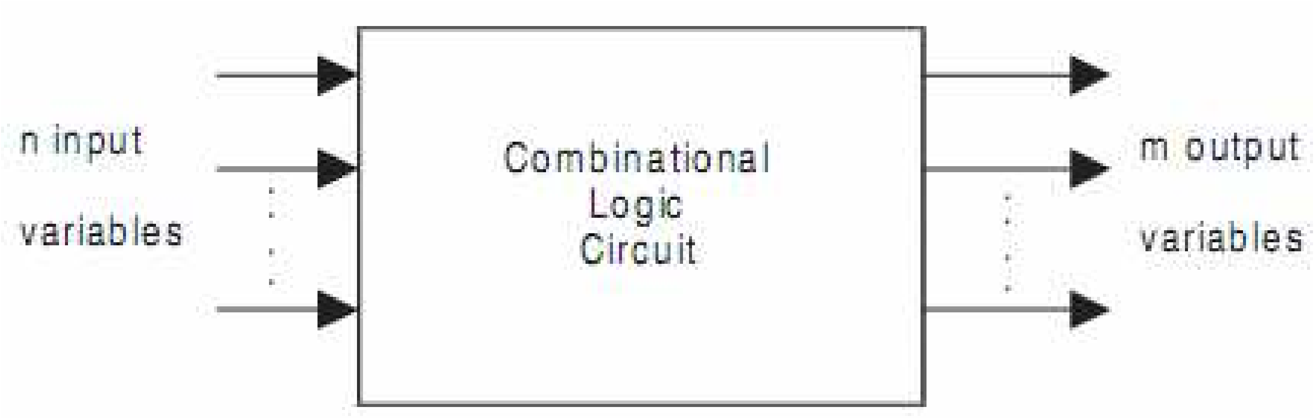
LOGIC CIRCUITS

Introduction

The digital system consists of two types of circuits, namely
(i) Combinational circuits and
(ii) Sequential circuit

A **combinational circuit** consists of logic gates, where outputs are at any instant and are determined only by the present combination of inputs without regard to previous inputs or previous state of outputs A combinational circuit performs a specific information-processing operation assigned logically by a set of Boolean functions



Sequential circuits contain logic gates as well as memory cells. Their outputs depend on the present inputs and also on the states of memory elements. Since the outputs of sequential circuits depend not only on the present inputs but also on past inputs, the circuit behavior must be specified by a time sequence of inputs and memory states.

DESIGN PROCEDURE

Any combinational circuit can be designed by the following steps of design procedure.

1. The problem is stated.
2. Identify the input variables and output functions.
3. The input and output variables are assigned letter symbols.
4. The truth table is prepared that completely defines the relationship between the input variables and output functions.
5. The simplified Boolean expression is obtained by any method of minimization—algebraic method, Karnaugh map method, or tabulation method.
6. A logic diagram is realized from the simplified expression using logic gates.

ADDERS

Addition of two binary digits is the most basic arithmetic operation. The simple addition consists of four possible elementary operations, which are
0+0 = 0, 0+1 = 1, 1+0 = 1, and 1+1 = 10.

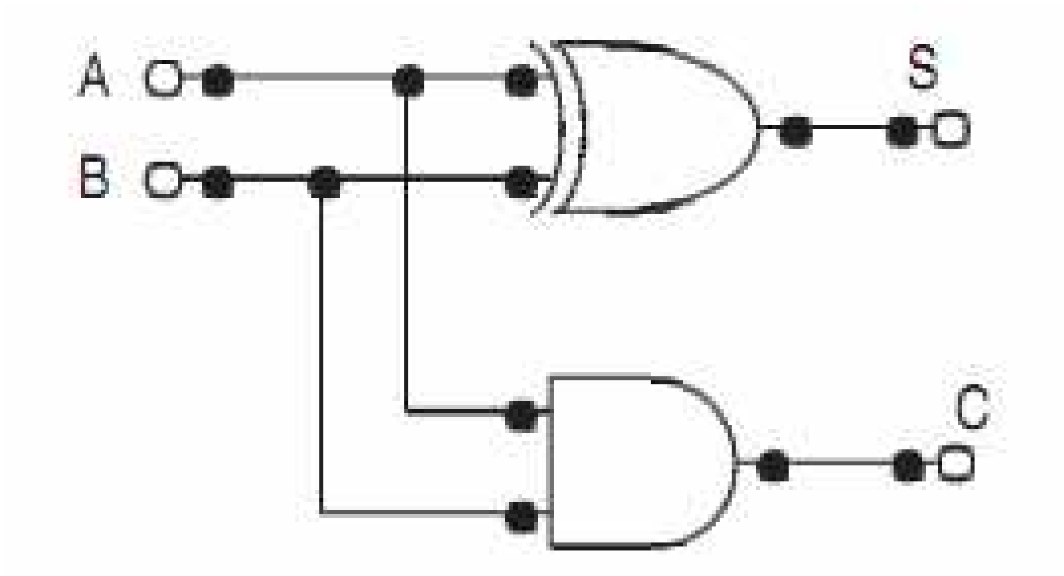
The first three operations produce a sum of one digit, but the fourth operation produces a sum consisting of two digits. The higher significant bit of this result is called the carry. A combinational circuit that performs the addition of two bits as described above is called a half-adder.

Design of Half-adders

As described above, a half-adder has two inputs and two outputs. Let the input variables augend and addend be designated as A and B, and output functions be designated as S for sum and C for carry. The truth table for the functions is below.

Input variables		*Output variables*	
A	*B*	*S*	*C*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = A'B + AB' \quad and$$
$$C = AB$$



Design of Full-adders

- A combinational circuit of full-adder performs the operation of addition of three bits—the augend, addend, and previous carry, and produces the outputs sum and carry.

