COMBINATIONAL LOGIC CIRCUITS

Prepared By:
Nuren Zabin Shuchi
Lecturer,
Department of Electrical and Electronic Engineering
Shahjalal University of Science and Technology

Outline

- INTRODUCTION
- COMBINATIONAL LOGIC CIRCUITS
- DESIGN PROCEDURE
- ADDERS
  - i. Design of Half-adders
  - ii. Design of Full-adders
- SUBTRACTORS
  - i. Design of Half-Subtractors
  - ii. Design of Full-Subtractors
- PRACTICE PROBLEMS

INTRODUCTION

The digital system consists of **two types** of circuits:

(*i*) Combinational circuits and

(*ii*) Sequential circuits

Combinational circuits: 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: 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**.

COMBINATIONAL LOGIC CIRCUITS

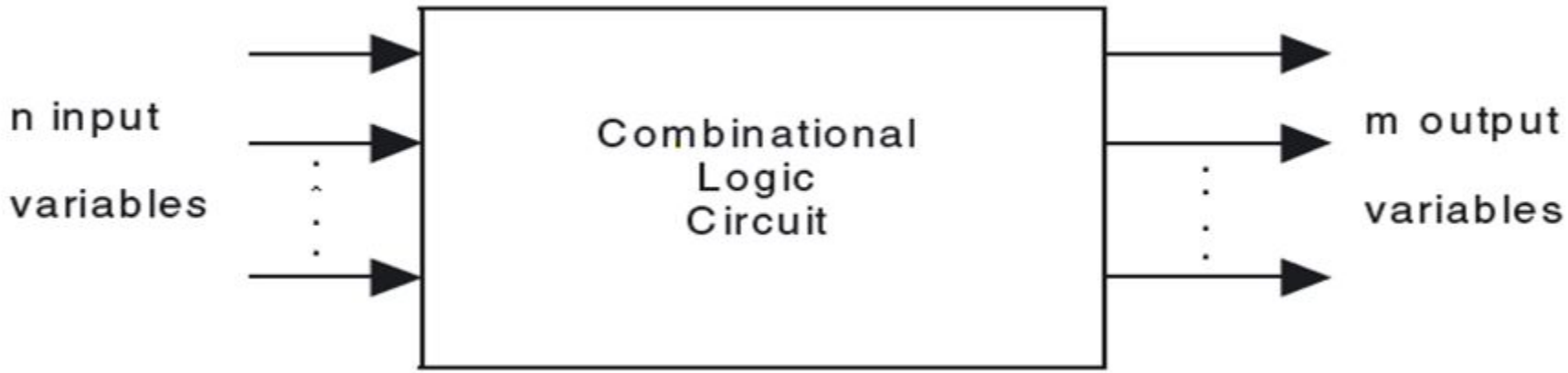


Fig-1: Block diagram of Combinational Logic circuit

- A combinational circuit consists of input variables, logic gates, and output variables.
- The logic gates accept signals from inputs and output signals are generated according to the logic circuits employed in it.
- Both input and output are obviously the binary signals, logic 1 and logic 0.
- For *n* number of input variables to a combinational circuit, 2^n possible combinations of binary input states are possible.
- For each possible combination, there is one and only one possible output combination.

DESIGN PROCEDURE

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

- The problem is stated.
- Identify the input variables and output functions.
- The input and output variables are assigned letter symbols.
- The truth table is prepared that completely defines the relationship between the input variables and output functions.
- The simplified Boolean expression is obtained by any method of minimization—algebraic method, Karnaugh map method, or tabulation method.
- 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
- 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***.

There are **two types** of adders:

- **Half Adder:** A combinational circuit that performs the addition of **two bits** is called a *half-adder*.
- **Full Adder:** A combinational circuit that performs the addition of **three bits** (two operands and a **carry bit**) is called a *full-adder*.

Design of Half-adders

Let the input variables as A, B and outputs sum as S and carry as C.

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

Fig-2: Truth Table

From the truth table in Figure-2, it can be seen that the outputs S and C functions are similar to Exclusive-OR and AND functions respectively. So that the Boolean expressions are:

