In this topic, We’ll learn

- Adders
- Subtractors
- Code convertors
- Analysis procedure

Introduction

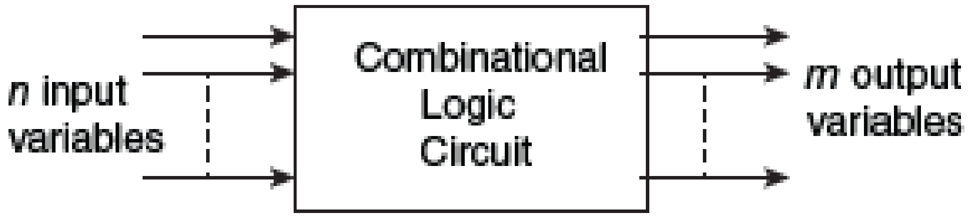
- Logical circuits

1. Combinational circuits
   - Made of logical gates
   - Outputs at any time are directly determined by the present combination of i/ps. (do not depend on previous i/ps)
   - Performs information processing operation specified by a set of Boolean functions.

2. Sequential circuits
   - Made of logical gates and memory elements (binary cells).
   - Outputs at any time are determined by the present combination of i/ps and the state of the memory elements.
   - State of memory elements is a function of previous i/ps.

Combinational circuit

- n i/p variables
  - 2^n combinations of binary i/p values possible.
  - for each i/p combination there is only and one o/p combination possible.
- A combinational circuit can be represented by
  - m Boolean functions, one for each o/p variable.
  - each o/p function is expressed in terms of n i/p variables.



- Each i/p variable to a combinational circuit may have
  - One wire (either normal or complement form)
  - or two wires (both normal and complement form)
- Flip Flops (binary cells for storage) have o/ps in both forms.

Design Procedure

1. The problem is **stated**.
2. The number of available **input** variables and **required output** variables is **determined**.
3. The input and output variables are **assigned** letter **symbols**.
4. The **truth table** that defines the required relationships between inputs and outputs is derived.
5. The **simplified** Boolean **function** for each output is **obtained**.
6. The **logic diagram** is drawn.

Simplified expressions

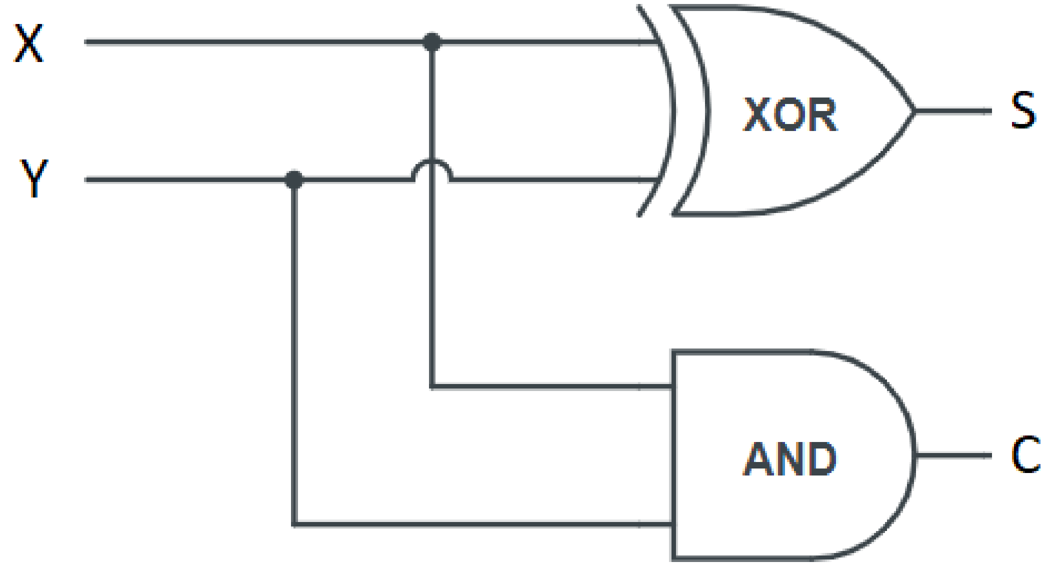
- Application specific.
- Depends on constraints such as
  - Minimum number of gates.
  - Minimum number of i/ps to a gate.
  - Minimum propagation time of the signal through the circuit.
  - Minimum number of interconnections.
  - Limitations on the driving capabilities of the gate.

Adders

- Digital computers perform a variety of information-processing tasks.
  - Includes various arithmetic operations.
- Most basic arithmetic operation
  - Addition of two bits
- Half-adder: Combinational circuit that performs addition of two bits.
- Full-adder: Performs addition of three bits.
  - Two half-adders can be employed to implement a full-adder.

Half-adder

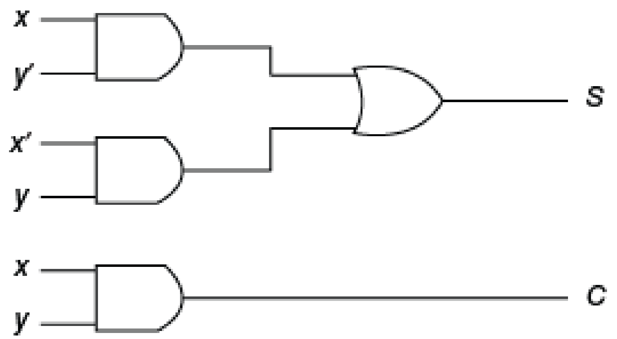
x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



