Digital Logic

From Boolean algebra to implementing logic circuits

Outline

- Boolean algebra
- Logic gates
- Truth table
- Logic expressions
- Implementing digital logic using gates
- Karnaugh map – logic minimization
- Using NAND, NOR gates only.
- Binary numbers
- ADC – Analog to digital conversion

Logic circuits

- Boolean variables
  - Can have values of either TRUE (1), FALSE(0)
  - Can represent
    - either truth or falsehood of a statement
    - ON or OFF states of a switch
    - High(5V) or low(0V) of a voltage level

Logic operation

AND (x.y)

x	y	x.y
0	0	0
0	1	0
1	0	0
1	1	1

OR(x+y)

x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

NOT(x)

x	$\bar{x}$
0	1
1	0

Basic laws of Boolean Algebra

$$0 + A = A$$

$$1 + A = 1$$

$$A + A = A$$

$$A + \overline{A} = 1$$

$$0.A = 0$$

$$1.A = A$$

Basic laws of Boolean Algebra

$$A.A = A$$

$$A.\overline{A} = 0$$

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

$$A + B = B + A$$

$$A.B = B.A$$

Basic laws of Boolean Algebra

$$A + (B + C) = (A + B) + C$$

$$(A.B).C = A.(B.C$$

$$A.(B + C) = A.B + A.C$$

