both alphabetic and decimal notation.

3. Given an algebraic expression for a function, expand it algebraically to obtain the minterm or maxterm form.

Learning Objectives

4. Given one of the following: minterm expansion for F, minterm expansion for F, maxterm expansion for F, or maxterm expansion for F, find any of the other three forms.

5. Write the general form of the minterm and maxterm expansion of a function of n variables.

6. Explain why some functions contain don't-care terms.

7. Explain the operation of a full adder and a full subtracter and derive logic equations for these modules. Draw a block diagram for a parallel adder or subtracter and trace signals on the block diagram.

Conversion of English Sentences to Boolean Equations

Introduction:

❖Logic design problems are often stated in terms of one or more English sentences.

❖The first step in designing a logic circuit is to translate these sentences into Boolean equations. Must break down each sentence into phrases and associate a Boolean variable with each phrase.

❖If a phrase can have a value of true or false, then we can represent that phrase by a Boolean variable.

❖Phrases can be either true or false, or have no truth value.

Conversion of English Sentences to Boolean Equations

Example 1:

Statement: The alarm will ring iff the alarm switch is turned on and the door is not closed, or it is after 6 p.m. and the window is not closed.

This statement can be broken up into the following phrases with Boolean variables A,B,C,D and Z:

$$\underbrace{\text{The alarm will ring}}_{Z} \quad iff \quad \underbrace{\text{the alarm switch is on}}_{A} \quad and$$

$$\underbrace{\text{the door is not closed}}_{B'} \quad or \quad \underbrace{\text{it is after 6 P.M.}}_{C} \quad and$$

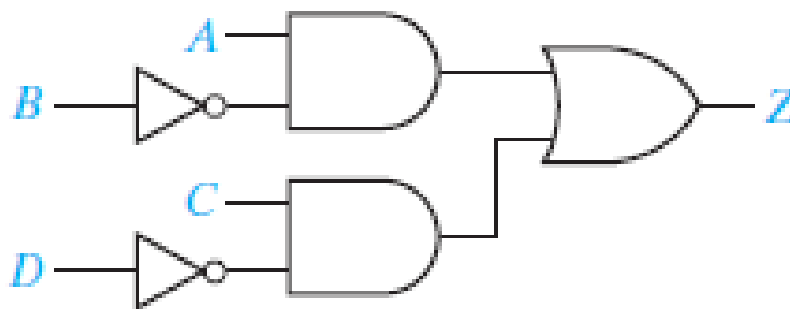$$\underbrace{\text{the window is not closed.}}_{D'}$$

Conversion of English Sentences to Boolean Equations

Example 1 (continued):

❖*Using this assignment of variables, the sentence can be translated into the following Boolean equation:*
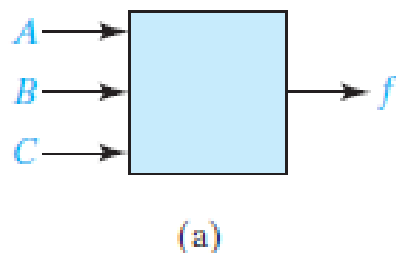
$$Z = AB + CD$$

❖This equation corresponds to the following circuit:

Combinational Logic Design Using a Truth Table

Logic Design with Truth Tables:

❖Take a three-input, one output system where A,B,C are inputs that represent the digits of binary number N and f is the output such that $f=1$ if $N>=011_2$ and $f=0$ if $N<011_2$.

A	B	C	f	f'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

(a)

(b)

Combinational Logic Design Using a Truth Table

Logic Design with Truth Tables:

❖ OR-ing the terms that yield value 1 will result in

$$f=A'BC+AB'C'+AB'C+ABC'+ABC$$

❖ Where the expression equals 1 if A,B, and C take on any of the five combinations of values 011, 100, 101, 110 or 111.

❖This can be simplified to

$$f=A'BC+AB'+AB=A'BC+A=A+BC$$

❖ The expression can also be written in terms of 0's:

$$f=(A+B+C)(A+B+C')(A+B'+C)$$

Minterm and Maxterm Expansions

Minterms:

❖A **minterm** of n variables is a product of n literals in which each variable appears exactly once in either true or complemented form, but not both.

❖Minterms are often written in abbreviated form—A′B′C′ is designated m0, A′B′C is designated m1, etc.

❖In general, the minterm which corresponds to row i of the truth table is designated mi (i is usually written in decimal).

