Combinational Logic

Rab Nawaz Khan Jadoon

Lecturer

COMSATS Lahore

Pakistan

Department of Computer Science

DCS

COMSATS Institute of
Information Technology

COMBINATIONAL LOGIC

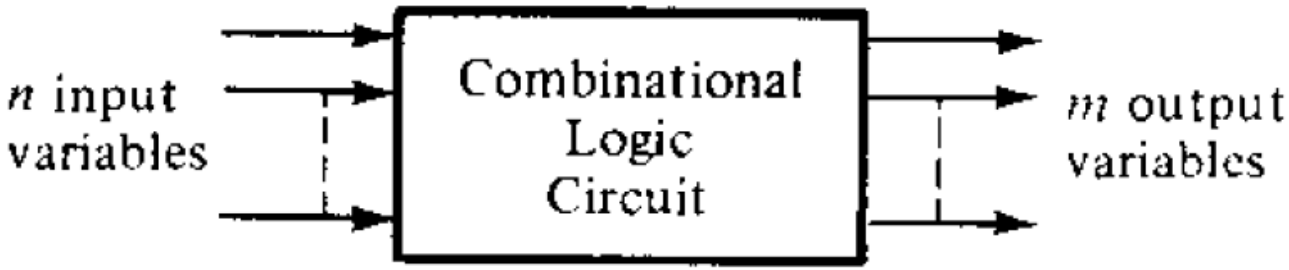
- Introduction
- Design Procedure
- Adders
  - Half Adder
  - Full Adder
  - Parallel Adder
- Subtractors
  - Half Subtractor
  - Full Subtractor
- Code Conversion

Combinational logic

- A combinational circuit consists of logic gates whose outputs at any time are determined directly from the present combination of inputs without regard to previous outputs.
- A combinational circuit performs a specific information processing operation fully specified logically by a set of Boolean functions.
- A Combinational circuit consists of input variables, logic gates, and output variables.

Combinational logic

- The logics gate accept signals from the inputs and generate signals to the outputs.
- This process transforms binary information from the given input data to the required output data.



Combinational logic

- Design Procedures
  - Starts from the verbal outline of the problem and ends in a logic circuit diagram.
  - The procedure involves the following step,
    - The problem is stated.
    - Input and required output variables are determined.
    - Assigned the variables letter symbols.
    - Make the truth table.
    - The simplified Boolean functions for each output is obtained.
    - The logic diagram is drawn.

Combinational logic

- Adders
  - Adders are important in computers and also in other types of digital systems in which numerical data are processed.
  - An understanding of the basic adder operation is fundamental to the study of digital systems.
  - The most basic operation is no doubt is the addition of two binary digits.

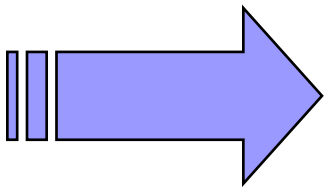
The Half Adder

- Half Adder
  - The combinational circuit that performs the additions of two bit is called Half adder.
  - One that performs the addition of three bits including two digits and one previous carry is a full adder.
  - Two half adders can be employed to form a full adder.

- Half Adder
  - It has two inputs and two outputs.
  - The input variables designates the augends and addend bits; the output variables produces the sum and carry.
  - It is necessary to specify two output variables because the result may consist of two binary digits.
  - A and B are two inputs binary variables while C and S used for carry and Sum to the outputs.

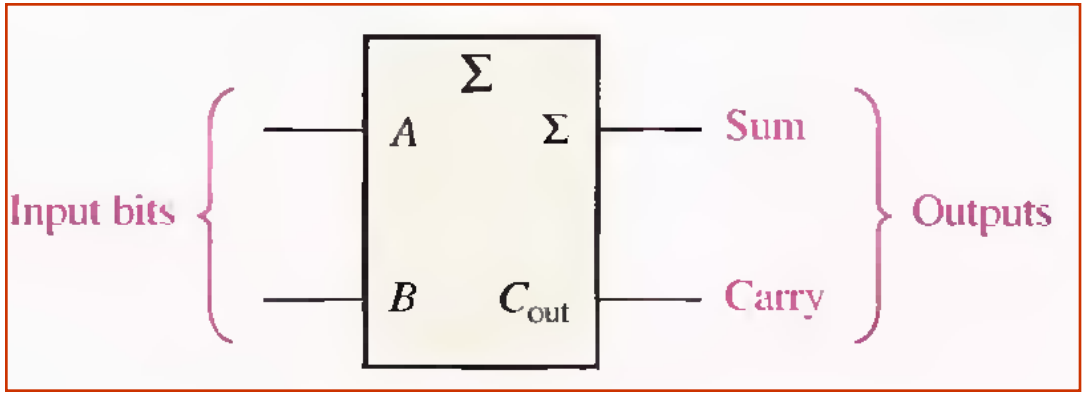
- Half Adder
  - The half-adder accepts two binary digits on its inputs and produces two binary digits on its outputs, a sum bit and a carry bit.
  - The truth table look like this,

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 10$$



A	B	C_{out}	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

- Half Adder
  - A half-adder is represented by the logic symbol in Figure below,



Half Adder

- Half-Adder Logic
  - Notice that the output Carry (C_{out}) is a 1 only when both A and B are 1s; therefore C_{out} can be expressed as the AND of the input variables.
  - $C_{out} = AB$
  - Now observe that the sum output (Σ) is a 1 only if the input variables. A and B, are not equal.
  - The sum can therefore be expressed as the exclusive-OR of the input variables.

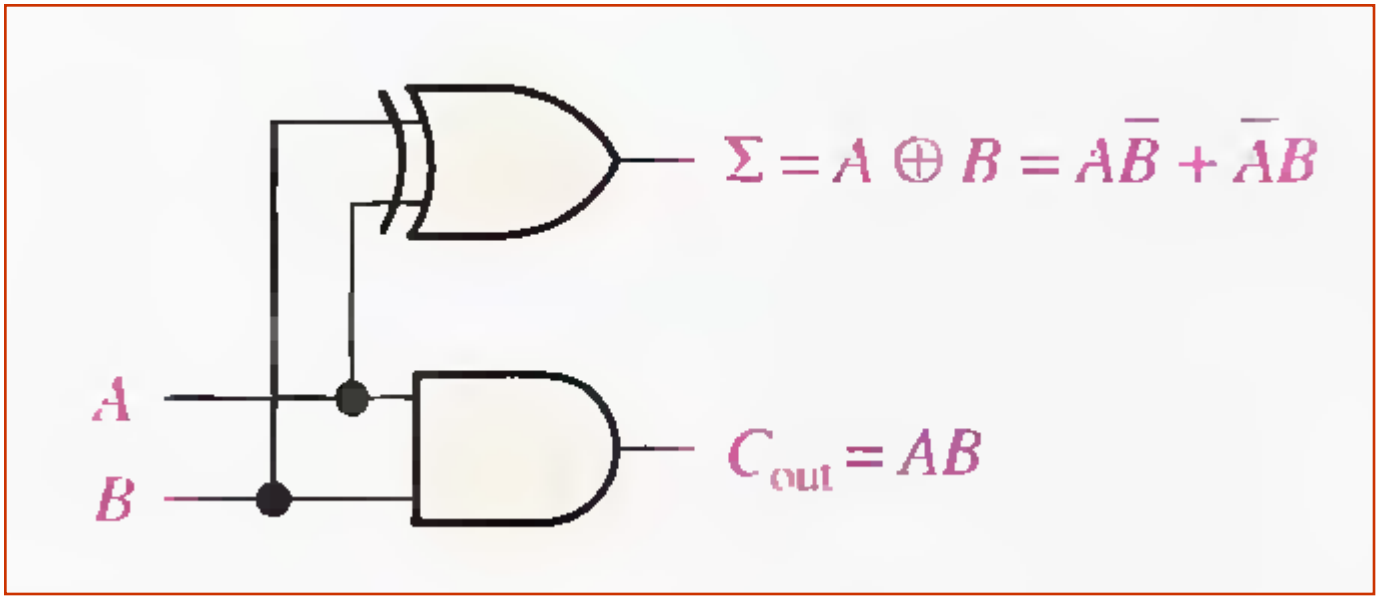
$$\Sigma = A \oplus B$$

Related Example

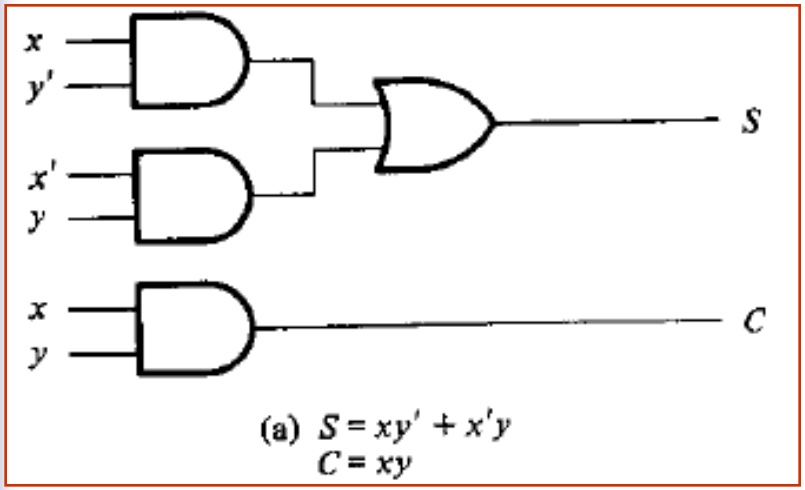
- Half-Adder Logic
  - The logic implementation required for the half adder function can be developed.
  - The output carry is produced with an AND gate with A and B on the inputs.
  - The sum output is generated with an exclusive-OR gate.