$$A + A.B = A$$

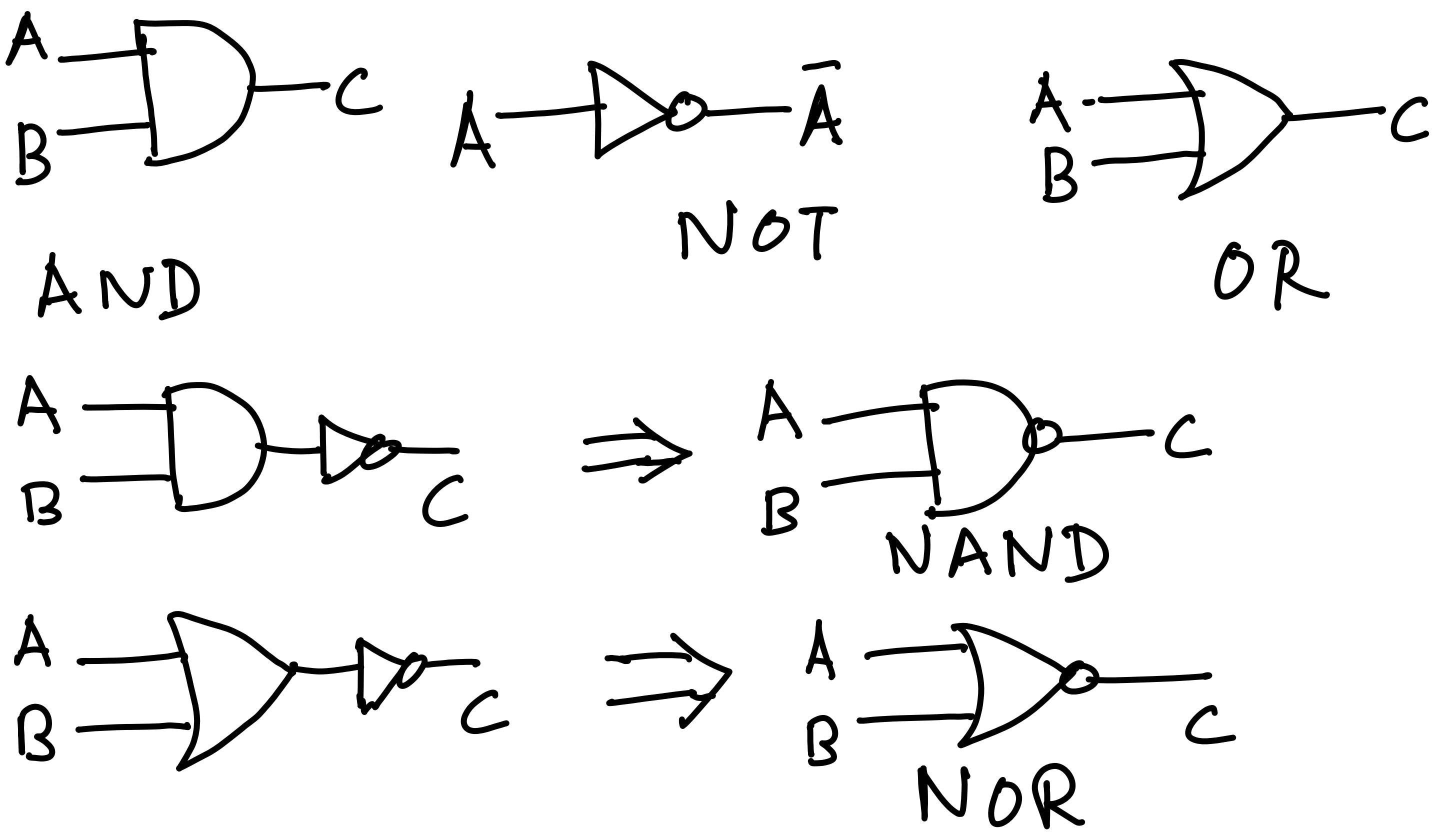
$$A.(A + B) = A$$

$$(A + B).(A + C) = A + B.C$$

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

Logic Gates

- Electronic circuits used to perform boolean logic operations are known as logic gates.

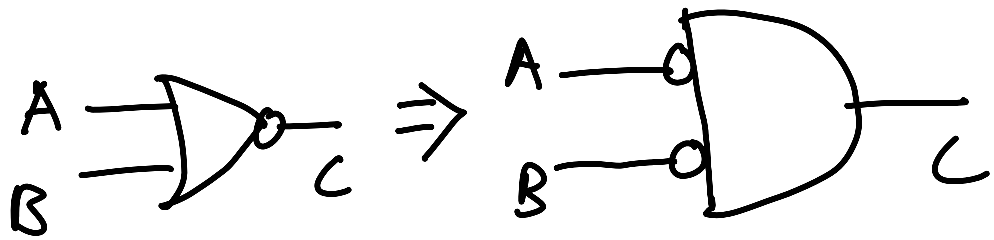


DeMorgan’s Theorem

$\overline{A+B} = \overline{A}.\overline{B}$ → Inverting a sum results in product of inverted signals.

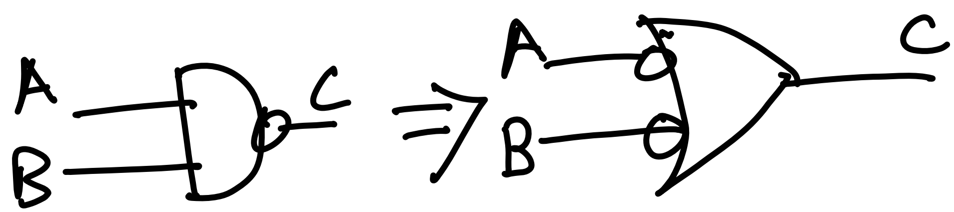
$\overline{A.B} = \overline{A} + \overline{B}$

NOR : $\overline{A+B} = \bar{A} \cdot \bar{B}$



(Alternate NOR)

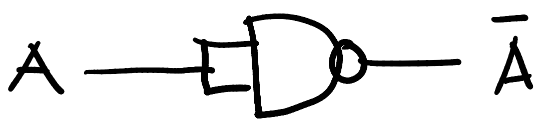
NAND $\overline{A \cdot B} = \bar{A} + \bar{B}$



(Alternate NAND).

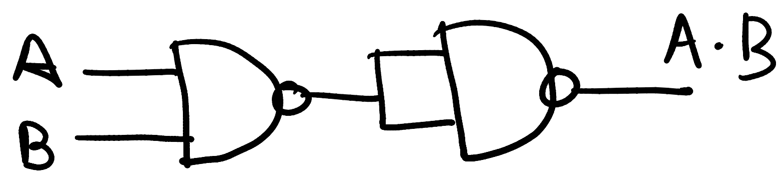
Universality of NAND gates

- AND, OR, NOT gates can be obtained from NAND gate

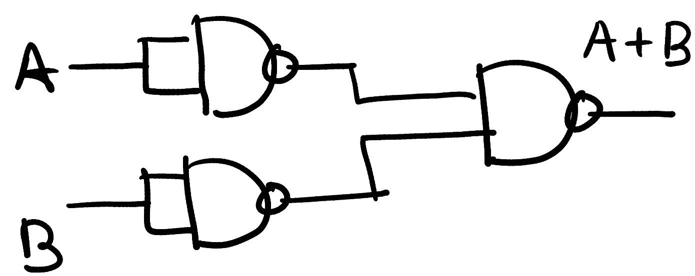


as $\overline{A.A} = \bar{A}$

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



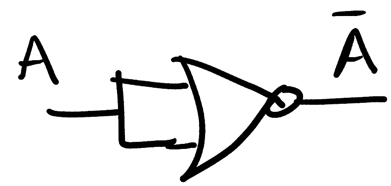
$A + B = \overline{\overline{A + B}} = \overline{\bar{A} \cdot \bar{B}}$



