



CE232 DIGITAL SYSTEM

Topic 7. Combinational Logic Circuit

Prepared by Nabila Husna Shabrina

Contact : nabila.husna@umn.ac.id

Subtopic

**7.1 Combinational
Circuits**

**7.2 Implementation
Procedures**

7.3 Adder

7.4 Subtractor

**7.5 Code
Conversion**



The background features several overlapping geometric shapes, primarily diamonds and parallelograms, in teal, yellow, and green colors. These shapes are arranged in a way that creates a sense of depth and movement, with some shapes appearing to be layered on top of others. The colors are vibrant and the shapes are sharp, contributing to a modern and abstract aesthetic.

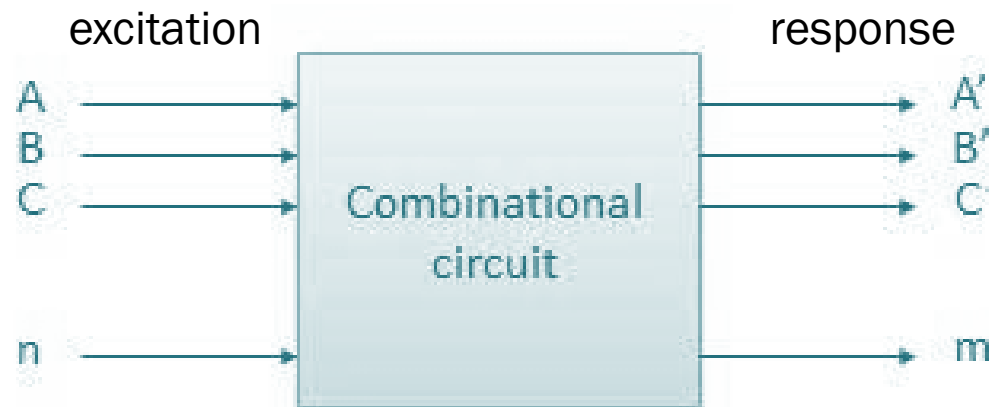
“

7.1 Combinational Circuits

7.1 Combinational Circuits

Intro to Combinational Circuits

- A circuits which output is dependent upon combination of input variables
- The circuits **does not use any memory**
- It can have n number of inputs and m number of outputs



7.1 Combinational Circuits



Example of Combinational circuits

- Adders and subtractors
- Decoders
- MUX
- Code Inverters
- Comparators
- Read Only Memory, Programmable Logic Array, Programmable Array Logic

The background features several large, overlapping geometric shapes, primarily diamonds and triangles, in teal, yellow, and green colors. These shapes are positioned in the top right and bottom left corners, creating a modern, abstract design.

“

7.2 Implementation Procedures

7.2 Implementation Procedures



Procedures

- Observe the problem definition
- Determine the required input and output variables
- Assign letters symbols to the input variables
- Make truth table that defines required relationship
- Determine the simplified Boolean expression using K-MAP
- Draw Logic Diagram

7.2 Implementation Procedures

Example.

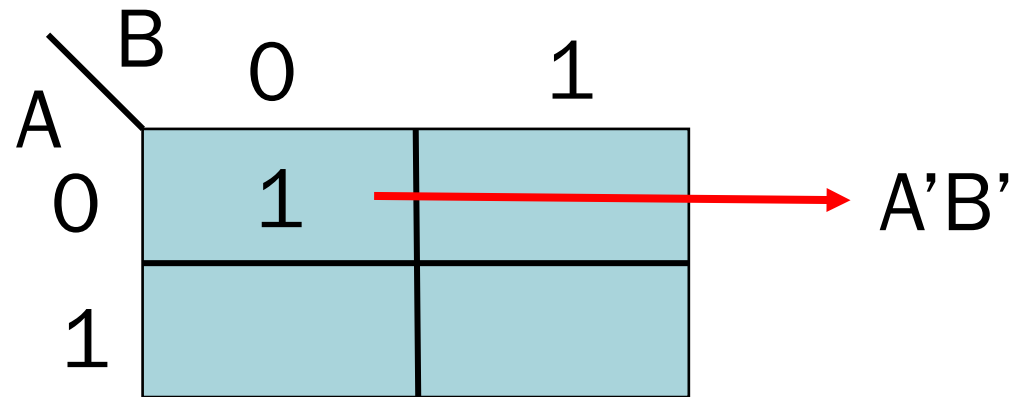
Design a combinational circuits with two input which produce output as logic 0 when any one input is 1

Inputs A, B output Y

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

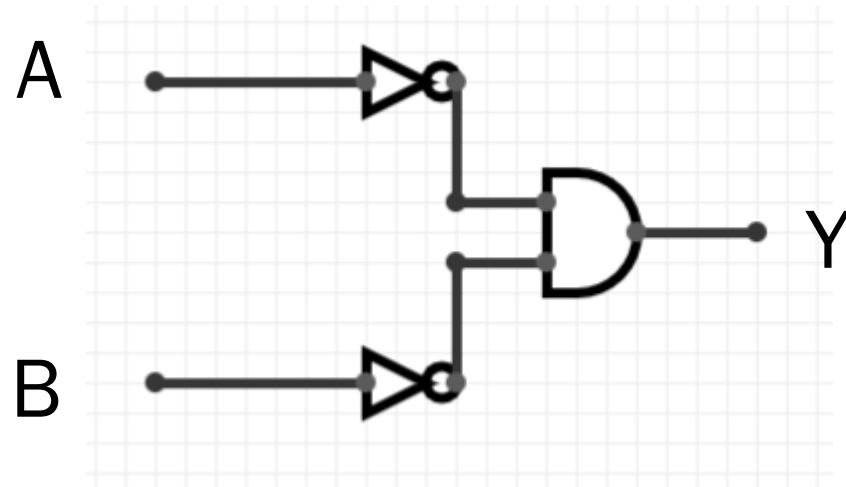
7.2 Implementation Procedures

Consider it as SOP form, then consider the '1' (if you choose POS form, consider '0')