- Let us designate the input variables augend as A, addend as B, and previous carry as X, and outputs sum as S and carry as C.

- As there are three input variables, eight different input combinations are possible.

Input variables			*Outputs*	
X	A	B	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

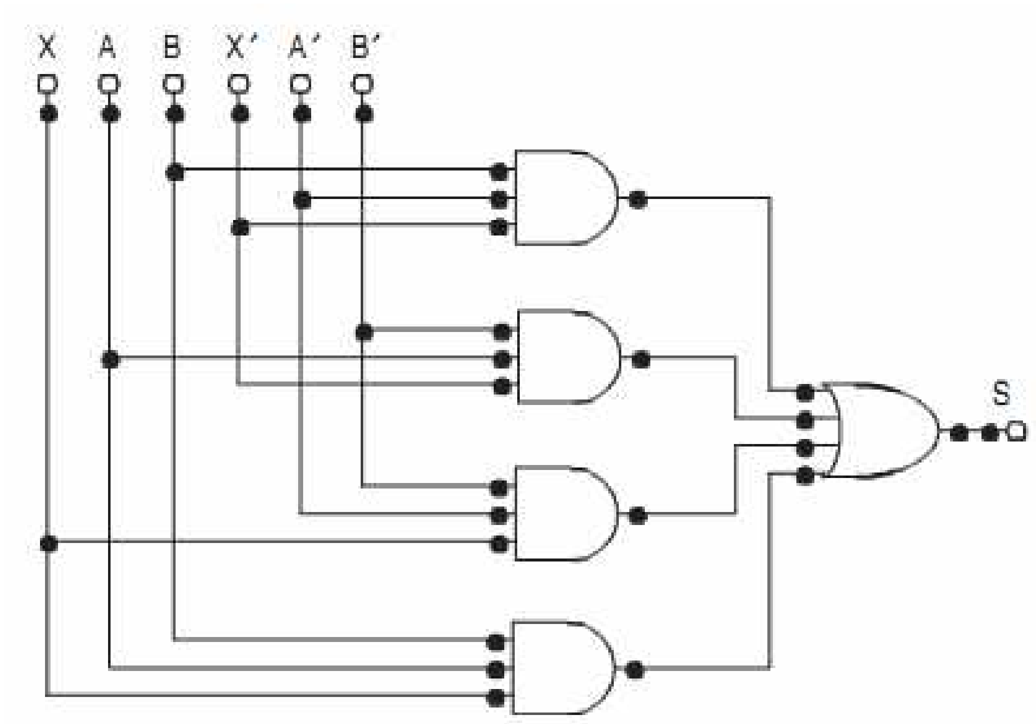
To derive the simplified Boolean expression from the truth table, the Karnaugh map method is adopted

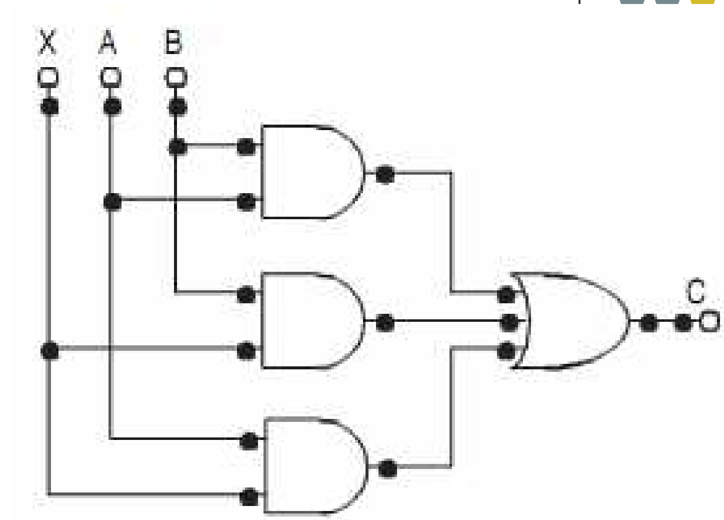
	A′B′	A′B	AB	AB′
X′		1		1
X	1		1	

	A′B′	A′B	AB	AB′
X′			1	
X		1	1	1

The simplified Boolean expressions of the outputs are

S = X′A′B + X′AB′ + XA′B′ + XAB and
C = AB + BX + AX.