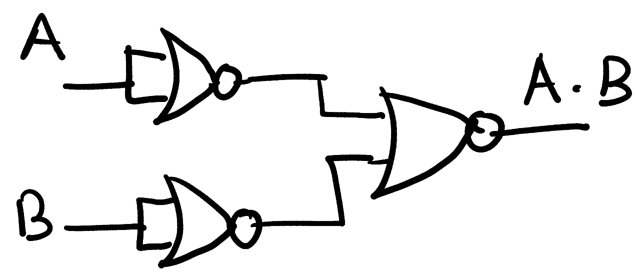
Universality of OR gates

- AND, OR, NOT gates can be obtained from NOR gate

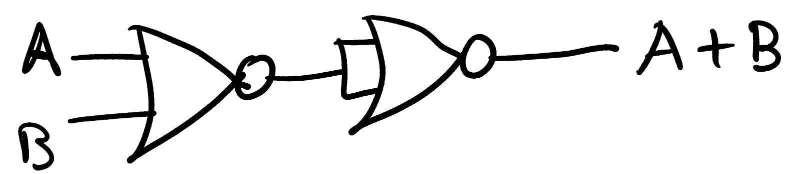
$\overline{A+A} = \bar{A}$ as $A+A=A$;



$A \cdot B = \overline{\overline{A \cdot B}} = \overline{\bar{A}+\bar{B}}$;



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



Truth table and logic expression

- Consider a logic circuit having two inputs A, B and one output C.
- The output becomes 1 when only one input is TRUE (1), but is FALSE(0) other wise.

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

TRUTH table

PRODUCT terms

- For each row in the Truth table, with ‘1’ at the output column, we can create a PRODUCT term of all inputs.
  - Take the input, if the entry for it is 1
  - Take the input with a bar, if the entry for it is 0
- The logic expression will be SUM of these PRODUCTs (SOP)

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

$\bar{A}B$

$A\bar{B}$

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

SUM terms

- For each row in the Truth table, with ‘0’ at the output column, we can create a SUM term of all inputs.
  - Take the input, if the entry for it is 0
  - Take the input with a bar, if the entry for it is 1
- The logic expression will be PRODUCT of SUMs (POS).

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

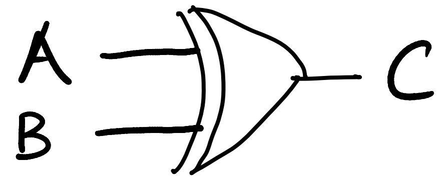
$(A+B)$

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

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

Exclusive OR

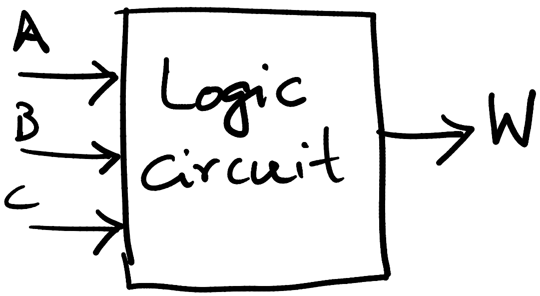
A	B	$C = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0



$$A \oplus B = \bar{A}B + A\bar{B}$$

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

Truth table – Logic expressions



A	B	C	W
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

$(A+B+C)$

$\bar{A}.\bar{B}.C$

$\bar{A}.B.\bar{C}$

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

$A.\bar{B}.\bar{C}$

$A.\bar{B}.C$

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

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

Sum of Products (SOP)

$$W = \overline{A}.\overline{B}.C + \overline{A}.B.\overline{C} + A.\overline{B}.\overline{C} + A.\overline{B}.C$$

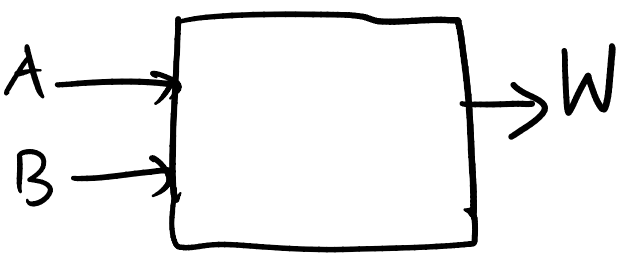
Product of Sums (POS)

$$\mathrm{W} = (A+B+C).(A+\overline{B}+\overline{C}).(\overline{A}+\overline{B}+C).(\overline{A}+\overline{B}+\overline{C})$$