Half Adder

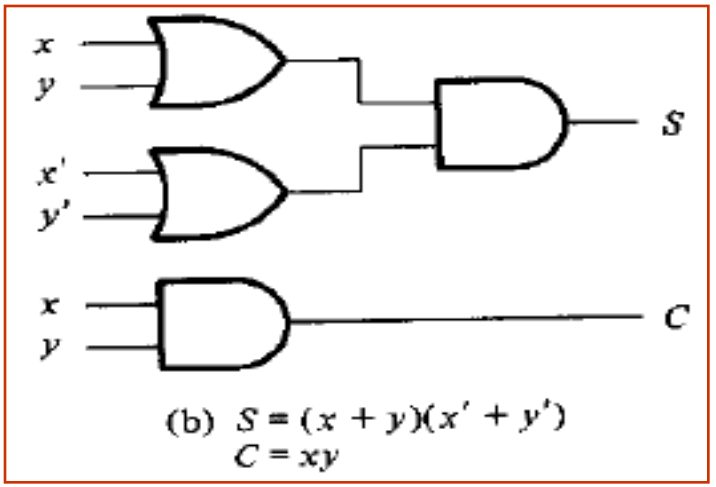
- Half-Adder Logic diagram



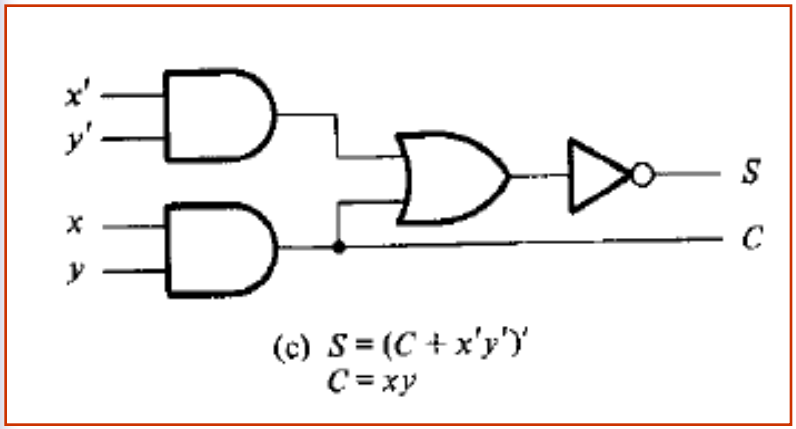
Solutions



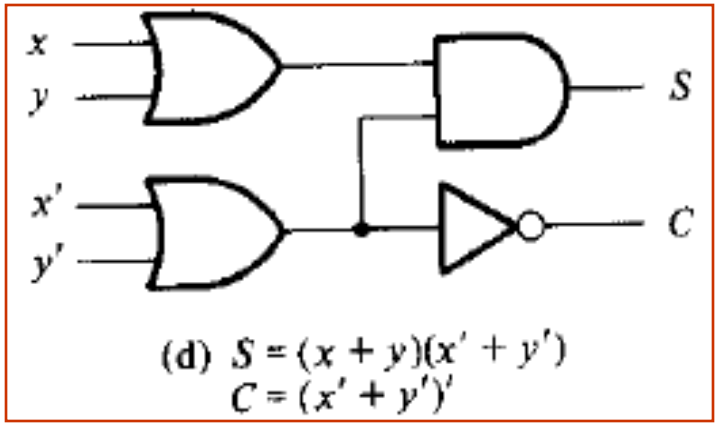
(a) $S = xy' + x'y$
$C = xy$



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



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

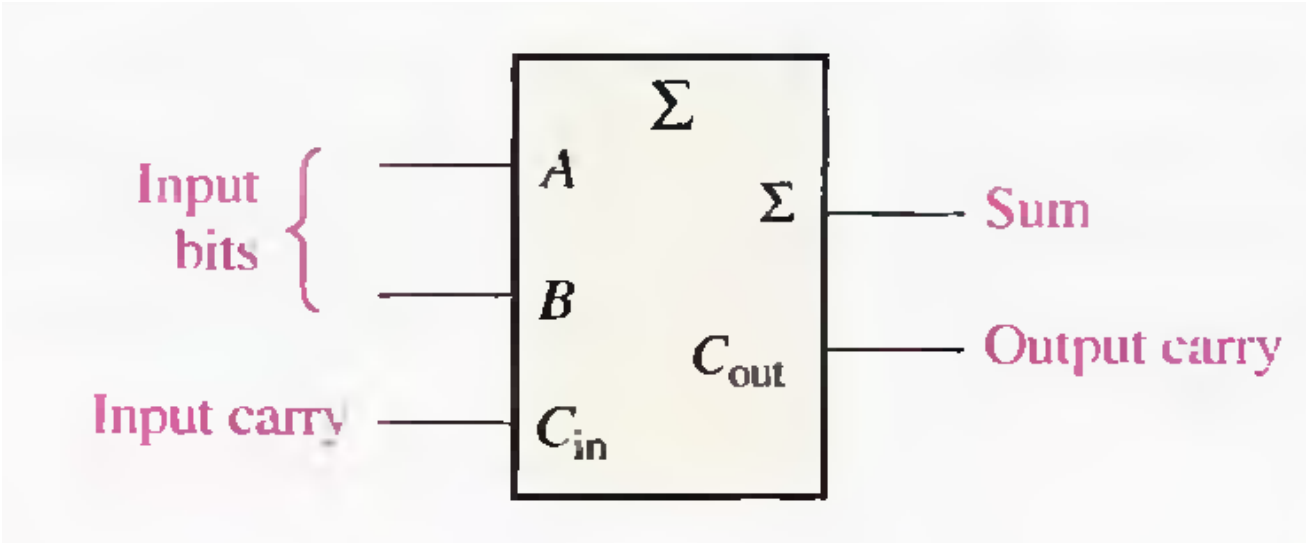


(d) $S = (x + y)(x' + y')$
$C = (x' + y')'$

- Full Adder
  - The second category of adder is the full-adder.
  - The full-adder accepts two input bits and an input carry and generates a sum output and an output carry.
  - The basic difference between a full-adder and a half-adder is that the full-adder accepts an input carry.

Full Adder

- Logical symbol for full adder is,



Full Adder

Truth Table

A	B	C_{in}	C_{out}	Σ
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

C_{in} = input carry, sometime designated as CI
C_{out} = output carry sometimes designated as CO
Σ=sum
A and B = input variables (operands)

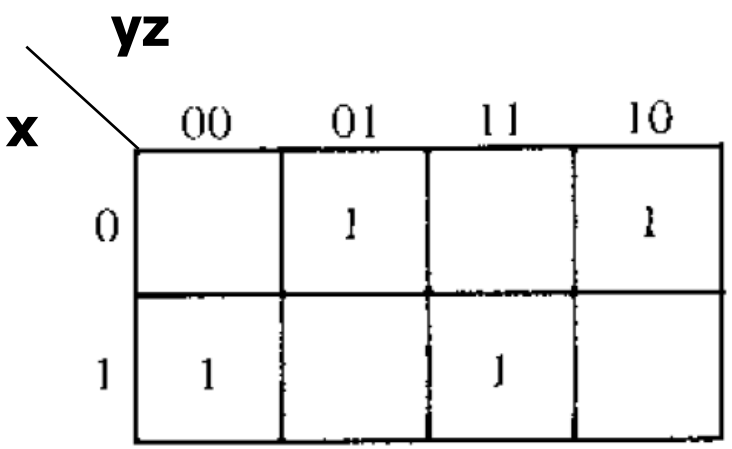
Full Adder

- Full Adder Logic
  - The full-adder must add the two input bits and the input carry.
  - From the half-adder you know that the sum of the input bits A and B is the exclusive-OR of those two variables, A xor B.
  - For the input carry (C_{in}) to be added to the input bits. it must be exclusive-ORed with A xor B, yielding the equation for the sum output of the full-adder.

$$\Sigma = (A \oplus B) \oplus C_{\text{in}}$$

Full Adder

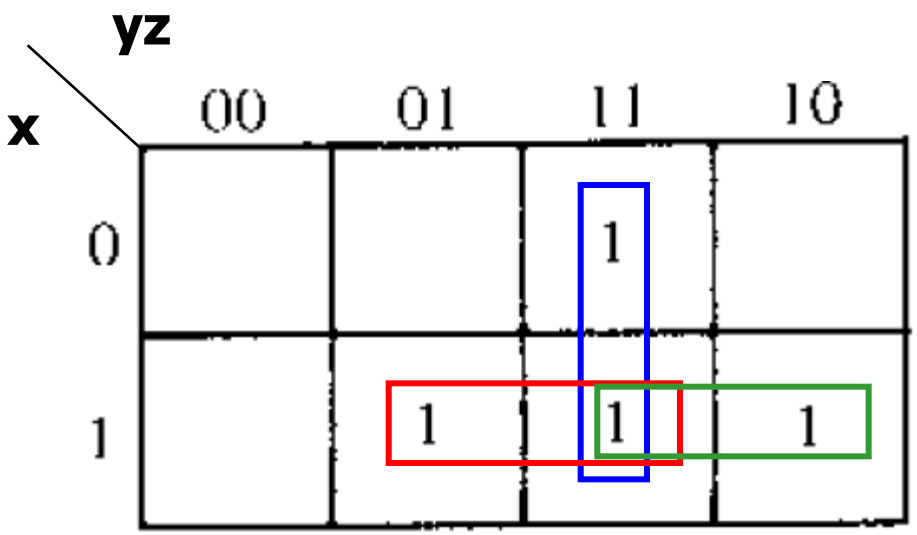
- Map for Full Adder (For Sum Function)