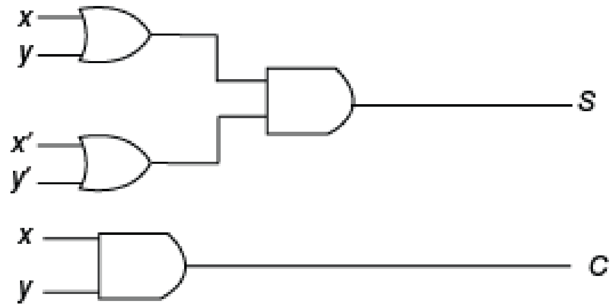
$$S = x'y + xy'$$

$$C = xy$$

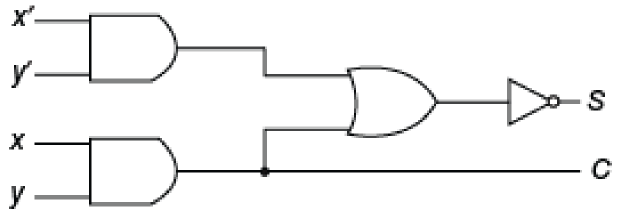
Various implementations of half-adder



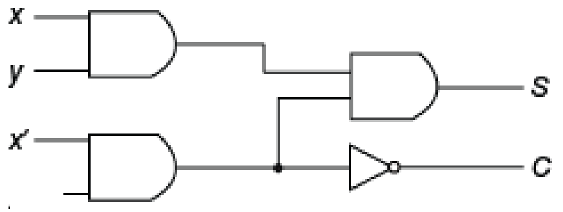
(a) $S = xy' + x'y$
$C = xy$



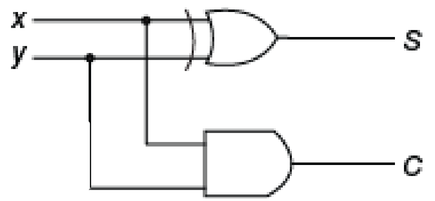
(b) $S = (x + y)(x' + y')$
$C = x\,y$



(c) $S = (C + x'y')'$
$C = xy$

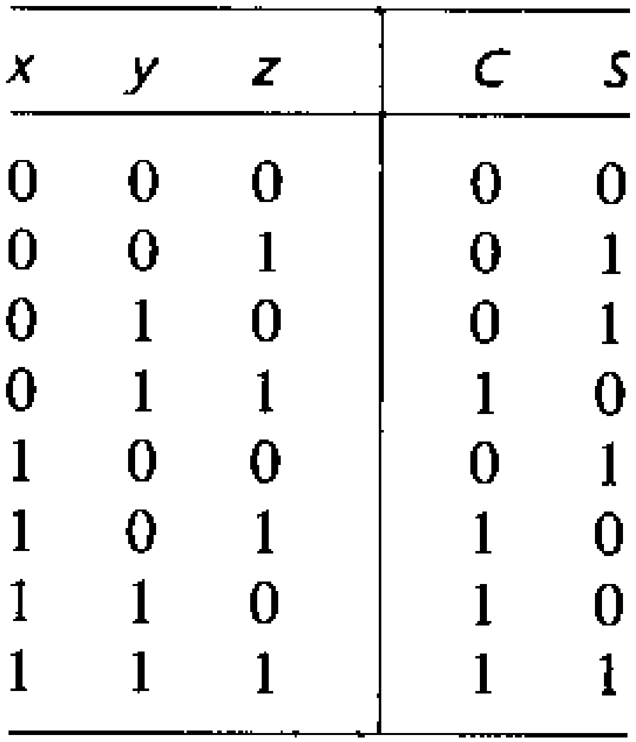


(d) $S = (x + y)(x' + y')$
$C = (x' + y')'$



(e) $S = x \oplus y$
$C = xy$

Full-adder

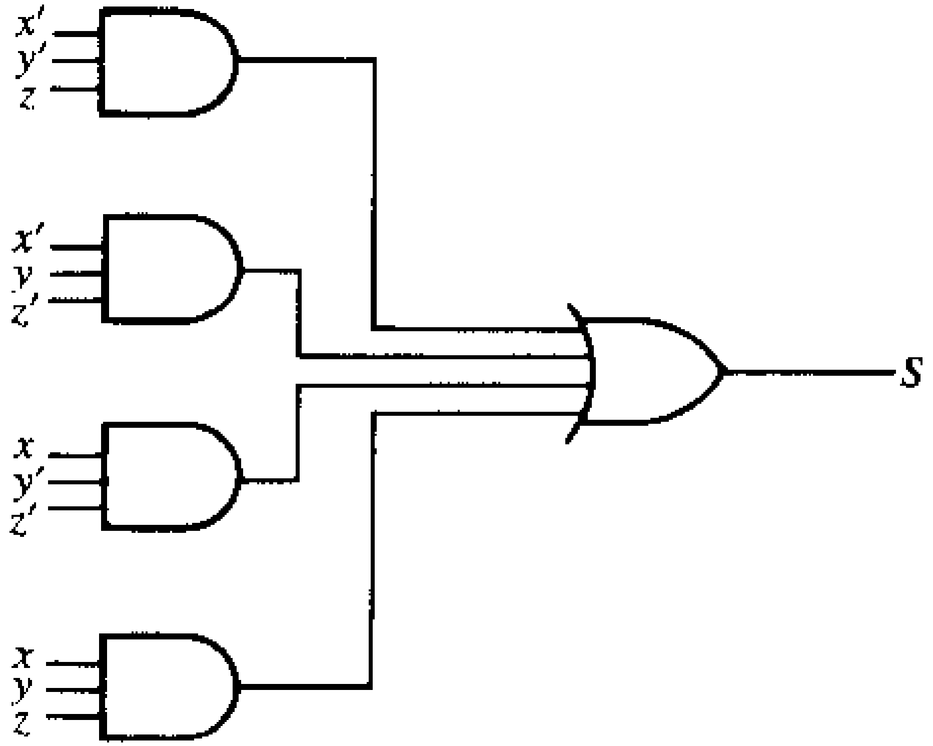


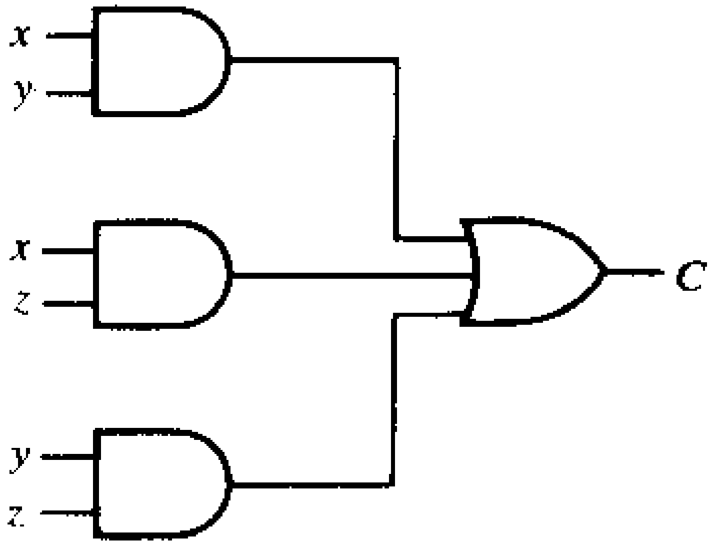
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = x'y'z + x'yz' + xy'z' + xyz$$