❖When a function f is written as a sum of minterms as in Equation (4-1), this is referred to as a **minterm expansion** or a **standard sum of products**.

$$f(A, B, C) = m_3 + m_4 + m_5 + m_6 + m_7$$

$$f(A, B, C) = \Sigma m(3, 4, 5, 6, 7)$$

Minterm and Maxterm Expansions

Maxterms:

❖A **maxterm** of n variables is a sum of n literals in which each variable appears exactly once in either true or complemented form, but not both.

❖A maxterm is the complement of the corresponding minterm.

❖Maxterms are written in M-notation.

❖When a function f is written as a product of maxterms, this is referred to as a **maxterm expansion** or a **standard product of sums**.

$$f(A, B, C) = M_0 M_1 M_2 \qquad\qquad f(A, B, C) = \Pi\, M(0, 1, 2)$$

Minterm and Maxterm Expansions

Table of Minterms and Maxterms for Three Terms:

Row No.	A B C	Minterms	Maxterms
0	0 0 0	$A'B'C' = m_0$	$A + B + C = M_0$
1	0 0 1	$A'B'C = m_1$	$A + B + C' = M_1$
2	0 1 0	$A'BC' = m_2$	$A + B' + C = M_2$
3	0 1 1	$A'BC = m_3$	$A + B' + C' = M_3$
4	1 0 0	$AB'C' = m_4$	$A' + B + C = M_4$
5	1 0 1	$AB'C = m_5$	$A' + B + C' = M_5$
6	1 1 0	$ABC' = m_6$	$A' + B' + C = M_6$
7	1 1 1	$ABC = m_7$	$A' + B' + C' = M_7$

Minterm and Maxterm Expansions

Example 2:

Find the minterm expansion of $f(a, b, c, d) = a'(b' + d) + acd'$.

$$f = a'b' + a'd + acd'$$
$$= a'b'(c + c')(d + d') + a'd(b + b')(c + c') + acd'(b + b')$$
$$= a'b'c'd' + a'b'c'd + a'b'cd' + a'b'cd + \cancel{a'b'c'd} + \cancel{a'b'cd}$$
$$+ a'bc'd + a'bcd + abcd' + ab'cd' \tag{4-9}$$

Duplicate terms have been crossed out, because $X + X = X$. This expression can then be converted to decimal notation:

$$f = a'b'c'd' + a'b'c'd + a'b'cd' + a'b'cd + a'bc'd + a'bcd + abcd' + ab'cd'$$
$$\quad\ 0\,0\,0\,0 \quad\ 0\,0\,0\,1 \quad\ 00\,10 \quad\ 0\,0\,11 \quad\ 0\,10\,1 \quad\ 0\,111 \quad\ 1110 \quad\ 10\,10$$
$$f = \Sigma\, m(0, 1, 2, 3, 5, 7, 10, 14) \tag{4-10}$$

The maxterm expansion for f can then be obtained by listing the decimal integers (in the range 0 to 15) which do not correspond to minterms of f:

$$f = \Pi\, M(4, 6, 8, 9, 11, 12, 13, 15)$$

Minterm and Maxterm Expansions

Example 3:

Show that $a'c + b'c' + ab = a'b' + bc + ac'$.

We will find the minterm expansion of each side by supplying the missing variables. For the left side,

$$a'c(b + b') + b'c'(a + a') + ab(c + c')$$

$$= a'bc + a'b'c + ab'c' + a'b'c' + abc + abc'$$

$$= m_3 + m_1 + m_4 + m_0 + m_7 + m_6$$

For the right side,

$$a'b'(c + c') + bc(a + a') + ac'(b + b')$$

$$= a'b'c + a'b'c + abc + a'bc + abc' + ab'c'$$

$$= m_1 + m_0 + m_7 + m_3 + m_6 + m_4$$

Because the two minterm expansions are the same, the equation is valid.

General Minterm and Maxterm Expansions

General definition:

❖ A general truth table for three variables:

❖ To completely specify a function, all a_i's must be assigned values.

❖ 2^8 ways of filling the F column.