S= x′y′z + x′yz′ + xy′z′ + xyz

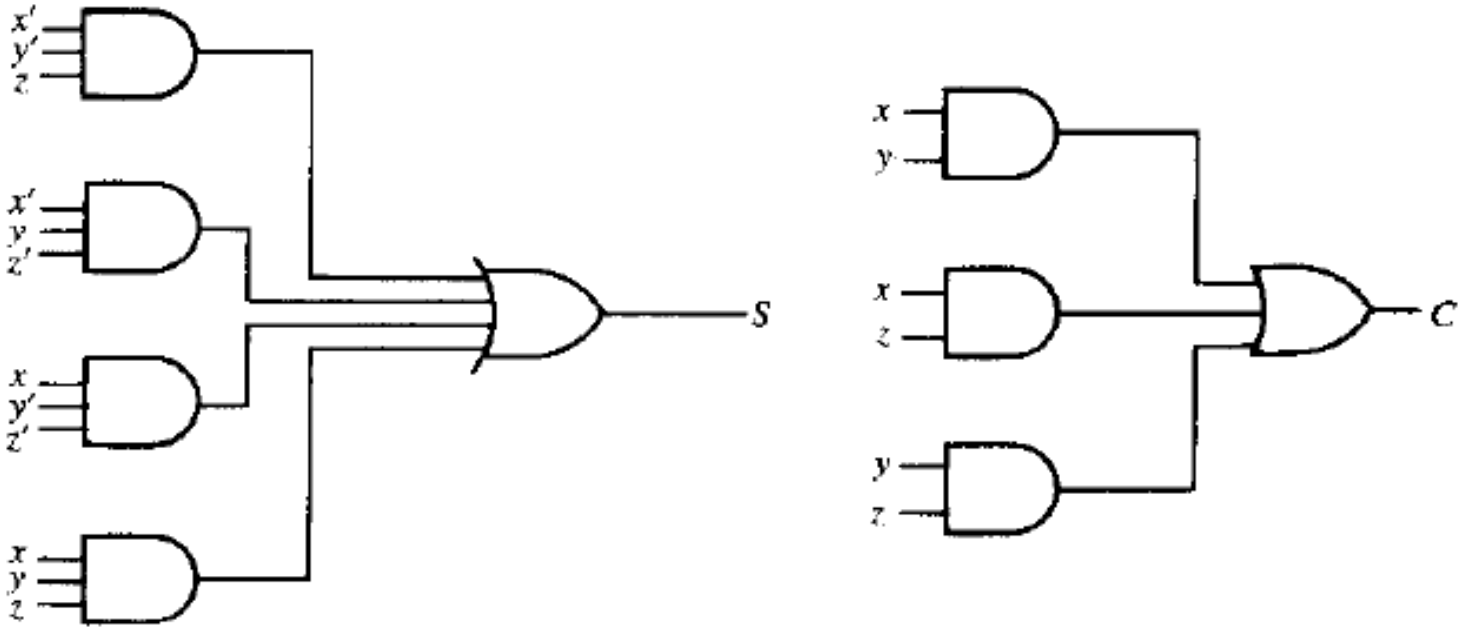
Full Adder

For Carry Simplified Expression



C= xy + xz + yz

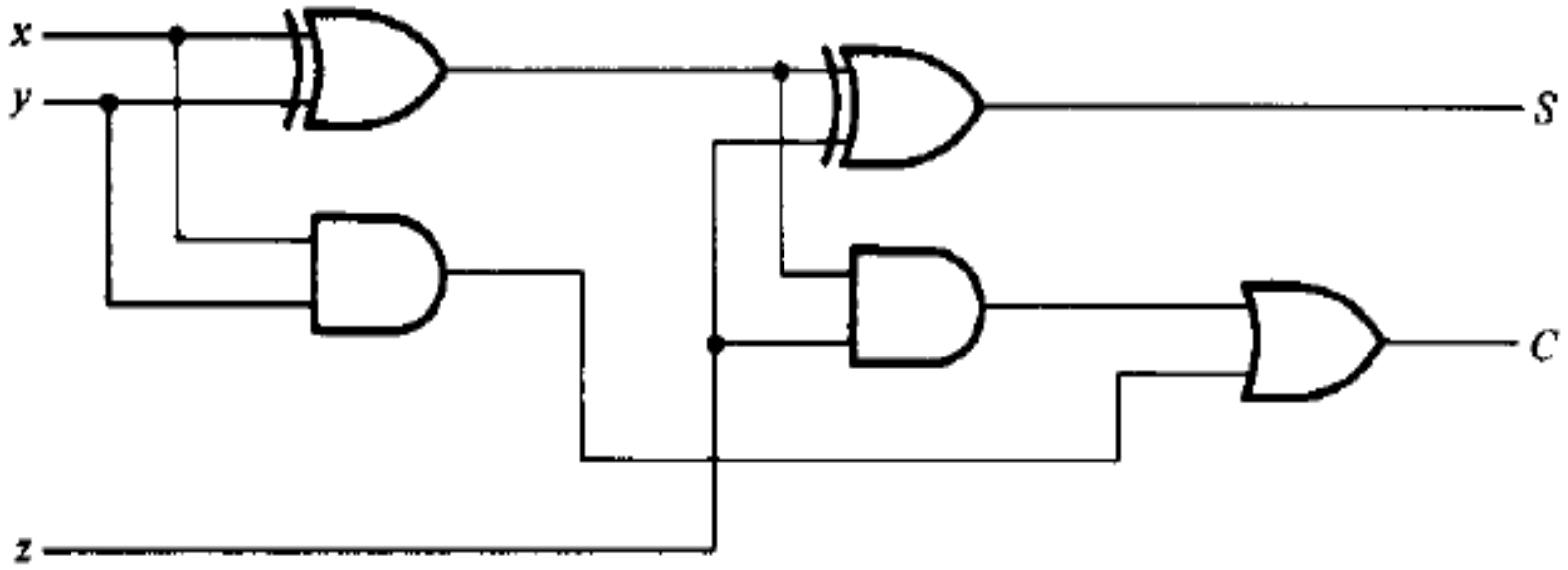
Implementation of Full Adder in SOP



Logic Diagram

Implementation of Full Adder

Implementation of a full adder with two half adders and an OR Gate



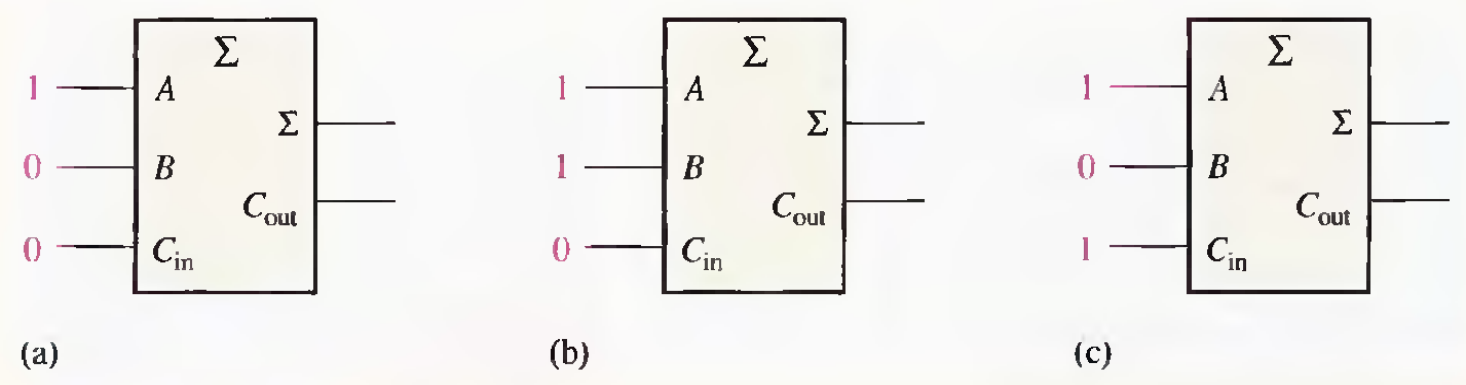
Adders

$$
\begin{aligned}
S &= z \oplus (x \oplus y) \\
&= z'(xy' + x'y) + z(xy' + x'y)' \\
&= z'(xy' + x'y) + z(xy + x'y') \\
&= xy'z' + x'yz' + xyz + x'y'z
\end{aligned}
$$

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

Problem

- For each of the three full-adders in Figure below, determine the outputs for the inputs shown.



Solution

(a) The input bits are $A = 1$, $B = 0$, and $C_{in} = 0$.

$$1 + 0 + 0 = 1 \text{ with no carry}$$

Therefore, $\Sigma = 1$ and $C_{out} = 0$.