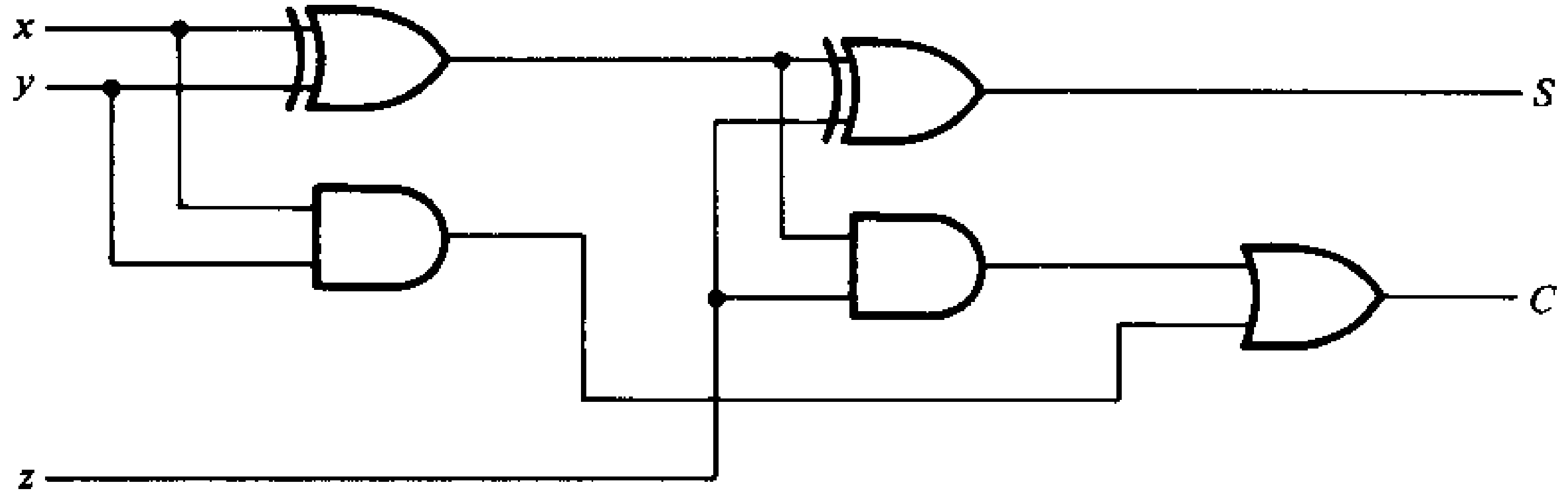
$$C = xy + xz + yz$$

Implementation of Full Adder Circuit





Implementation of Full-adder with 2 HA.



$$S = z \oplus (x \oplus y)$$
$$= z'(xy' + x'y) + z(xy' + x'y)'$$
$$= z'(xy' + x'y) + z(xy + x'y')$$
$$= xy'z' + x'yz' + xyz + x'y'z$$

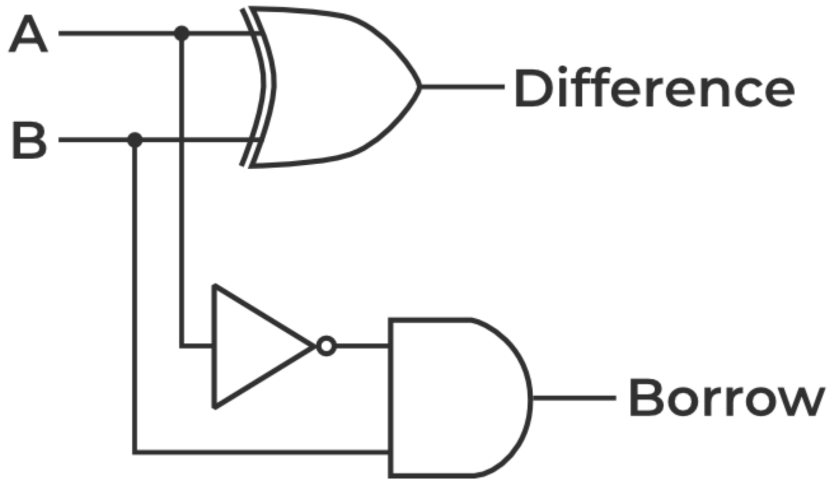
$$C = z(xy' + x'y) + xy = xy'z + x'yz + xy$$

Subtractors

- If the minuend (1st) bit is smaller than the subtrahend (2nd) bit
  - A **1** is borrowed from the next significant position.
  - Conveyed to next pair of higher bits by Borrow bit (B).