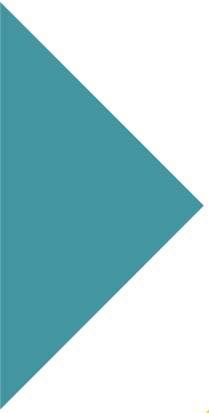
7.2 Implementation Procedures

Draw the logic diagram



“

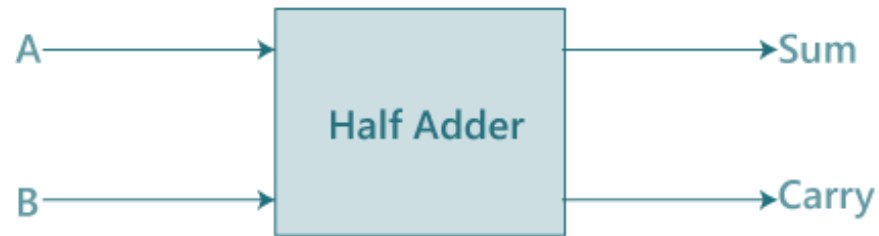
7.3 Adder



7.3 Adder

Half Adder

- Half adder is a combinational logic circuit designed to add two single bit numbers
- It contains two inputs and two outputs



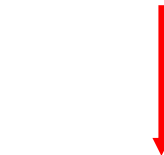
| Inputs | | Outputs | |
|--------|---|---------|-------|
| A | B | Sum | Carry |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

7.3 Adder

K-MAP for Sum

| A \ B | 0 | 1 |
|-------|---|---|
| 0 | | 1 |
| 1 | 1 | |

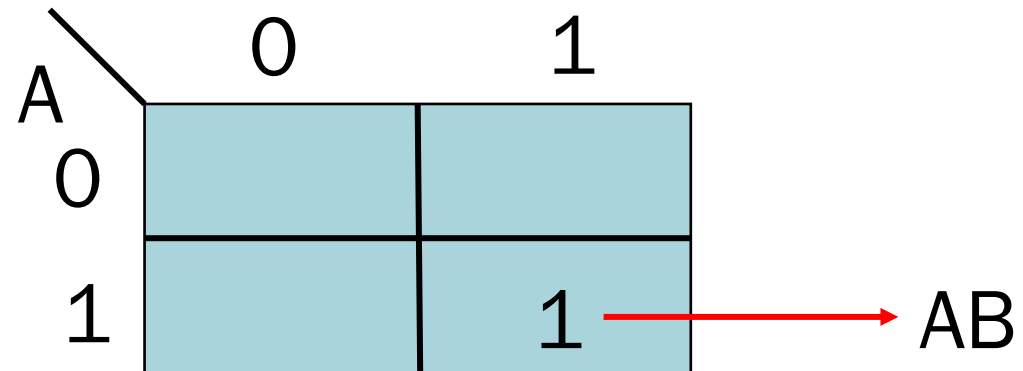
$$A'B + AB'$$



$$A \text{ XOR } B$$

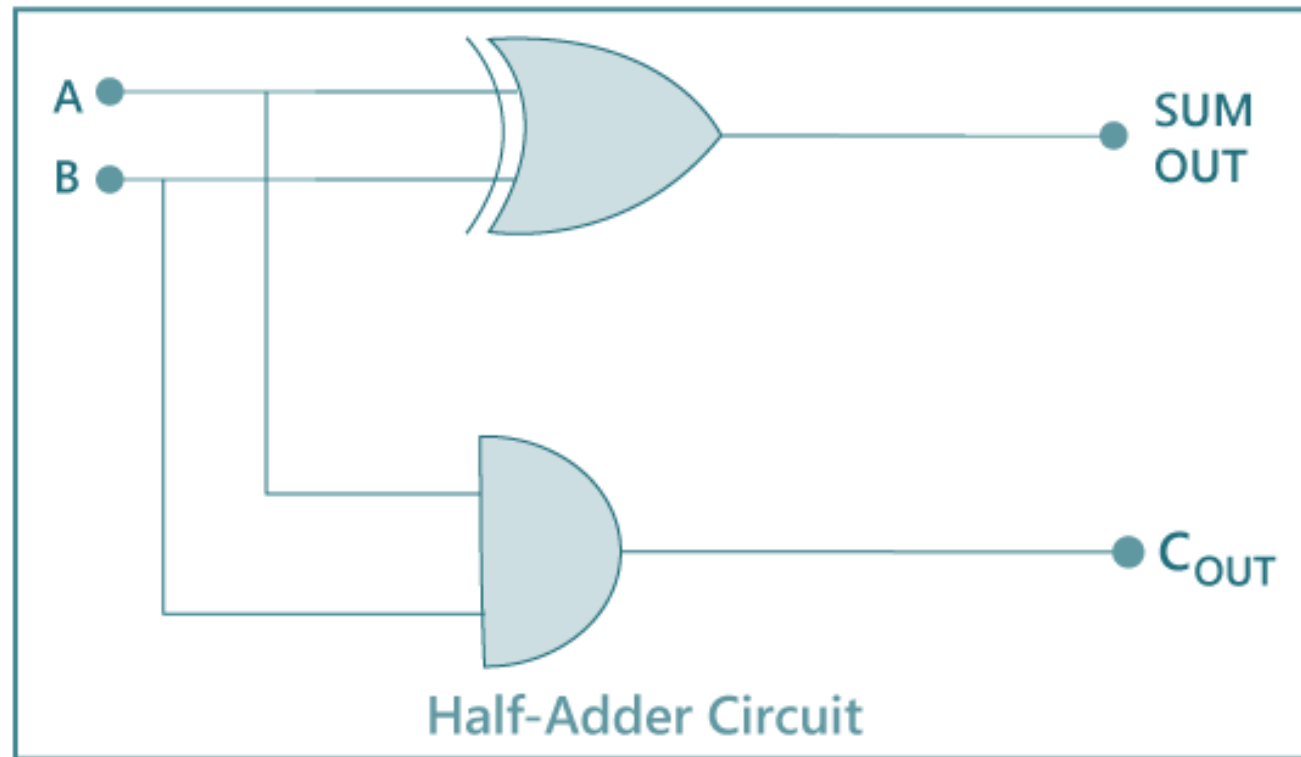
7.3 Adder

K-MAP for Carry



7.3 Adder

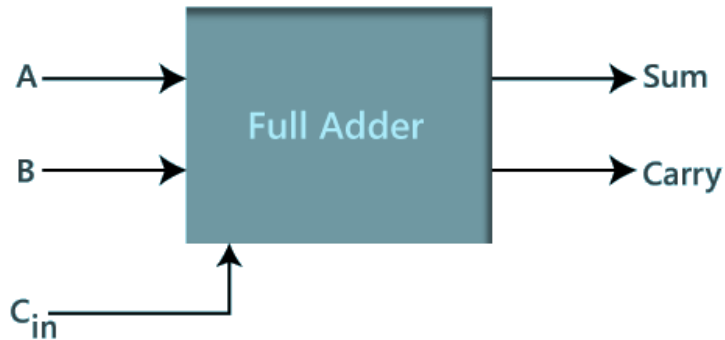
Half adder circuit



7.3 Adder

Full Adder

- Full adder is arithmetic logic circuit designed to add single bit numbers with a carry