$S = A'B + AB'$ and

$C = AB$

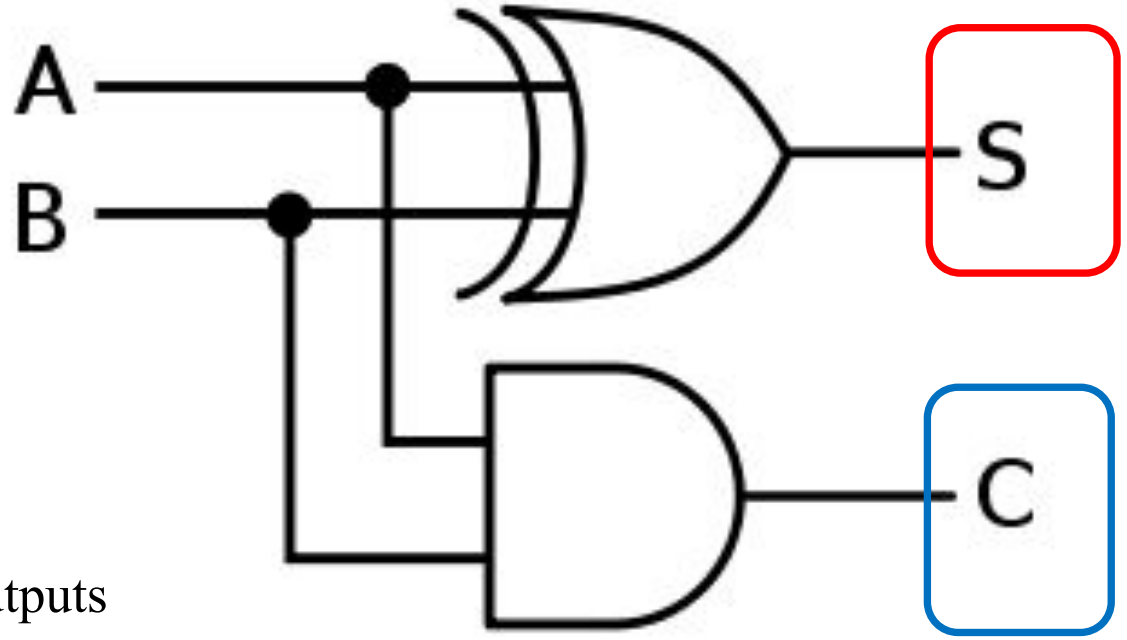


Fig-3: Logic diagram of Half-Adder

Design of Full-adders

Let the input variables as A, B and previous carry as X, and outputs sum as S and carry as C.

Input variables			Output variables	
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

Fig-4: Truth table

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

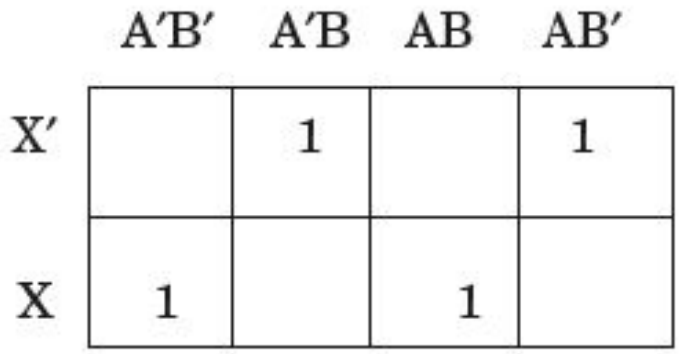


Fig-5: Map for function S.

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

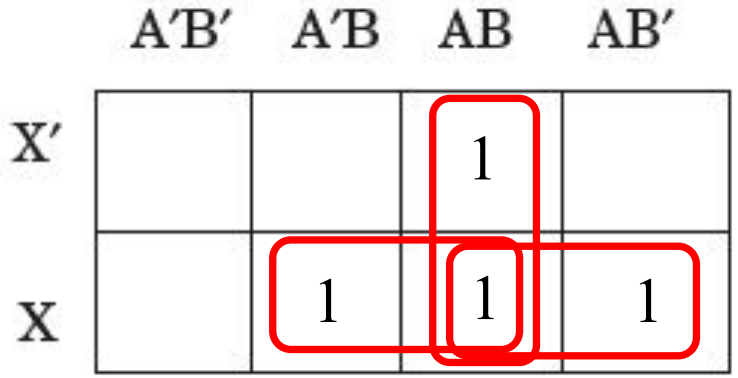


Fig-6: Map for function C.

$$C = AB + BX + AX.$$

Design of Full-adders (cont.)

The logic diagram for the functions is shown below:

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

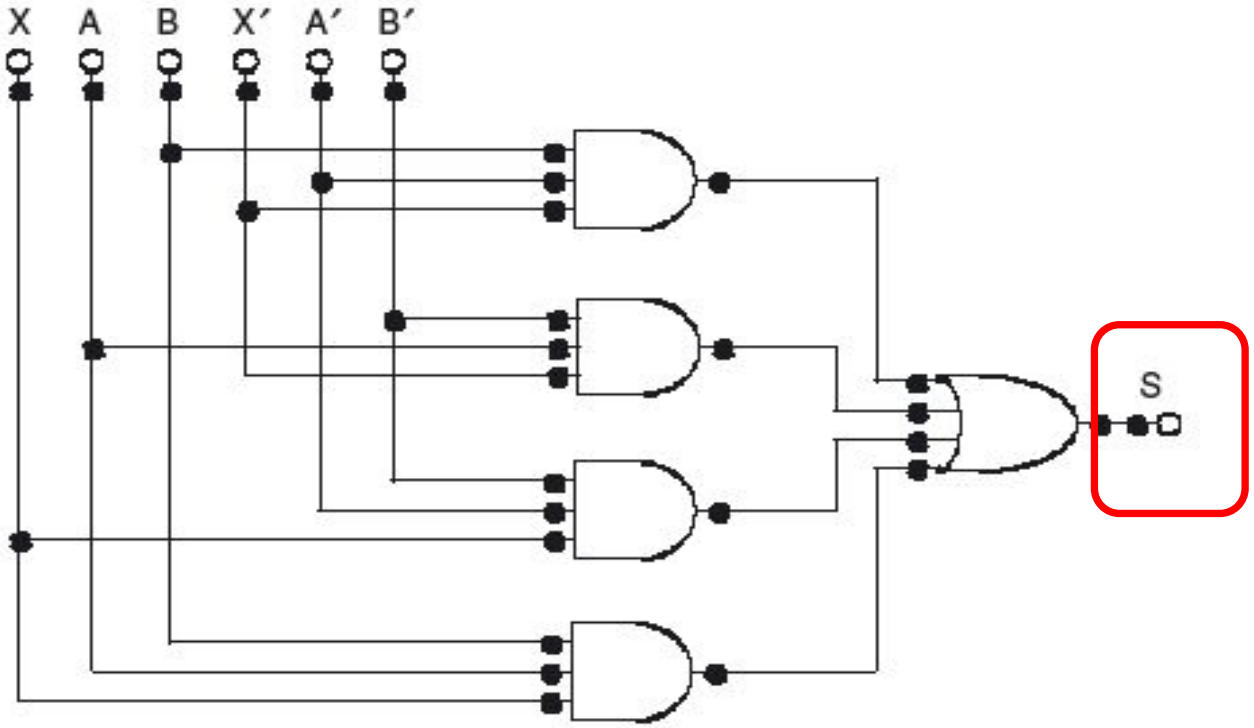


Fig-7: Logic Diagram for function S.

$$C = AB + BX + AX.$$

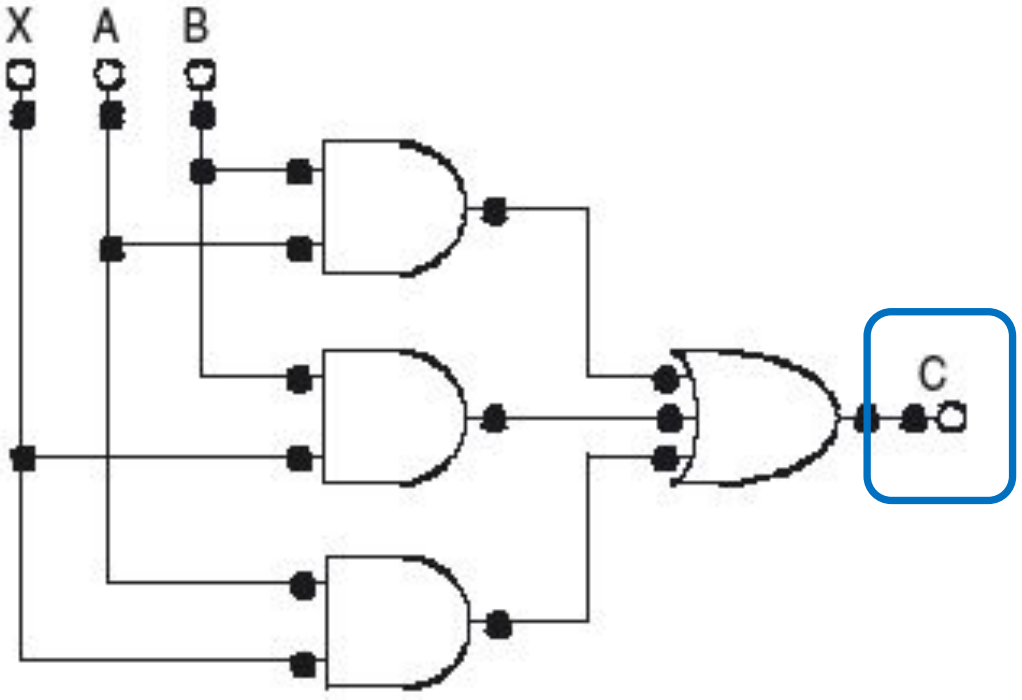


Fig-8: Logic Diagram for function C.

Design of Full-adders (cont.)