$$= X'A'B + X'AB' + XA'B' + XAB$$

$$= X'(A'B + AB') + X(A'B' + AB)$$

$$= X'(A \oplus B) + X(A \oplus B)'$$

$$= X \oplus A \oplus B$$

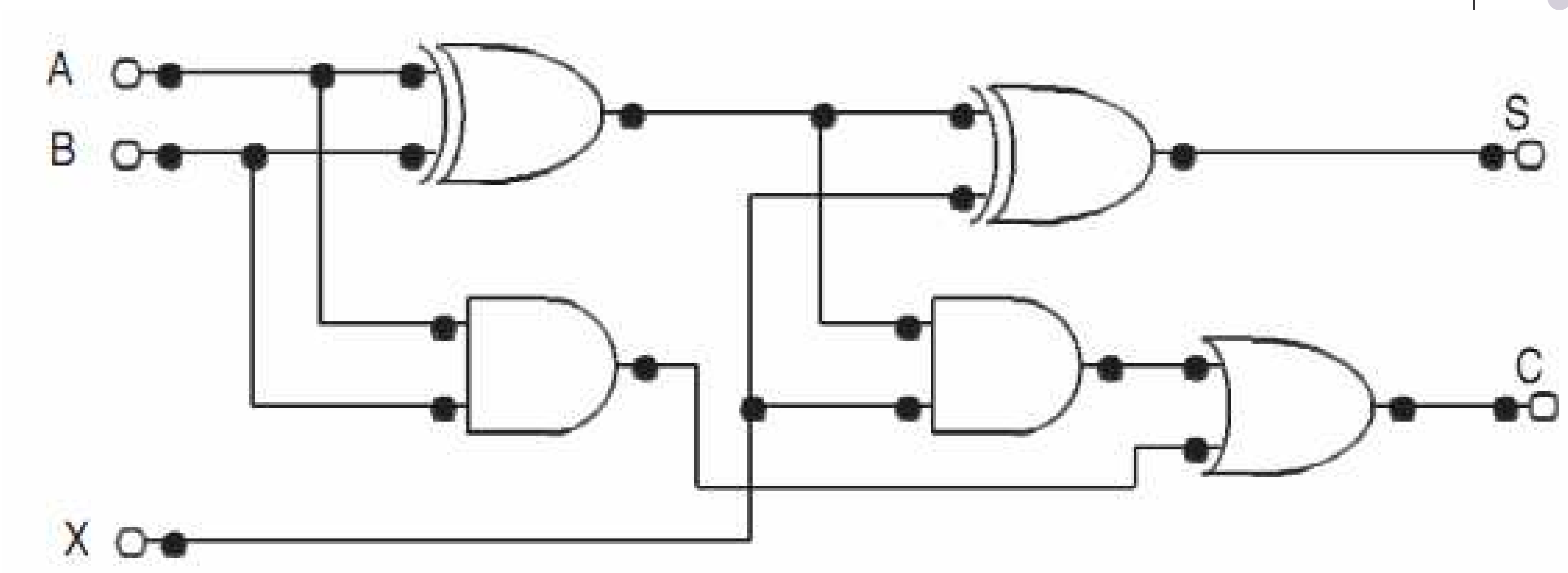
$$C = AB + BX + AX = AB + X(A + B)$$

$$= AB + X(AB + AB' + AB + A'B)$$

$$= AB + X(AB + AB' + A'B)$$

$$= AB + XAB + X(AB' + A'B)$$

$$= AB + X(A \oplus B)$$



The full-adder developed in above Figure consists of two 2-input AND gates, two 2-input XOR (Exclusive-OR) gates and one 2-input OR gate.

SUBTRACTORS

- Subtraction is the other basic function of arithmetic operations of information-processing tasks of digital computers.

- Similar to the addition function, subtraction of two binary digits consists of four possible elementary operations, which are 0–0 = 0, 0–1 = 1 with borrow of 1, 1–0 = 1, and 1–1 = 0.

- The first, third, and fourth operations produce a subtraction of one digit, but the second operation produces a difference bit as well as a borrow bit.

- The borrow bit is used for subtraction of the next higher significant bit. A combinational circuit that performs the subtraction of two bits as described above is called a half-subtractor

- The digit from which another digit is subtracted is called the minuend and the digit which is to be subtracted is called the subtrahend.

- When the minuend and subtrahend numbers contain more significant digits, the borrow obtained from the subtraction of two bits is subtracted from the next higher-order pair of signifi cant bits. Here the subtraction operation involves three bits—the minuend bit, subtrahend bit, and the borrow bit, and produces a different result as well as a borrow.

- The combinational circuit that performs this type of addition operation is called a full-subtractor.

- Similar to an adder circuit, a full-subtractor combinational circuit can be developed by using two half-subtractors.