| Inputs | | | Outputs | |
|--------|---|-----------------|---------|-------|
| A | B | C _{in} | Sum | Carry |
| 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 |

7.3 Adder

K-MAP for Sum

| | | BC | | | |
|---|---|----|----|----|----|
| | | 00 | 01 | 11 | 10 |
| A | 0 | | 1 | | 1 |
| | 1 | 1 | | 1 | |

$A \oplus B \oplus C$

7.3 Adder

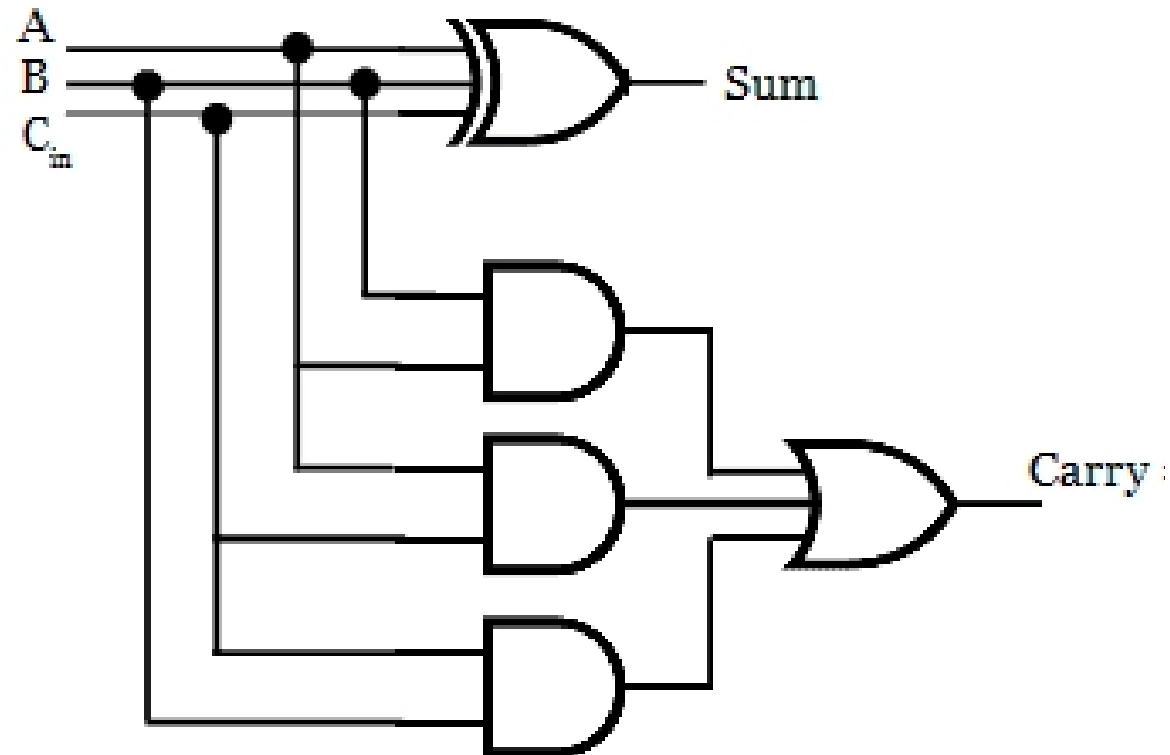
K-MAP for Carry

| BC | | 00 | 01 | 11 | 10 |
|----|---|----|----|----|----|
| A | 0 | | | 1 | |
| | 1 | | 1 | 1 | 1 |

$$AC + BC + AB$$

7.3 Adder

Full adder circuit



The background features several overlapping geometric shapes, primarily diamonds and parallelograms, in teal, yellow, and green colors. These shapes are arranged in a way that creates a sense of depth and movement, with some shapes appearing to be layered on top of others. The colors are vibrant and the shapes are sharp, contributing to a modern and abstract aesthetic.

“

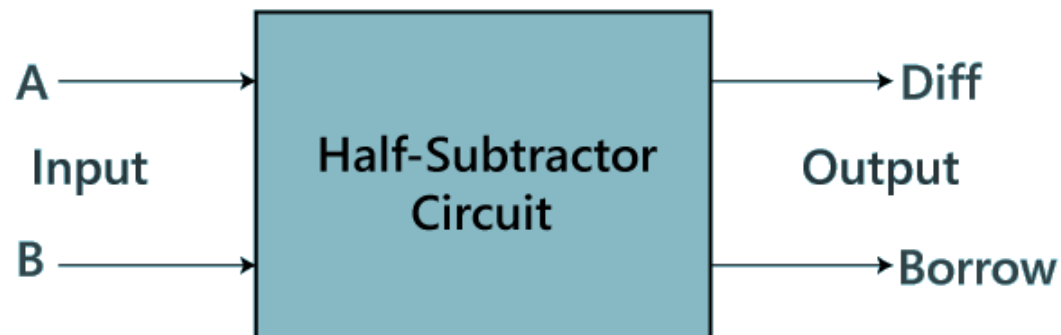
7.4 Subtractor

7.4 Subtractor

Half Subtractor

- Half subtractor is a combinational circuit used to get the difference between two single bit

| Inputs | | Outputs | |
|--------|---|---------|--------|
| A | B | Diff | Borrow |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |



7.4 Subtractor

K-MAP for Difference

| A \ B | 0 | 1 |
|-------|---|---|
| 0 | | 1 |
| 1 | 1 | |

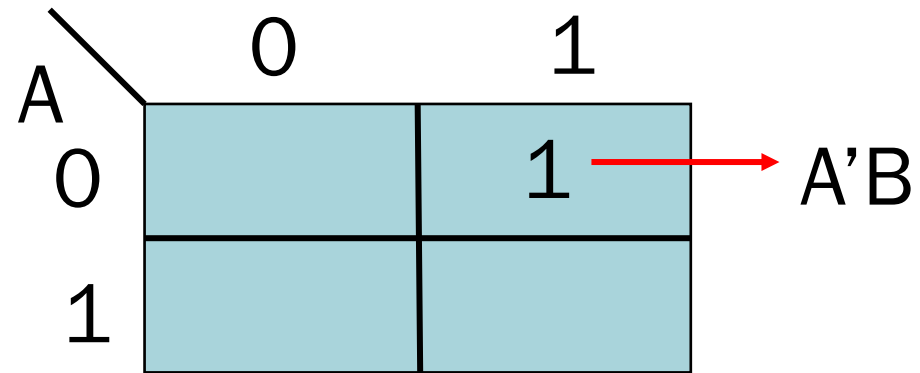
$$A'B + AB'$$



$$A \text{ XOR } B$$

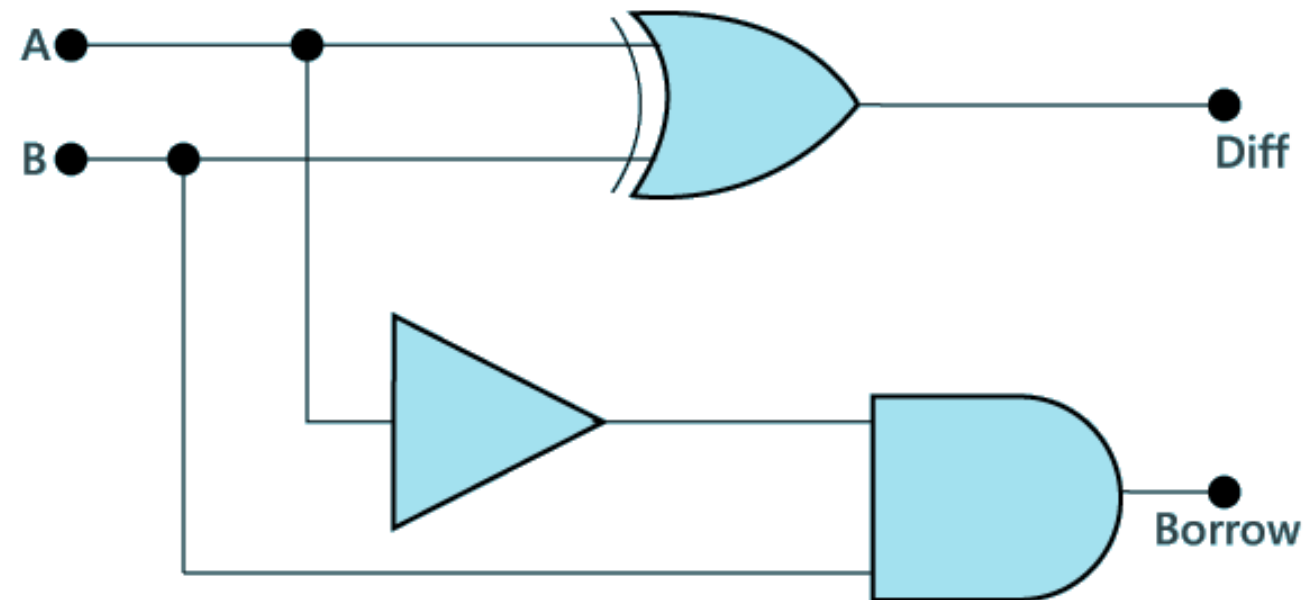
7.4 Subtractor

K-MAP for Borrow



7.4 Subtractor

Half Subtractor circuit

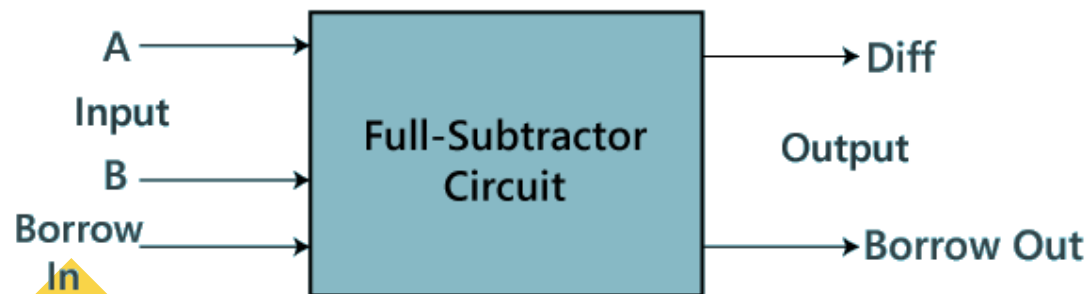


Half-Subtractor Circuit

7.4 Subtractor

Full Subtractor

- Full subtractor is a combinational circuit used to perform subtraction among 3 bit



| Inputs | | | Outputs | |
|--------|---|----------------------|---------|--------|
| A | B | Borrow _{in} | Diff | Borrow |
| 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 |

7.4 Subtractor

K-MAP for Difference

| | | B Bin | | | |
|---|---|-------|----|----|----|
| | | 00 | 01 | 11 | 10 |
| A | 0 | | 1 | | 1 |
| | 1 | 1 | | 1 | |