Half-Subtractor

x	y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0



$$D = x'y + xy'$$

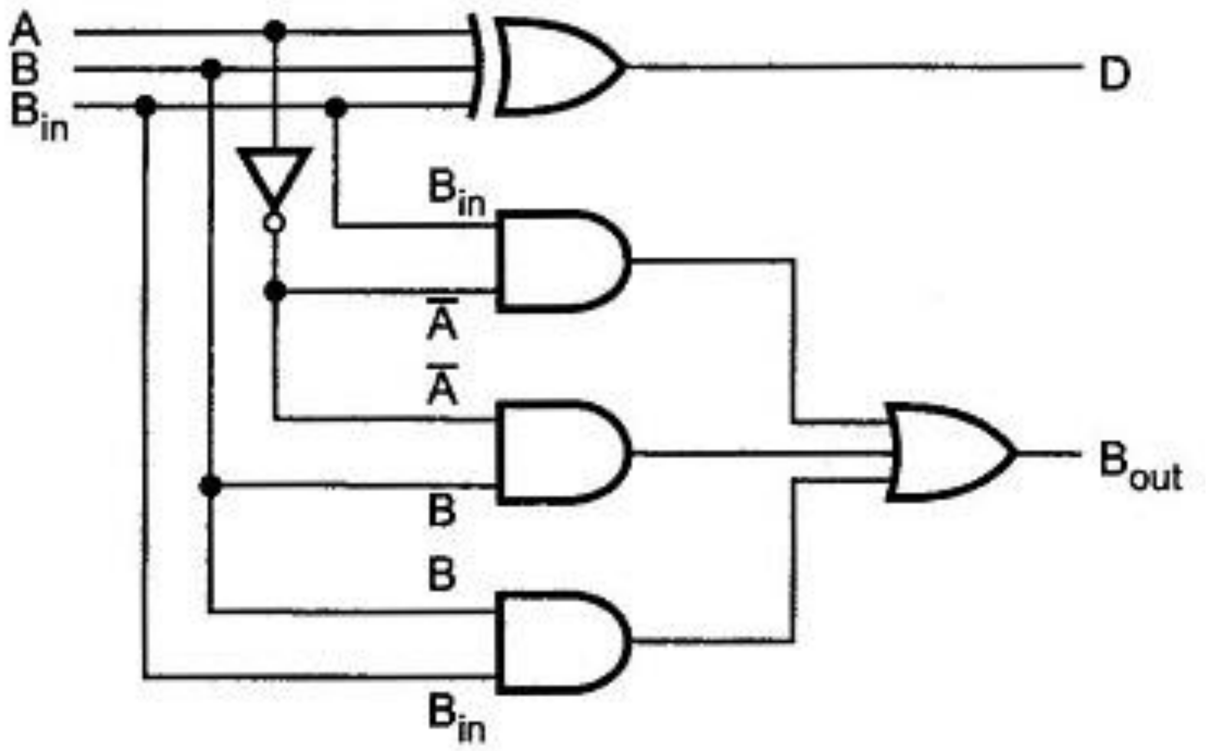
$$B = x'y$$

Full-subtractor

- Performs a subtraction between two bits
  - Takes in to account a 1 may have been borrowed by a lower significant stage.
  - 3 i/ps and 2 o/ps.