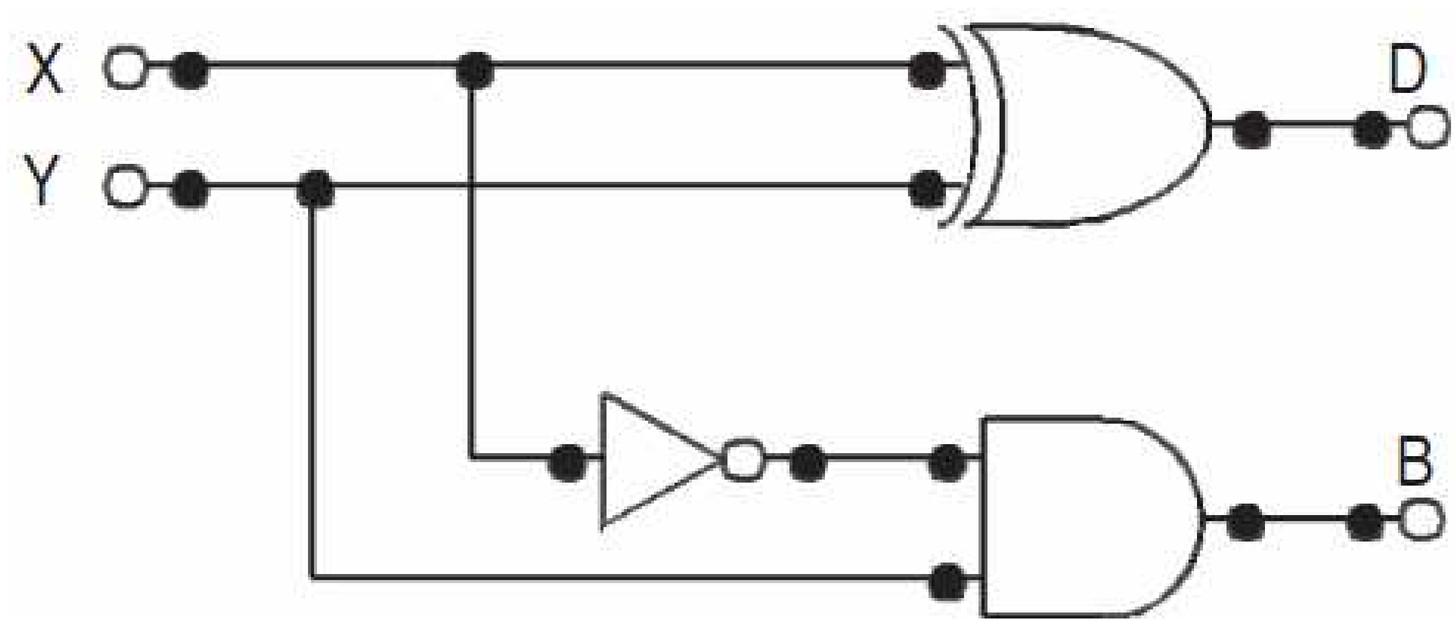
Design of Half-subtractors

A half-subtractor has two inputs and two outputs. Let the input variables minuend and subtrahend be designated as X and Y respectively, and output functions be designated as D for difference and B for borrow. The truth table of the functions is as follows.

Input variables		*Output variables*	
X	*Y*	*D*	*B*
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

The Boolean expressions of the outputs D and B functions can be written as

$D = X'Y + XY'$ and
$B = X'Y$.



Design of Full-subtractors

Input variables			Outputs	
X	Y	Z	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

- A combinational circuit of full-subtractor performs the operation of subtraction of three bits—the minuend, subtrahend, and borrow generated from the subtraction operation of previous significant digits and produces the outputs difference and borrow.

- Let us designate the input variables minuend as X, subtrahend as Y, and previous borrow as Z, and outputs difference as D and borrow as B. Eight different input combinations are possible for three input variables. The truth table is shown

	Y'Z'	Y'Z	YZ	YZ'
X'		1		1
X	1		1	

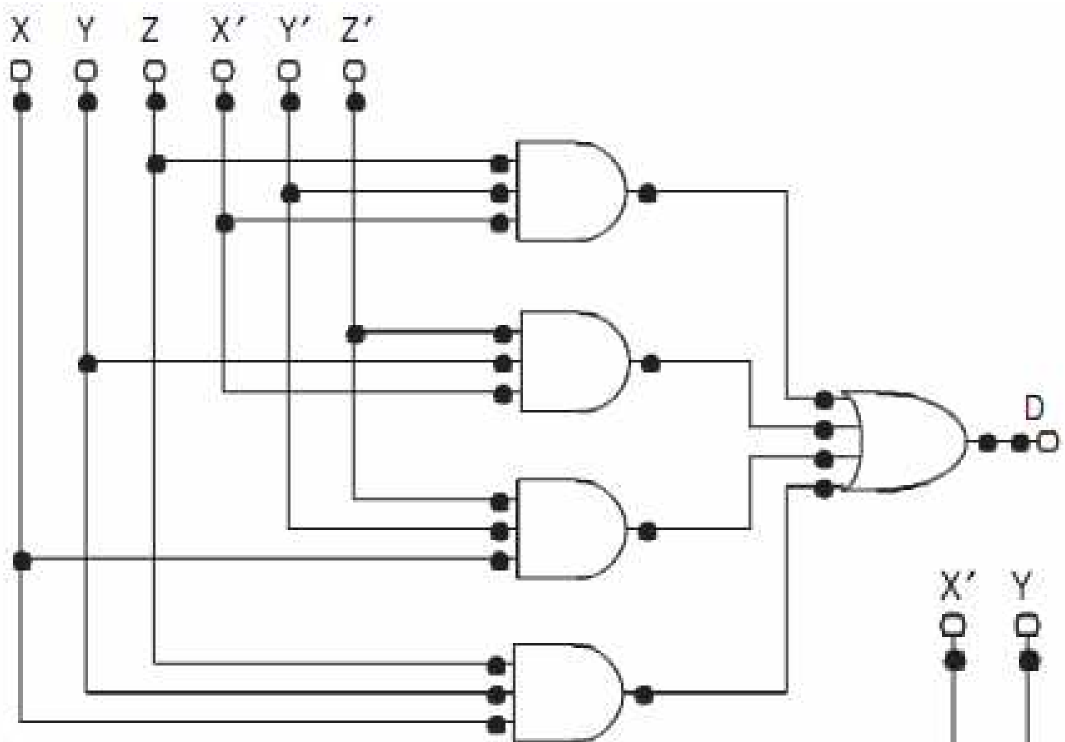
Map for function D.

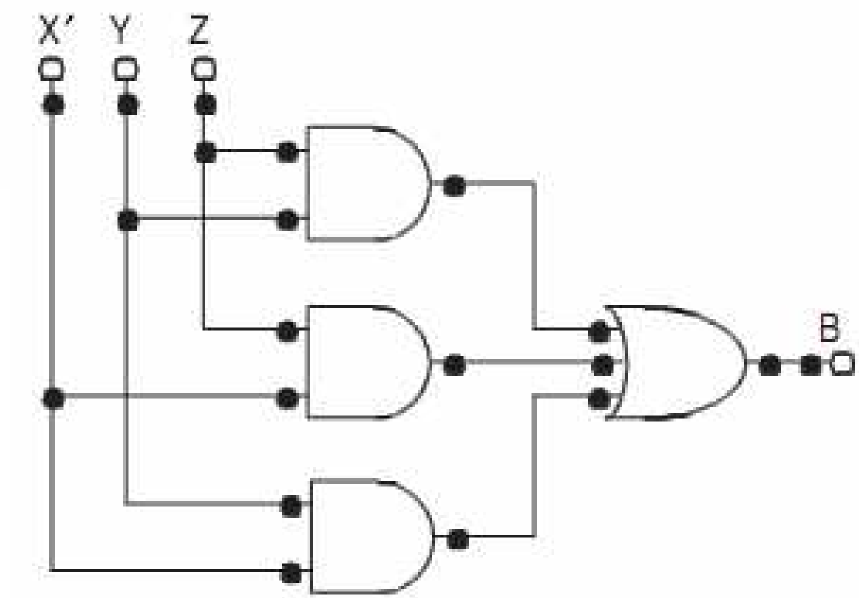
	Y'Z'	Y'Z	YZ	YZ'
X'		1	1	1
X			1	

Map for function B.