(b) The input bits are $A = 1$, $B = 1$, and $C_{in} = 0$.

$$1 + 1 + 0 = 0 \text{ with a carry of 1}$$

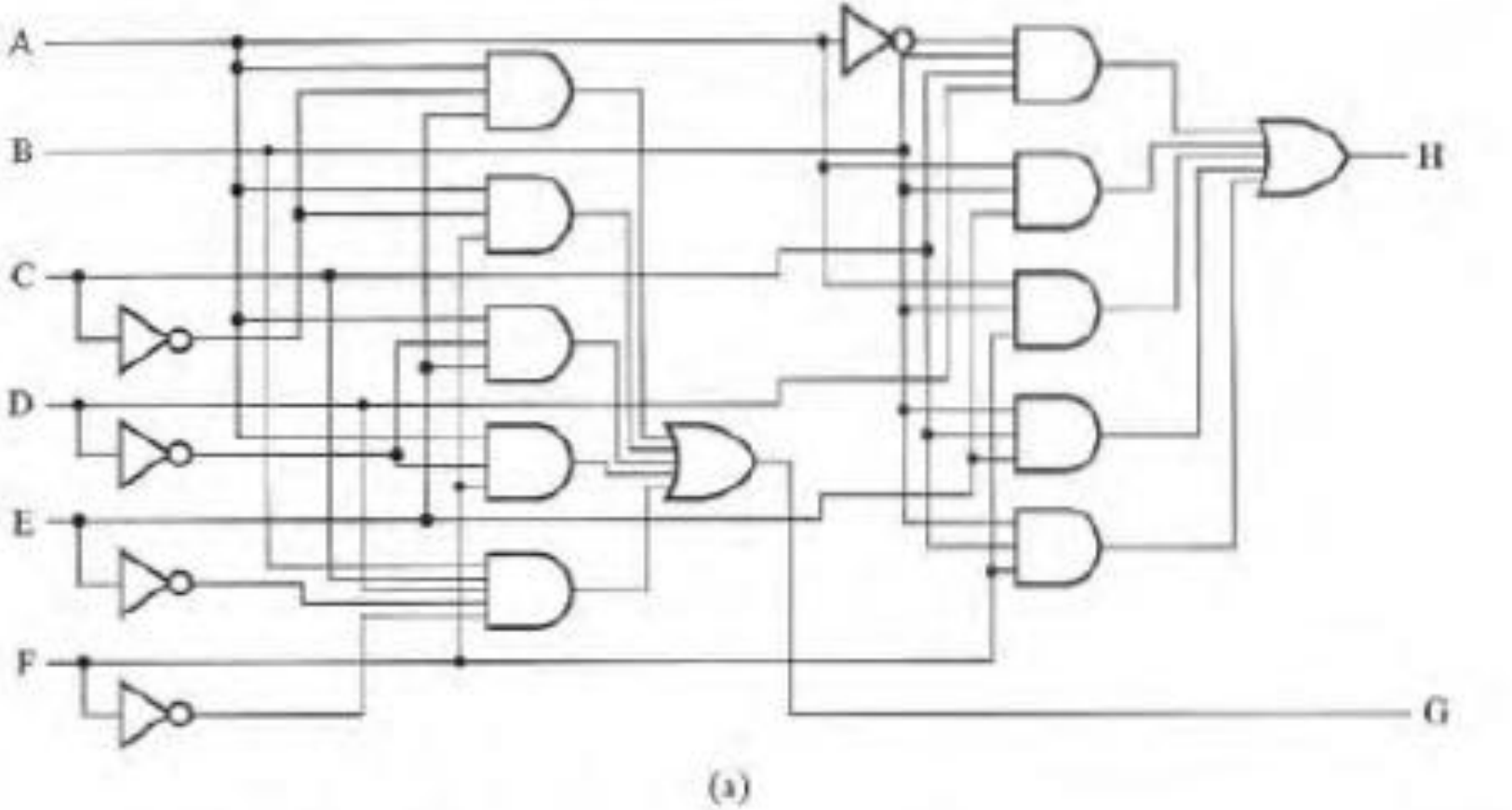
Therefore, $\Sigma = 0$ and $C_{out} = 1$.

(c) The input bits are $A = 1$, $B = 0$, and $C_{in} = 1$.

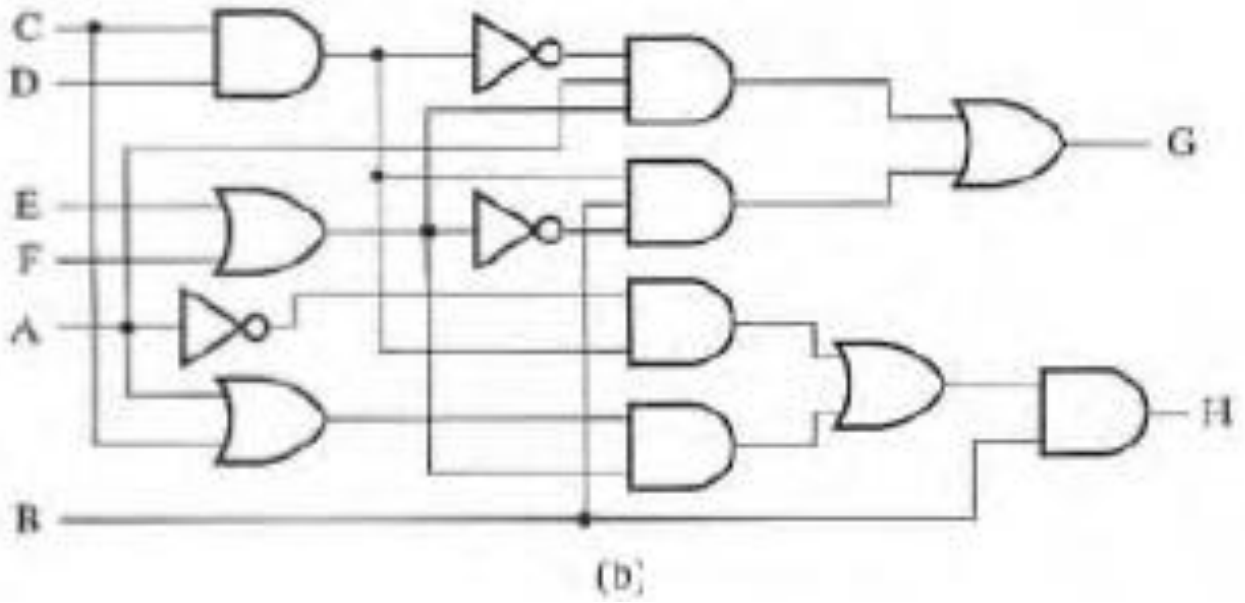
$$1 + 0 + 1 = 0 \text{ with a carry of 1}$$

Therefore, $\Sigma = 0$ and $C_{out} = 1$.

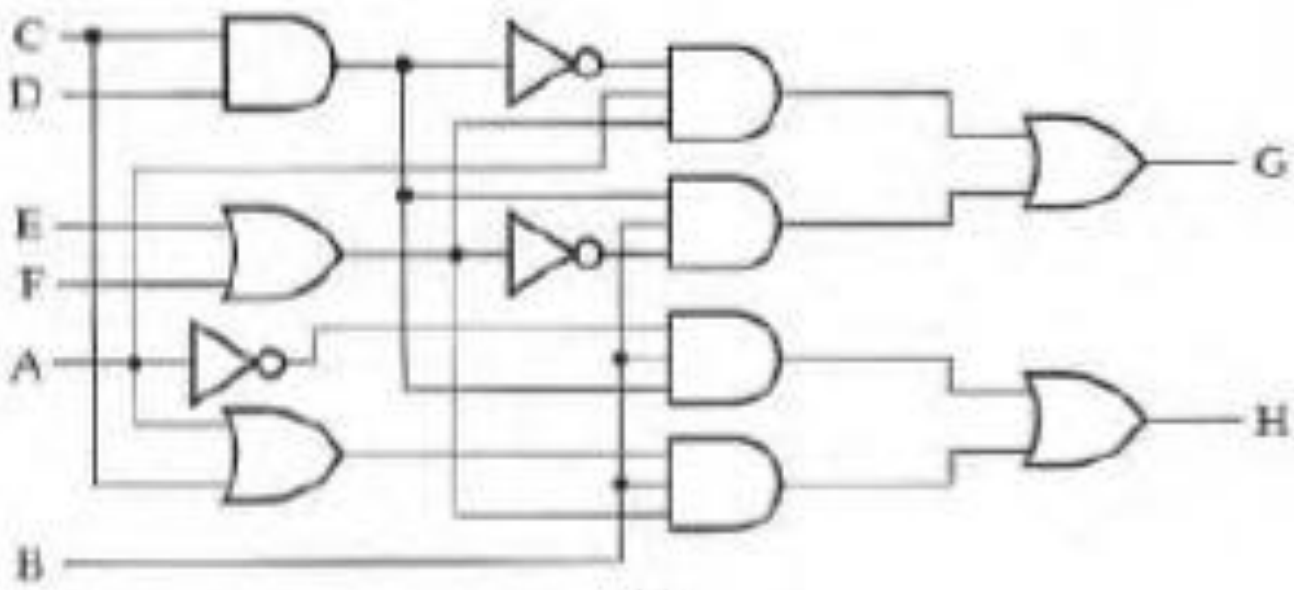
Quiz



(a)