A	B	C	F
0	0	0	a_0
0	0	1	a_1
0	1	0	a_2
0	1	1	a_3
1	0	0	a_4
1	0	1	a_5
1	1	0	a_6
1	1	1	a_7

❖ Minterm expansion of a general function:

$$F = a_0 m_0 + a_1 m_1 + a_2 m_2 + \cdots + a_7 m_7 = \sum_{i=0}^{7} a_i m_i$$

❖ Maxterm expansion of a general function:

$$F = (a_0 + M_0)(a_1 + M_1)(a_2 + M_2) \cdots (a_7 + M_7) = \prod_{i=0}^{7} (a_i + M_i)$$

General Minterm and Maxterm Expansions

Conversion of Forms (Table 4-3):

	DESIRED FORM			
GIVEN FORM	Minterm Expansion of F	Maxterm Expansion of F	Minterm Expansion of F'	Maxterm Expansion of F'
Minterm Expansion of F	_____	maxterm nos. are those nos. not on the minterm list for F	list minterms not present in F	maxterm nos. are the same as minterm nos. of F
Maxterm Expansion of F	minterm nos. are those nos. not on the maxterm list for F	_____	minterm nos. are the same as maxterm nos. of F	list maxterms not present in F

General Minterm and Maxterm Expansions

Application of Conversion of Forms (Table 4-4):

	DESIRED FORM			
GIVEN FORM	Minterm Expansion of f	Maxterm Expansion of f	Minterm Expansion of f'	Maxterm Expansion of f'
$f = \Sigma m(3, 4, 5, 6, 7)$	_____	$\Pi M(0, 1, 2)$	$\Sigma m(0, 1, 2)$	$\Pi M(3, 4, 5, 6, 7)$
$f = \Pi M(0, 1, 2)$	$\Sigma m(3, 4, 5, 6, 7)$	_____	$\Sigma m(0, 1, 2)$	$\Pi M(3, 4, 5, 6, 7)$

Incompletely Specified Functions

Introduction:

❖ In some systems, certain combinations of inputs will never occur. For these combinations, we "don't care" what the value of F is. The function F is then considered **incompletely** specified.

❖ Truth table with don't-cares:

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

❖ d, and D specify "don't cares" in minterm and maxterm expansions respectively.

Examples of Truth Table Construction

Example 4:

We will design a simple binary adder that adds two 1-bit binary numbers, a and b, to give a 2-bit sum. The numeric values for the adder inputs and output are as follows:

a	b	Sum	
0	0	00	$(0 + 0 = 0)$
0	1	01	$(0 + 1 = 1)$
1	0	01	$(1 + 0 = 1)$
1	1	10	$(1 + 1 = 2)$

Examples of Truth Table Construction

Example 4 (continued):

We will represent inputs to the adder by the logic variables A and B and the 2-bit sum by the logic variables X and Y, and we construct a truth table:

A	B	X	Y
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Because a numeric value of 0 is represented by a logic 0 and a numeric value of 1 by a logic 1, the 0's and 1's in the truth table are exactly the same as in the previous table. From the truth table,

$$X = AB \quad \text{and} \quad Y = A'B + AB' = A \oplus B$$

Examples of Truth Table Construction

Example 5:

The four inputs to a circuit (A, B, C, D) represent an 8-4-2-1 binary-coded-decimal digit. Design the circuit so that the output (Z) is 1 iff the decimal number represented by the inputs is exactly divisible by 3. Assume that only valid BCD digits occur as inputs.

The digits 0, 3, 6, and 9 are exactly divisible by 3, so $Z = 1$ for the input combinations $ABCD = 0000, 0011, 0110,$ and 1001. The input combinations 1010, 1011, 1100, 1101, 1110, and 1111 do not represent valid BCD digits and will never occur, so Z is a don't-care for these combinations. This leads to the following truth table:

A	B	C	D	Z
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Examples of Truth Table Construction

Example 5 (continued):

The corresponding output function is

$$Z = \Sigma\, m(0, 3, 6, 9) + \Sigma\, d(10, 11, 12, 13, 14, 15)$$

In order to find the simplest circuit which will realize Z, we must choose some of the don't-cares (X's) to be 0 and some to be 1. The easiest way to do this is to use a Karnaugh map as described in Unit 5.

❖ See pages 104-107 for more examples
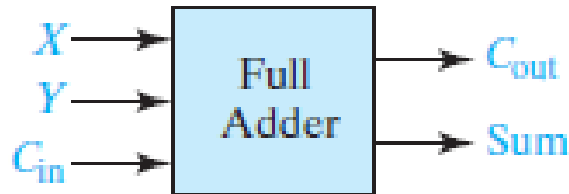
Design of Binary Adders and Subtracters

Full Adder:

❖Below, the carry output from the first full adder serves as the carry input to the second full adder, etc.