The simplified Boolean expressions of the outputs are

S = X'Y'Z + X'YZ' + XY'Z' + XYZ and
B = X'Z + X'Y + YZ.


X
Y
Z
X′
Y′
Z′
D


X′
Y
Z
B

$= X'Y'Z + X'YZ' + XY'Z' + XYZ$

$= X'(Y'Z + YZ') + X(Y'Z' + YZ)$

$= X'(Y \oplus Z) + X(Y \oplus Z)'$

$= X \oplus Y \oplus Z$

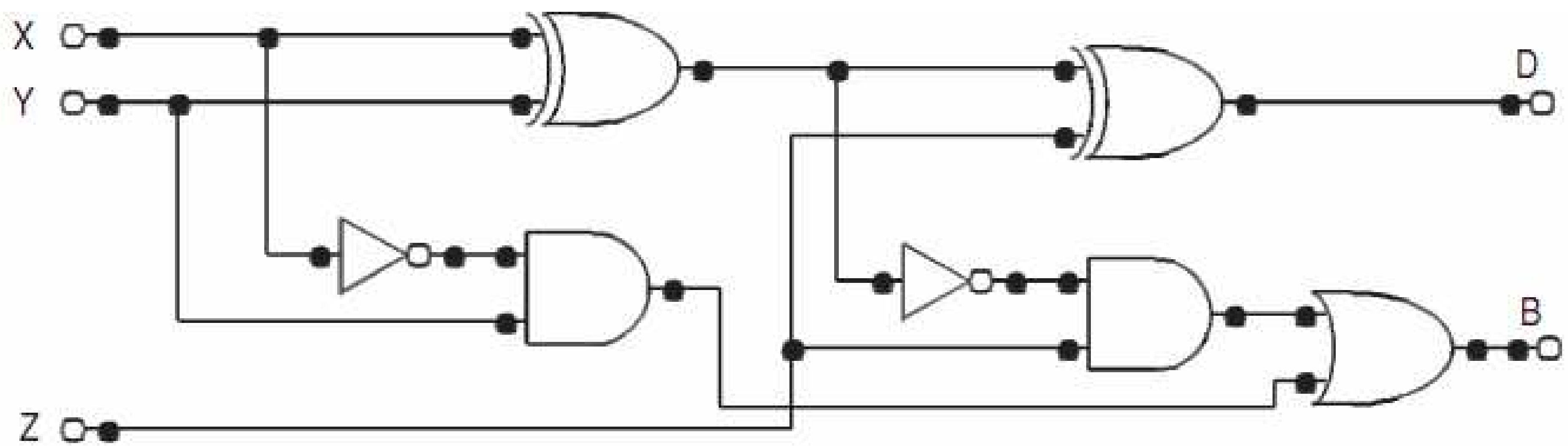
$B = X'Z + X'Y + YZ = X'Y + Z(X' + Y)$

$= X'Y + Z(X'Y + X'Y' + XY + X'Y)$

$= X'Y + Z(X'Y + X'Y' + XY)$

$= X'Y + X'YZ + Z(X'Y' + XY)$

$= X'Y + Z(X \oplus Y)'$



CODE CONVERSION

Binary-to-gray Converter

- The bit combinations 4-bit binary code and its equivalent bit combinations of gray code are listed in the table.

- The four bits of binary numbers are designated as A, B, C, and D, and gray code bits are designated as W, X, Y, and Z.

- For transformation of binary numbers to gray, A, B, C, and D are considered as inputs and W, X, Y, and Z are considered as outputs.

Binary				*Gray*			
A	*B*	*C*	*D*	*W*	*X*	*Y*	*Z*
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

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

Karnaugh map for W.

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

Karnaugh map for X.