Full-adder to Full-subtractor

x	*y*	*z*	*B*	*D*
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



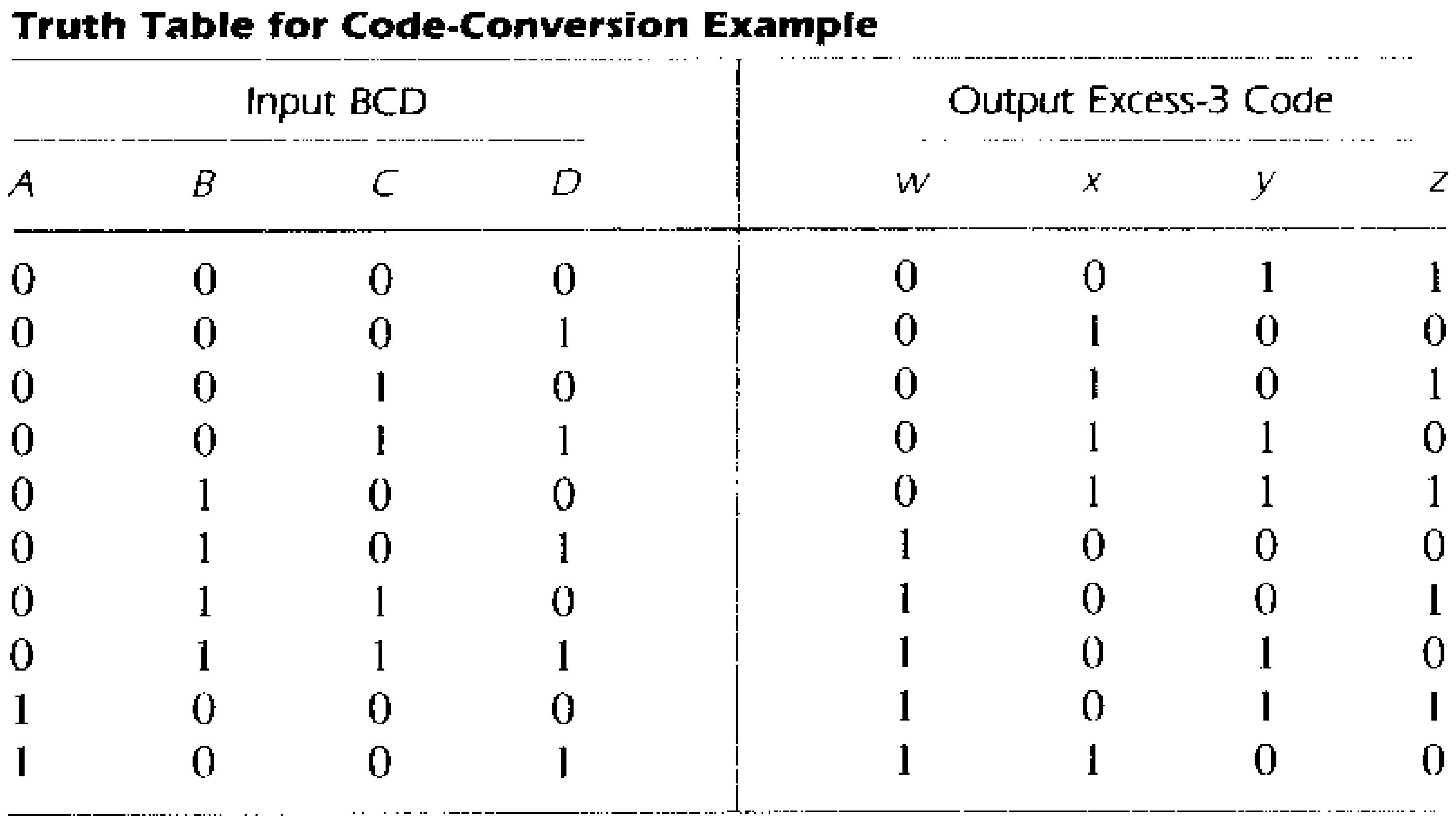
$$D = x'y'z + x'yz' + xy'z' + xyz$$

$$B = x'y + x'z + yz$$

Code Conversion

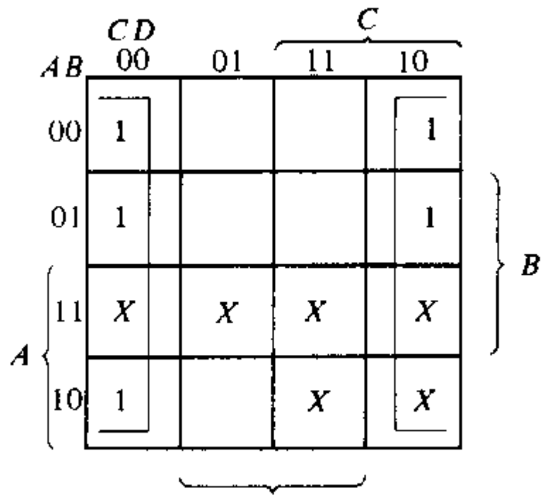
- A large variety of codes are available for the same discrete elements.
  - Different codes are used by different digital systems.
- Code conversion is used when connecting two such digital systems to make them compatible.
- To convert code A to code B
  - i/p is A.
  - o/p generated corresponding bit combination of code B.
  - Combinational circuits using logical gates is used for this transformation.

BCD to excess-3 code

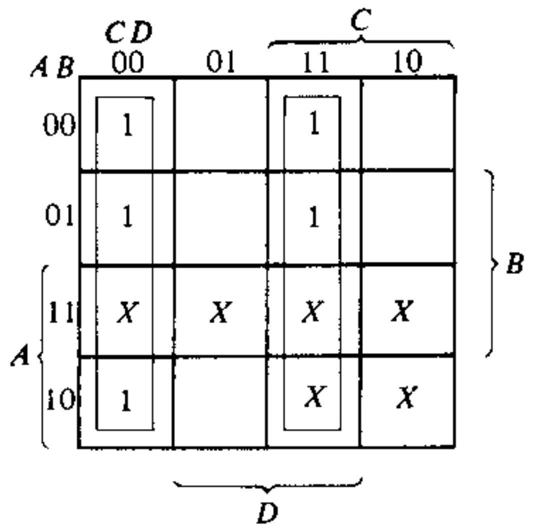


Truth Table for Code-Conversion Example

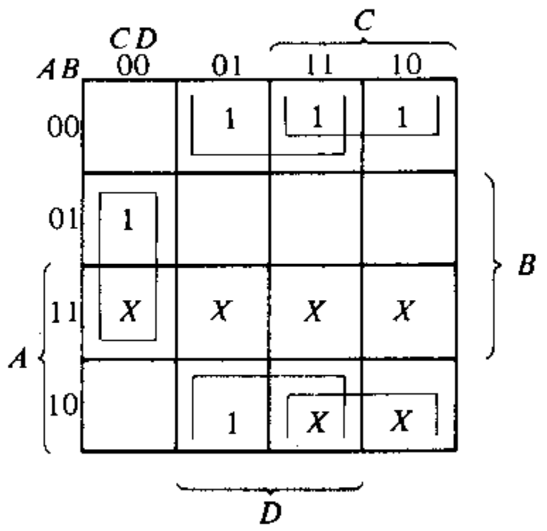
Input BCD				Output Excess-3 Code			
A	*B*	*C*	*D*	*w*	*x*	*y*	*z*
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0



