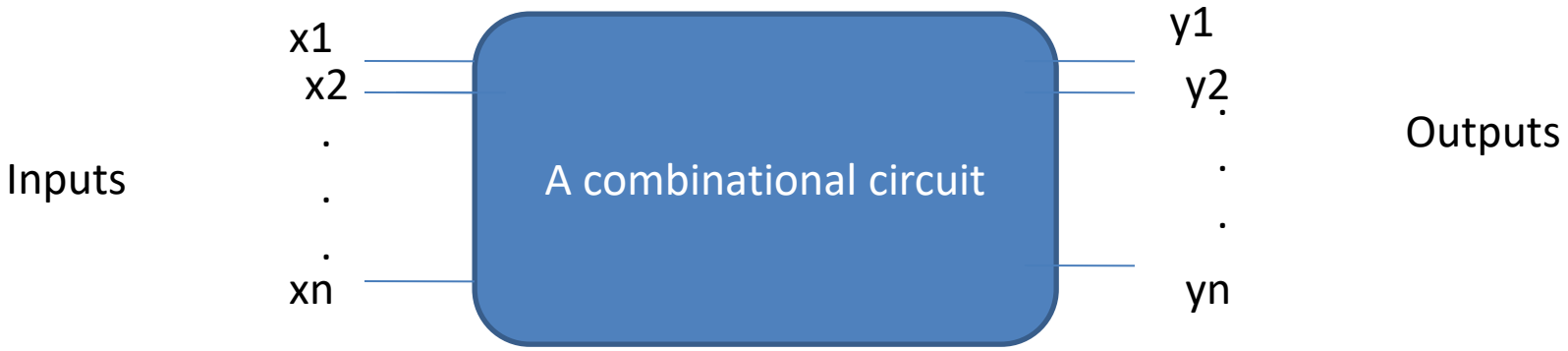
Introduction:

- Logic circuit for digital system may be:
  - Combinational Logic
  - Sequential Logic

Combinational logic

- Output at any times depends only on present combination of input.
- Combinational circuit does not need any memory.
- Combinational circuits are faster in operation than sequential circuits.



Block diagram

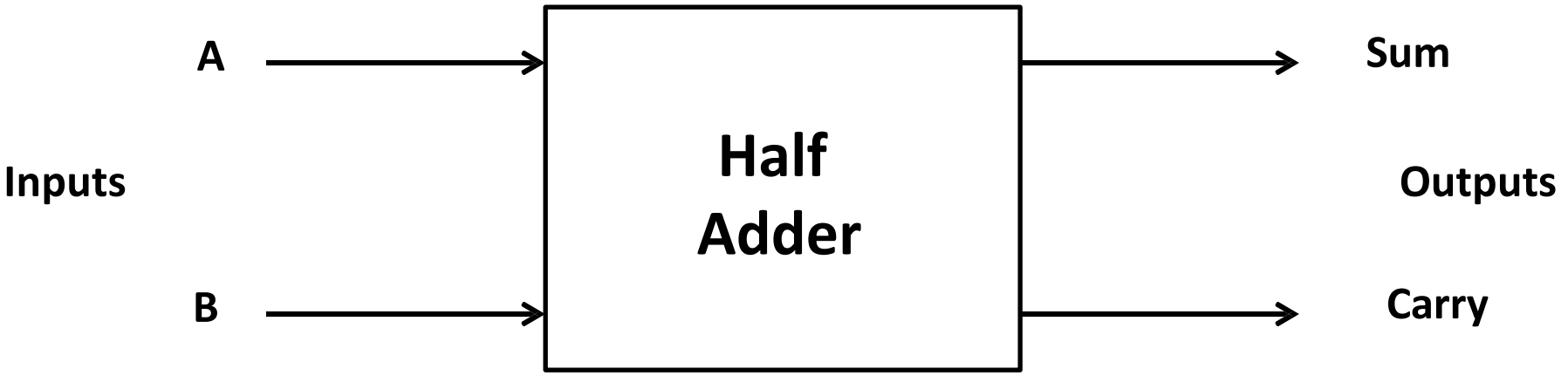
Design procedure:

The design steps for combinational logic are:

- Question is given to you.
- Find the number of i/p and o/p.
- Draw the truth table.
- Simplify the o/p using K-map. Simplification is done for o/p only.
- Draw the logic diagram.

Half Adder

- ✓ Half adder is a combinational logic circuit with two inputs and two outputs.
- ✓ It is a basic building block for addition of two single bit numbers.



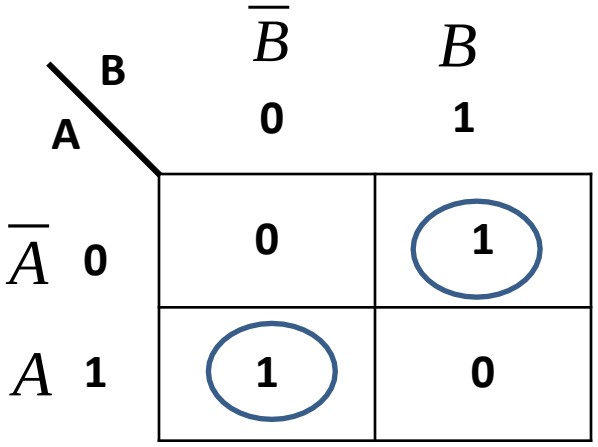
Half Adder

Truth Table for Half Adder

Input		Output	
A	**B**	**Sum (S)**	**Carry (C)**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Half Adder

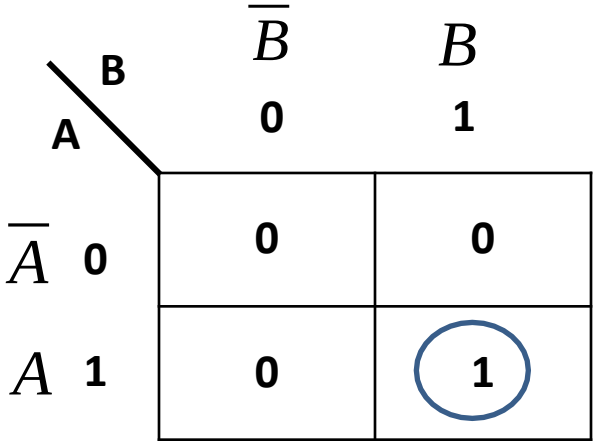
K-map for Sum Output:



$$S = \overline{A}B + A\overline{B}$$

$$S = A \oplus B$$

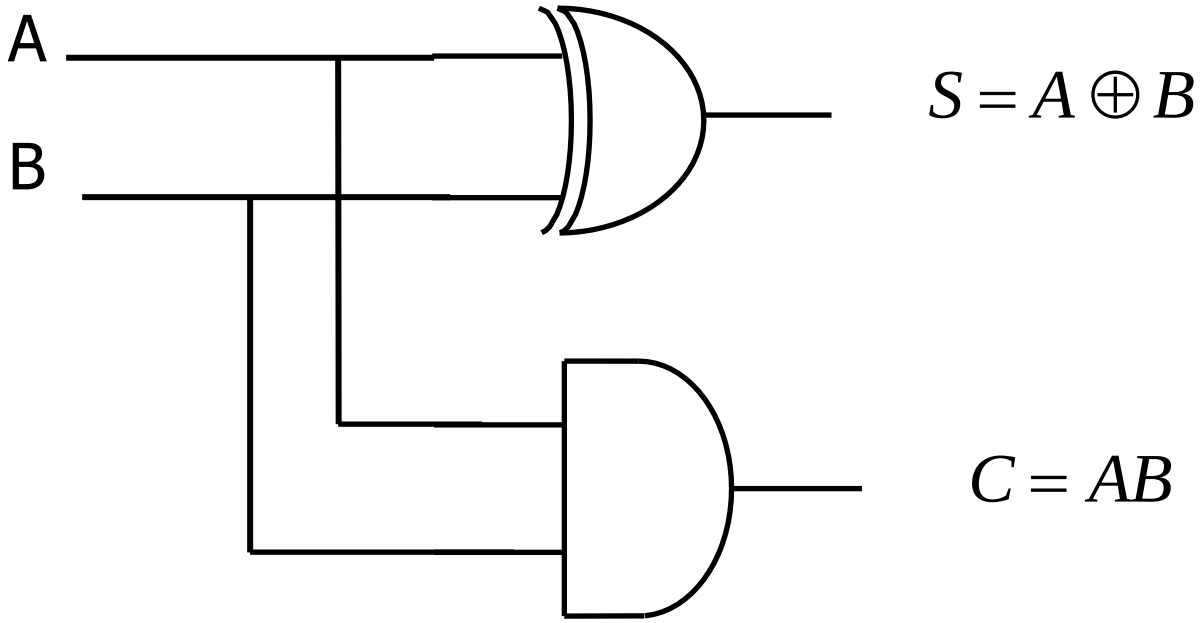
K-map for Carry Output:



$$C = AB$$

Half Adder

Logic Diagram:

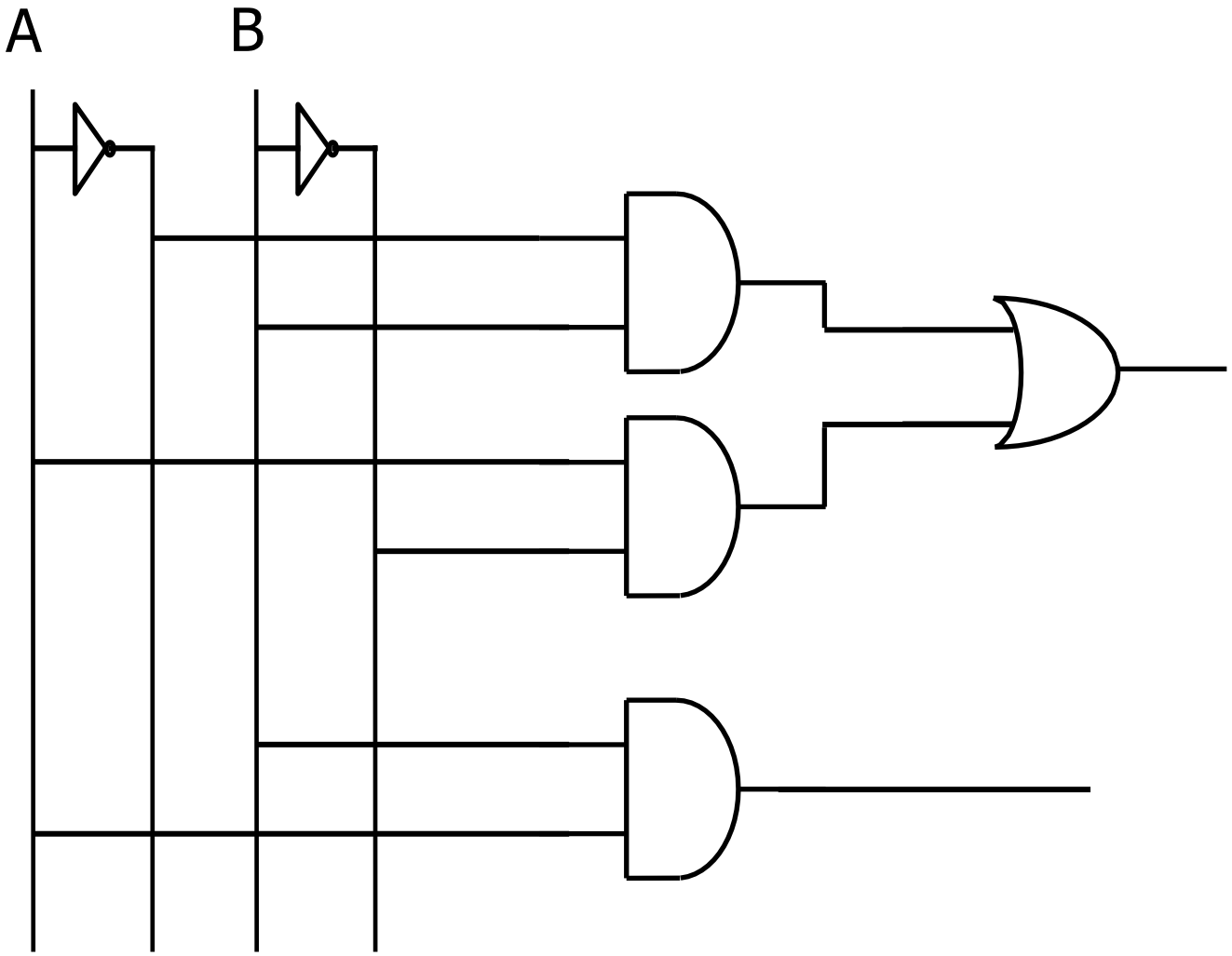


$$S = A \oplus B$$

$$C = AB$$

Half Adder

Logic Diagram using Basic Gates:

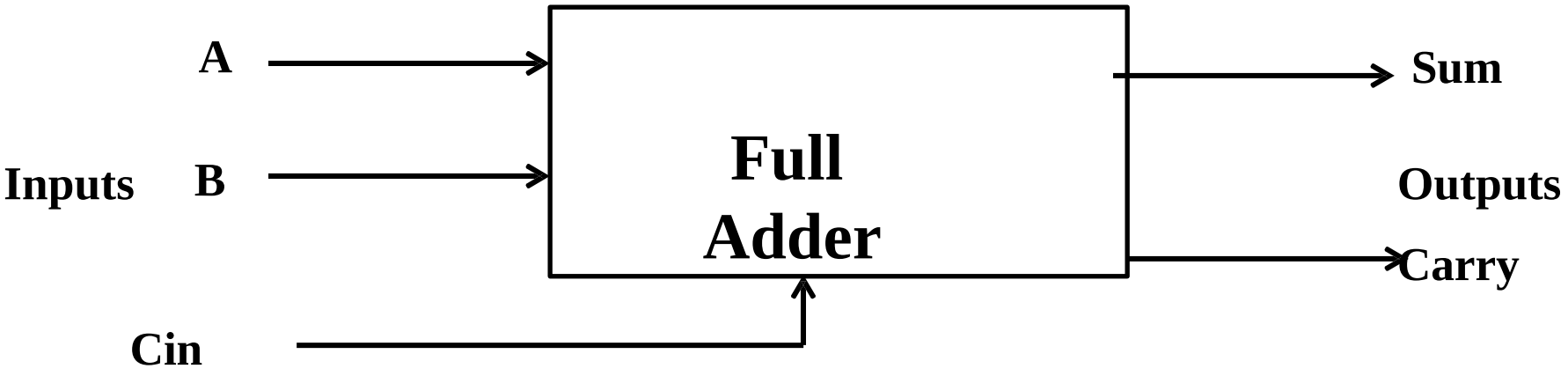


$$S = A \oplus B$$

$$C = AB$$

Full Adder

- ✓ Full adder is a combinational logic circuit with three inputs and two outputs.