Other configurations can also be developed where **number** and **type** of gates are **reduced**.
For this, the Boolean expressions of S and C are **modified** as follows:

$$S = X'A'B + X'AB' + XA'B' + XAB$$
$$= X' (A'B + AB') + X (A'B' + AB)$$
$$= X' (A \oplus B) + X (A \oplus B)'$$
$$= X \oplus A \oplus B$$

And

$$C = AB + BX + AX$$
$$= AB + X (A + B)$$
$$= AB + X (AB + AB' + AB + A'B)$$
$$= AB + X (AB + AB' + A'B)$$
$$= AB + XAB + X (AB' + A'B)$$
$$= AB + X (A \oplus B)$$

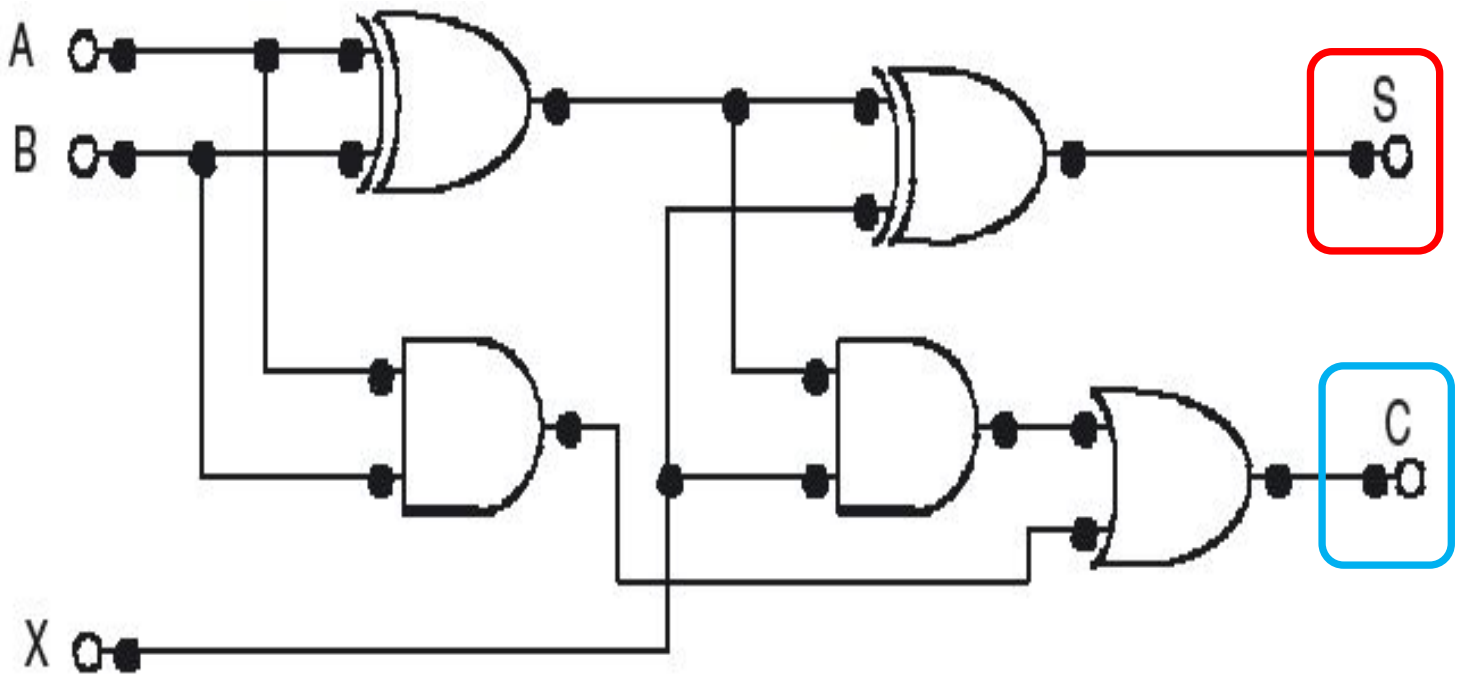


Fig- 9: Logic diagram for the modified expression

SUBTRACTORS

Subtraction is another basic arithmetic operation.
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
- 1–1 = 0

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

There are two types of Subtractors:

- **Half-Subtractor:** A combinational circuit that performs the subtraction of two bits is called a *half-subtractor*.
- **Full-Subtractor:** A combinational circuit that performs the subtraction of three bits (the minuend bit, subtrahend bit, and the borrow bit) is called a *full-subtractor*.

Design of Half-Subtractors

Let the input variables minuend as X, subtrahend as Y and outputs difference as D and borrow as B.

