Paper 7, TDC Part-3
Chapter– 4, Combinational Logic Design
Lecture - 2

By:

Mayank Mausam

Assistant Professor (Guest Faculty)

Department of Electronics

L.S. College, BRA Bihar University,

Muzaffarpur, Bihar

Combinational Logic Design

Example :- 5.1 Given Function $Y = (A + BC)(B + \bar{C}A)$

(a) Design a circuit using gates to realise this function Y.

(b) Find out whether it is possible to design the circuit with only one type of gates (NAND or NOR). If, yes design the circuits.

(c) Find out whether it is possible to simplify this equation. If, yes, simplify it.

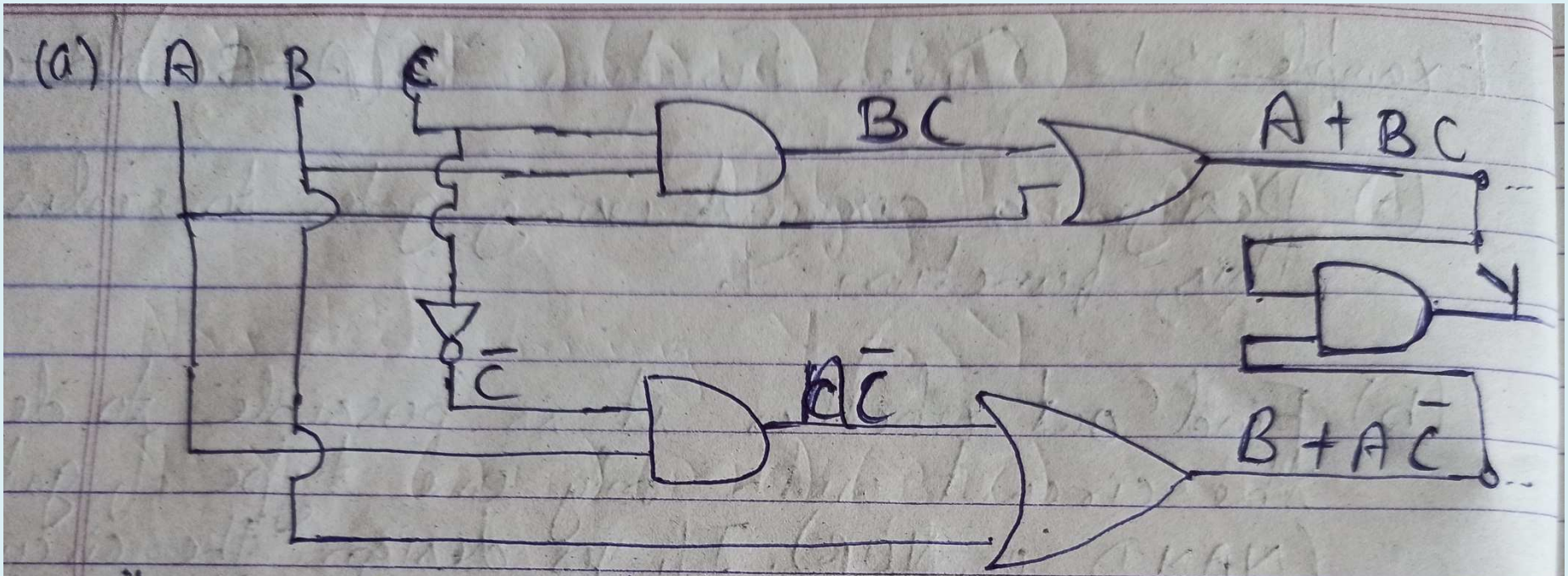
Combinational Logic Design

(d) Now design the circuit using the simplified expression obtained in part (c).

(e) Compare the circuits obtained in parts (a), (b) and (d) from the point of view of number of gates, number of inputs for the gates, types of gates and propagation delay.

Soln:- $Y = (A + BC)(B + \bar{C}A)$

Combinational Logic Design



Logic Circuits for the Realisation of Function.