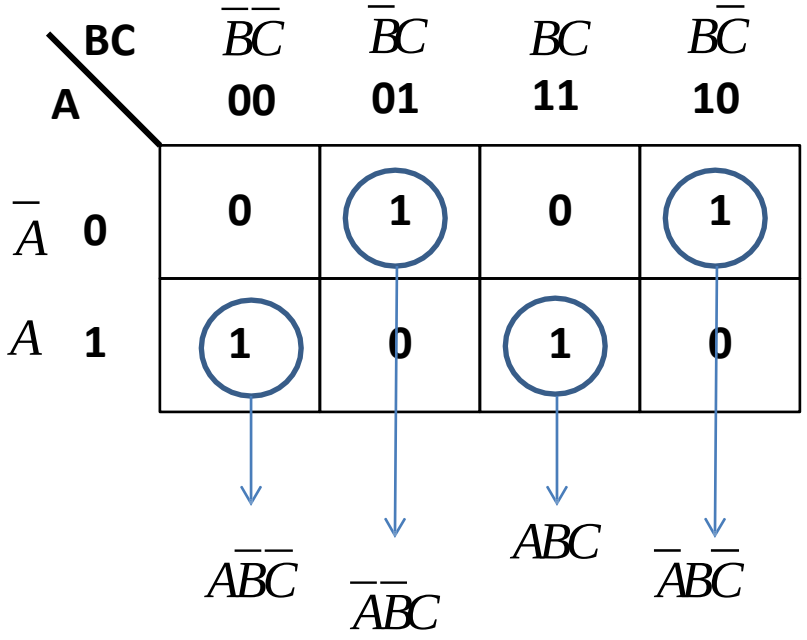
Full Adder

Truth Table

Inputs			Outputs	
A	B	Cin	Sum (S)	Carry (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

Full Adder

K-map for Sum Output:



$$S = \overline{A}\overline{B}C + \overline{A}B\overline{C} + ABC + A\overline{B}\overline{C}$$

$$S = \overline{A}\overline{B}C + ABC + \overline{A}B\overline{C} + A\overline{B}\overline{C}$$

$$S = C(\overline{A}\overline{B} + AB) + \overline{C}(\overline{A}B + A\overline{B})$$

$$Let \quad \overline{A}B + A\overline{B} = X$$

$$\therefore S = C(\overline{X}) + \overline{C}(X)$$

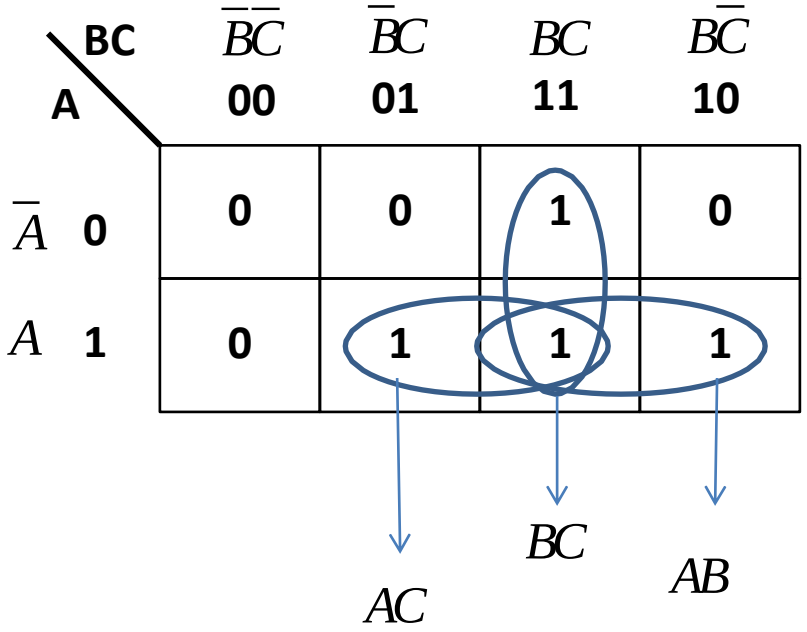
$$S = C \oplus X$$

$$Let \quad X = A \oplus B$$

$$\therefore S = C \oplus A \oplus B$$

Full Adder

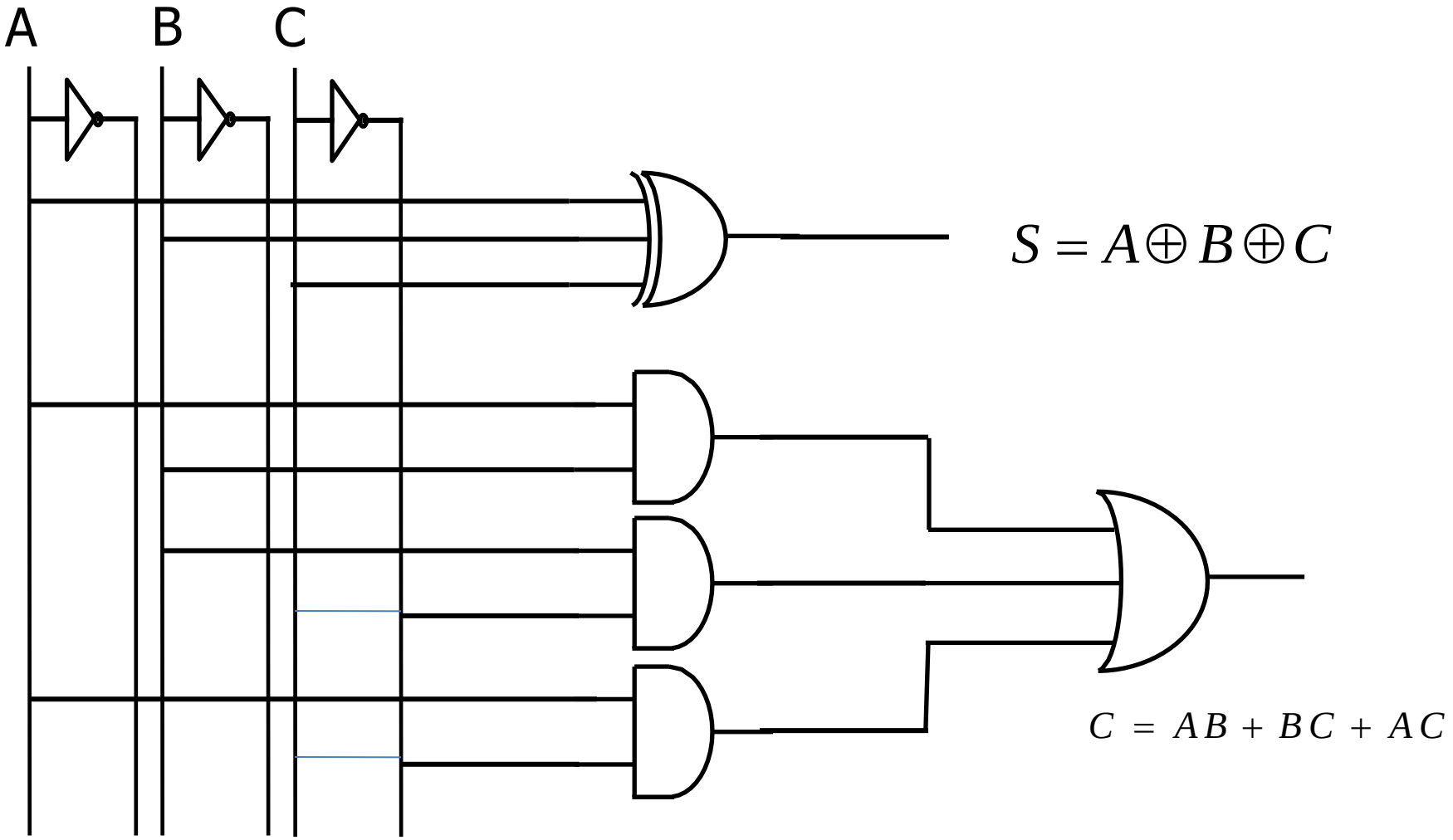
K-map for Carry Output:



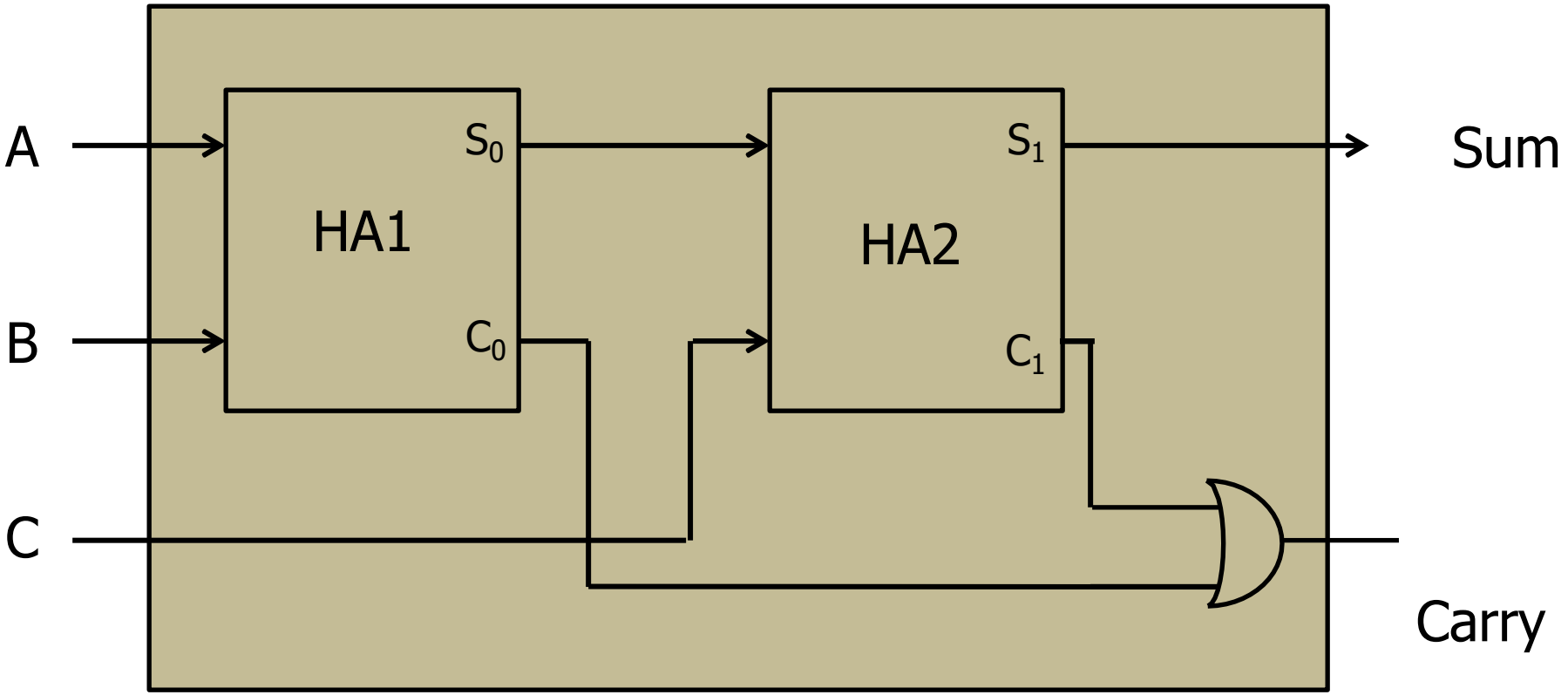
$$C = AB + BC + AC$$

Full Adder

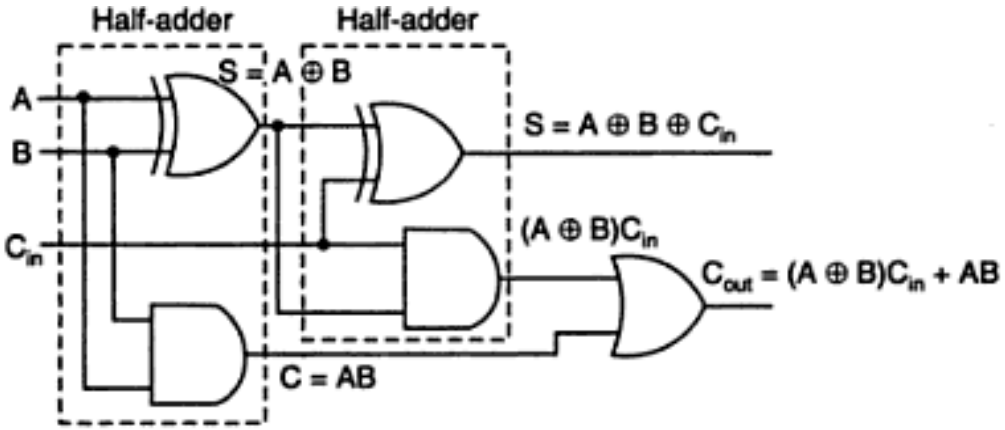
Logic Diagram:



Full Adder using Half Adders

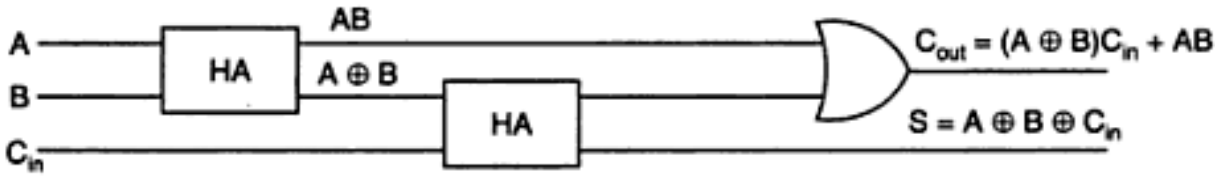


Draw the full adder circuit using two half adders.



Logic diagram of a full-adder using two half-adders.

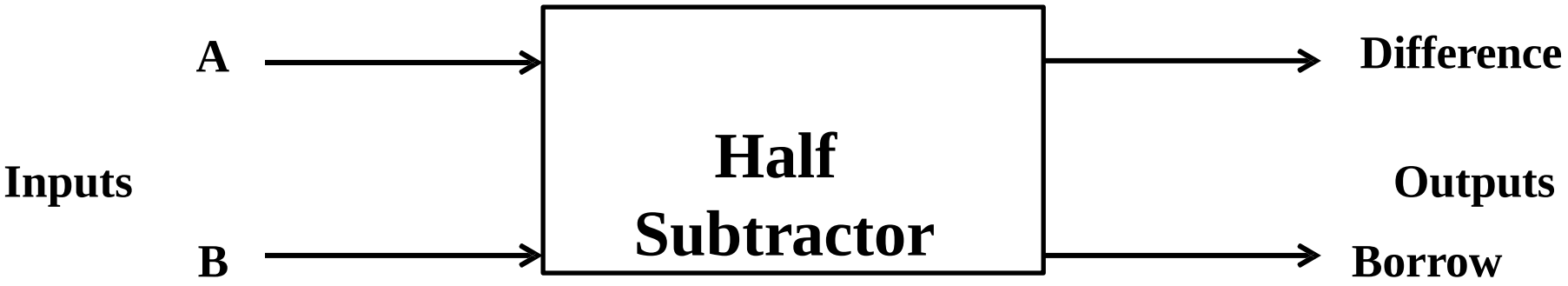
The block diagram of a full-adder using two half-adders is :



Block diagram of a full-adder using two half-adders.

Half Subtractor

- ✓ Half subtractor is a combinational logic circuit with two inputs and two outputs.
- ✓ It is a basic building block for subtraction of two single bit numbers.



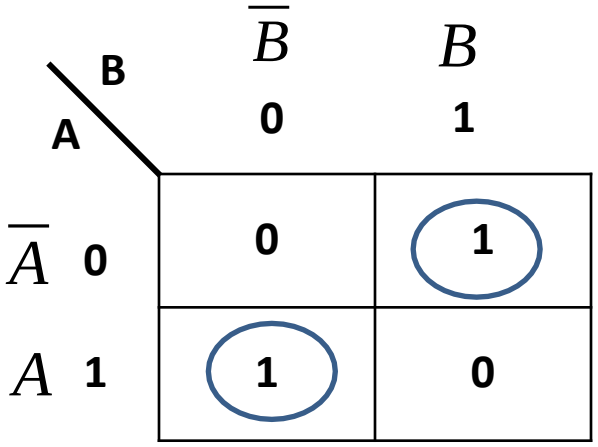
Half Subtractor

Truth Table

Input		Output	
A	B	Difference (D)	Borrow (B)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Half Subtractor

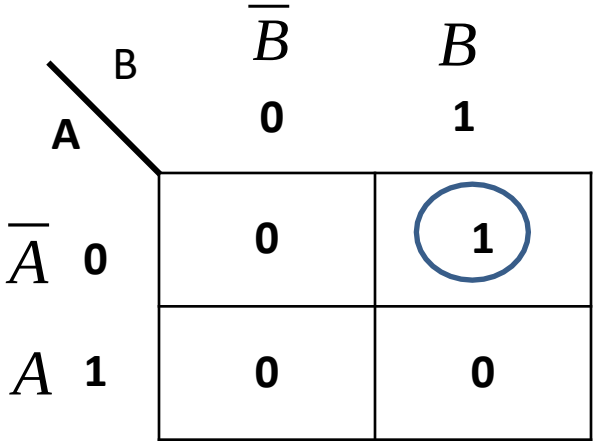
K-map for Difference Output:



$$D = \overline{A}B + A\overline{B}$$

$$D = A \oplus B$$

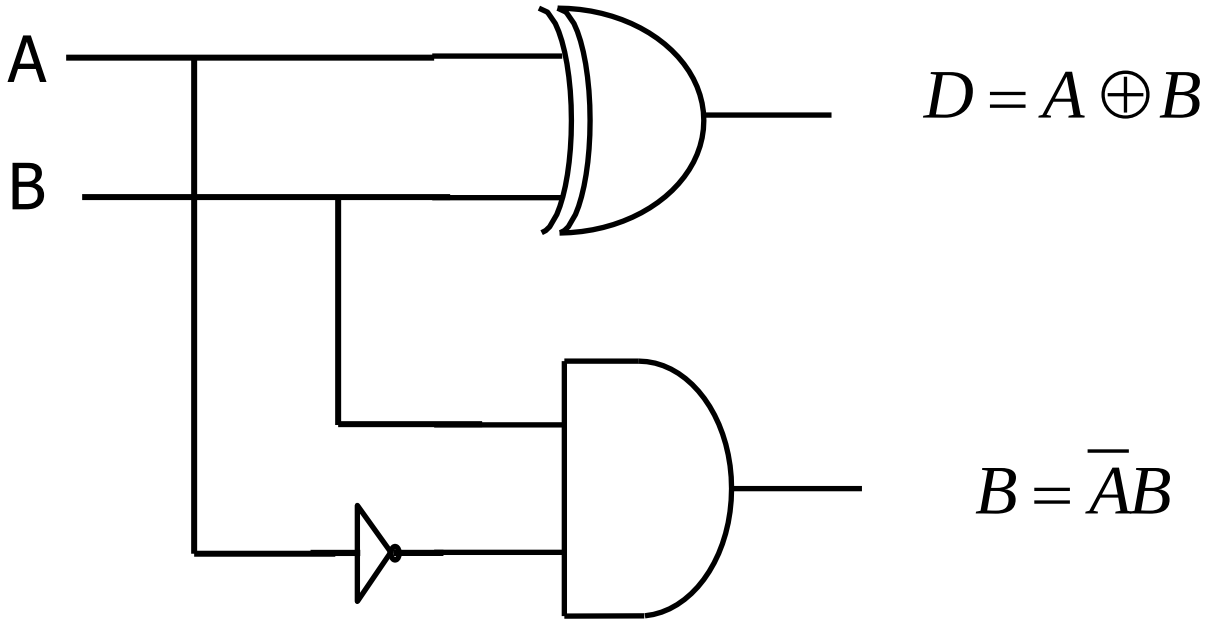
K-map for Borrow Output:



$$B = \overline{A}B$$

Half Subtractor

Logic Diagram:

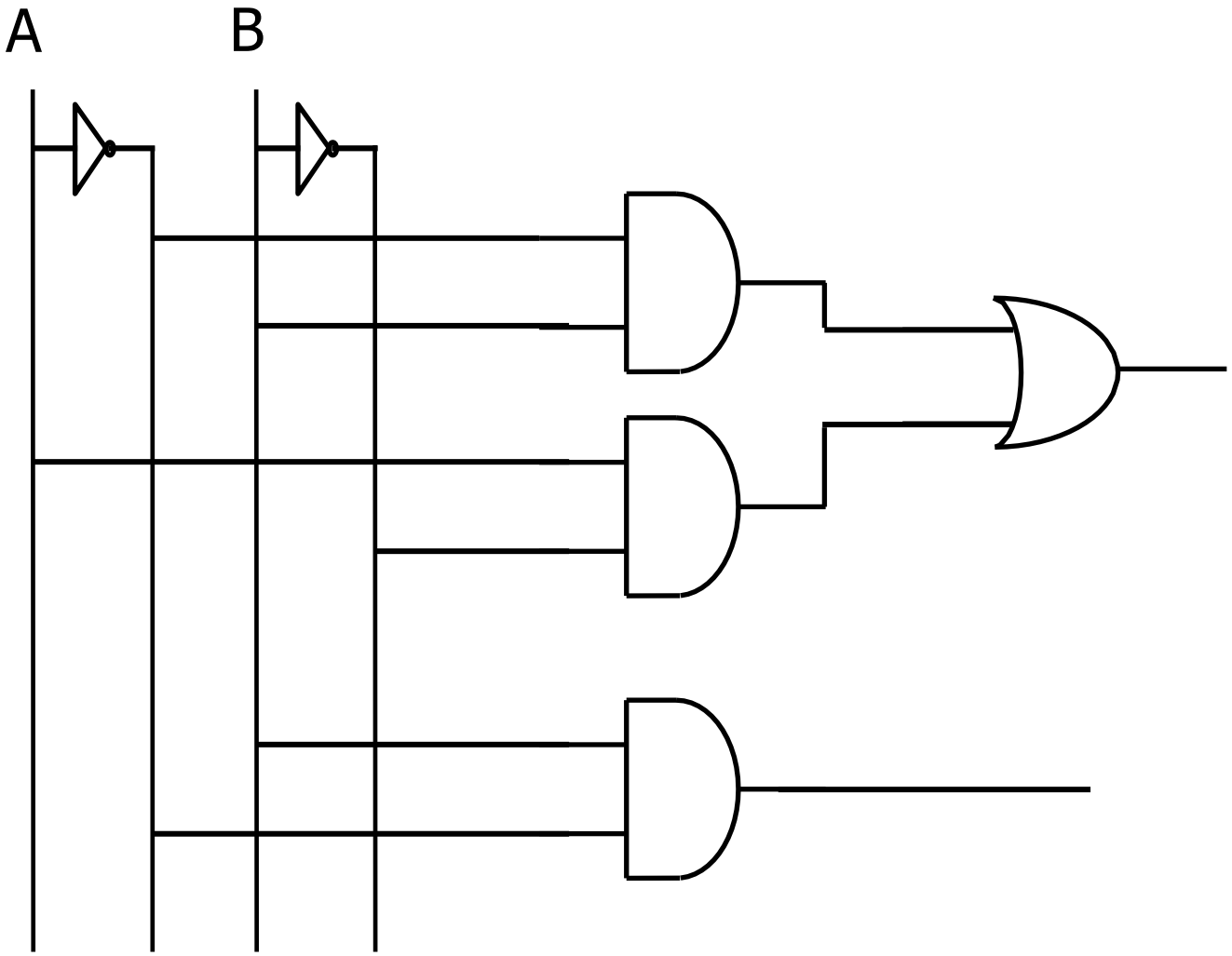


$$D = A \oplus B$$

$$B = \overline{A}B$$

Half Subtractor

Logic Diagram using Basic Gates:

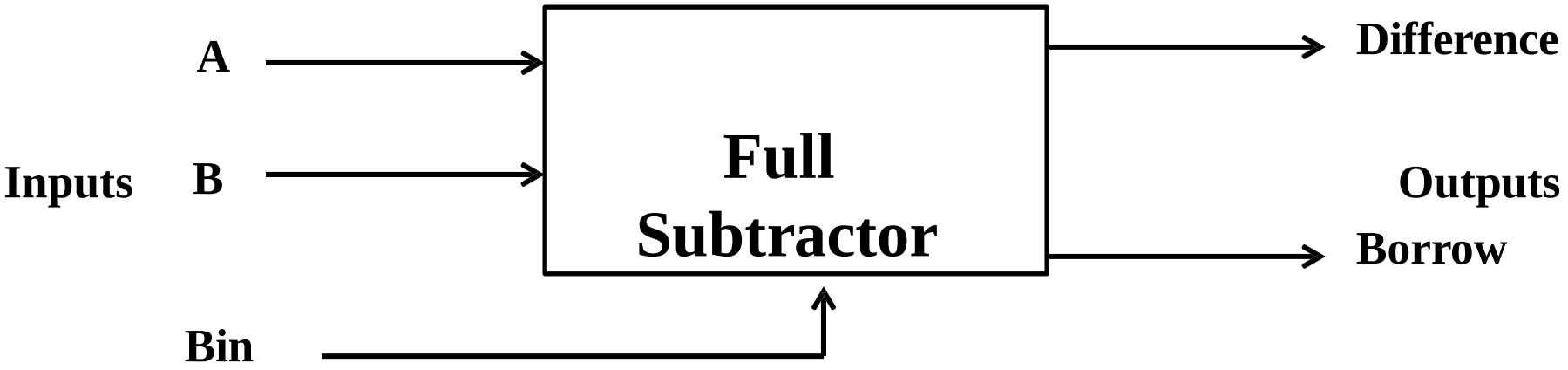


$$D = A \oplus B$$

$$B = \overline{A}B$$

Full Subtractor

- ✓Full subtractor is a combinational logic circuit with three inputs and two outputs.