end-around carry for 1's complement
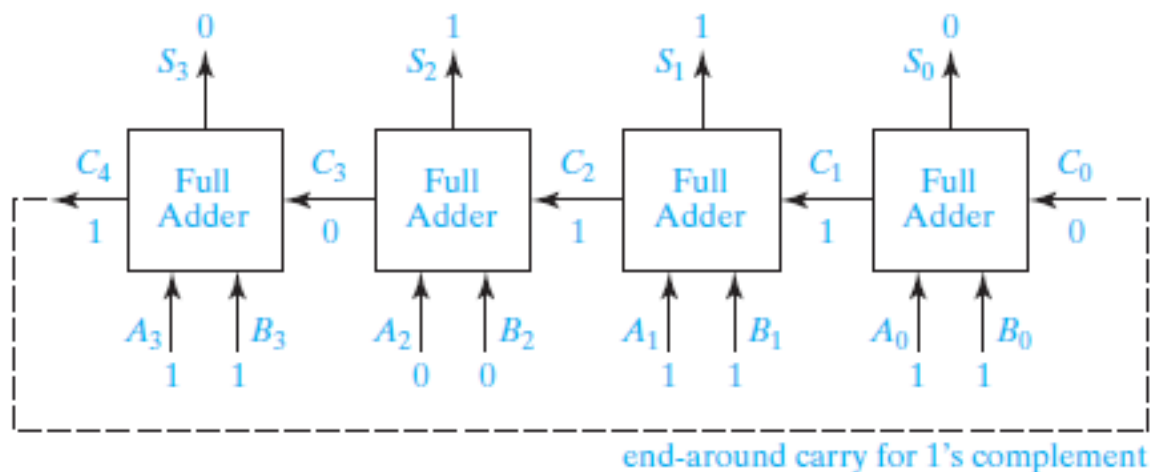
Design of Binary Adders and Subtracters

Full Adder Truth Table:

X	Y	C_{in}	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Design of Binary Adders and Subtracters

Logic Equations for Full Adder:

$$Sum = X'Y'C_{in} + X'YC'_{in} + XY'C'_{in} + XYC_{in}$$

$$= X'(Y'C_{in} + YC'_{in}) + X(Y'C'_{in} + YC_{in}) \qquad (4\text{-}20)$$

$$= X'(Y \oplus C_{in}) + X(Y \oplus C_{in})' = X \oplus Y \oplus C_{in}$$

$$C_{out} = X'YC_{in} + XY'C_{in} + XYC'_{in} + XYC_{in}$$

$$= (X'YC_{in} + XYC_{in}) + (XY'C_{in} + XYC_{in}) + (XYC'_{in} + XYC_{in}) \quad (4\text{-}21)$$

$$= YC_{in} + XC_{in} + XY$$
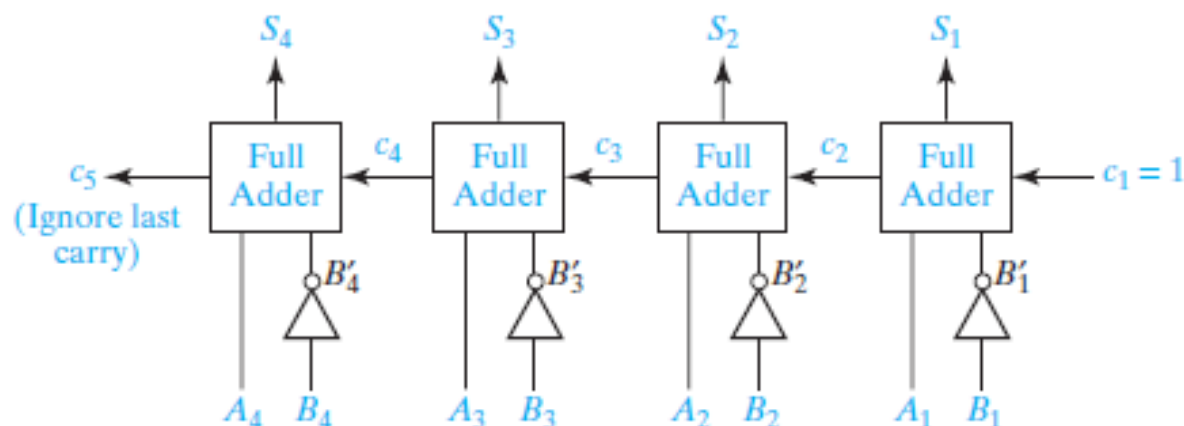
Design of Binary Adders and Subtracters