Logic

The above circuits design requires 1 NOT Gate (Inverter), two 2-input OR Gate and three 2-input AND Gate.

Combinational Logic Design

(b) (i) Design of function using only gate that is either through NAND gate or NOR gate.

To design using NAND gate, we write the function Y in the SOP form.

$$Y = (A + BC)(B + A\bar{C})$$

$$= A \cdot B + A \cdot A\bar{C} + BC \cdot B + BC \cdot A\bar{C}$$

$$= AB + A\bar{C} + BC + 0$$

$A \cdot A = A$

$B \cdot B = B$

$C \cdot \bar{C} = 0$

$$Y = AB + A\bar{C} + BC$$

Combinational Logic Design

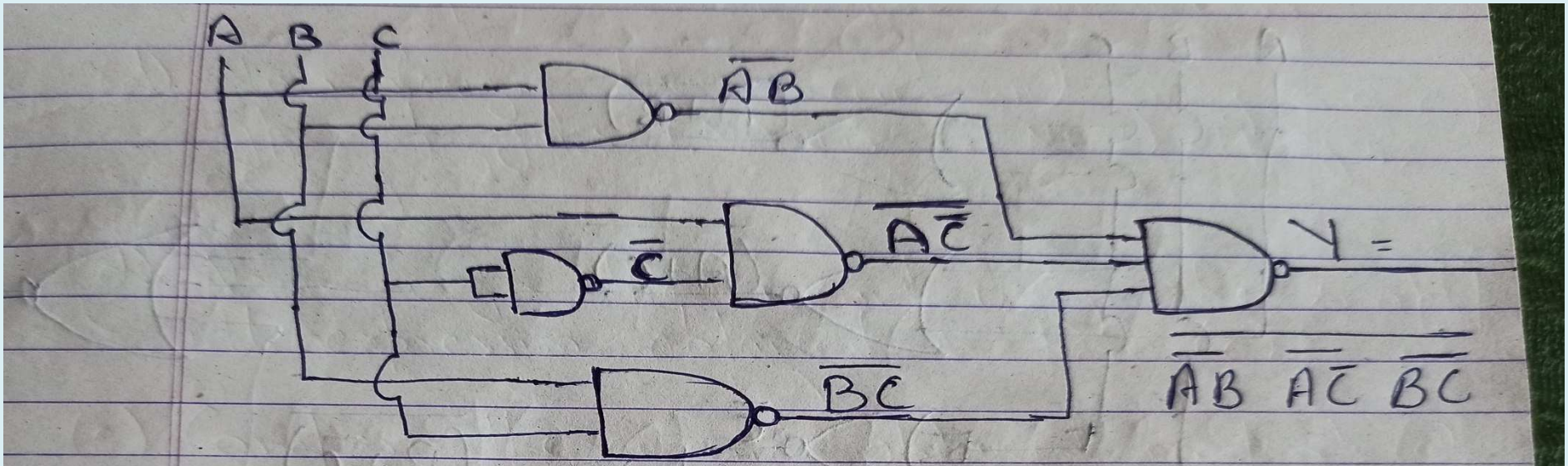
Now write function Y in complemented form, i.e.,

$$\overline{\overline{Y}} = \overline{\overline{AB + B\bar{C} + BC}}$$

$$Y = \overline{\overline{AB} \cdot \overline{A\bar{C}} \cdot \overline{BC}}$$

Now function Y can be realized using NAND gate only as shown below.