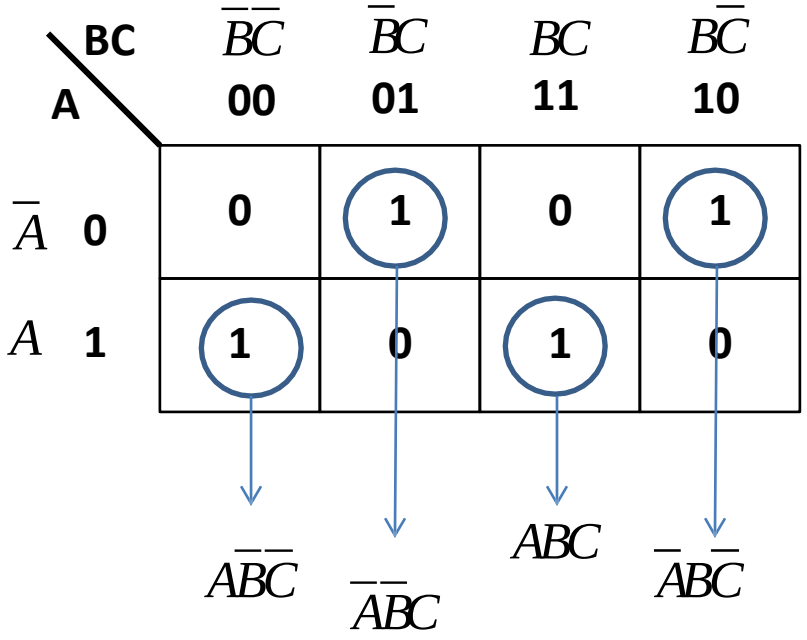
Full Subtractor

Truth Table

Inputs			Outputs	
A	B	Bin (C)	Difference (D)	Borrow (B0)
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

Full Subtractor

K-map for Difference Output:



$$D = \overline{A}\overline{B}C + \overline{A}B\overline{C} + ABC + A\overline{B}\overline{C}$$

$$D = \overline{A}\overline{B}C + ABC + \overline{A}B\overline{C} + A\overline{B}\overline{C}$$

$$D = C(\overline{A}\overline{B} + AB) + \overline{C}(\overline{A}B + A\overline{B})$$

$$Let \quad \overline{A}B + A\overline{B} = X$$

$$\therefore D = C(\overline{X}) + \overline{C}(X)$$

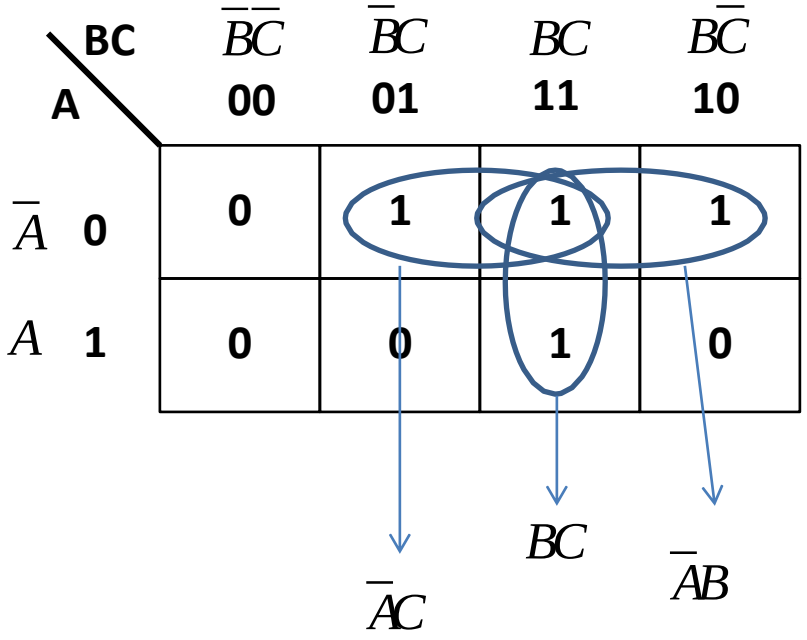
$$D = C \oplus X$$

$$Let \quad X = A \oplus B$$

$$\therefore D = C \oplus A \oplus B$$

Full Subtractor

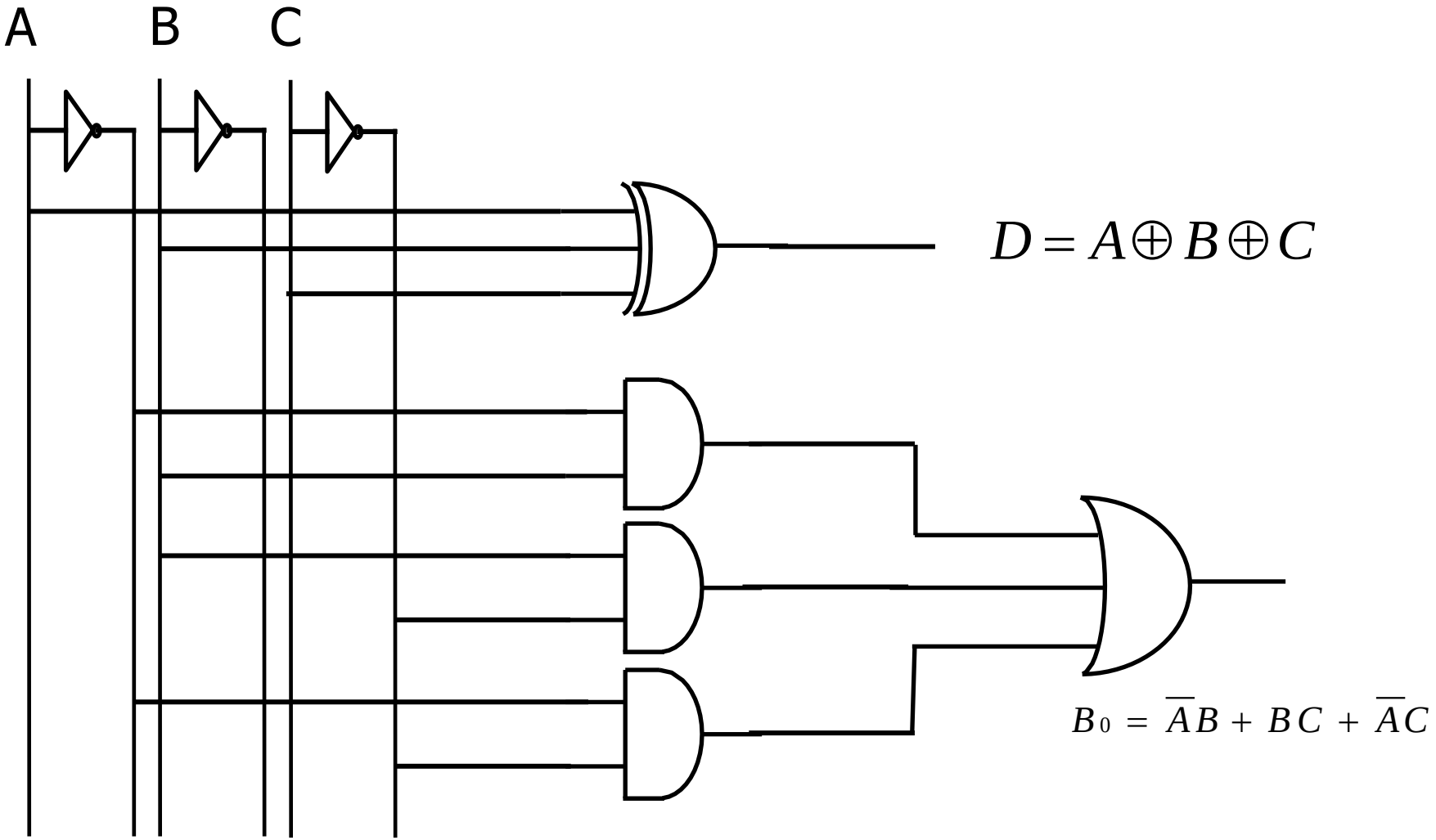
K-map for Borrow Output:



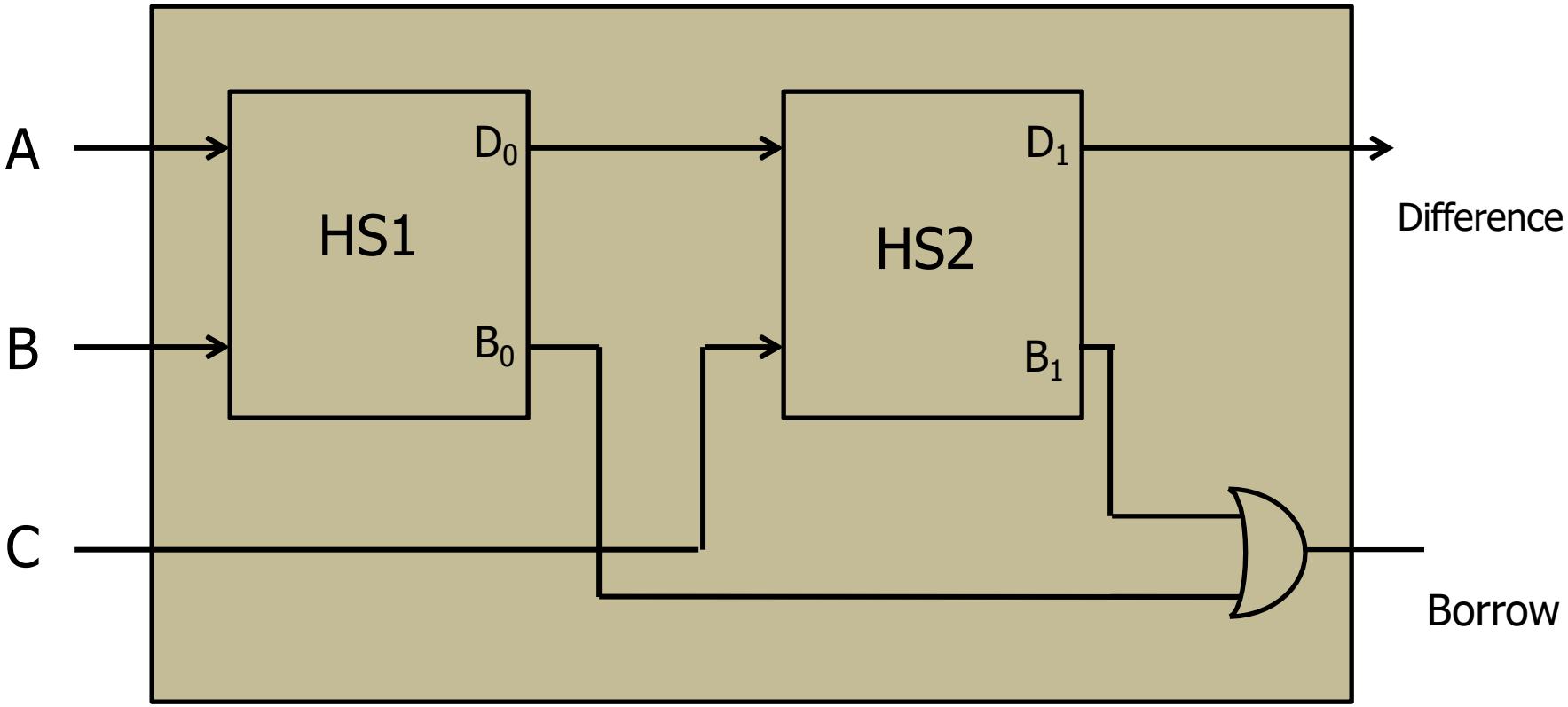
$$B_0 = \overline{A}B + BC + \overline{A}C$$

Full Subtractor

Logic Diagram:

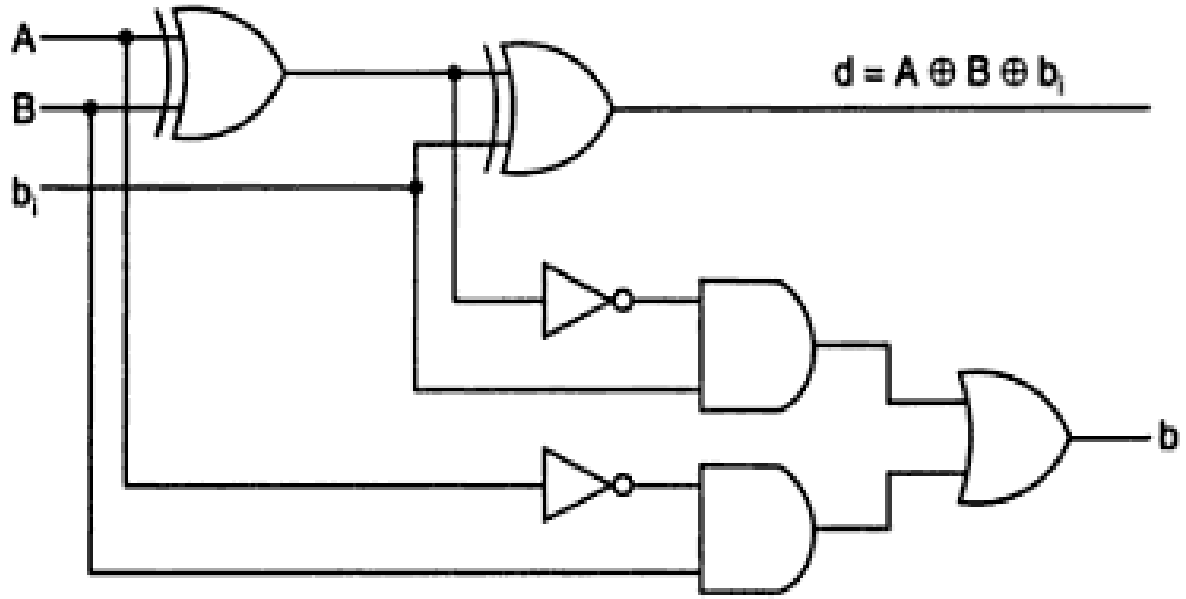


Full Subtractor using Half Subtractor



Draw a full subtractor circuit using two half subtractors.

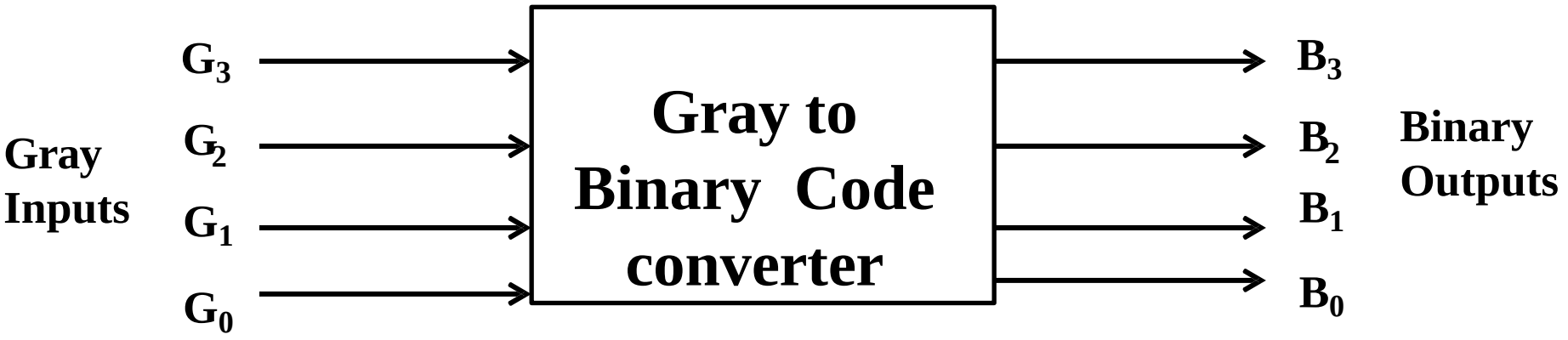
The circuit diagram is as:



Logic diagram of a full-subtractor.

Design of Gray to Binary Code Converter

Block Diagram:



Design of Gray to Binary Code Converter

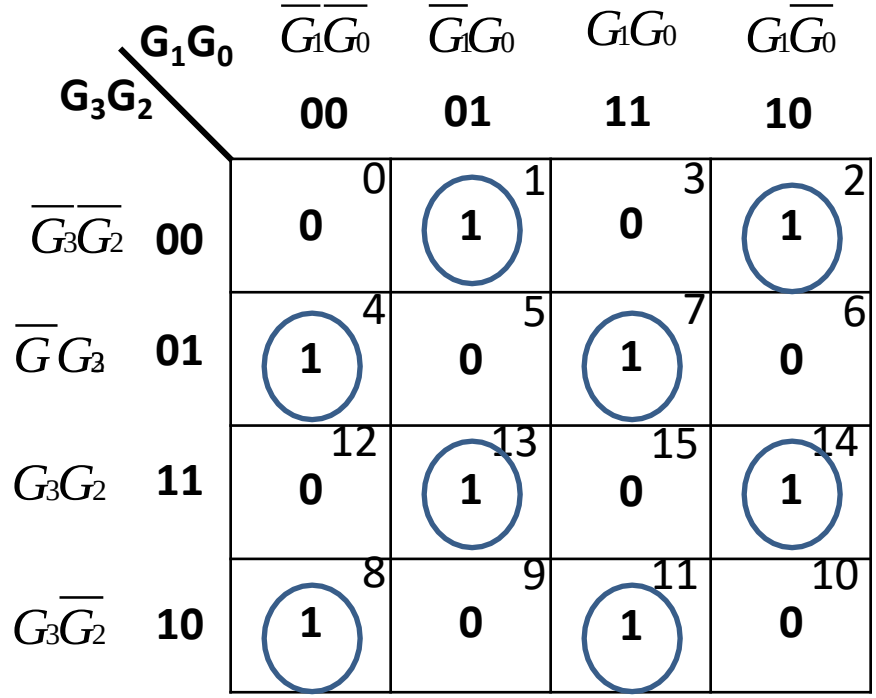
Truth Table :

Gray Inputs			Binary Outputs			
G_2	G_1	G_0	B_3	B_2	B_1	B_0
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	1	1
0	1	1	0	0	1	0
1	0	0	0	1	1	0
1	0	1	0	1	1	1
1	1	0	0	1	0	1
1	1	1	0	1	0	0

Gray Inputs				Binary Outputs			
G_3	G_2	G_1	G_0	B_3	B_2	B_1	B_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

Design of Gray to Binary Code Converter

K-map for B_0:

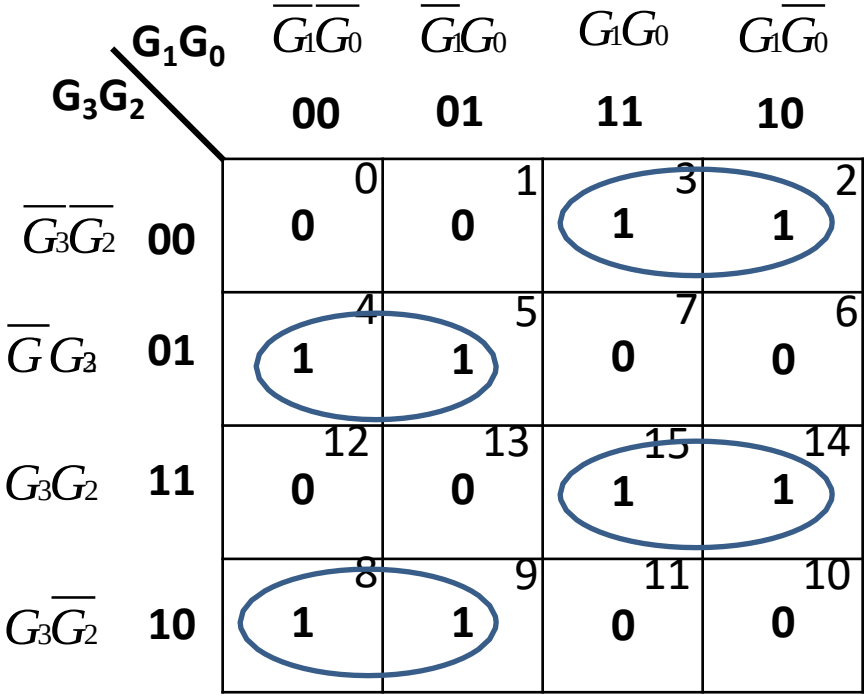


$$B_0 = \overline{G_3}\,\overline{G_2}\,\overline{G_1}G_0 + \overline{G_3}\,\overline{G_2}G_1\overline{G_0} + \overline{G_3}G_2\overline{G_1}\,\overline{G_0} + \overline{G_3}G_2G_1G_0$$
$$+ G_3\overline{G_2}\,\overline{G_1}G_0 + G_3\overline{G_2}G_1G_0 + G_3G_2\overline{G_1}G_0 + G_3G_2G_1\overline{G_0}$$

$$B_0 = G_3 \oplus G_2 \oplus G_1 \oplus G_0$$

Design of Gray to Binary Code Converter

K-map for B_1:

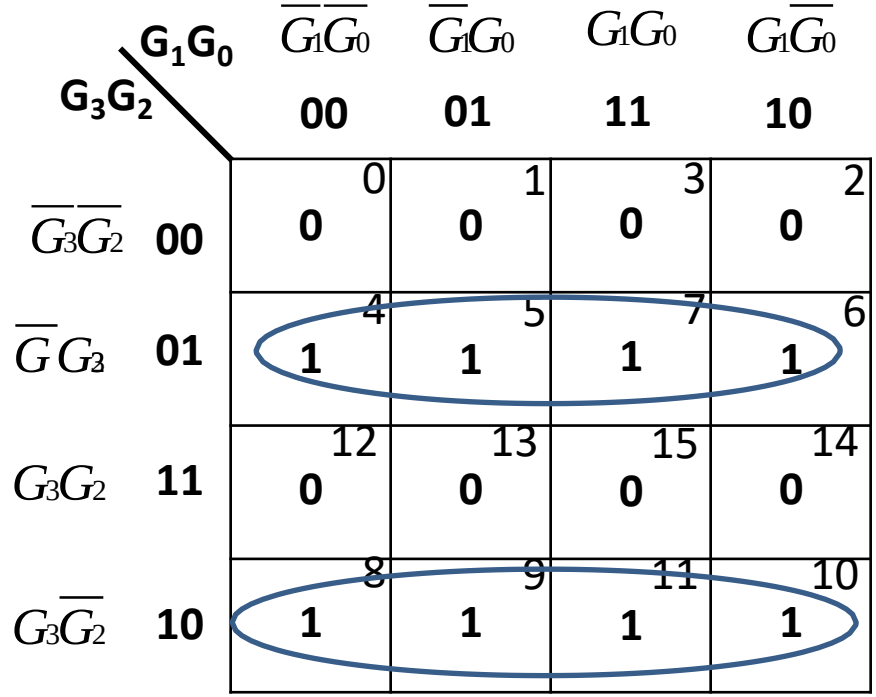


$$B_1 = \overline{G_3}\,\overline{G_2}G_1 + \overline{G_3}G_2\overline{G_1} + G_3G_2G_1 + G_3\overline{G_2}\,\overline{G_1}$$

$$B_1 = G_3 \oplus G_2 \oplus G_1$$

Design of Gray to Binary Code Converter

K-map for B_2:

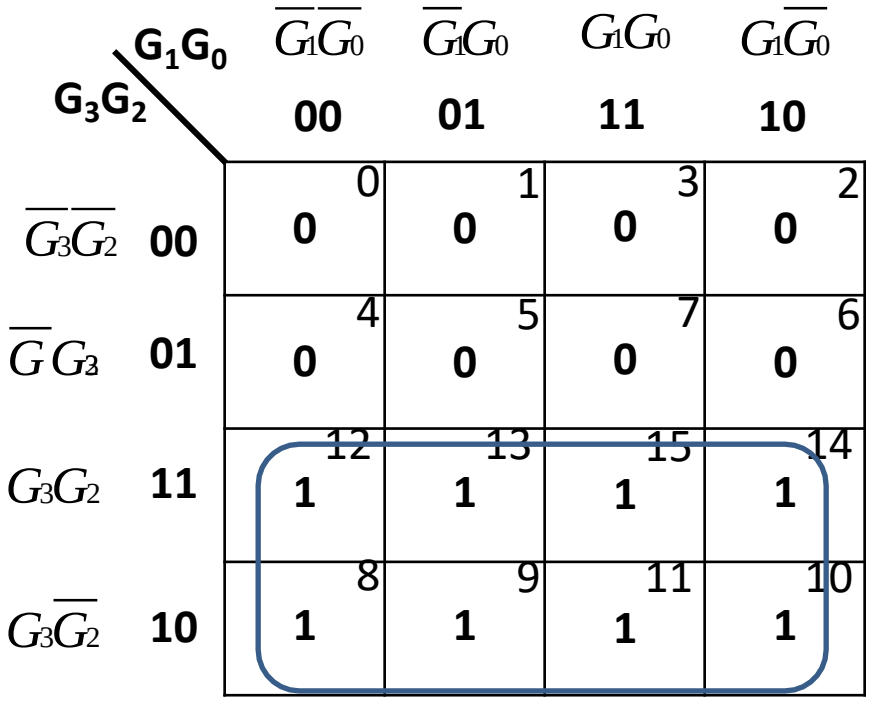


$$B_2 = \overline{G_3}G_2 + G_3\overline{G_2}$$

$$B_1 = G_3 \oplus G_2$$

Design of Gray to Binary Code Converter

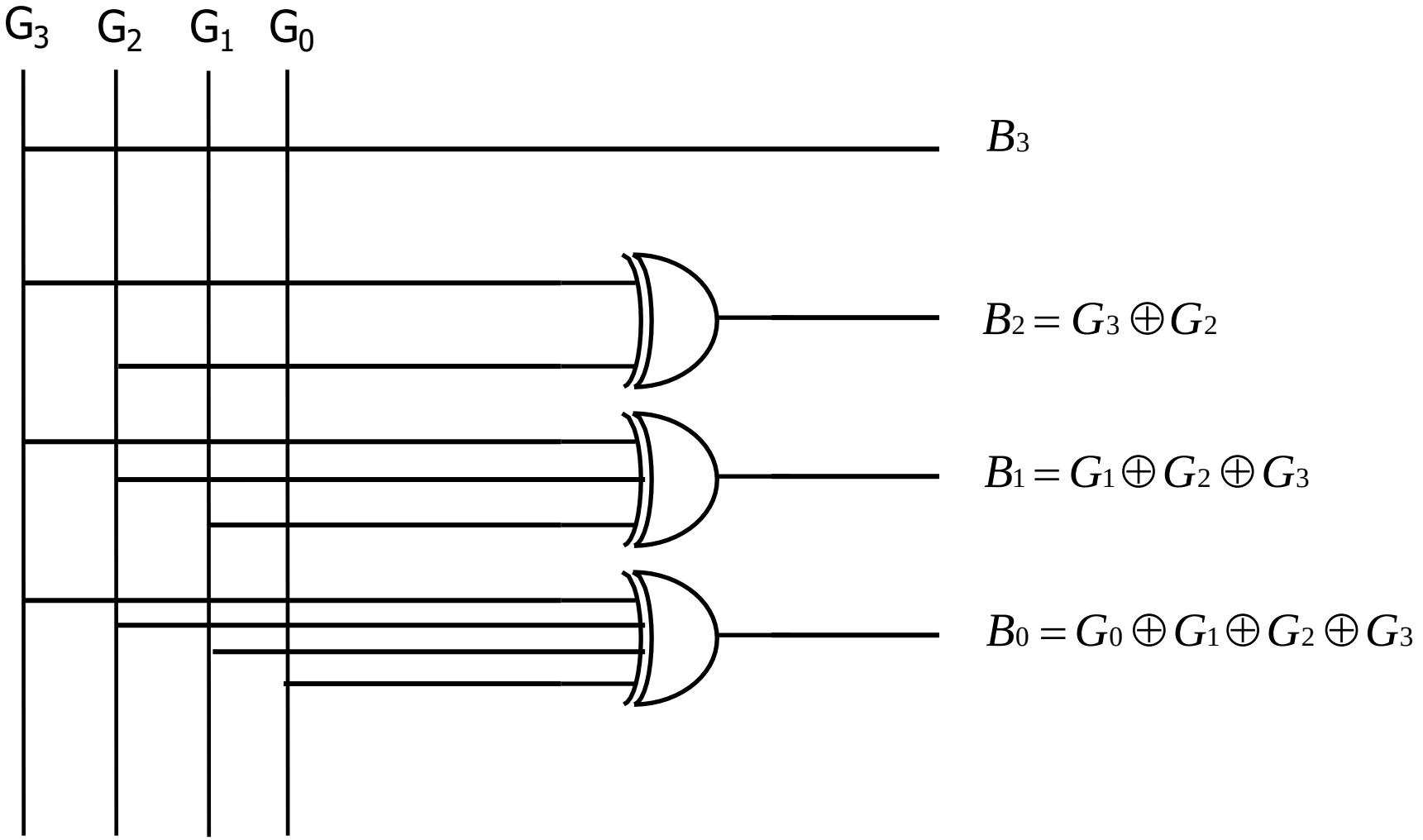
K-map for B_3:



$$B_3 = G_3$$

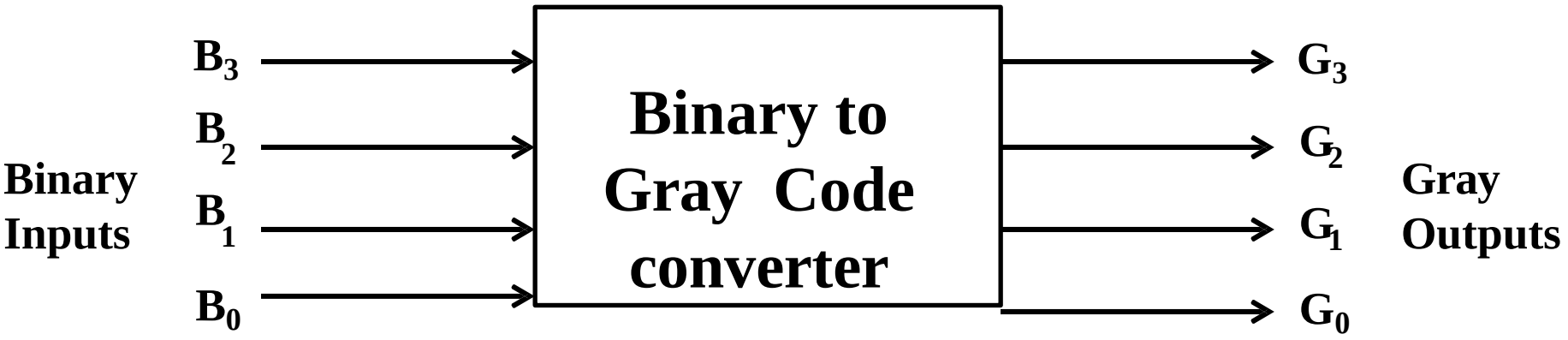
Design of Gray to Binary Code Converter

Logic Diagram:



Design of Binary to Gray Code Converter

Block Diagram:



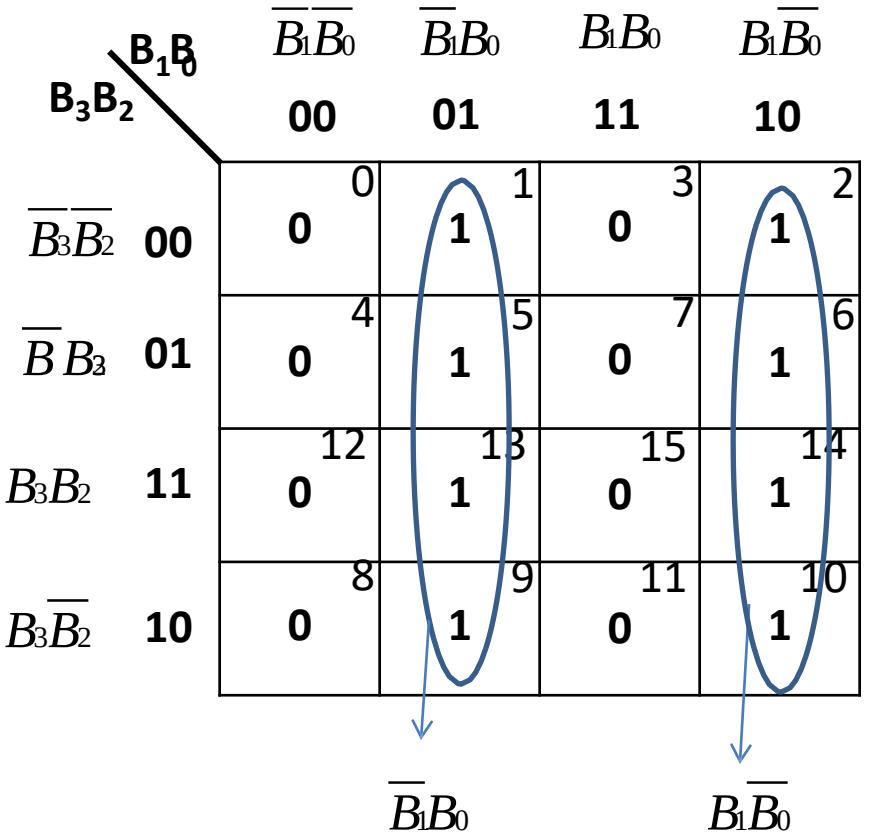
Design of Binary to Gray Code Converter

Truth Table :

Binary Inputs				Gray Outputs			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
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

Design of Binary to Gray Code Converter

K-map for G_0:

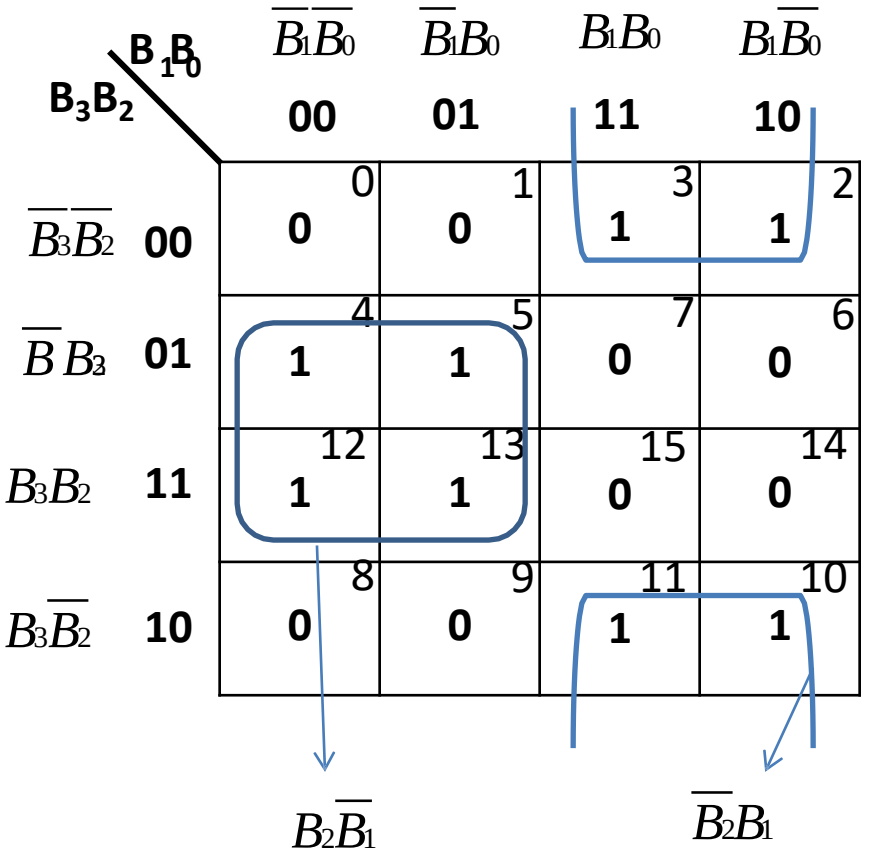


$$G_0 = \overline{B_1}B_0 + B_1\overline{B_0}$$

$$\therefore G_0 = B_0 \oplus B_1$$

Design of Binary to Gray Code Converter

K-map for G_1:

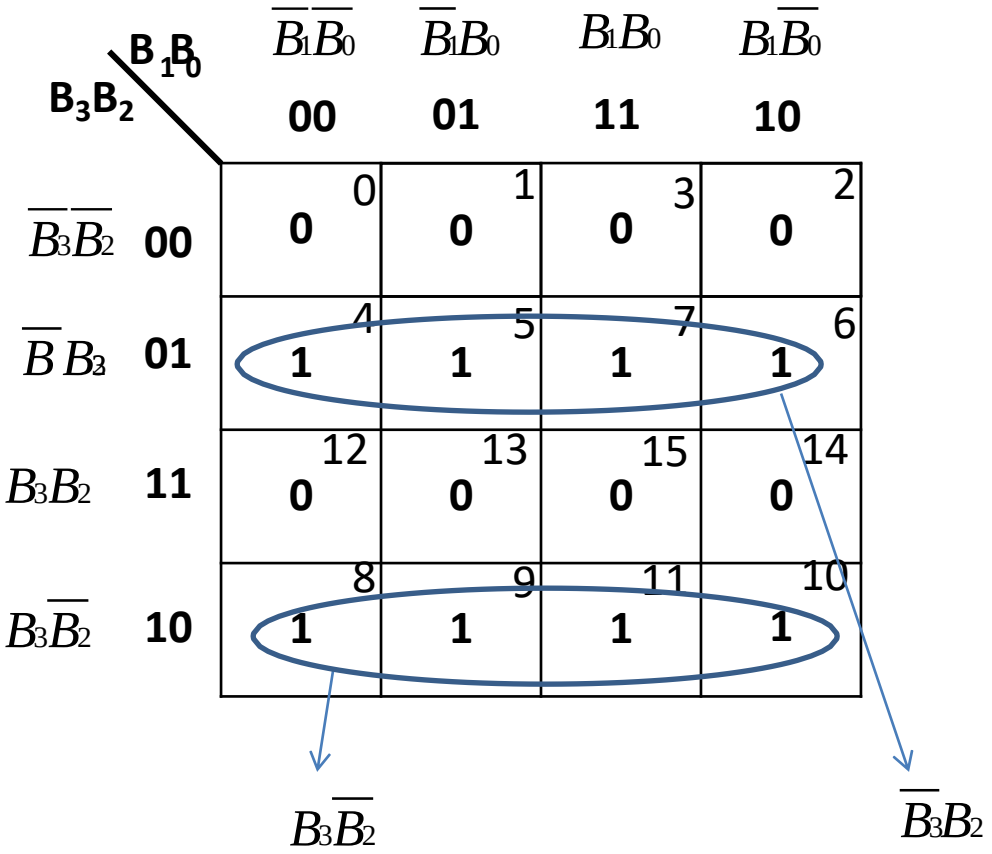


$$G_1 = \overline{B_2}B_1 + B_2\overline{B_1}$$

$$\therefore G_1 = B_2 \oplus B_1$$

Design of Binary to Gray Code Converter

K-map for G_2:

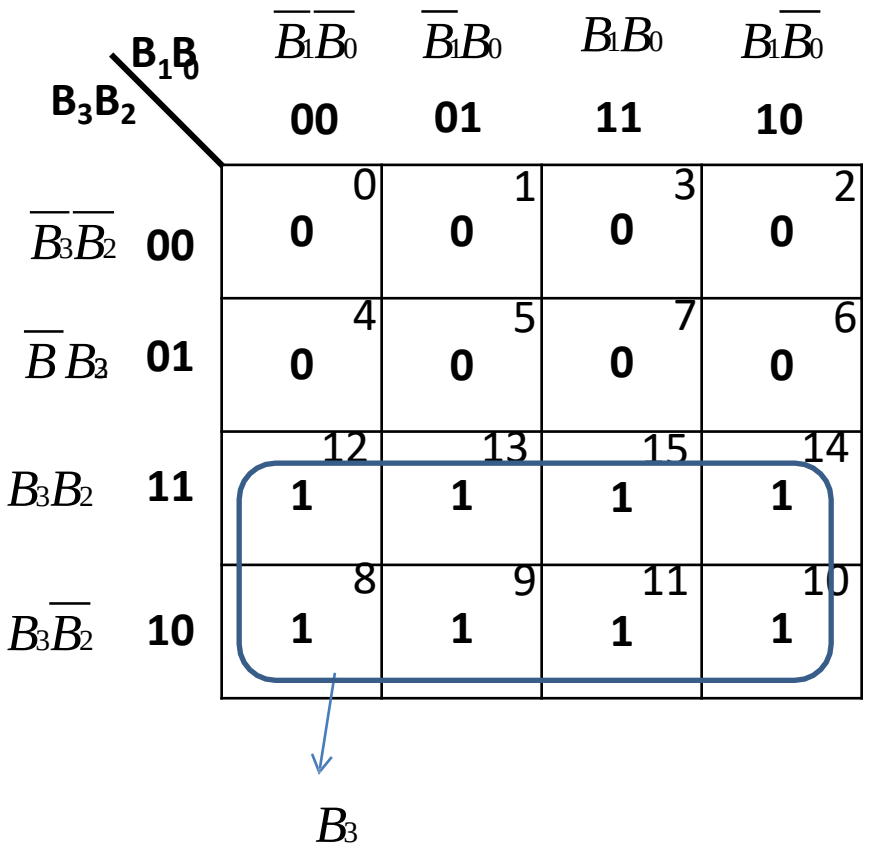


$$G_2 = \overline{B_3}B_2 + B_3\overline{B_2}$$

$$\therefore G_2 = B_3 \oplus B_2$$

Design of Binary to Gray Code Converter

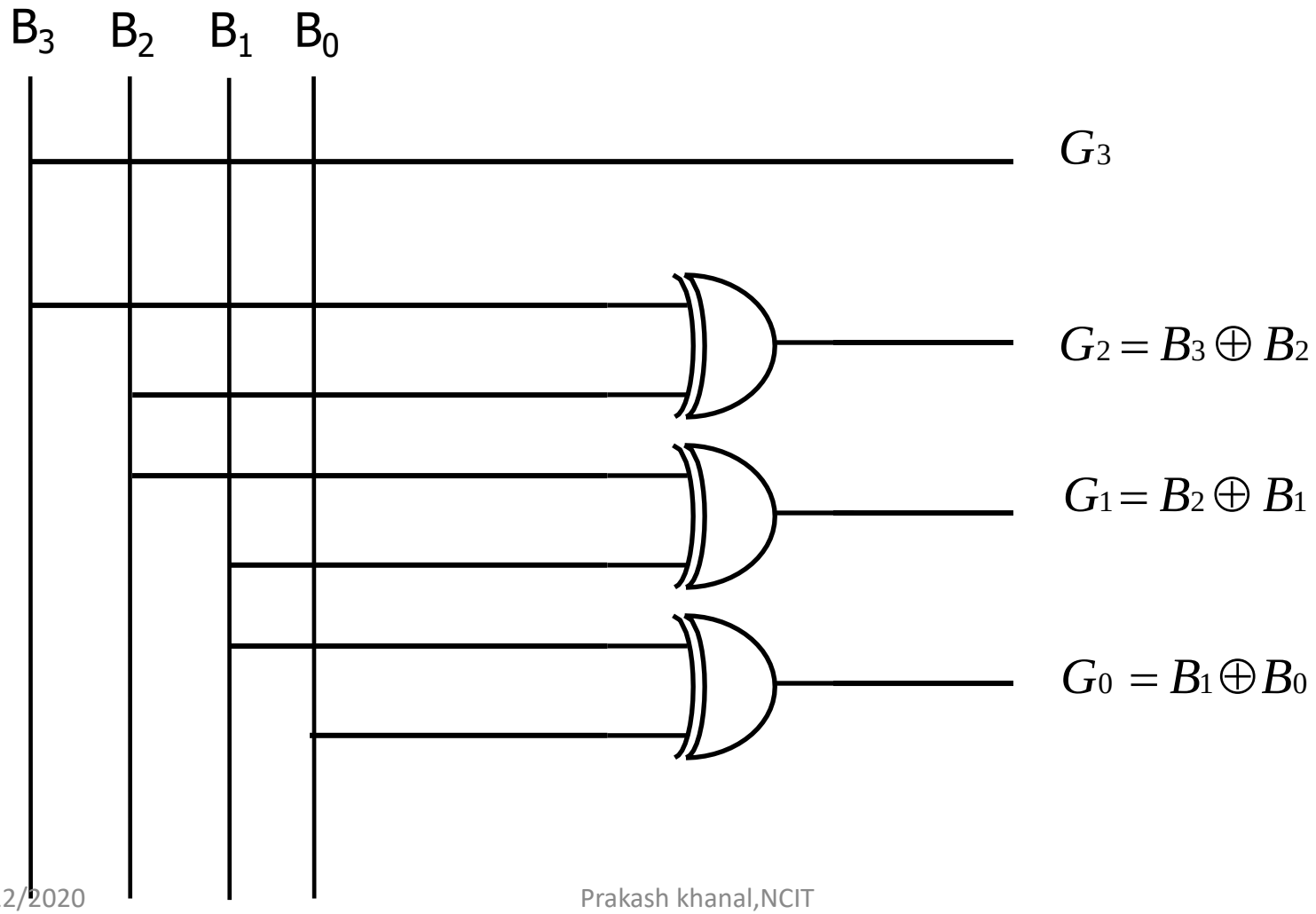
K-map for G_3:



$$G_3 = B_3$$

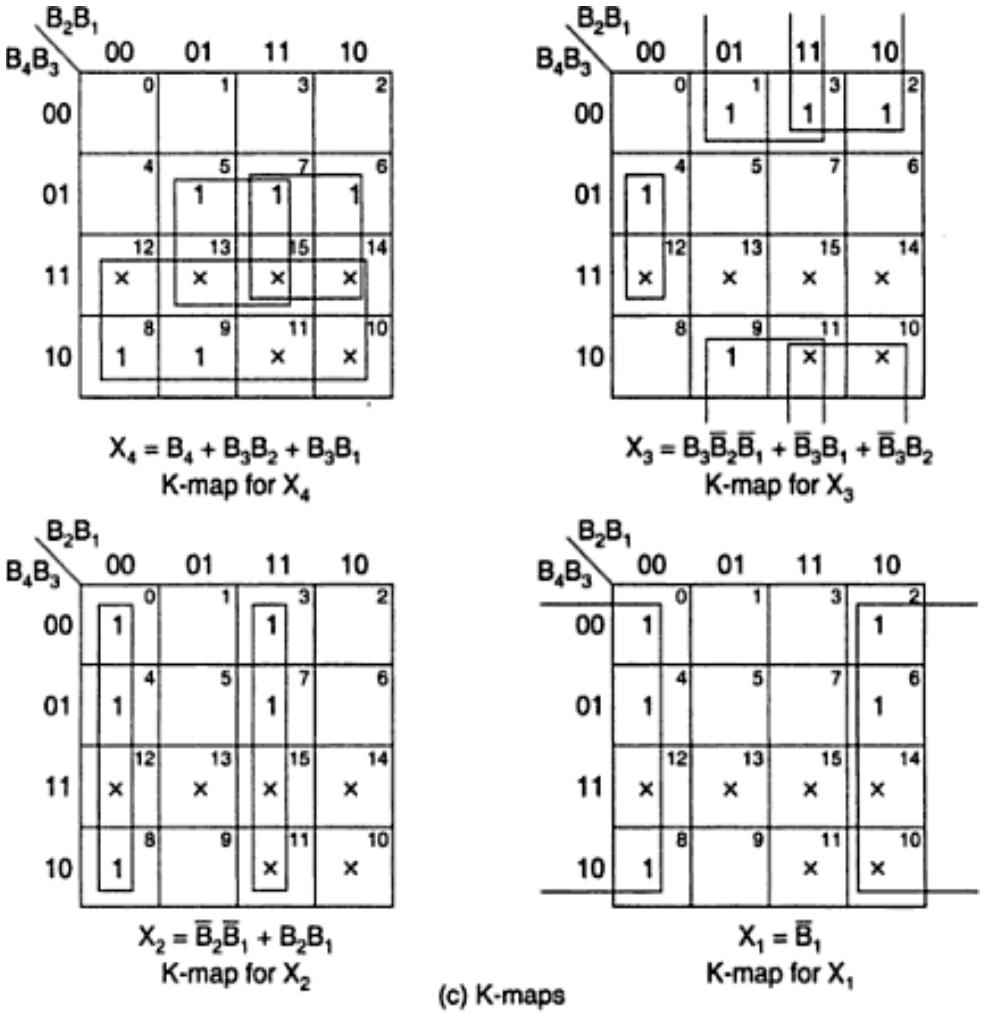
Design of Binary to Gray Code Converter

Logic Diagram:



Design of a 4-bit BCD to XS-3 code converter:

8421 code				XS-3 code			
B_4	B_3	B_2	B_1	X_4	X_3	X_2	X_1
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



(c) K-maps

4-bit BCD-to-XS-3 code converter.

The o/p functions and logic diagram is as:

The minimal expressions are

$$X_4 = B_4 + B_3B_2 + B_3B_1$$

$$X_3 = B_3\bar{B}_2\bar{B}_1 + \bar{B}_3B_1 + \bar{B}_3B_2$$

$$X_2 = \bar{B}_2\bar{B}_1 + B_2B_1$$

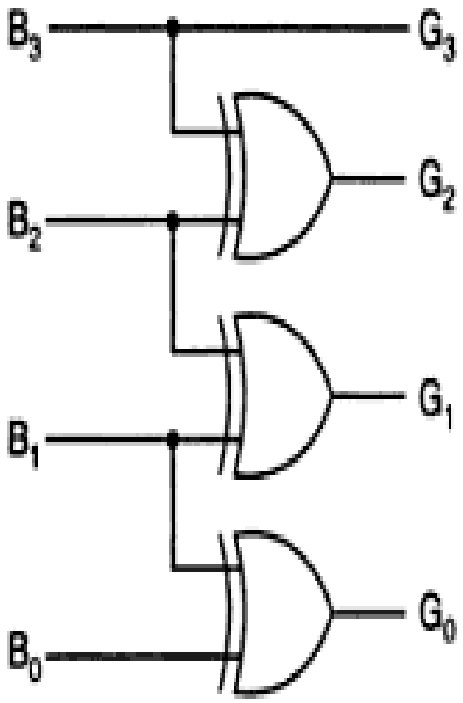
$$X_1 = \bar{B}_1$$

- Draw the logic diagram yourself.