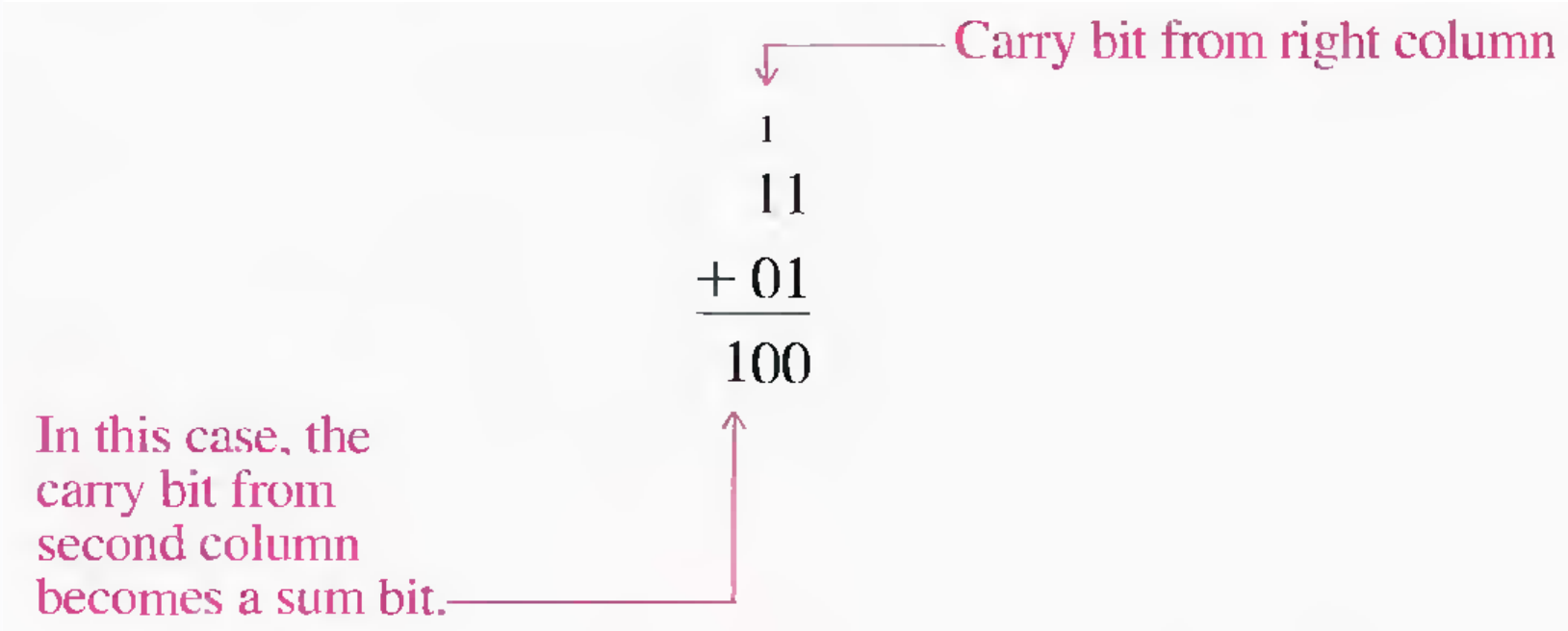
(b)



(c)

Parallel Binary Adders

- Two or more full-adders are connected to form parallel binary adders.
- a single full-adder is capable of adding two 1-bit numbers and an input carry.
- To add binary numbers with more than one bit, you must use additional full-adders.
- When one binary number is added to another, each column generates a sum bit and a 1 or 0 carry bit to the next column to the left.


Carry bit from right column
1
11
+01
100
In this case, the carry bit from second column becomes a sum bit.

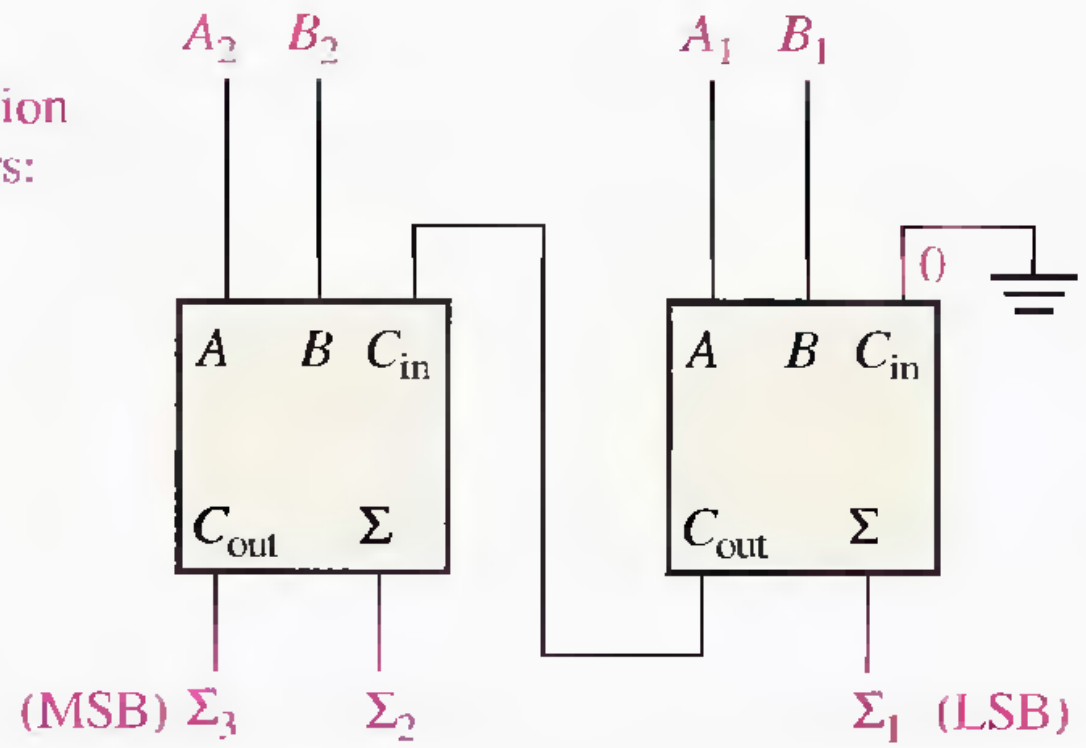
Parallel Binary Adder

- To add two binary numbers, a full-adder is required for each bit in the numbers.
- So for 2-bit numbers, two adders are needed.
- For 4-bit numbers, four adders are used; and so on.
- The carry output of each adder is connected to the carry input of the next higher-order adder.
- Notice that either a half-adder can be used
- for the least significant position or the carry input of a full-adder can be made 0 (grounded) because there is no carry input to the least significant bit position.

Parallel Binary Adder

General format, addition of two 2-bit numbers:

$$\begin{array}{r} A_2A_1 \\ +\ B_2B_1 \\ \hline \Sigma_3\Sigma_2\Sigma_1 \end{array}$$

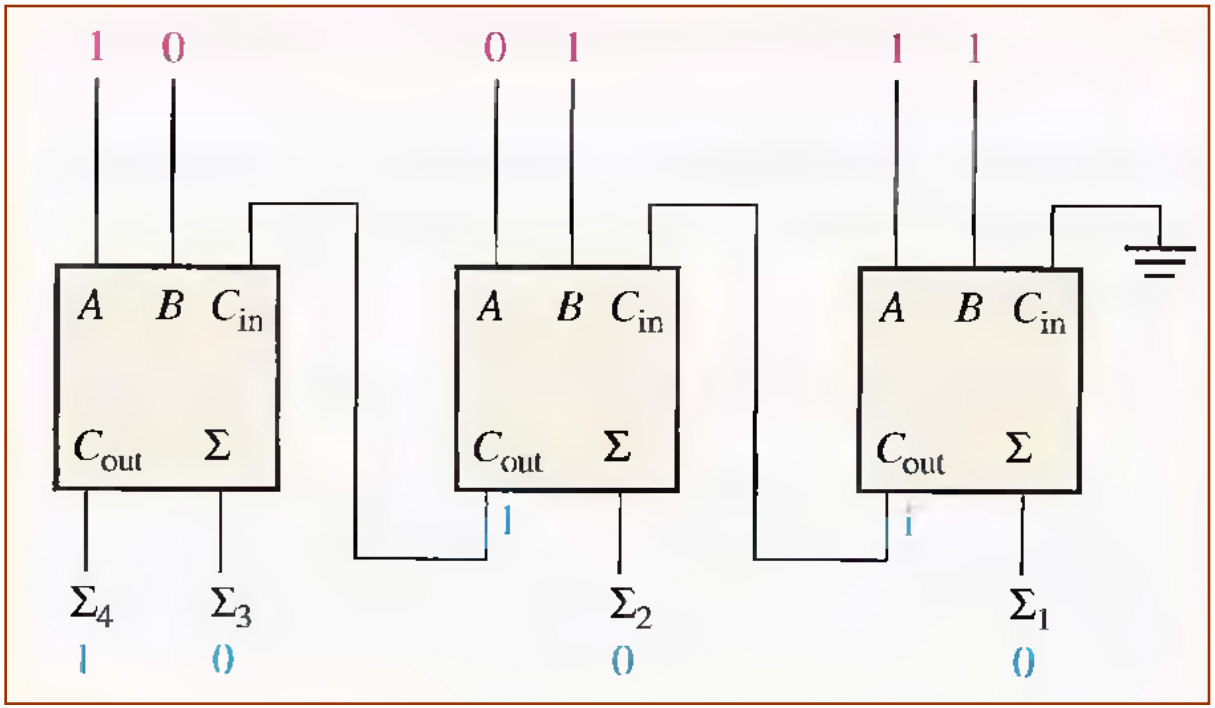


Parallel Binary Adder

- In Figure the least significant bits (LSB) of the two numbers are represented by A_1 and B_1.
- The next higher-order bits are represented by A_2 and B_2 .
- The three sum bits are Σ_1,Σ_2 and Σ_3.
- Notice that the output carry from the left-most full-adder becomes the most significant bit (MSB) in the sum, Σ_3.