Combinational Logic Design



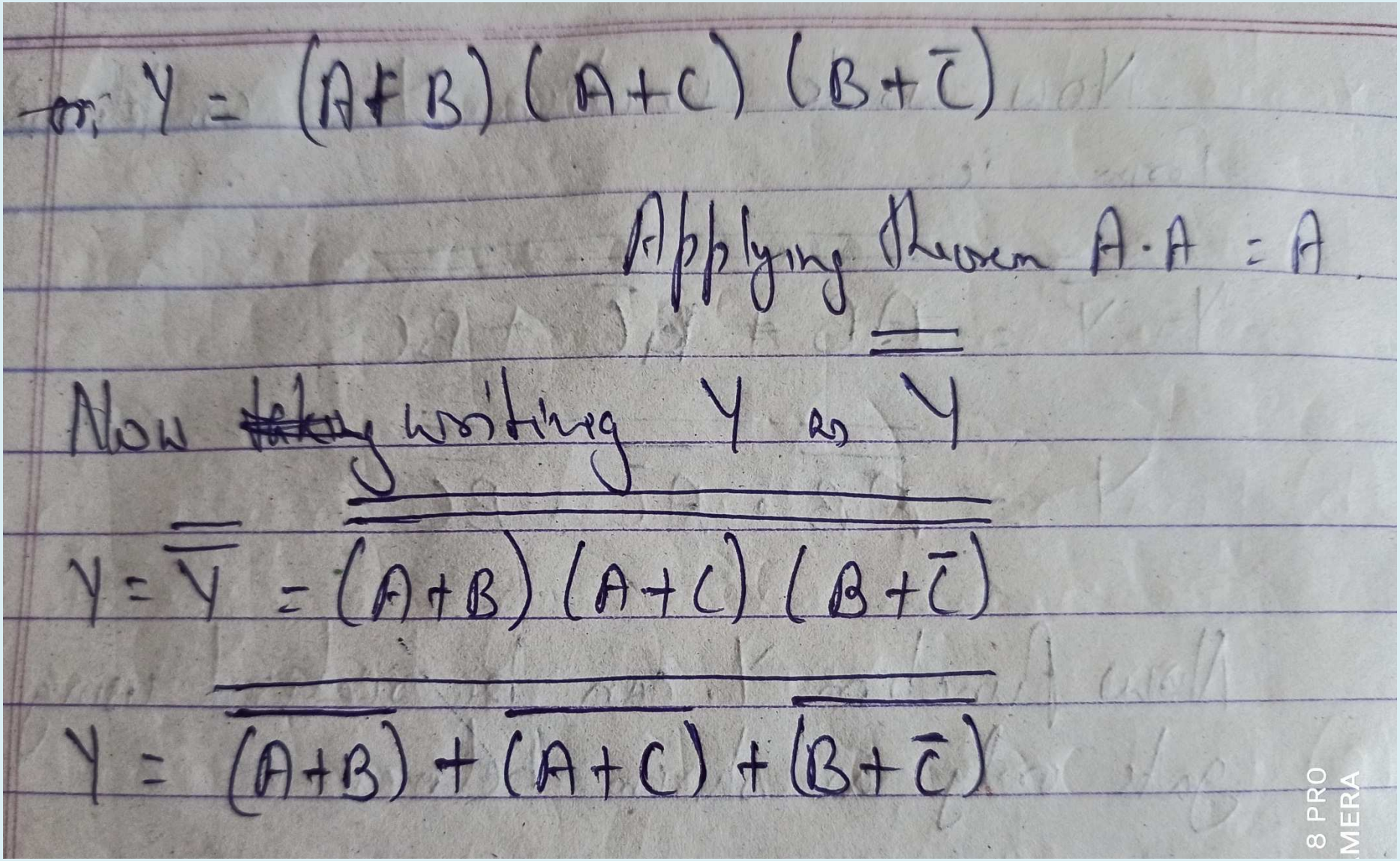
b(ii) Design of function Y using NOR gate only.

To design using NOR gate only we write the function Y in the POS form.