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

Fig-10: Truth Table

By considering the minterms of the truth table in Figure 10, the Boolean expressions of the outputs D and B functions can be written as:

$D = X'Y + XY'$ and

$B = X'Y$

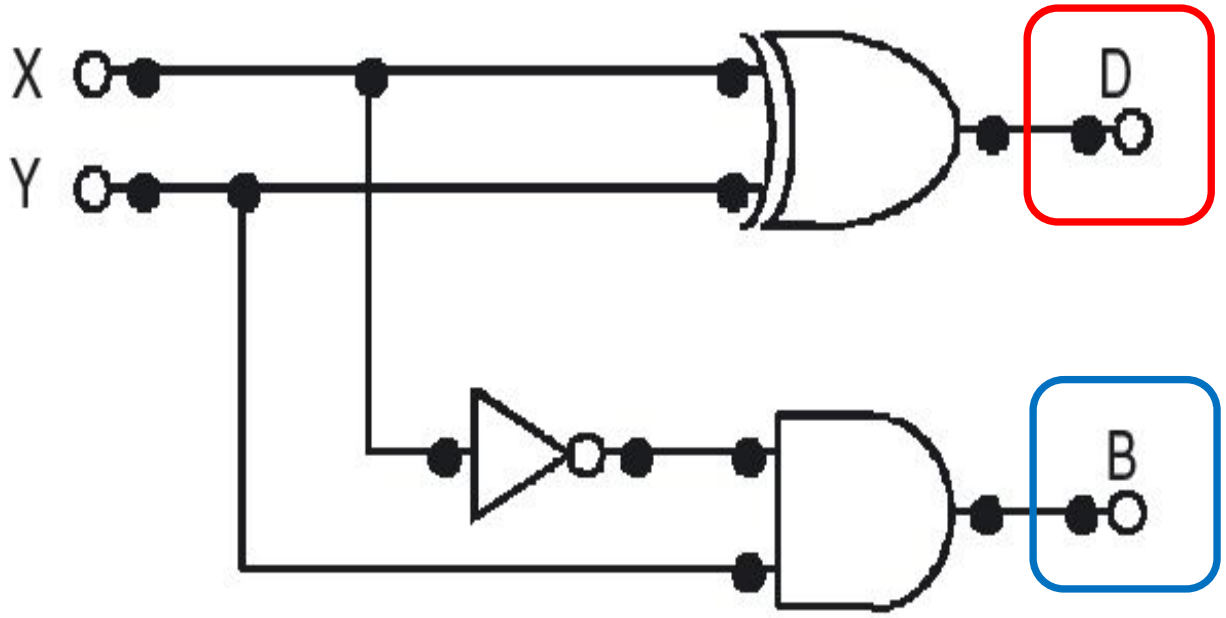


Fig-11: Logic diagram of Half-Subtractor

Design of Full-Subtractors

Let the input variables minuend as X, subtrahend as Y, and previous borrow as Z, and outputs difference as D and borrow as B.

Input variables			Output variables	
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

Fig-12: Truth table

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

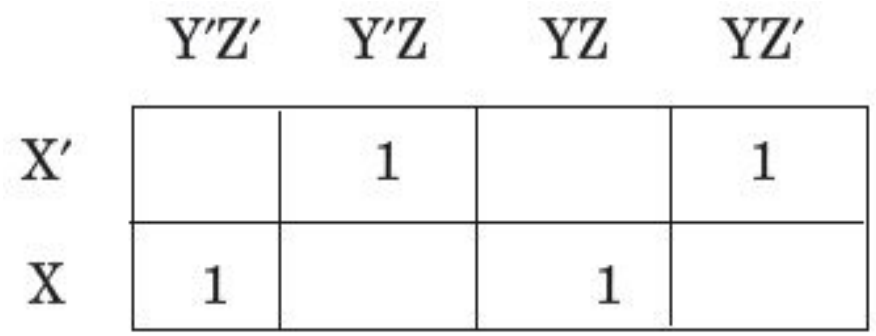


Fig-13: Map for function D.

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

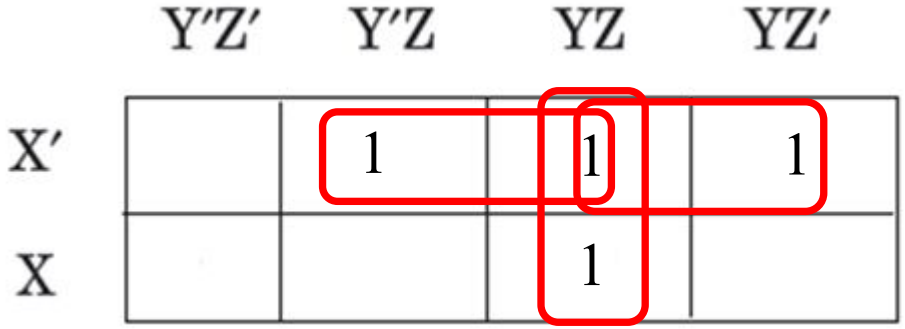


Fig-14: Map for function B.

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

Design of Full-Subtractors (Cont.)