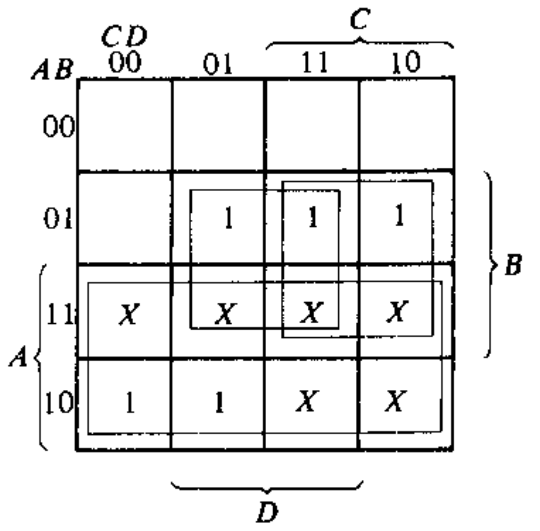
$z = D'$



$y = CD + C'D'$



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



$w = A + BC + BD$

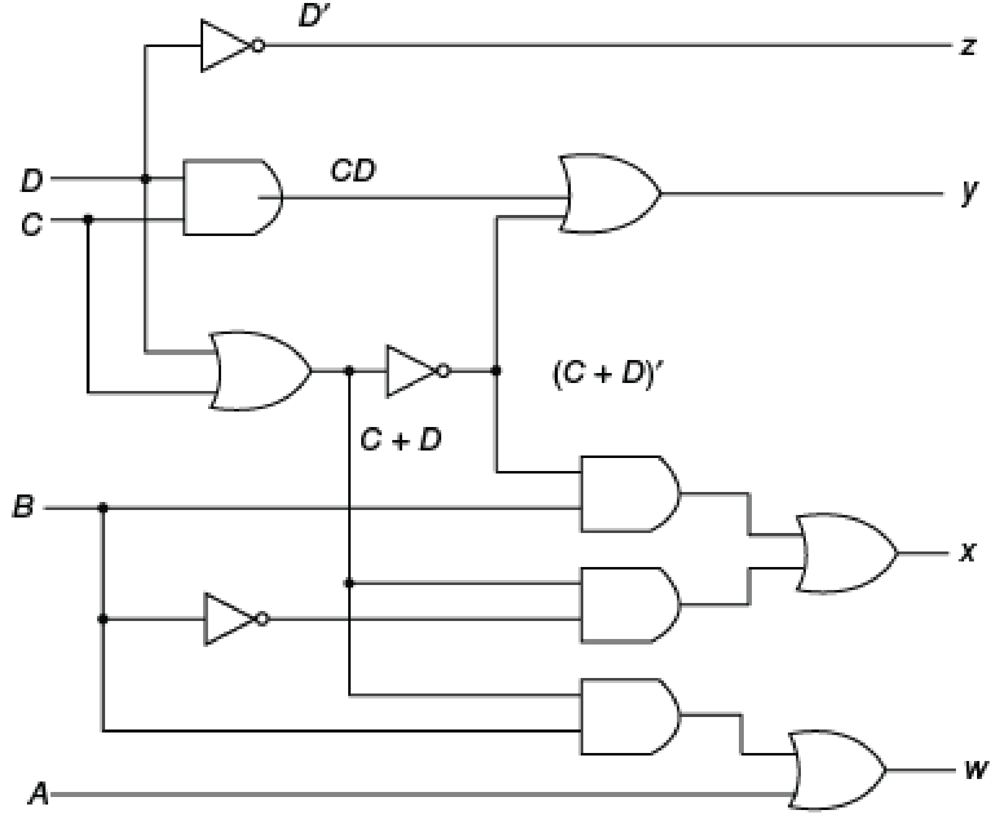
$z = D'$

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

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

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

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

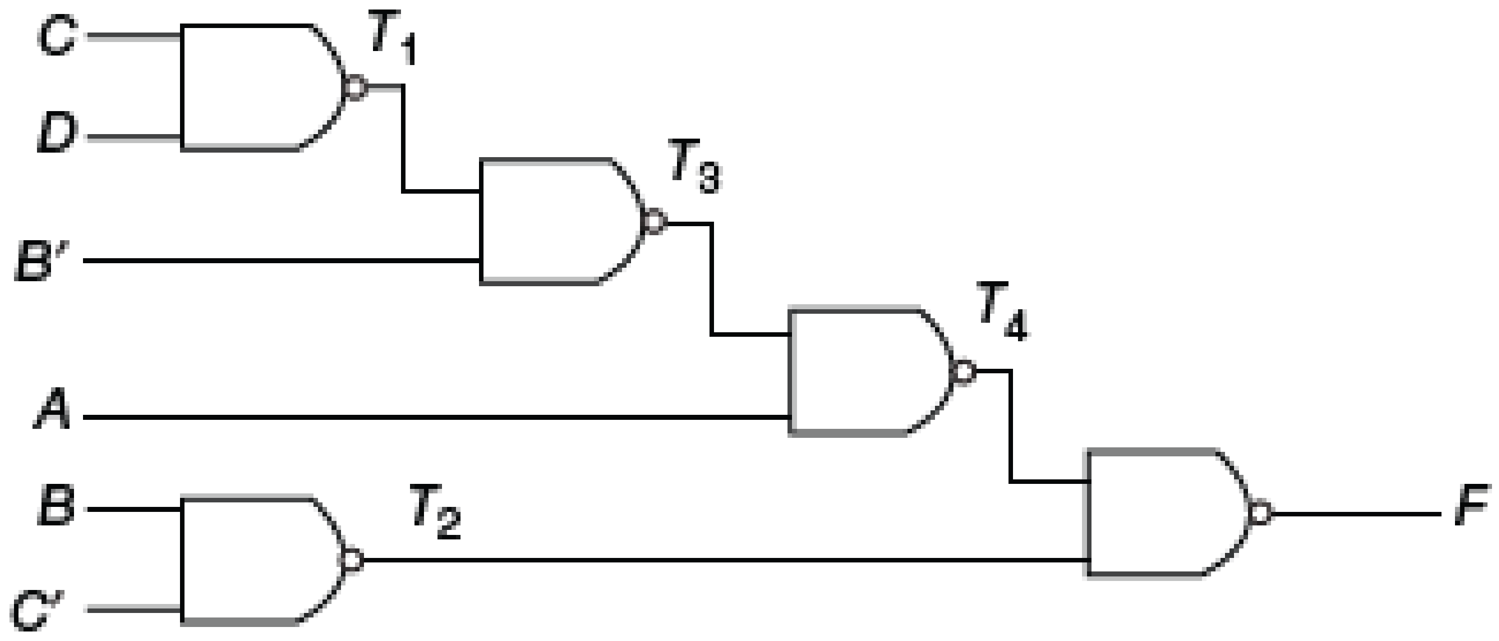


Analysis Procedure

- Reverse problem
  - Starts with a logic diagram
  - Culminates with a Boolean expression or truth table.