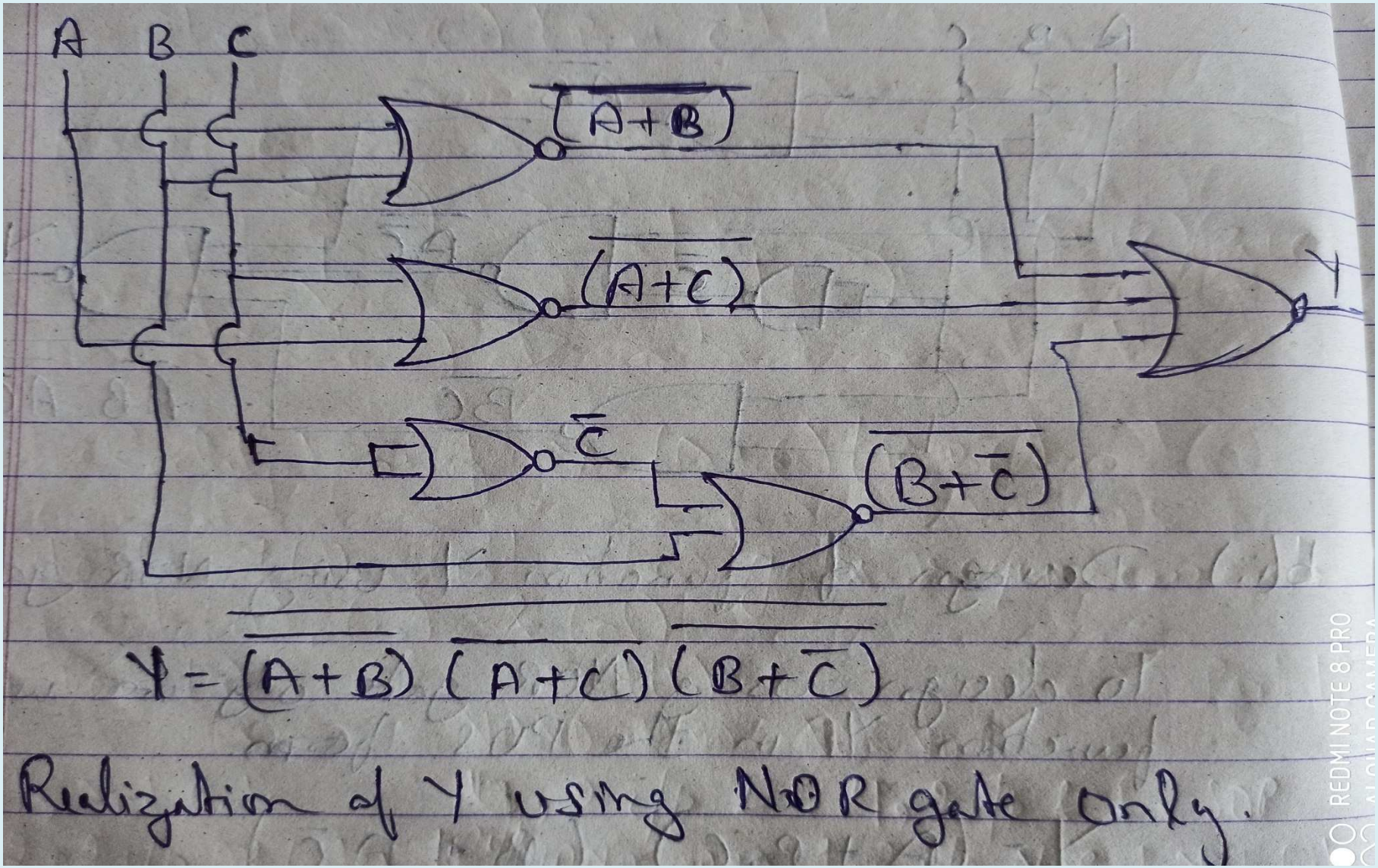
$$Y = (A + BC)(B + A\bar{C})$$
$$= (A+B)(A+C)(B+A)(B+\bar{C})$$

Applying theorem $(A+BC) = (A+B)(A+C)$

Combinational Logic Design



Combinational Logic Design



Combinational Logic Design

Hence if we can express the expression in the SOP form we can always design the circuit using only one type of gates i.e. "NAND" gate, and if we can expression the expression in the POS form we can always design the circuit using only one type of gates i.e. "NOR" gate.

c)(i) $Y = (A + BC)(B + \bar{C}A)$

$= AB + A\bar{C} + BC + \underbrace{BC\bar{C}A}_{0}$

$= AB + A\bar{C} + BC$

$= A\bar{C} + BC$ as per theorem (A)

Combinational Logic Design

So it is possible to simplify this equation.