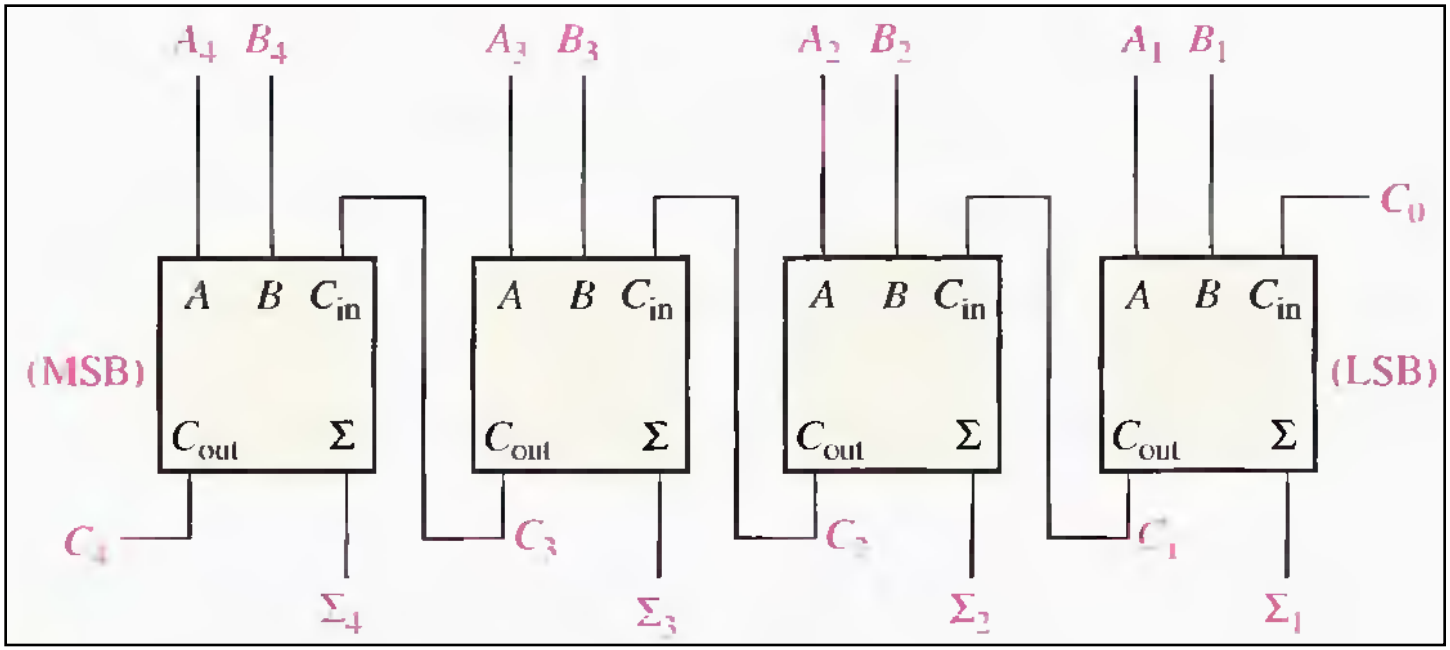
Example

- Determine the sum generated by the 3-bit parallel adder and show the intermediate carries when the binary numbers 101 and 011 are being added.

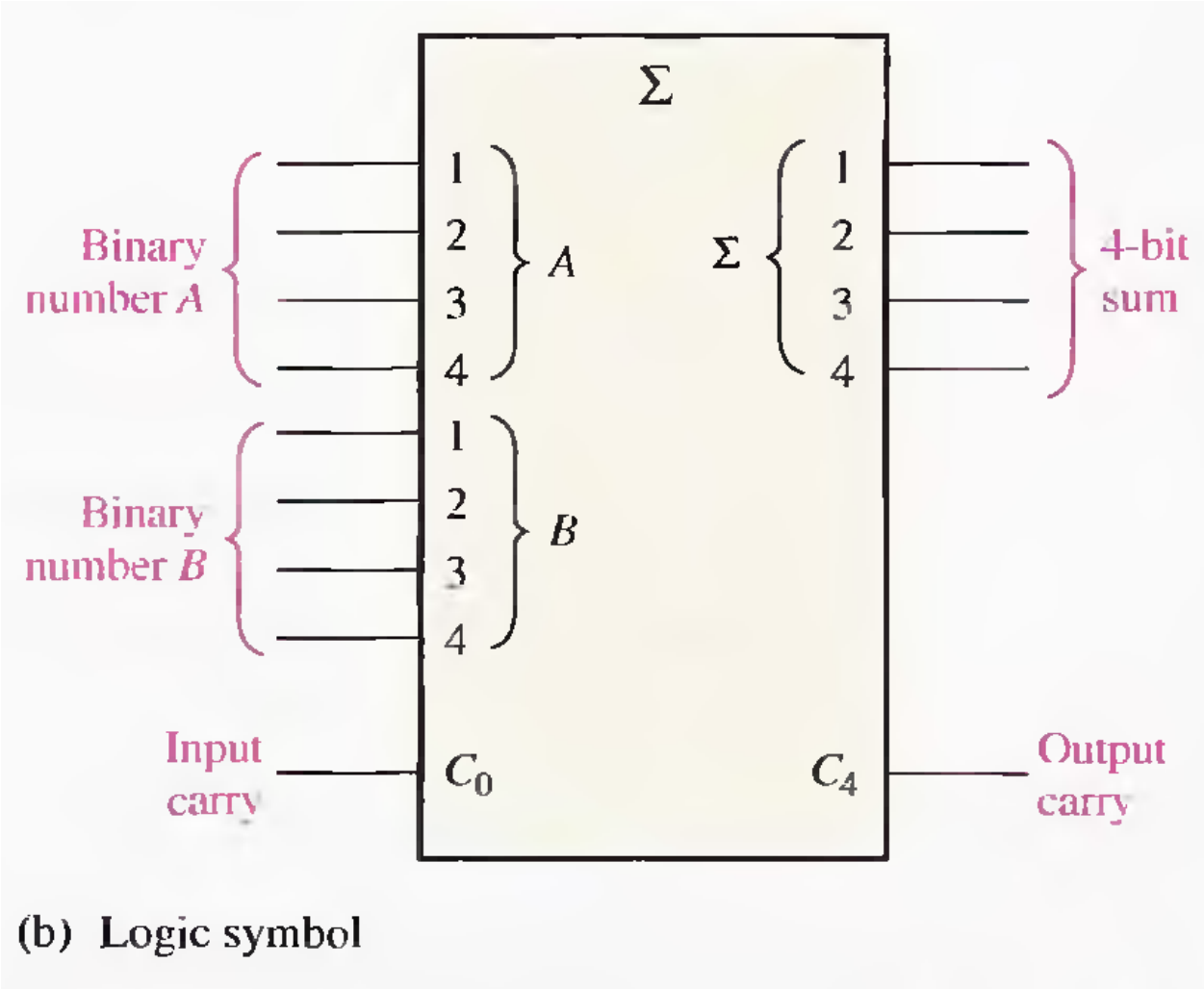


Four Bit Parallel Adder

- A group of four bits is called a nibble.
- A basic 4-bit parallel adder is implemented with four full-adder stages as shown in Figure .