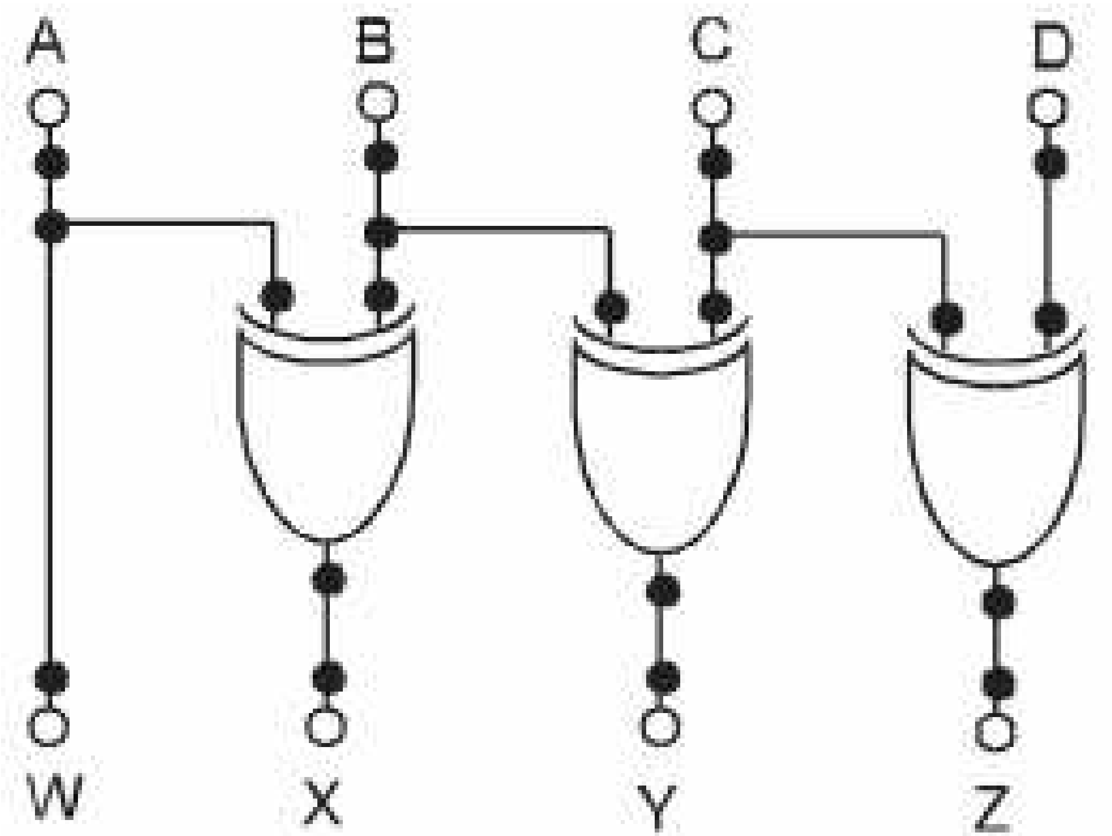
	C′D′	C′D	CD	CD′
A′B′			1	1
A′B	1	1		
AB	1	1		
AB′			1	1

Karnaugh map for Y.

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

Karnaugh map for Z.

$W = A, \quad X = A'B + AB' = A \oplus B,$
$Y = BC' + B'C = B \oplus C, \quad \text{and} \quad Z = C'D + CD' = C \oplus D.$



Gray-to-binary Converter

	Y'Z'	Y'Z	YZ	YZ'
W'X'				
W'X				
WX	1	1	1	1
W'X'	1	1	1	1

Karnaugh map for A.

	Y'Z'	Y'Z	YZ	YZ'
W'X'				
W'X	1	1	1	1
WX				
WX'	1	1	1	1

Karnaugh map for B.

	Y'Z'	Y'Z	YZ	YZ'
W'X'			1	1
W'X	1	1		
WX			1	1
WX'	1	1		

Karnaugh map for C.

	Y'Z'	Y'Z	YZ	YZ'
W'X'		1		1
W'X	1		1	
WX		1		1
WX'	1		1	

Karnaugh map for D.

The Boolean expressions

$A = W$

$B = W'X + WX' = W \oplus X$

$C = W'X'Y + W'XY' + WXY + WX'Y'$

$= W'(X'Y + XY') + W(XY + X'Y')$

$= W'(X \quad Y) + W(X \quad Y)'$

$= W \quad X \quad Y \quad$ or, $C = B \quad Y$

$D = W'X'Y'Z + W'X'YZ' + W'XY'Z' + W'XYZ + WXY'Z + WXYZ' + WX'Y'Z' + WX'YZ$

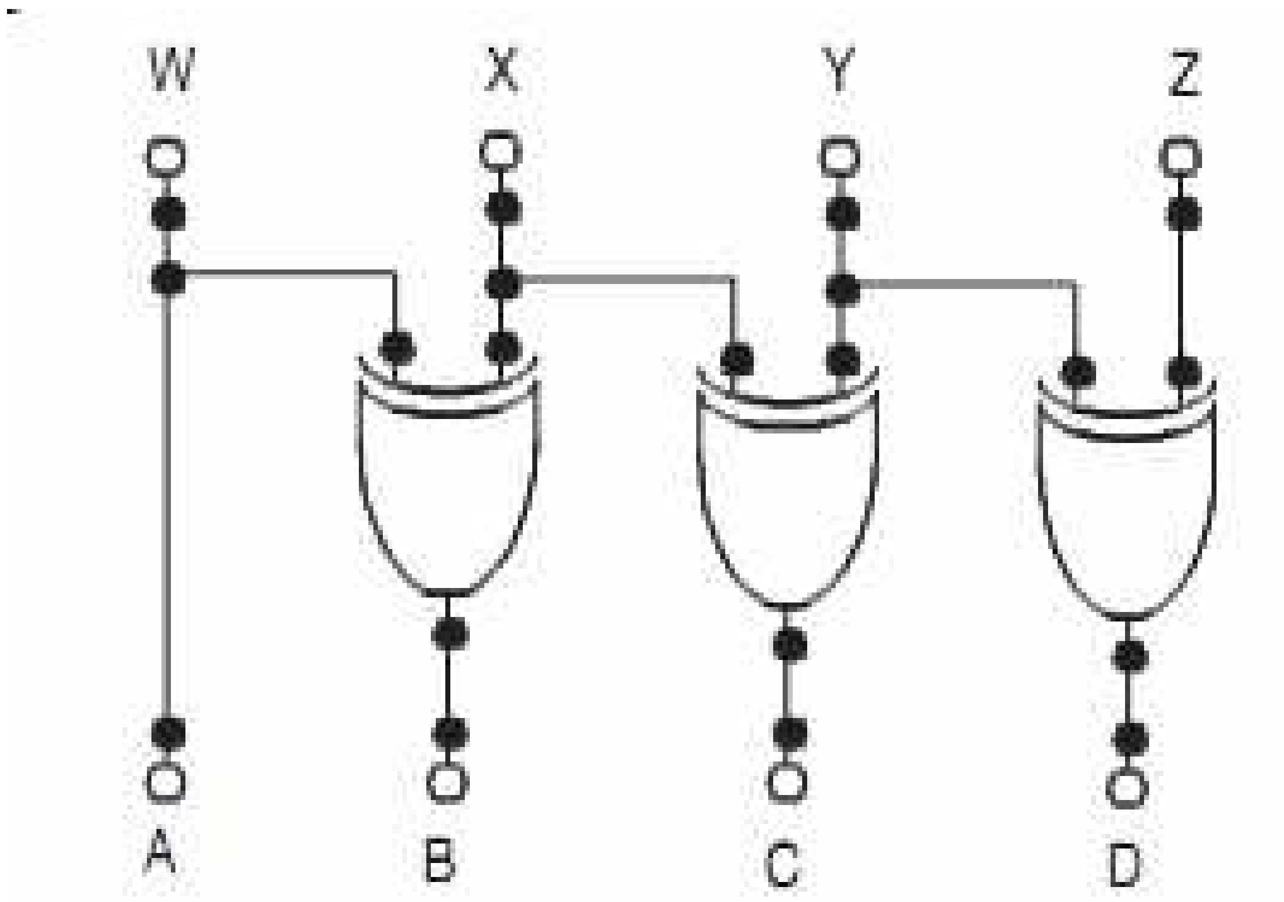
$= W'X'(Y'Z + YZ') + W'X(Y'Z' + YZ) + WX(Y'Z + YZ') + WX'(Y'Z' + YZ)$