(d) (i) $Y = (A + BC)(B + \bar{C}A)$

$= (A+B)(A+C)(A+B)(B+\bar{C})$

$= (A+B)(A+C)(B+\bar{C})$

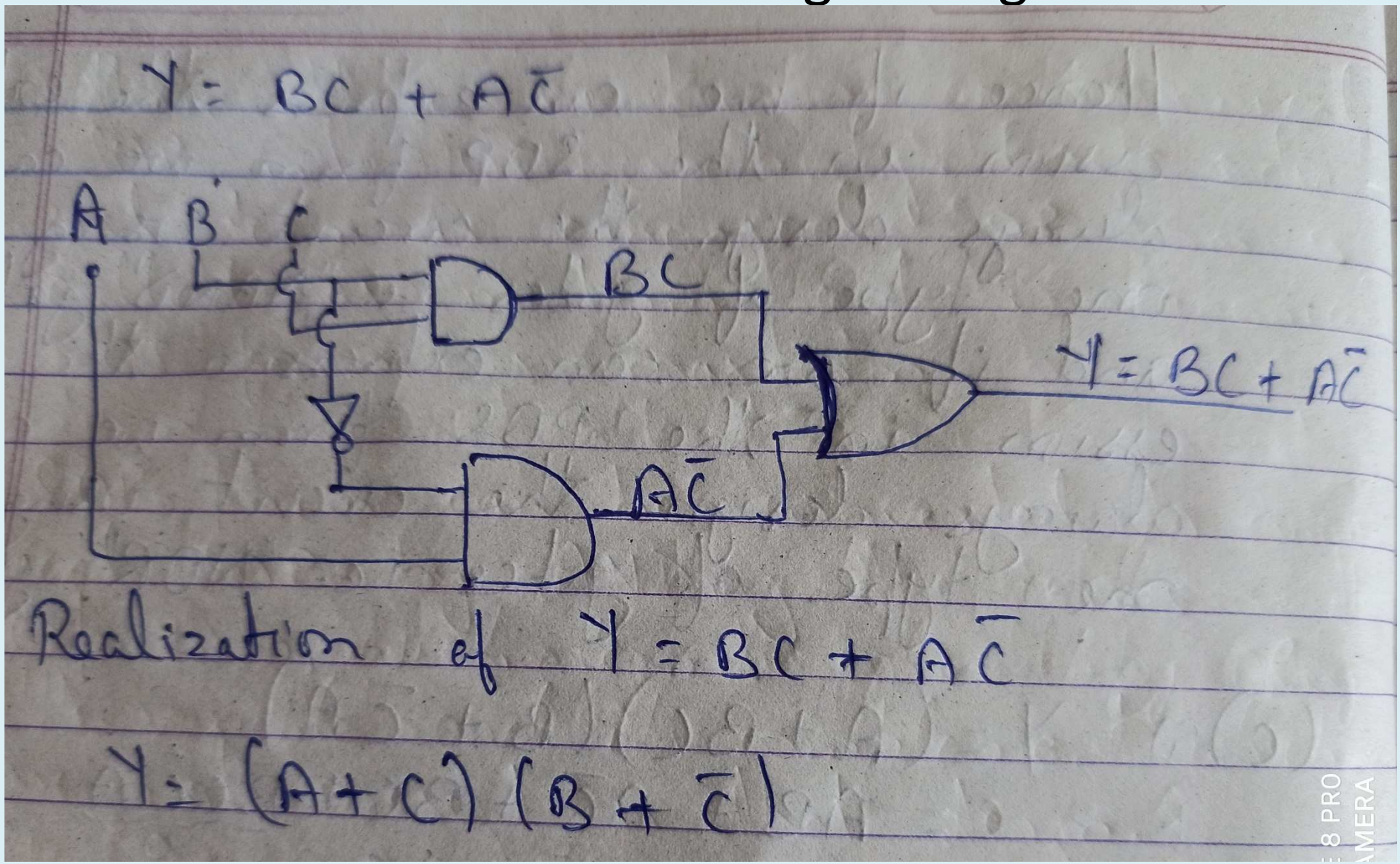
$= (A+C)(B+\bar{C})$ as per theorem.