Design of a BCD to gray code converter:

BCD code				Gray code			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
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

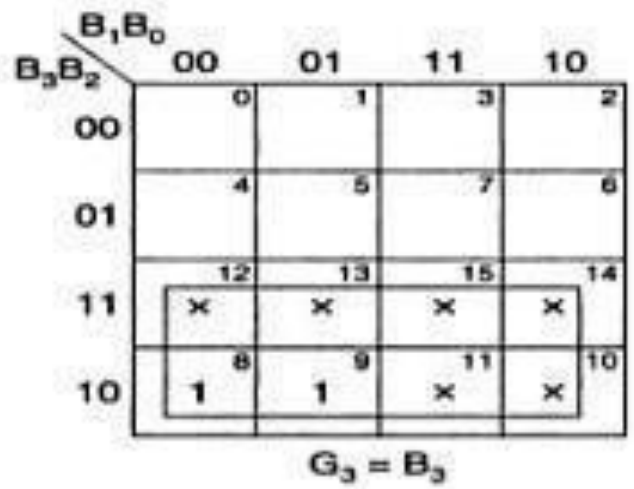
(a) BCD-to-Gray code conversion table



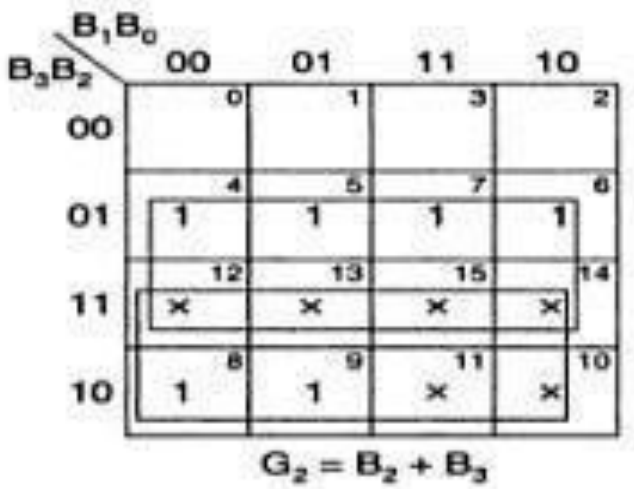
(b) Logic diagram

BCD-to-Gray code converter.

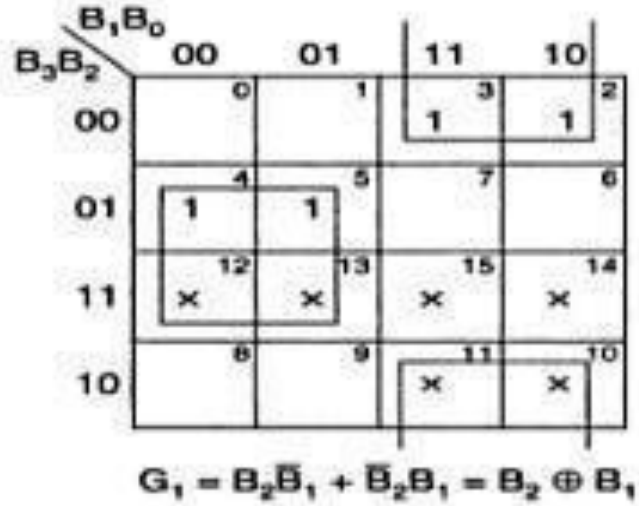
Using K-map,



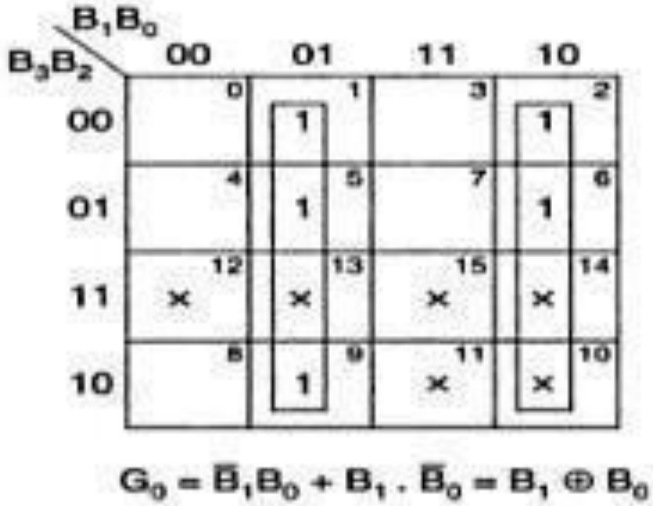
$G_3 = B_3$



$G_2 = B_2 + B_3$



$G_1 = B_2\bar{B}_1 + \bar{B}_2B_1 = B_2 \oplus B_1$



$G_0 = \bar{B}_1B_0 + B_1 \cdot \bar{B}_0 = B_1 \oplus B_0$

K-maps for a BCD-to-Gray code converter.

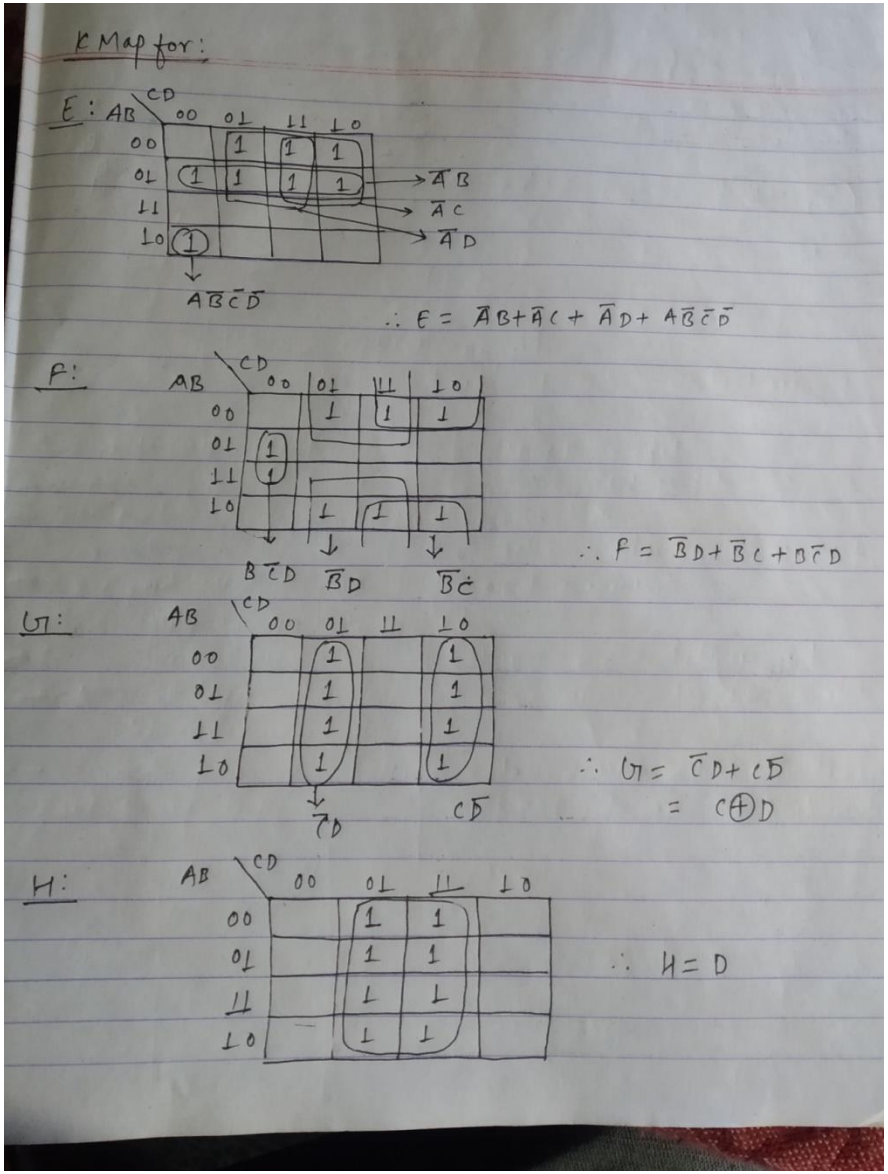
Design of a Combinational circuit to produce the 2's complement of a 4-bit binary number:

Input				Output			
A	B	C	D	E	F	G	H
0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

(a) Conversion table

Conversion table and K-maps for the circuit

- Simplify the o/p and draw the logic diagram yourself.


K Map for:
E: AB CD 00 01 11 10
$\bar{A}B$
$\bar{A}C$
$\bar{A}D$
$A\bar{B}\bar{C}\bar{D}$
$\therefore E = \bar{A}B + \bar{A}C + \bar{A}D + A\bar{B}\bar{C}\bar{D}$
F: AB CD 00 01 11 10
$B\bar{C}D$
$\bar{B}D$
$\bar{B}\bar{C}$
$\therefore F = \bar{B}D + \bar{B}C + B\bar{C}D$
G: AB CD 00 01 11 10
$\bar{C}D$
$C\bar{D}$
$\therefore G = \bar{C}D + C\bar{D}$
$= C \oplus D$
H: AB CD 00 01 11 10
$\therefore H = D$

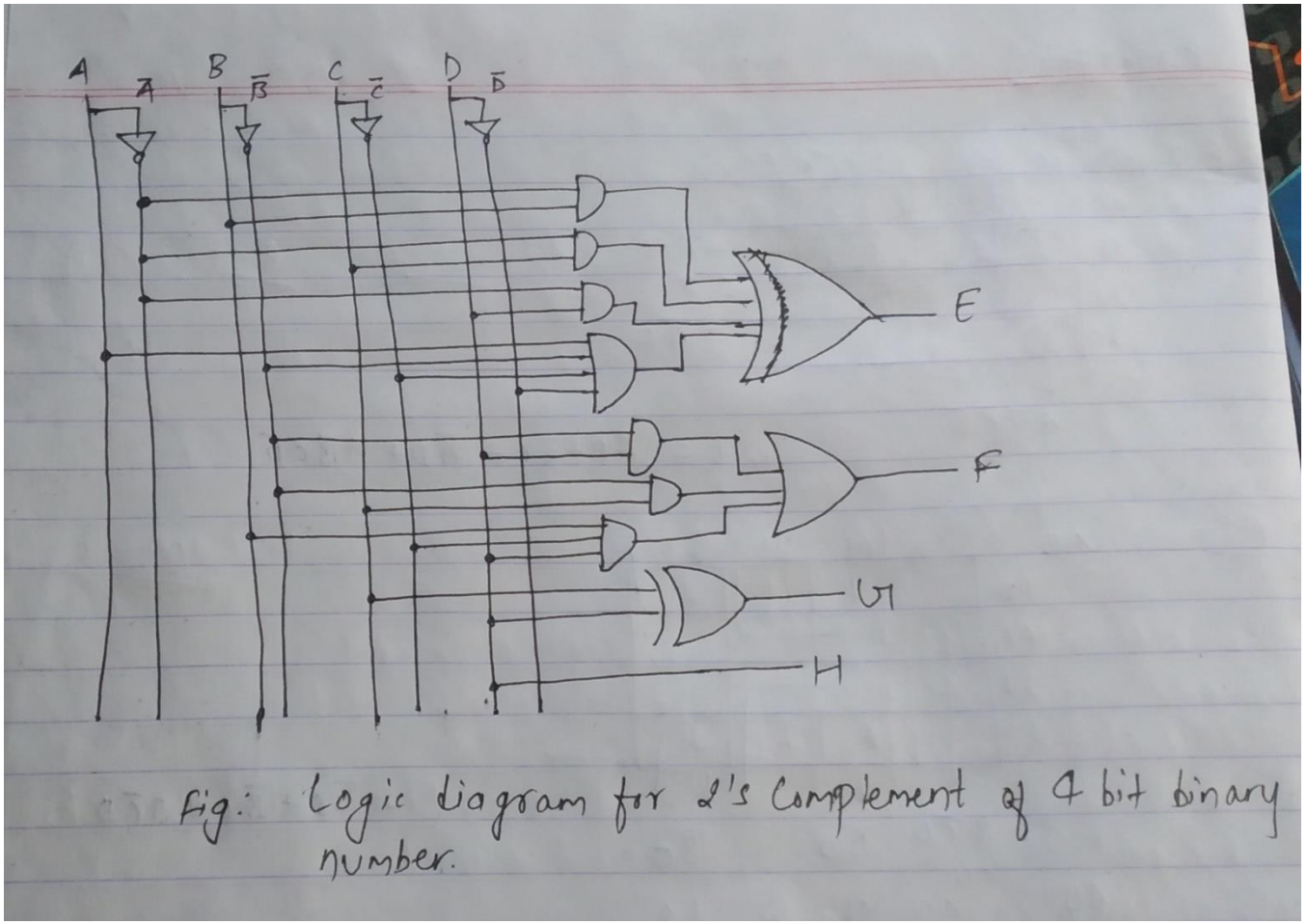


Fig: Logic diagram for 2's Complement of 4 bit binary number.

Parity method for error detection and correction:

- Parity bit is an extra bit included with a binary message to make the number of 1's either odd or even.
- The message including the parity bit, is transmitted and checked at the receiving end for errors.
- An error is detected if the checked parity does not correspond to the one transmitted.
- The circuit that generates the parity bit in the transmitter is called a parity generator, the circuit that checks the parity in the receiver is called a parity checker.

Design a circuit for parity generator for 3 bit message with even and odd parity:

X	y	z	Odd Parity bit generated(p)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Simplify the parity using k map and design the circuit.

Design a circuit for odd parity check:

Four-bits received				Parity-error check
x	y	z	P	C
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	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Simplify the k map for output and draw the logic daigram.

Four-bits received				Parity-error check
x	y	z	P	C
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	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1