$A \oplus B \oplus \text{Bin}$

7.4 Subtractor

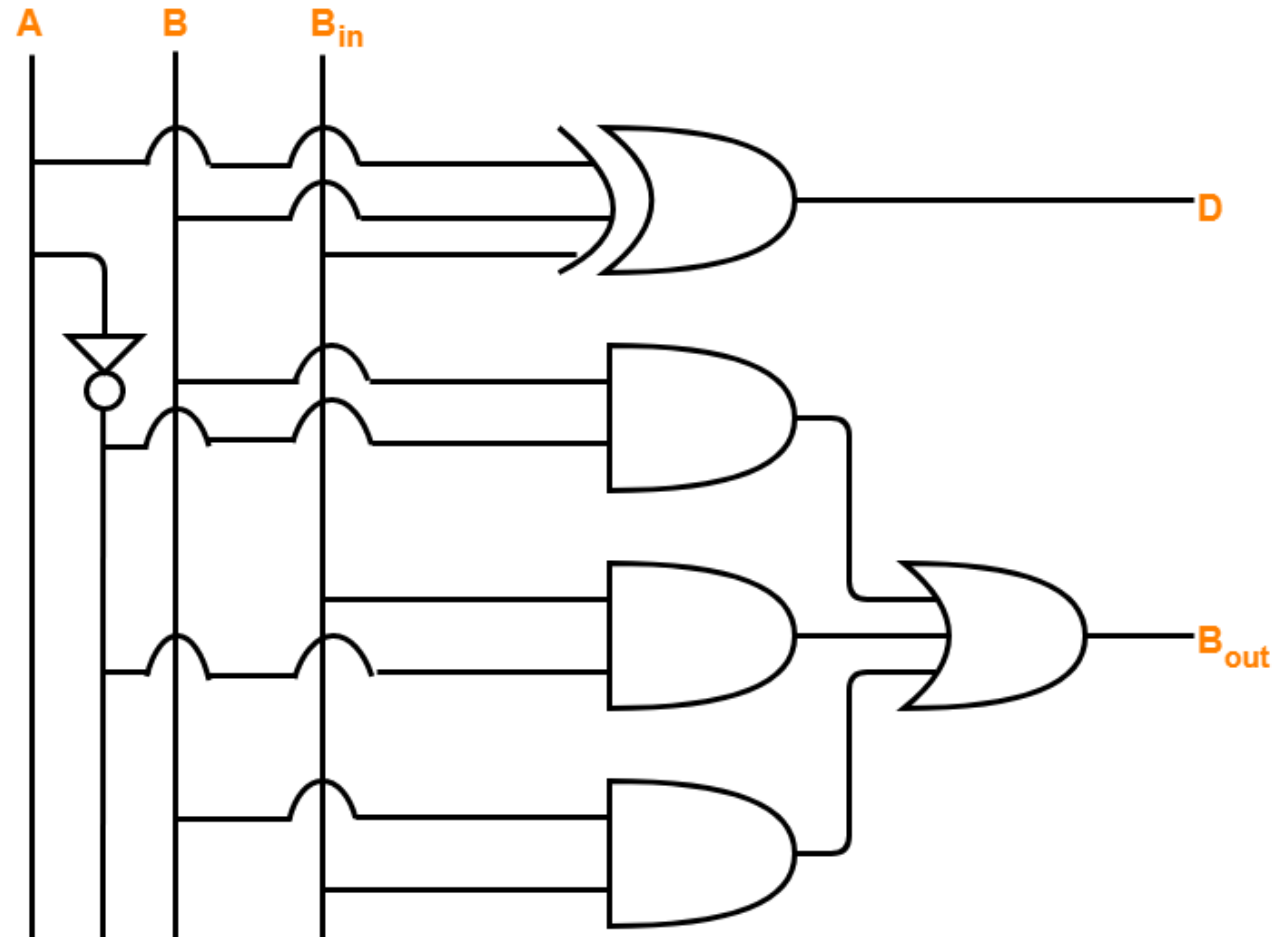
K-MAP for Borrow out

| | | B Bin | | | |
|---|---|-------|----|----|----|
| | | 00 | 01 | 11 | 10 |
| A | 0 | | 1 | 1 | 1 |
| | 1 | | | 1 | |

$$B \text{ Bin} + A'B + A'\text{Bin}$$

7.4 Subtractor

Full Subtractor circuit



The background features several large, overlapping geometric shapes, primarily diamonds and triangles, in teal, yellow, and green colors. These shapes are positioned in the top right and bottom left corners, creating a modern, abstract design.

“

7.5 Code Conversion

7.5 Code Conversion

- Code converter is used to convert one type of binary code to another
- There are 3 different types of binary codes, like BCD code, gray code, and excess-3 code
- In this topic, we will learn 2 different converter
 - Binary to Gray Code Converter
 - Binary to BCD Code Converter

7.5 Code Conversion

Binary to Gray Code Converter

| Decimal | Binary | Gray Code |
|---------|--------|-----------|
| 0 | 000 | 000 |
| 1 | 001 | 001 |
| 2 | 010 | 011 |
| 3 | 011 | 010 |
| 4 | 100 | 110 |
| 5 | 101 | 111 |
| 6 | 110 | 101 |
| 7 | 111 | 100 |