Logical Symbol for 4 bit Parallel Adder

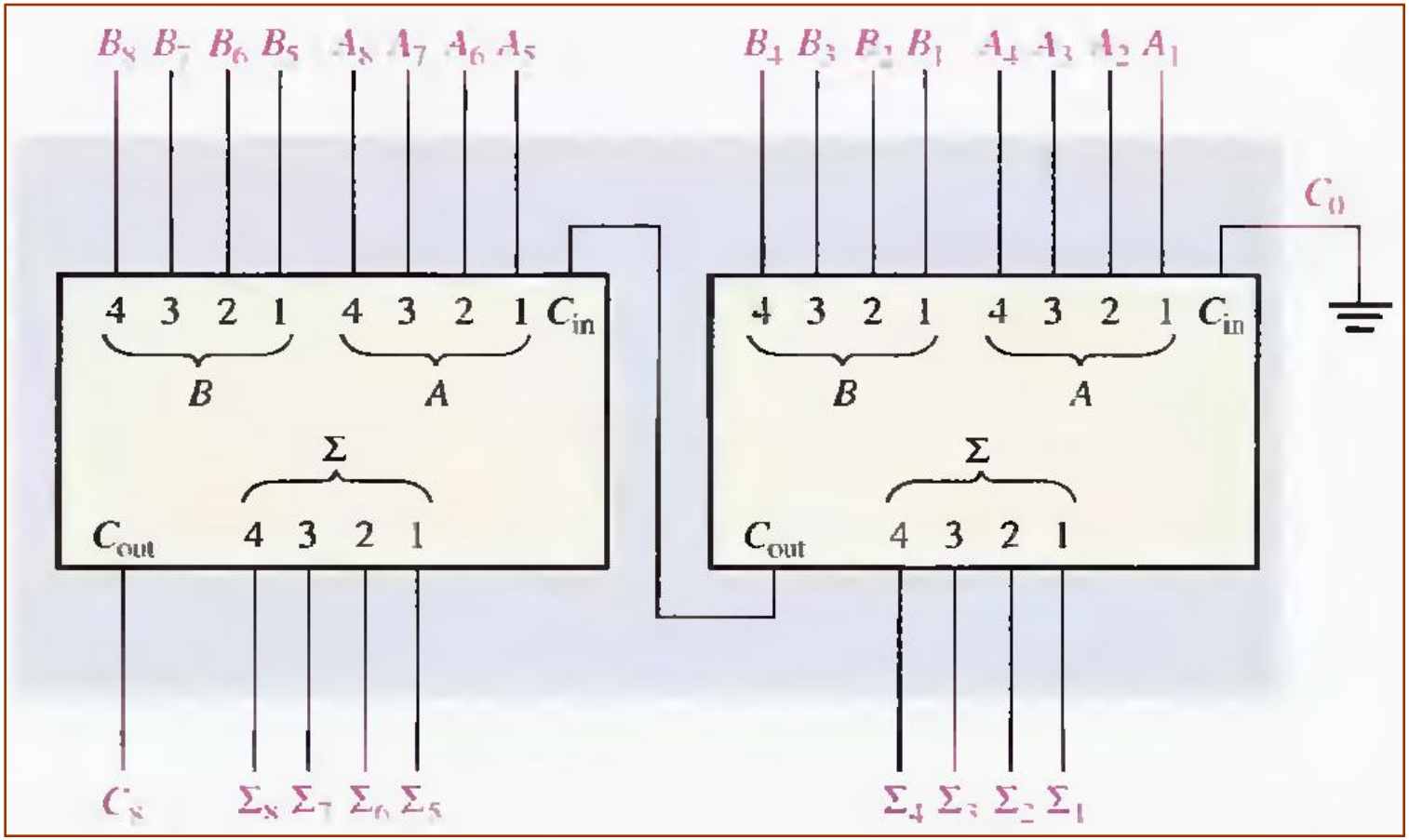


(b) Logic symbol

4 bit parallel adder

- The input labeled Co is the input carry to the least significant bit adder.
- C_4 in the case of four bits, is the output carry of the most significant bit adder; and Σ_1 (LSB) through Σ_4 (MSB) are the sum outputs.
- The 4-bit parallel adder can be expanded to handle the addition of two 8-bit numbers by using two 4-bit adders.

8 Bit Adder



Cascading of two 4-bit adders to form an 8-bit adder

Subtractors

- Subtraction of two binary number is accomplished by taking the complement of the subtrahend and adding it to the minuend.
- Logically it can be done through direct method.
- In this method each bit of the subtrahend is subtracted from its corresponding significant minuend bit to form a difference bit.
- If the minuend bit is smaller then a 1 borrow is taken from the next higher pair of the bits.

Subtractors

- Half Subtractor
  - It subtract two bits and produces their difference.
  - It also has an output to specify if a 1 has been borrowed.
  - x and y are minuend and subtrahend veriable.
  - For subtraction we check the relative magnitude of the x and y.
  - If x>=y then no issue.
  - If x<y then it is necessary to take a borrow from the next higher stage.

Half subtractor

- Truth table of half subtractor is,

X	Y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

D=x′ y + x y′
B=x′ y

Problem

- Draw a circuit diagram against Difference D and Borrow B.

Full Subtractor

- It performs a subtraction between two bits, taking in to account that a 1 may have been borrowed by a lower significant stage.
- It has three inputs and two outputs.
- Three inputs x, y and z shows the minuend, subtrahend and previous borrow respectively.
- B and D represents the output borrow and Difference.

Full Subtractor

- Truth table is as under,

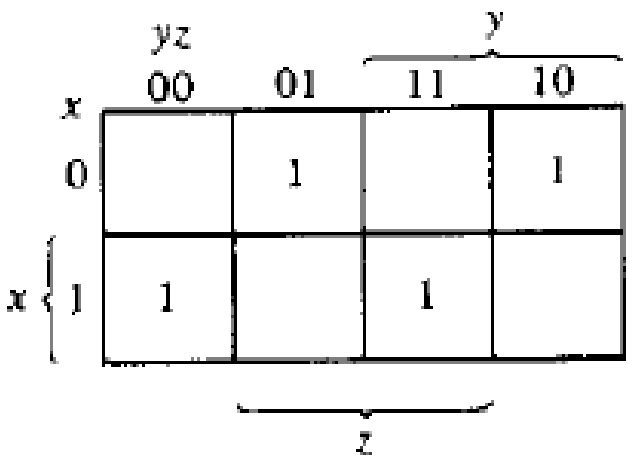
x	y	z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Full Subtractor

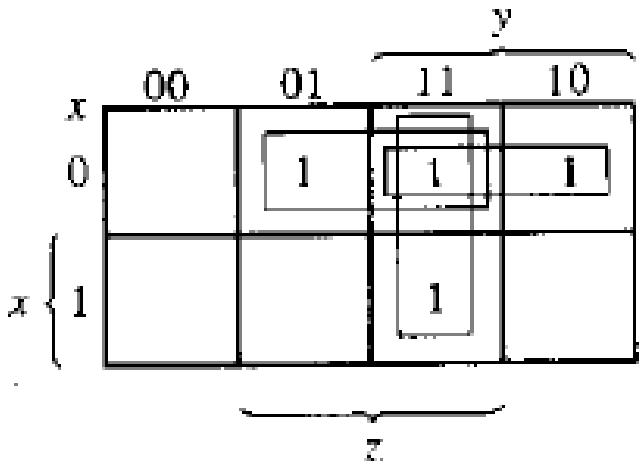
- The function against B and D are,

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

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



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



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

Code conversion

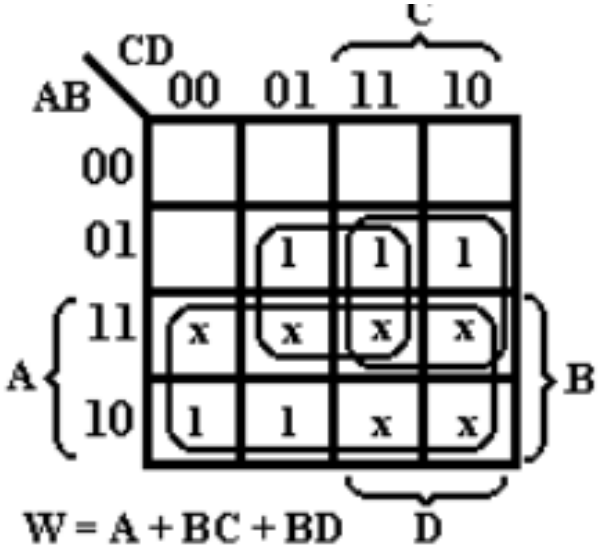
- Some times the output of the one system as the input to another.
- If both the system uses different coding system, then code convertor is needed between them.
- Thus a code convertor is a circuit that makes the two system compatible.
- To convert from binary code A to B,
  - Input lines supply the bit combination of elements by Code A and the output lines must generate the corresponding bit combination for code B.

- Code conversion from BCD to Excess 3 is illustrated below,

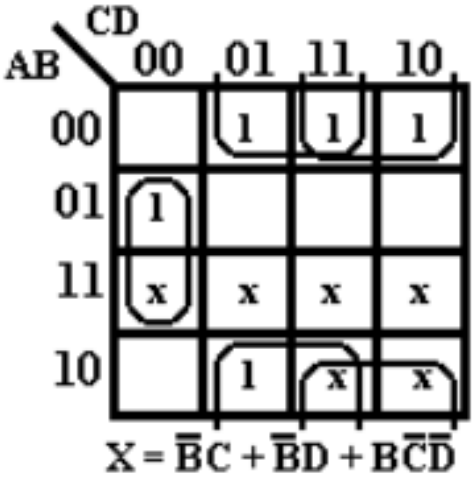
1) BCD code: ABCD
2) Axcess-3 code: WXYZ
3) Truth table:

Decimal	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
10	1	0	1	1	x	x	x	x
:	:	:	:	:	:	:	:	:
15	1	1	1	1	x	x	x	x