Implementation of Full Adder:
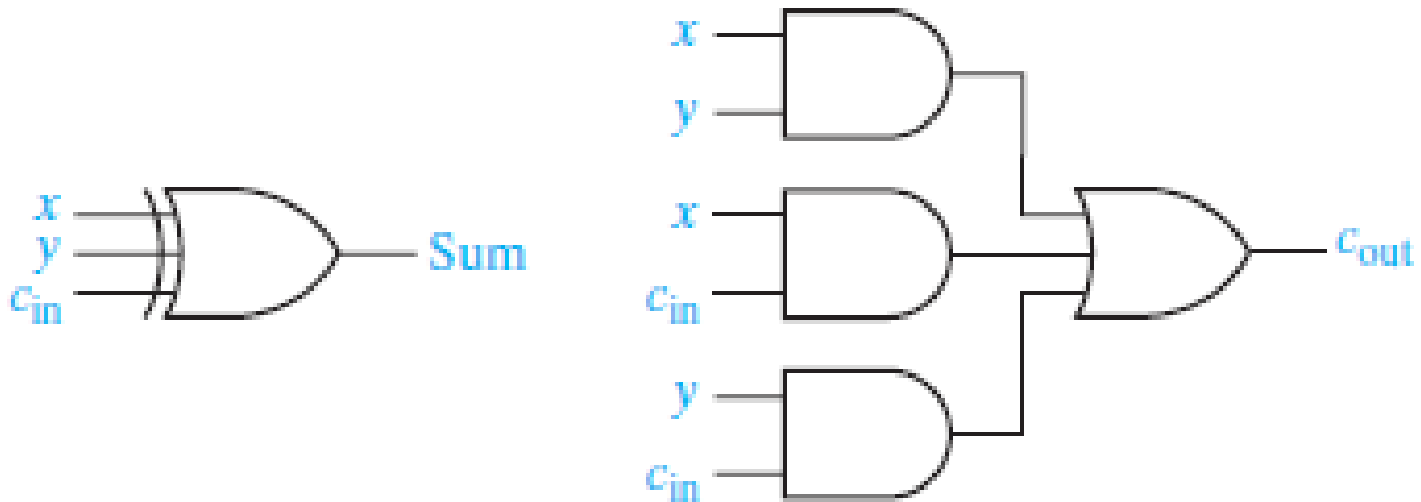
Design of Binary Adders and Subtracters

Binary Subtracter:

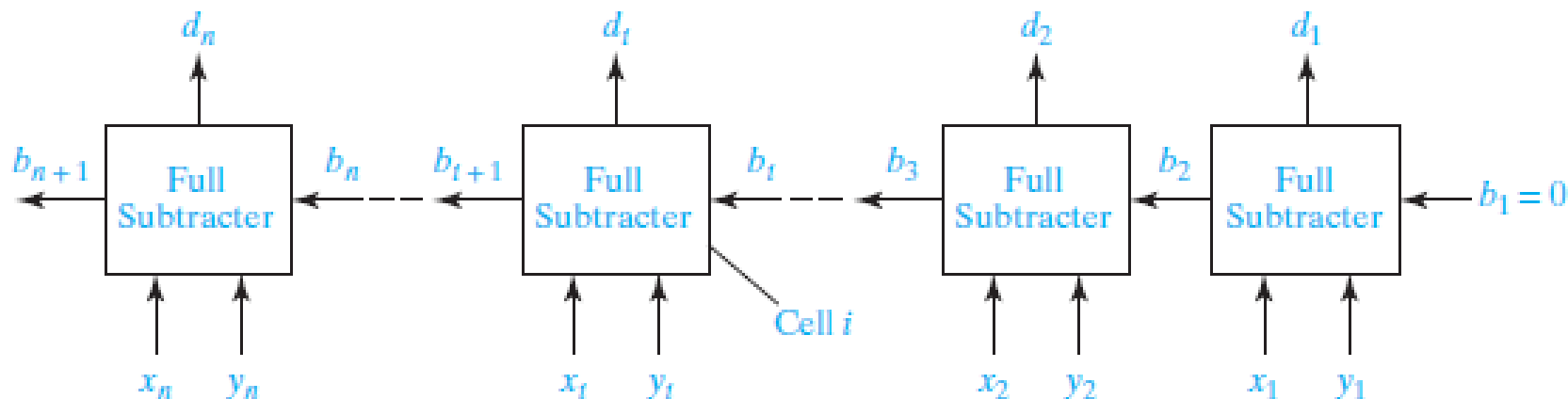❖ A binary subtracter can be made using full adders as shown below:

x_i	y_i	b_i	b_{i+1}	d_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Design of Binary Adders and Subtracters

Parallel Subtracter:

❖ A parallel subtracter can be made using full subtracters as shown below:

Design of Binary Adders and Subtracters

Carry Lookahead Adder:

In the parallel adder the carry out of the ith stage can be written as:

$$C_{i+1} = A_i B_i + C_i(A_i + B_i) = A_i B_i + C_i(A_i \oplus B_i) = G_i + P_i C_i$$

where $G_i = A_i B_i$ indicates the condition for the *i*th stage to *generate* a carry out and $P_i = A_i \oplus B_i$ (or $P_i = A_i + B_i$) indicates the condition for the *i*th stage to propagate a carry in to the carry out. Then C_{i+2} can be expressed in terms of C_i.

$$C_{i+1} = G_i + P_i C_i$$
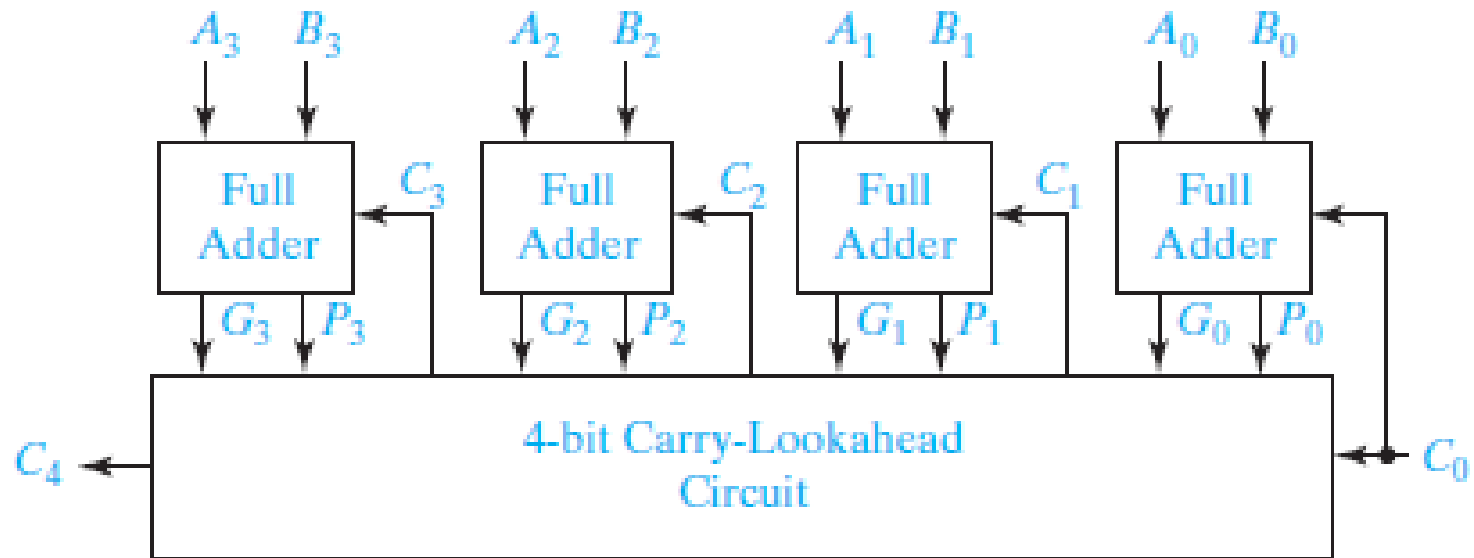$$C_{i+2} = G_{i+1} + P_{i+1}G_i + P_{i+1}P_i C_i$$
$$C_{i+3} = G_{i+2} + P_{i+2}G_{i+1} + P_{i+2}P_{i+1}G_i + P_{i+2}P_{i+1}P_i C_i \qquad (4\text{-}23)$$
$$C_{i+4} = G_{i+3} + P_{i+3}G_{i+2} + P_{i+3}P_{i+2}G_{i+1} + P_{i+3}P_{i+2}P_{i+1}G_i + P_{i+3}P_{i+2}P_{i+1}P_i C_i$$