7.5 Code Conversion

K-MAP for D

| A \ B C | 00 | 01 | 11 | 10 |
|---------|----|----|----|----|
| | 0 | 0 | 1 | 1 |
| 0 | | | | |
| 1 | 1 | 1 | 1 | 1 |

$$D = A$$

K-MAP for E

| A \ B C | 00 | 01 | 11 | 10 |
|---------|----|----|----|----|
| | 0 | 0 | 1 | 1 |
| 0 | | | 1 | 1 |
| 1 | 1 | 1 | | |

$$E = A'B + AB' = A \text{ XOR } B$$

7.5 Code Conversion

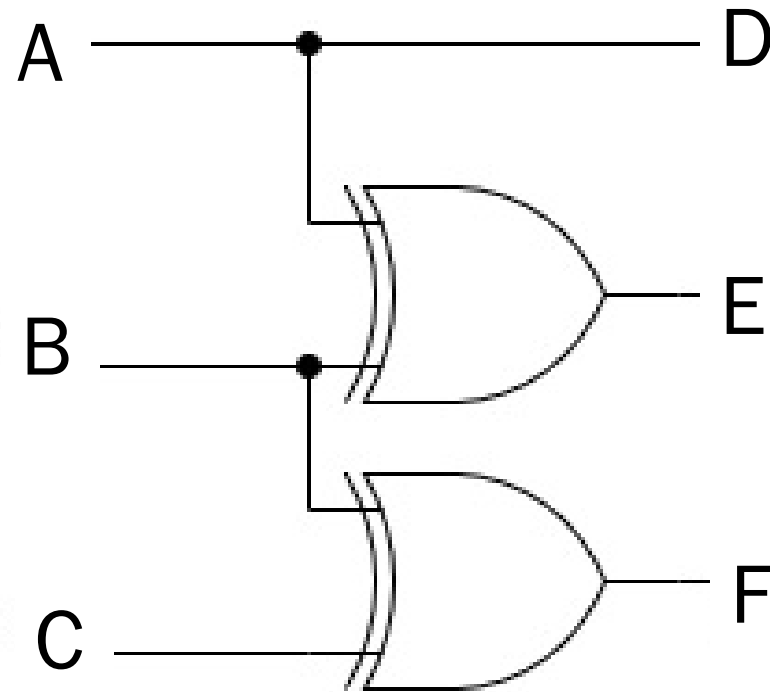
K-MAP for F

| B C | | 00 | 01 | 11 | 10 |
|-----|---|----|----|----|----|
| A | 0 | | 1 | | 1 |
| | 1 | | 1 | | 1 |

$$F = B'C + BC' = B \text{ XOR } C$$

7.5 Code Conversion

Binary to Gray Code
Converter Circuit



7.5 Code Conversion

Binary to BCD Converter

| | Binary Code (Input) | | | | BCD Code (Output) | | | | |
|---|---------------------|----------|----------|----------|-------------------|----------|----------|----------|----------|
| | <i>A</i> | <i>B</i> | <i>C</i> | <i>D</i> | <i>W</i> | <i>X</i> | <i>Y</i> | <i>Z</i> | <i>E</i> |
| 0 | | | | | | | | | |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 4 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 5 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 7 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

7.5 Code Conversion

Cont..

| | Binary Code (Input) | | | | BCD Code (Output) | | | | |
|----|---------------------|----------|----------|----------|-------------------|----------|----------|----------|----------|
| | <i>A</i> | <i>B</i> | <i>C</i> | <i>D</i> | <i>W</i> | <i>X</i> | <i>Y</i> | <i>Z</i> | <i>E</i> |
| 8 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 11 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 12 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 13 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 15 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |

7.5 Code Conversion

K-MAP for W

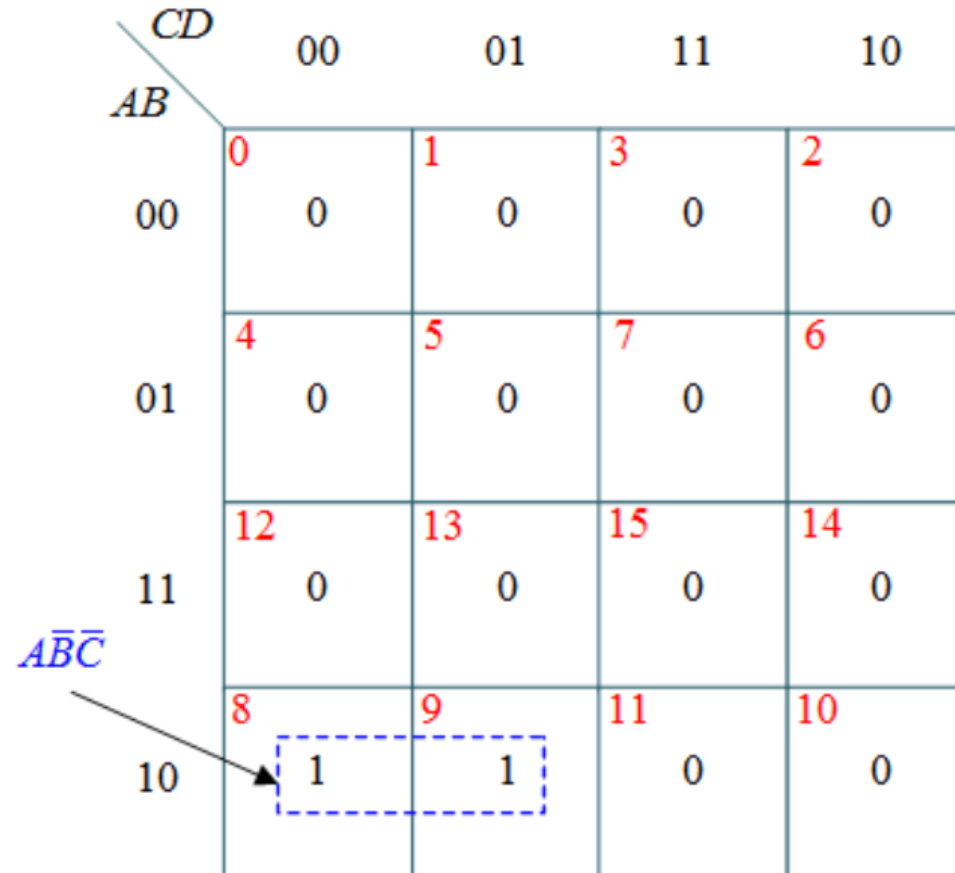
$$W = AB + AC$$

| <i>CD</i> | 00 | 01 | 11 | 10 |
|-----------|----|----|----|----|
| <i>AB</i> | | | | |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 0 | 0 | 1 | 1 |