The logic diagram for the functions is shown below:

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

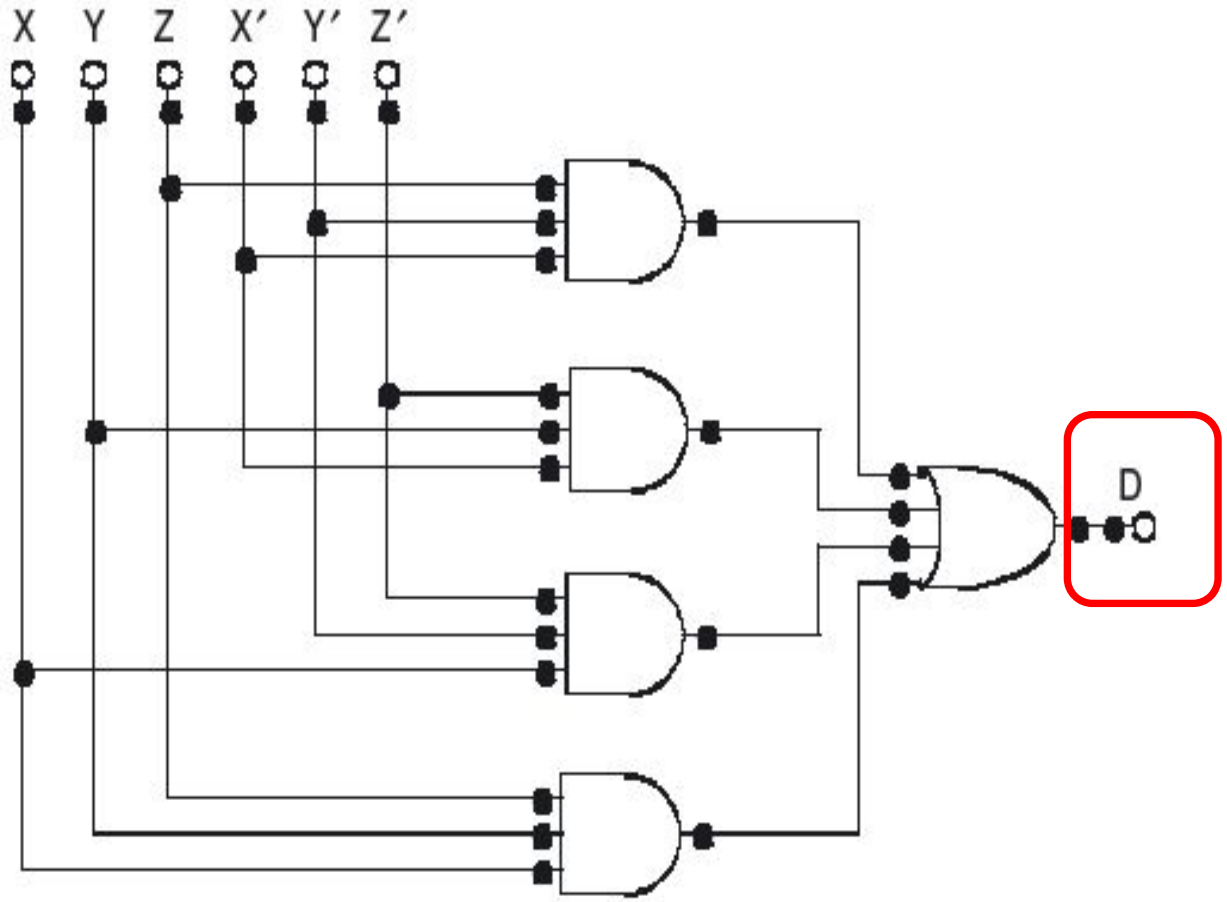


Fig-15: Logic Diagram for function S.