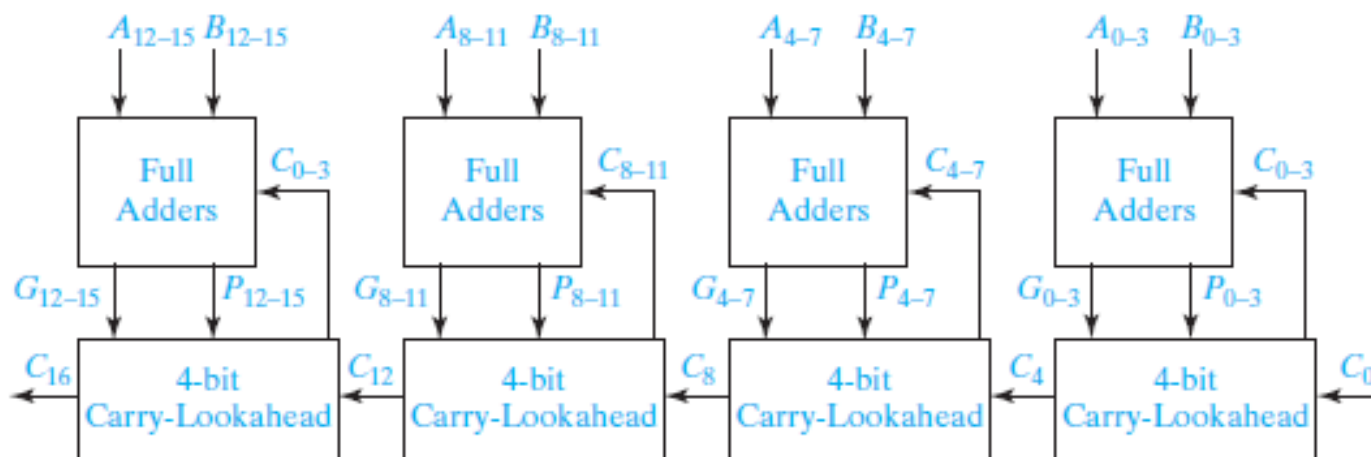# Design of Binary Adders and Subtracters

4 Bit Adder with Carry-Lookahead:

Design of Binary Adders and Subtracters

16-bit adder with Carry-Lookahead:

❖To reduce the delay of the adder without increasing the size of the carry-lookahead circuit, a second level of carry-lookahead circuits can be connected to the first level carry-lookahead circuits.

Design of Binary Adders and Subtracters

16-bit adder with Carry-Lookahead:

$C_4 = \boldsymbol{G_0} + \boldsymbol{P_0}C_0$

  where $\boldsymbol{G_0} = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0$ and $\boldsymbol{P_0} = P_3P_2P_1P_0$

$C_8 = \boldsymbol{G_4} + \boldsymbol{P_4}C_4$

  where $\boldsymbol{G_4} = G_7 + P_7G_6 + P_7P_6G_5 + P_7P_6P_5G_4$ and $\boldsymbol{P_4} = P_7P_6P_5P_4$

$C_{12} = \boldsymbol{G_8} + \boldsymbol{P_8}C_8$

  where $\boldsymbol{G_8} = G_{11} + P_{11}G_{10} + P_{11}P_{10}G_9 + P_{11}P_{10}P_9G_8$ and $\boldsymbol{P_8} = P_{11}P_{10}P_9P_8$

$C_{16} = \boldsymbol{G_{12}} + \boldsymbol{P_{12}}C_{12}$

  where $\boldsymbol{G_{12}} = G_{15} + P_{15}G_{14} + P_{15}P_{14}G_{13} + P_{15}P_{14}P_{13}G_{12}$ and $\boldsymbol{P_{12}} = P_{15}P_{14}P_{13}P_{12}$

Now these equations for C_4, C_8, C_{12}, and C_{16} can be written in terms of C_0.

$$C_4 = \boldsymbol{G_0} + \boldsymbol{P_0}C_0$$
$$C_8 = \boldsymbol{G_4} + \boldsymbol{P_4}\boldsymbol{G_0} + \boldsymbol{P_4}\boldsymbol{P_0}C_0$$
$$C_{12} = \boldsymbol{G_8} + \boldsymbol{P_8}\boldsymbol{G_4} + \boldsymbol{P_8}\boldsymbol{P_4}\boldsymbol{G_0} + \boldsymbol{P_8}\boldsymbol{P_4}\boldsymbol{P_0}C_0$$
$$C_{16} = \boldsymbol{G_{12}} + \boldsymbol{P_{12}}\boldsymbol{G_8} + \boldsymbol{P_{12}}\boldsymbol{P_8}\boldsymbol{G_4} + \boldsymbol{P_{12}}\boldsymbol{P_8}\boldsymbol{P_4}\boldsymbol{G_0} + \boldsymbol{P_{12}}\boldsymbol{P_8}\boldsymbol{P_4}\boldsymbol{P_0}C_0$$

Design of Binary Adders and Subtracters

16-bit adder with Second Level Carry-Lookahead:

33
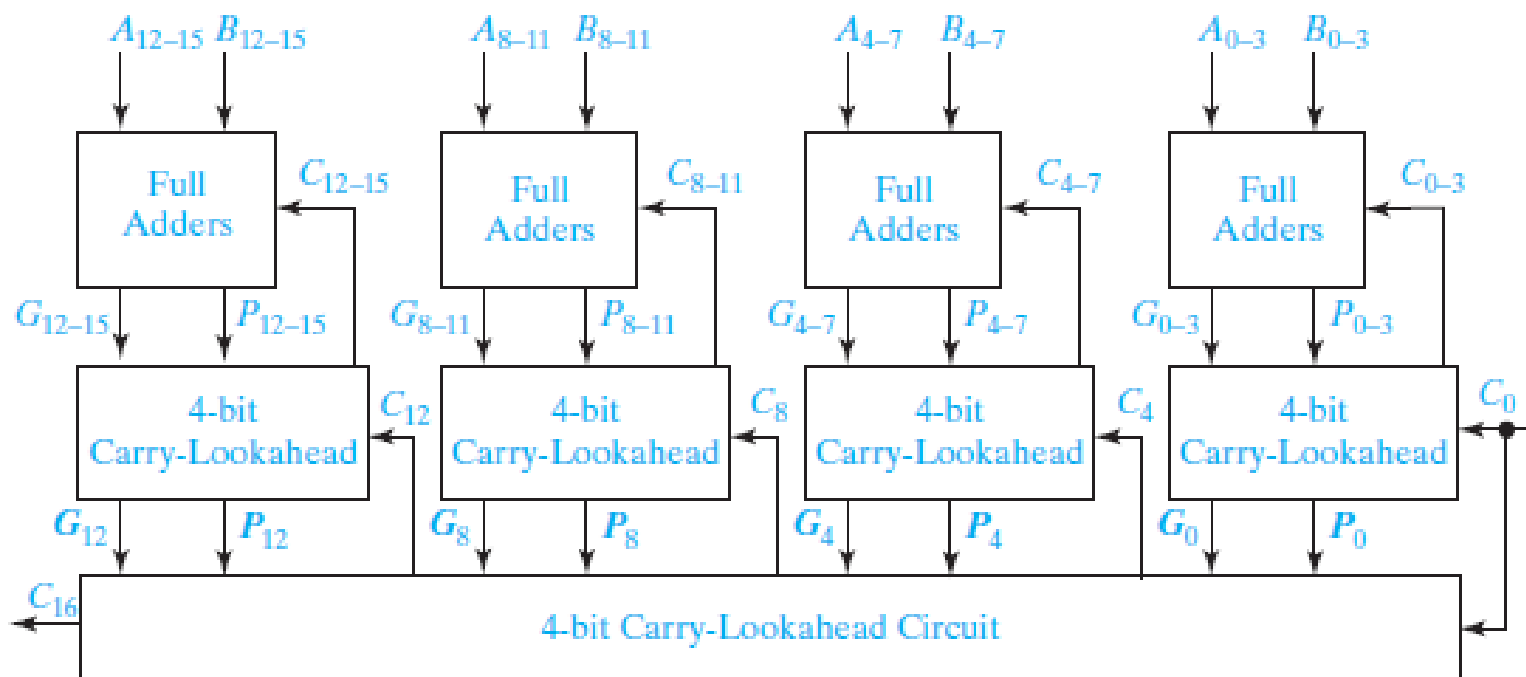
Design of Binary Adders and Subtracters

16-bit adder with Second Level Carry-Lookahead:

$$C_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} G_{i-2} + P_i P_{i-1} P_{i-2} G_{i-3} + P_i P_{i-1} P_{i-2} P_{i-3} C_{i-3}$$

$$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0 = \boldsymbol{G}_0 + \boldsymbol{P}_0 C_0$$

This expression can be expanded to express C_{i+1} in terms of C_{i-1}.

$$C_{i+1} = G_i + C_i P_i + G_i = (G_{i-1} + C_{i-1} P_{i-1}) P_i = G_i + P_i G_{i-1} + P_i P_{i-1} C_{i-1}$$

This procedure can be continued to obtain

$$C_{i+1} = G_i + C_i P_i$$
$$C_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} C_{i-1}$$
$$C_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} G_{i-2} + P_i P_{i-1} P_{i-2} C_{i-2}$$
$$C_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} G_{i-2} + P_i P_{i-1} P_{i-2} G_{i-3} + P_i P_{i-1} P_{i-2} P_{i-3} C_{i-3}$$

and so on.

34