7.5 Code Conversion

K-MAP for X

$$X = AB'C'$$



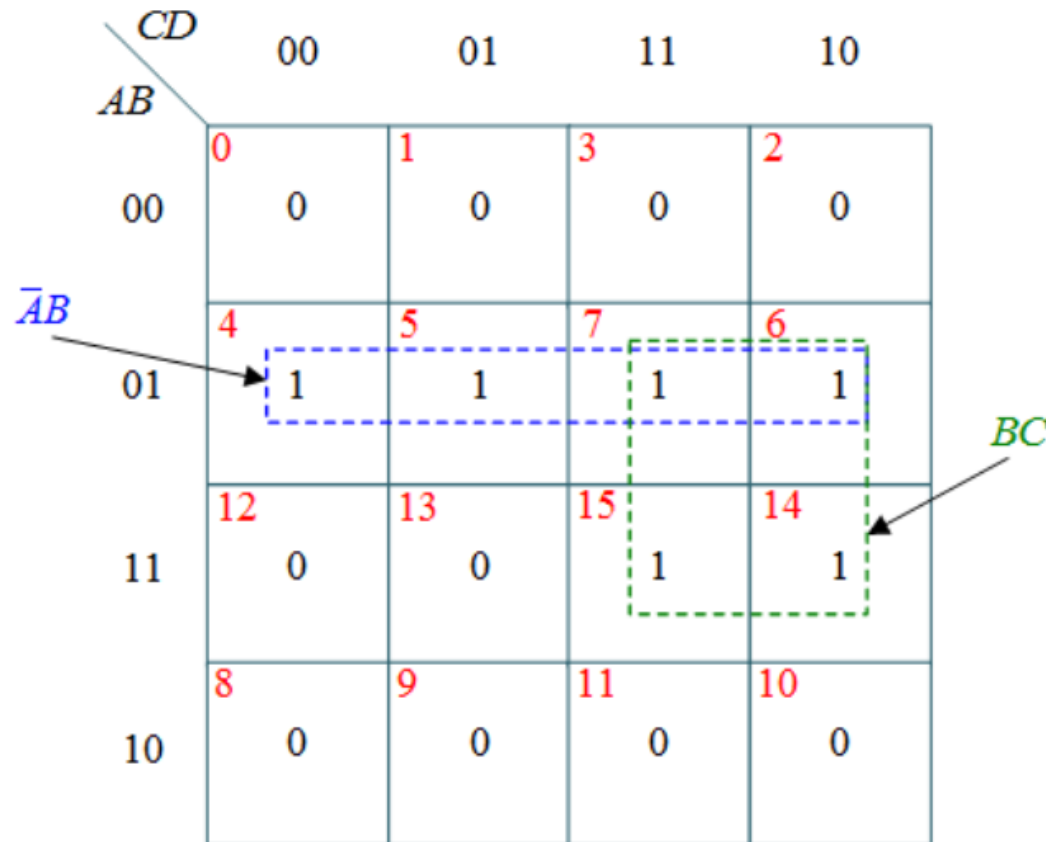
A 4x4 Karnaugh Map for the function $X = AB'C'$. The vertical axis is labeled AB with values 00, 01, 11, 10. The horizontal axis is labeled CD with values 00, 01, 11, 10. Each cell contains a decimal index (in red) and a value (in black). The values are 0 for all cells except for the two cells at $AB=10$ and $CD=00$ and $CD=01$, which contain 1. A dashed blue box highlights these two cells, and a blue arrow labeled $A\bar{B}\bar{C}$ points to the cell at $AB=10, CD=00$.

| $AB \backslash CD$ | 00 | 01 | 11 | 10 |
|--------------------|---------|---------|---------|---------|
| 00 | 0 0 | 1 0 | 3 0 | 2 0 |
| 01 | 4 0 | 5 0 | 7 0 | 6 0 |
| 11 | 12 0 | 13 0 | 15 0 | 14 0 |
| 10 | 8 1 | 9 1 | 11 0 | 10 0 |