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

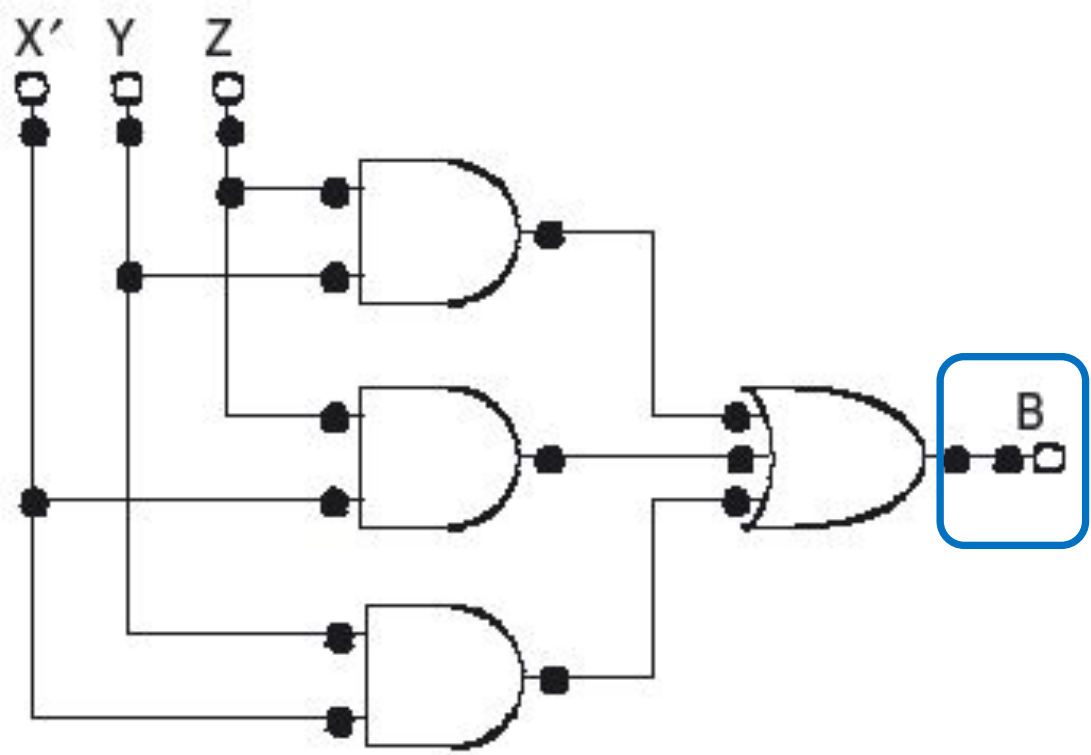


Fig-16: Logic Diagram for function S.

Design of Full-Subtractors (Cont.)

Other configurations can also be developed where **number** and **type** of gates are **reduced**. For this, the Boolean expressions of D and B are **modified** as follows:

$$D = X'Y'Z + X'YZ' + XY'Z' + XYZ$$
$$= X'(Y'Z + YZ') + X(Y'Z' + YZ)$$
$$= X'(Y \oplus Z) + X(Y \oplus Z)'$$
$$= X \oplus Y \oplus Z$$

And

$$B = X'Z + X'Y + YZ$$
$$= X'Y + Z(X' + Y)$$
$$= X'Y + Z(X'Y + X'Y' + XY + X'Y)$$
$$= X'Y + Z(X'Y + X'Y' + XY)$$
$$= X'Y + X'YZ + Z(X'Y' + XY)$$
$$= X'Y + Z(X \oplus Y)'$$

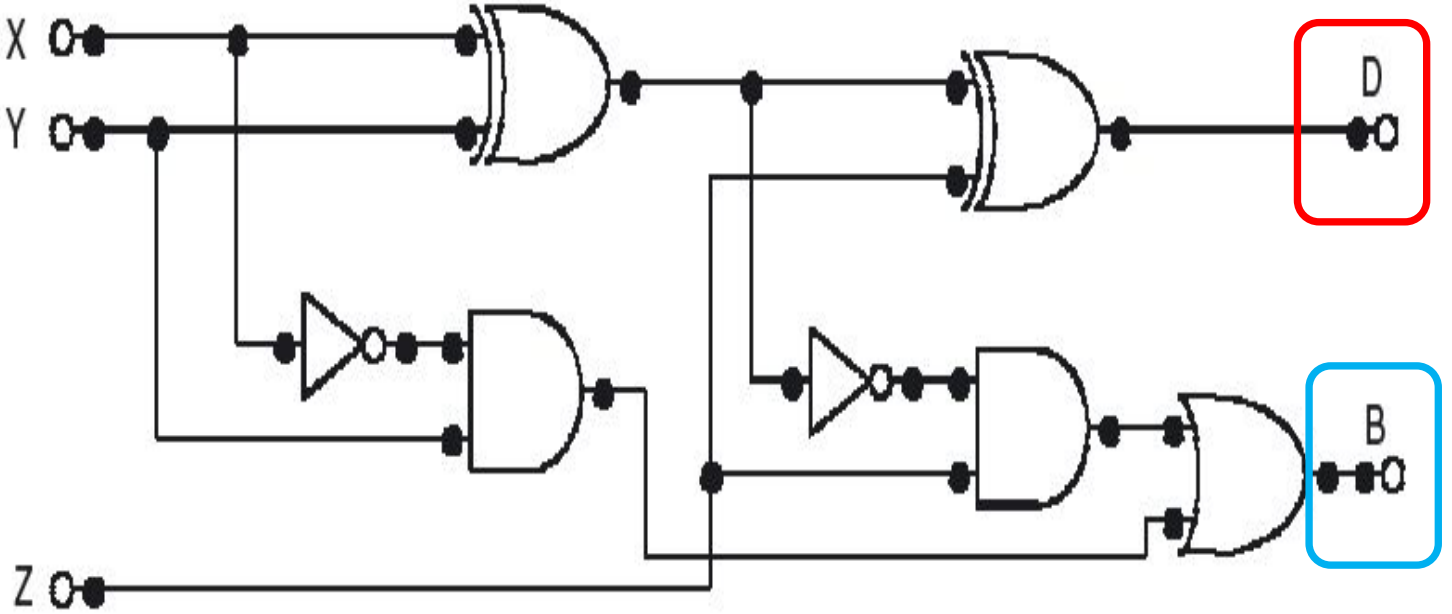


Fig-17: Logic diagram for the modified expression

Practice Problems

1. Given two input bits A and B, produce three outputs X, Y, and Z so that

- X is 1 only when only when A > B,
- Y is 1 only when A < B, and
- Z is 1 only when A = B