$= W'X'(Y \quad Z) + W'X(Y \quad Z)' + WX(Y \quad Z) + WX'(Y \quad Z)'$

$= (W'X + WX')(Y \quad Z)' + (W'X' + WX)\ (YZ)$

$= (W \square X)\ (Y \square Z)' + (W \square X)'\ (Y \quad Z)$

$= W \square X \square Y \square Z \quad$ or, $D = C \square Z$.


W
X
Y
Z
A
B
C
D

BCD-to-excess-3 Code Converter

- The bit combinations of both the BCD (Binary Coded Decimal) and Excess-3 codes represent decimal digits from 0 to 9. Therefore each of the code systems contains four bits and so there must be four input variables and four output variables.

- Figure provides the list of the bit combinations or truth table and equivalent decimal values. The symbols A, B, C, and D are designated as the bits of the BCD system, and W, X, Y, and Z are designated as the bits of the Excess-3 code system.

- It may be noted that though 16 combinations are possible from four bits, both code systems use only 10 combinations. The rest of the bit combinations never occur and are treated as don’t-care conditions.

Decimal Equivalent	BCD code				Excess-3 code			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

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

Karnaugh map for W

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

Karnaugh map for X.

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

Karnaugh map for Y.

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

Ki

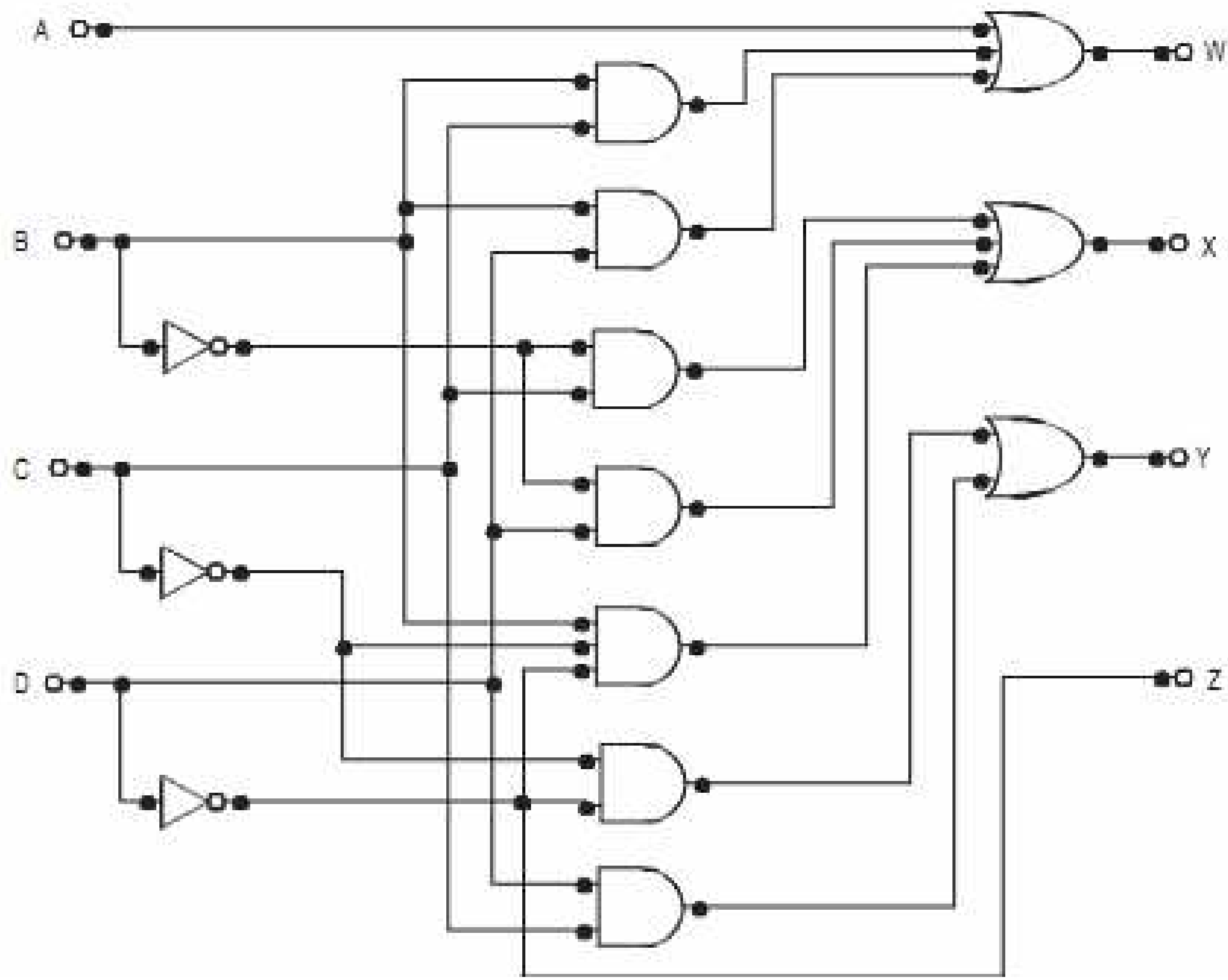
Karnaugh map for Z.