——— (B)

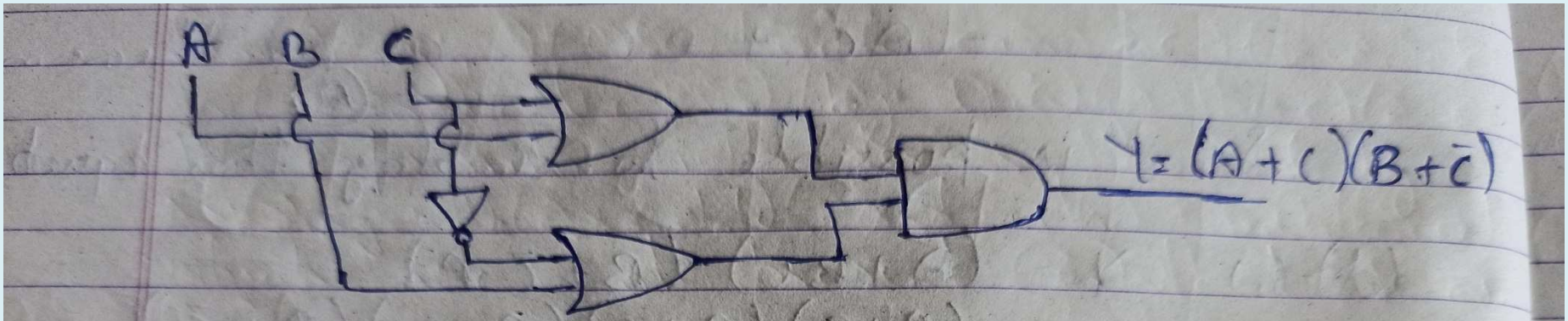
So it is possible to simply and reduce this equation is SOP and POS ~~form~~ form.

(ii) Realisation of eqn. A & eqn. B is shown in figure next page.

Combinational Logic Design



Combinational Logic Design



Realization of $Y = (A + C)(B + \bar{C})$

(e) Table 1(a) list the gates requirements of Part (a), (b) & (d)

Part(a)	Part(b) NAND-NAND	Part(b) NOR-NOR	Part(d)
1, Not Gate 2, 2 i/p OR Gates 3, 2 i/p AND Gates	4, 2 i/p & 1, 3 i/p NAND Gates	4, 2 i/p & 1, 3 i/p NOR Gate	For SOP - 1 NOT Gate 2 AND Gates 1 - OR Gate

Combinational Logic Design

For POS –
1, NOT Gate,
2, OR Gates &
1, AND Gate

The realisation of part (a) needs maximum number of gates and also it is 3-level realization which leads to increase in propagation delay, so it decrease the speed of operation.

Realization of part (b) i.e. using either NAND gates or NOR gates only is very convenient because a numbers of similar gates are available in same IC's.

Combinational Logic Design

~~Beta~~ Realization of part (d) uses less number of gates due to simplication of the ~~output~~ function, this reduces the circuit cost as well as complexity of the circuit.

We notice that in the SOP form and the POS form of function Y, all the individual terms do not involve all the 3 literals. If each term in SOP and POS forms contains all the liter

Combinational Logic Design

then these are known as canonical SOP and POS respectively. Each individual term in canonical SOP form is called as minterm and in canonical POS form as maxterm.

SOP form can be converted to canonical SOP by ANDing the terms in the expression, with terms formed by ORing the variable and its complement which are not present in that term.

Combinational Logic Design

Similarly, POS form can be converted to canonical POS by ORing the terms in the expression with terms formed by ANDing the variable and its complement which are not present in that terms.

We will the conversion SOP function into canonical SOP form and POS form into canonical POS form with example

Number Systems and Codes

Refer book- Modern Digital Electronics by RP Jain.

Thank You