✔ Input Variables: A & B
Output Variables: X,Y & Z

Input		output		
A	B	X	Y	Z
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

✔ Each Output variable will generate a different function.
✔ Number of K-Maps would equal the number of Output variables

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

X=AB’

Y=A’B

Z=AB+A’B’= A XNOR B

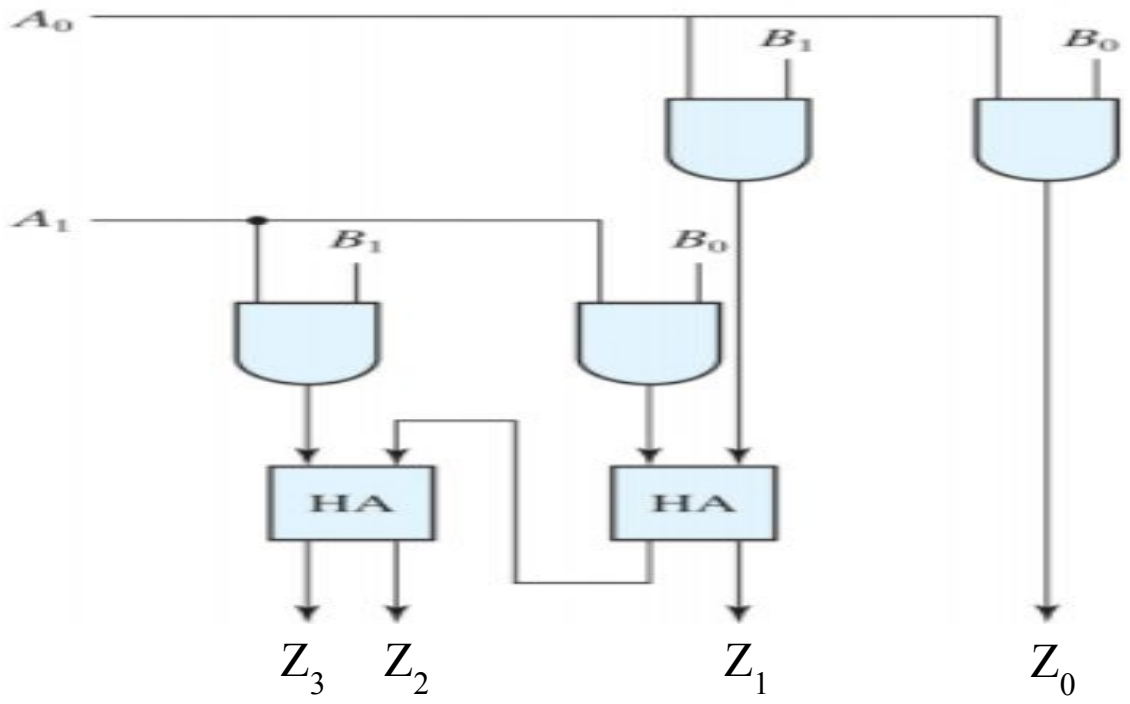
Practice Problems

Design a combinational circuit that multiplies two 2 bit numbers.

Input: $A=A_1A_0$ & $B=B_1B_0$

- ✔ Preparing a Truth Table is a bit tedious in this case.
- ✔ Let us see what actually happens when $A=A_1A_0$ & $B=B_1B_0$ are multiplied.

$$
\begin{array}{r}
A_1A_0 \\
\times\ B_1B_0 \\
\hline
A_1B_0\ \ A_0B_0 \\
+ \\
A_1B_1\ \ A_0B_1 \\
\hline
Z_3\ \ Z_2\ \ Z_1\ \ Z_0
\end{array}
$$



- Carry of the first HA flows to the next HA as an input bit.
- The Sum bit of the second HA is Z_2 and Carry bit is Z_3 .

Practice Problems

2. Design a circuit that has a 3-bit binary input and a single output that

   output 1 if it is a prime number. eg 2_{10}, 3_{10}, 5_{10}, 7_{10}
   otherwise output 0.

3. Design a circuit that has a 3-bit binary input and a single output (Z) specified as follows:

   Z = 0, when the input is less than 5_{10}
   Z = 1, otherwise

4. Design a Full Adder using half adders.
5. Implement a full subtractor using half subtractors.
6. Design a combinational circuit that multiplies two 3 bit numbers.
7. Design a magnitude comparator (4-bit).

Any Questions?