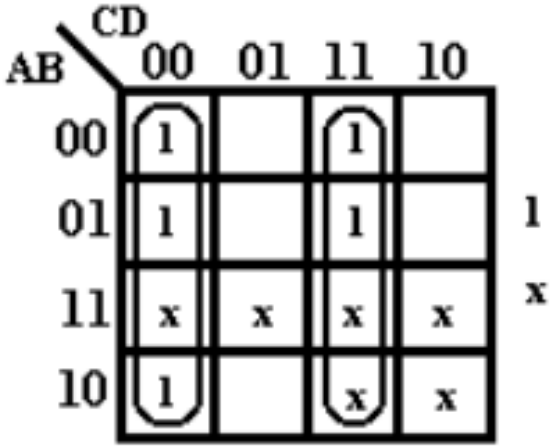
Outputs Simplification



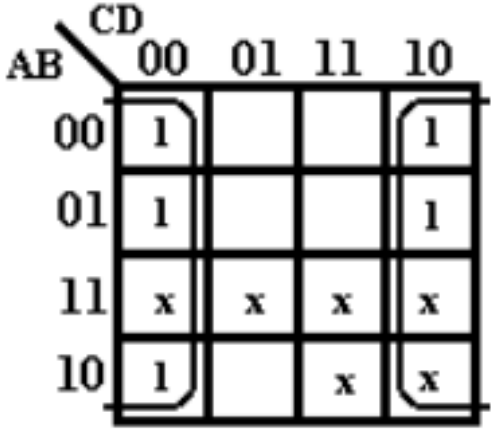
$W = A + BC + BD$



$X = \overline{B}C + \overline{B}D + B\overline{C}\overline{D}$

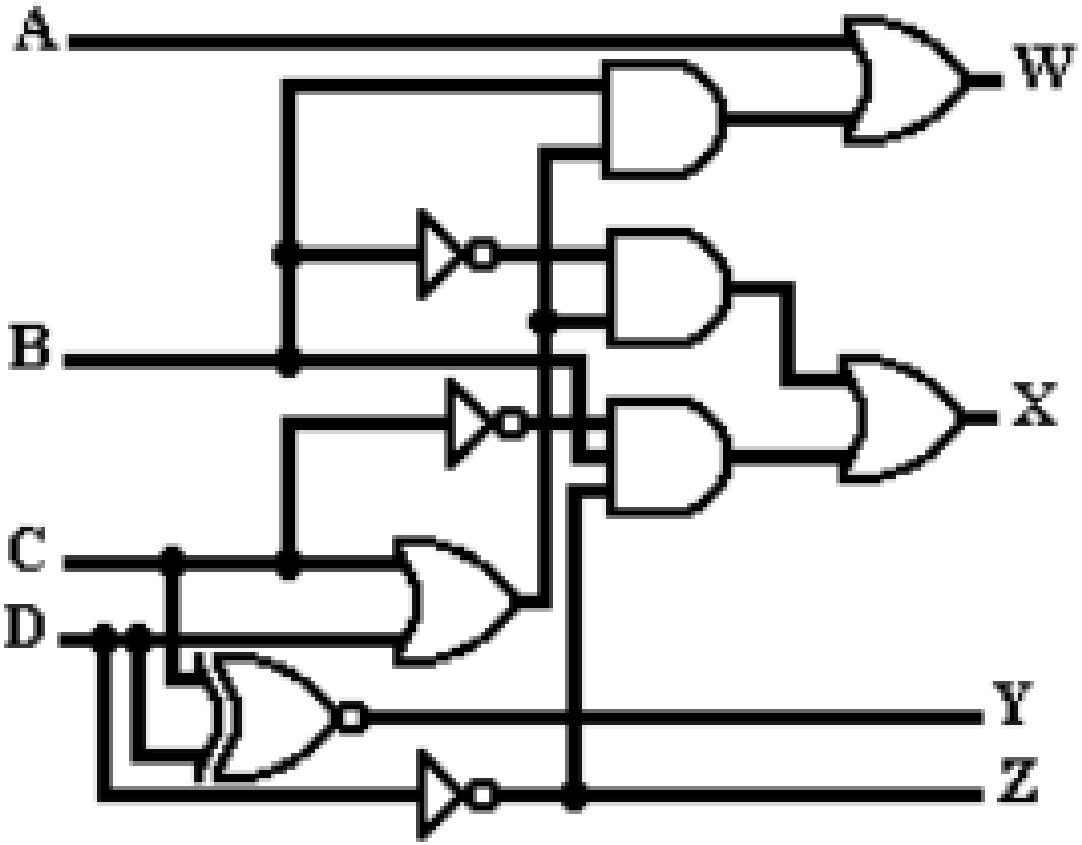


$Y = CD + \overline{C}\overline{D}$

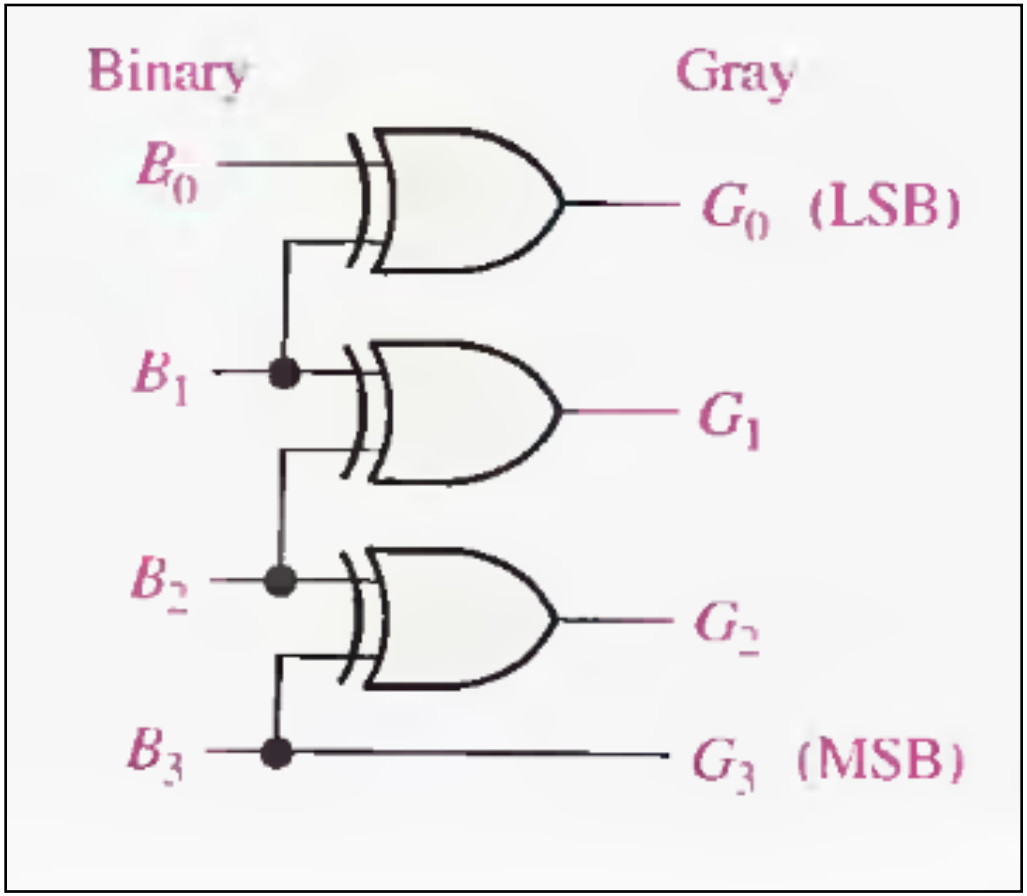


$Z = \overline{D}$

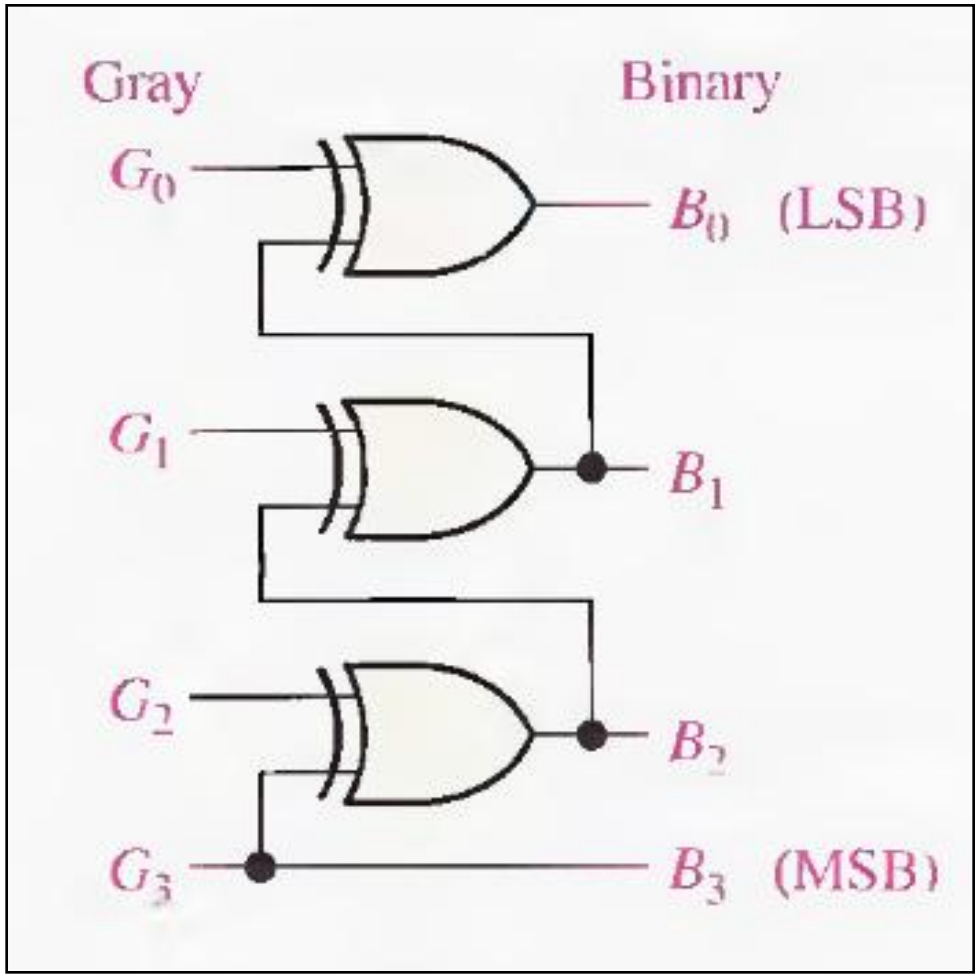
Circuit Diagram



Binary to Gray code



Gray to Binary



Related Problem

How many exclusive-OR gates are required to convert 8-bit binary to Gray?


thanks
for the
tolerance