$W = A + BC + BD$

$X = B'C + B'D + BC'D'$

$Y = CD + C'D'$

$Z = D'$



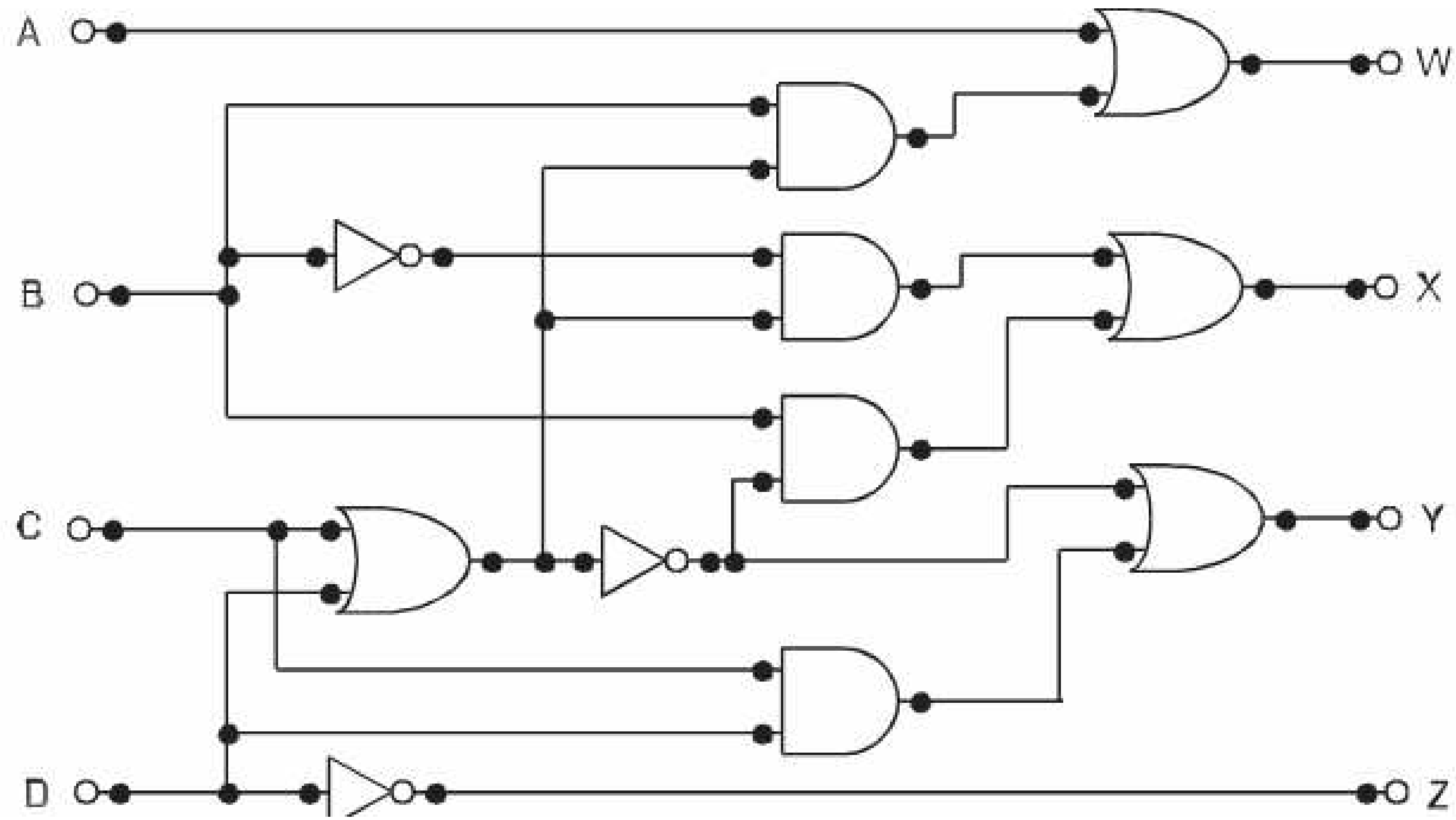
$$W = A + BC + BD = A + B(C + D)$$

$$X = B'C + B'D + BC'D' = B'(C + D) + BC'D'$$

$$= B'(C + D) + B(C + D)'$$

$$Y = CD + C'D' = CD + (C + D)'$$

$$Z = D'$$



Excess-3 - to – BCD Code Converter

?

Assignment