- Methods used
  - Derivation of the Boolean Function by Algebraic Manipulation
  - Derivation of the Truth Table
  - Block Diagram Transformation

Derivation of the Boolean Function by Algebraic Manipulation



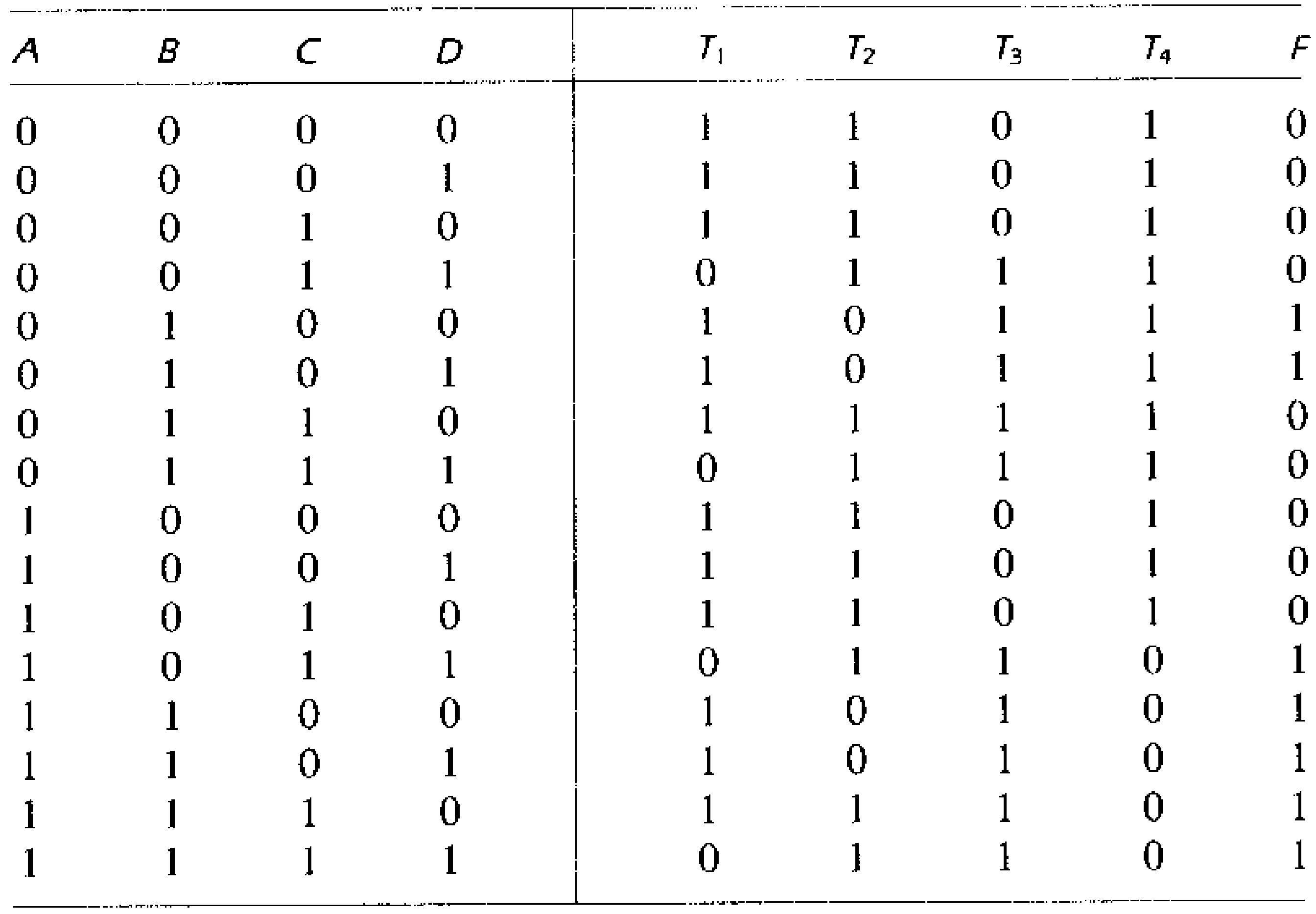
$$T_1 = (CD)' = C' + D'$$

$$T_2 = (BC')' = B' + C$$

$$\begin{aligned} T_3 &= (B'T_1)' = (B'C' + B'D')' \\ &= (B + C)(B + D) = B + CD \\ T_4 &= (AT_3)' = [A(B + CD)]' \\ F &= (T_2T_4)' = \{(BC')'[A(B + CD)]'\}' \\ &= BC' + A(B + CD) \end{aligned}$$