7.5 Code Conversion

K-MAP for Y

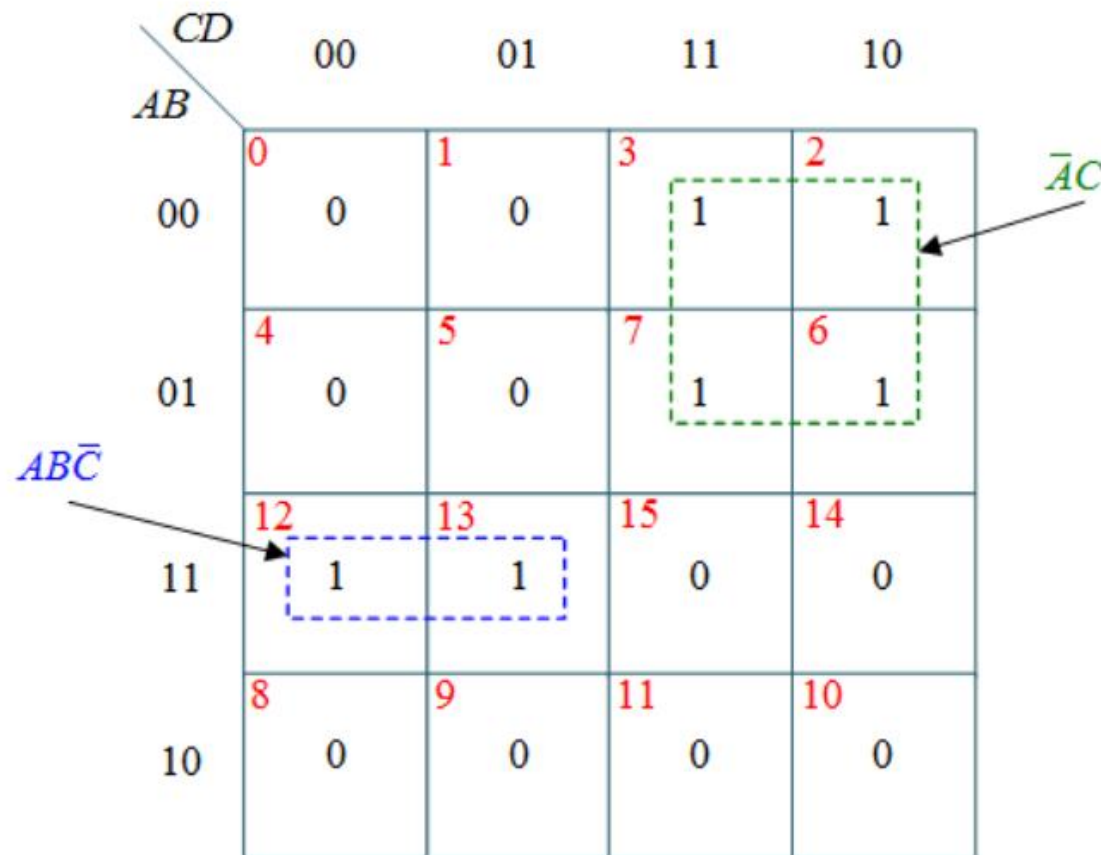
$$Y = A'B + BC$$



7.5 Code Conversion

K-MAP for Z

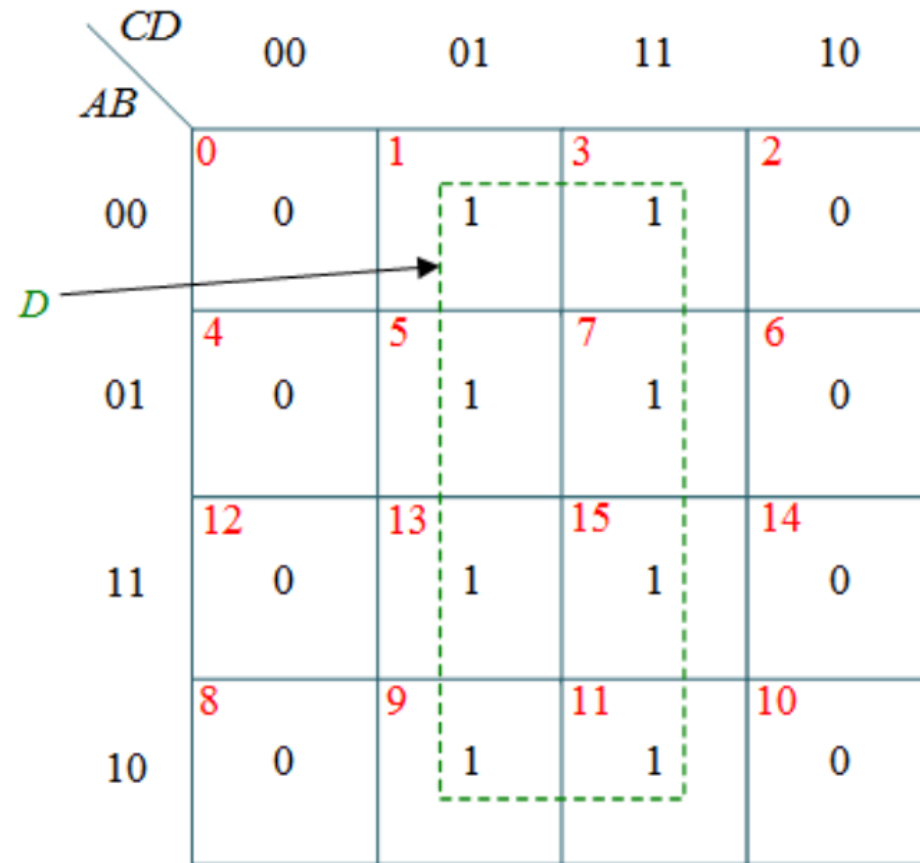
$$Z = ABC' + A'C$$



7.5 Code Conversion

K-MAP for E

$E=D$

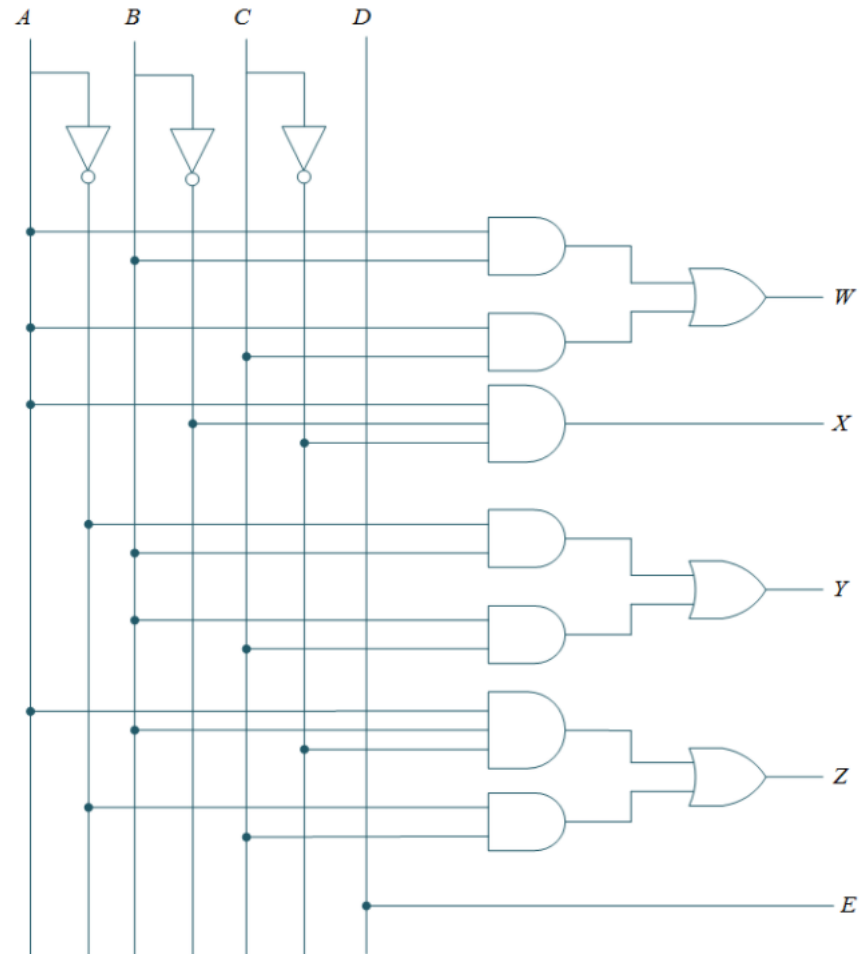


A 4x4 Karnaugh map for the function E=D. The map is labeled with AB on the vertical axis and CD on the horizontal axis. The cells contain 0s and 1s. A green dashed box highlights a group of four 1s in the middle two columns (CD=01 and CD=11) across all four rows (AB=00, 01, 11, 10). A green label 'D' with an arrow points to this group.

| $AB \backslash CD$ | 00 | 01 | 11 | 10 |
|--------------------|----|----|----|----|
| 00 | 0 | 1 | 1 | 0 |
| 01 | 0 | 1 | 1 | 0 |
| 11 | 0 | 1 | 1 | 0 |
| 10 | 0 | 1 | 1 | 0 |

7.5 Code Conversion

Binary to BCD Converter





References

M. Morris Mano, Digital Design, 5th ed, Prentice Hall, 2012, Chapter 4



Next Topic : Combinational Logic Circuit (2)