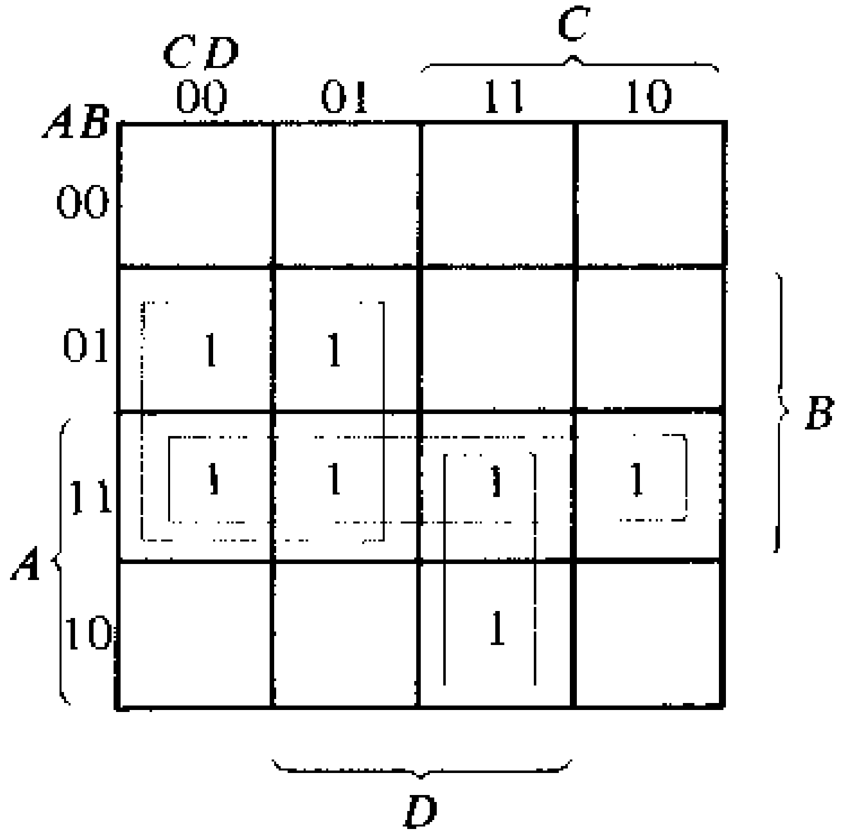
Derivation of the Truth Table

TABLE 4-3
Truth Table for the Circuit of Figure 4-14

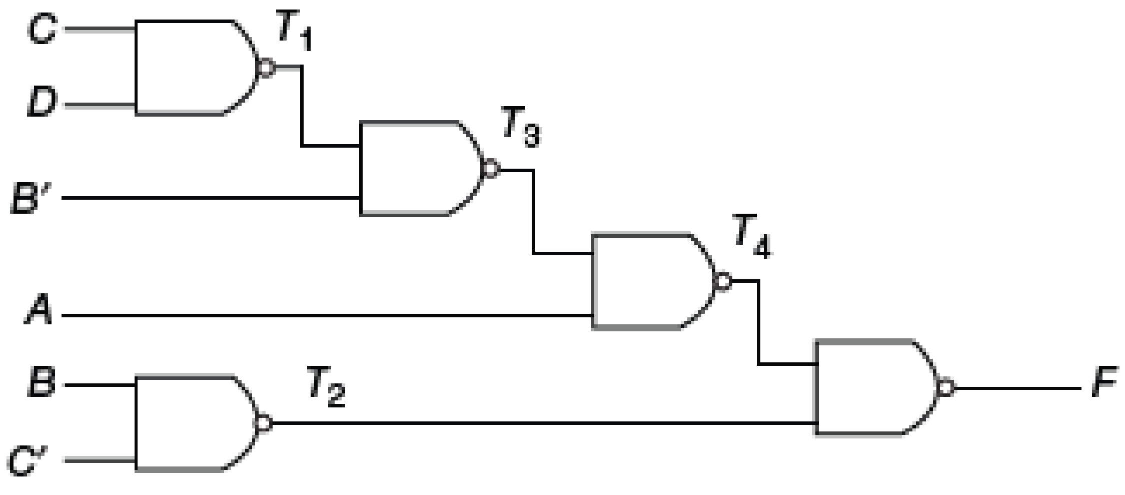


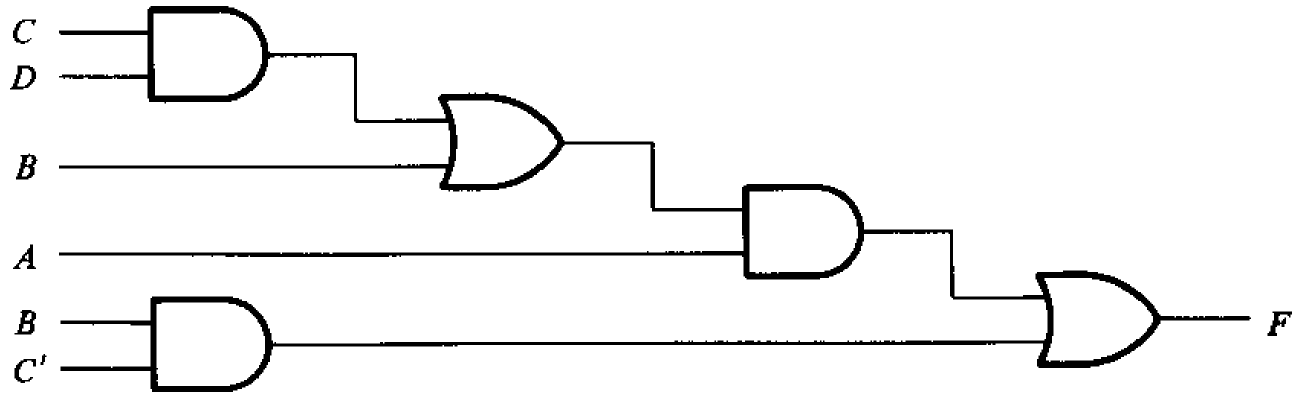
A	B	C	D	T_1	T_2	T_3	T_4	F
0	0	0	0	1	1	0	1	0
0	0	0	1	1	1	0	1	0
0	0	1	0	1	1	0	1	0
0	0	1	1	0	1	1	1	0
0	1	0	0	1	0	1	1	1
0	1	0	1	1	0	1	1	1
0	1	1	0	1	1	1	1	0
0	1	1	1	0	1	1	1	0
1	0	0	0	1	1	0	1	0
1	0	0	1	1	1	0	1	0
1	0	1	0	1	1	0	1	0
1	0	1	1	0	1	1	0	1
1	1	0	0	1	0	1	0	1
1	1	0	1	1	0	1	0	1
1	1	1	0	1	1	1	0	1
1	1	1	1	0	1	1	0	1

Block Diagram Transformation



$F = AB + BC' + ACD$





Review

- Solve unsolved problems for the topics covered.