Example – sop, pos

①

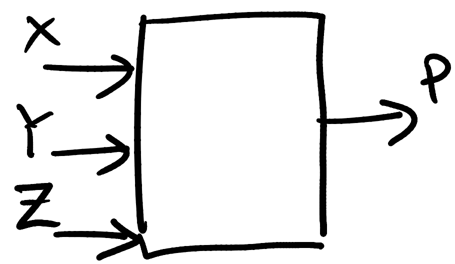
A	B	W
0	0	1
0	1	0
1	0	1
1	1	1



Sop for W?
POS

②

X	Y	Z	P
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0



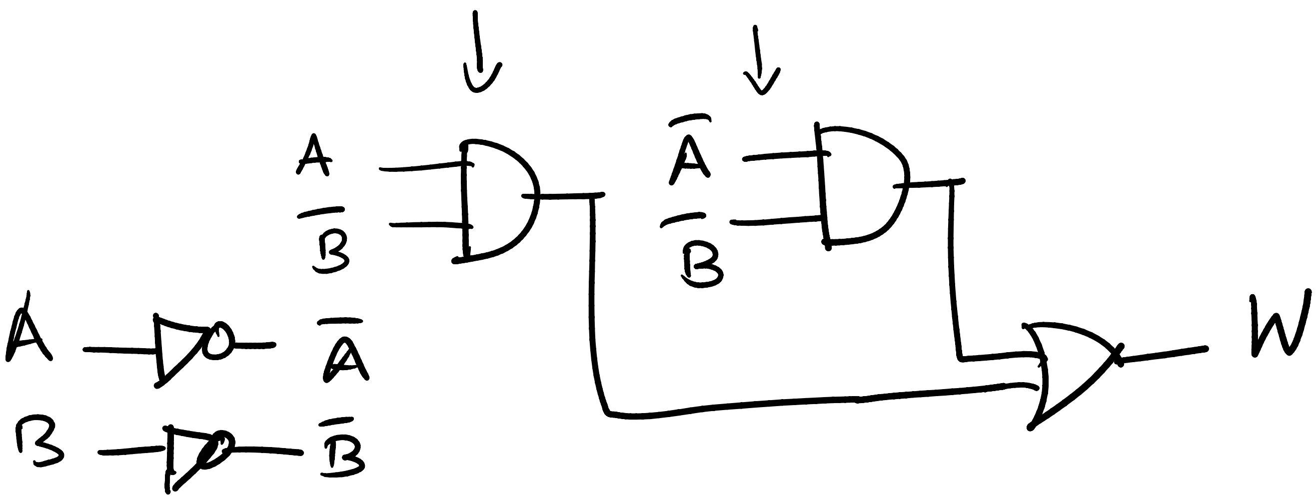
Sop for P?
POS

Design logic circuits: Implementing logic expressions

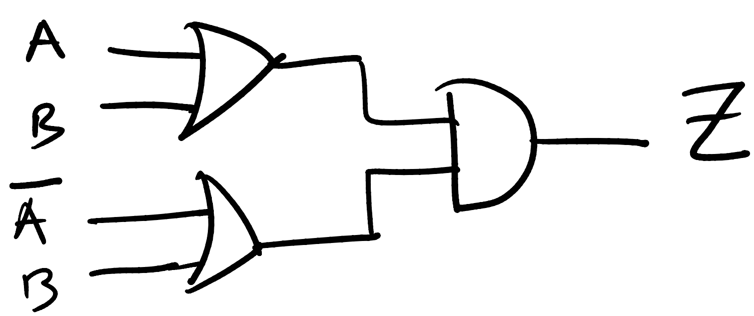
- Logic expressions are in either Sum of Product (SOP) or Product of Sums (POS) form
- Implement the sum term using OR gates and product term using AND gates
- Hence, using AND, OR and NOT gates we can implement any logic expressions

Implementing sop and pos

① $W = A\cdot\bar{B} + \bar{A}\cdot\bar{B}$



② $Z = (A+B)\cdot(\bar{A}+B)$



Design logic circuits: Implementing logic expressions

- As designer, our aim will be to use minimum number of gates.
- We need to minimize the SOP and POS expressions.
- There are two ways for this:
  - Boolean algebra laws
  - Graphical method – Karnaugh map

Karnaugh Maps

- It is a graphical representation of the truth table – each row in truth table corresponds to one cell in the K-map.
- The adjacent (horizontal or vertical) cells have only one variable changing from 0 to 1 or vice versa.

T.T.

A	B	C
0	0	1
0	1	1
1	0	0
1	1	0

K.M.

B \ A	0	1
0	1	0
1	1	0

Truth table to K-map more examples

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

AB	00	01	11	10
0	0	0	0	1
1	0	1	0	1

BC \ A	0	1
00	0	1
01	0	1
11	1	0
10	0	0

Grouping adjacent cells

- Group adjacent cells containing same values
- Group in powers of 2 (2^n) i.e. in 2, 4 ,8 etc.
- The bigger the group smaller the term – a group with 2^n terms has n terms less

CD \ AB	00	01	11	10
00	0	1	1	0
01	0	0	0	0
11	1	1	1	1
10	0	0	0	0

CD

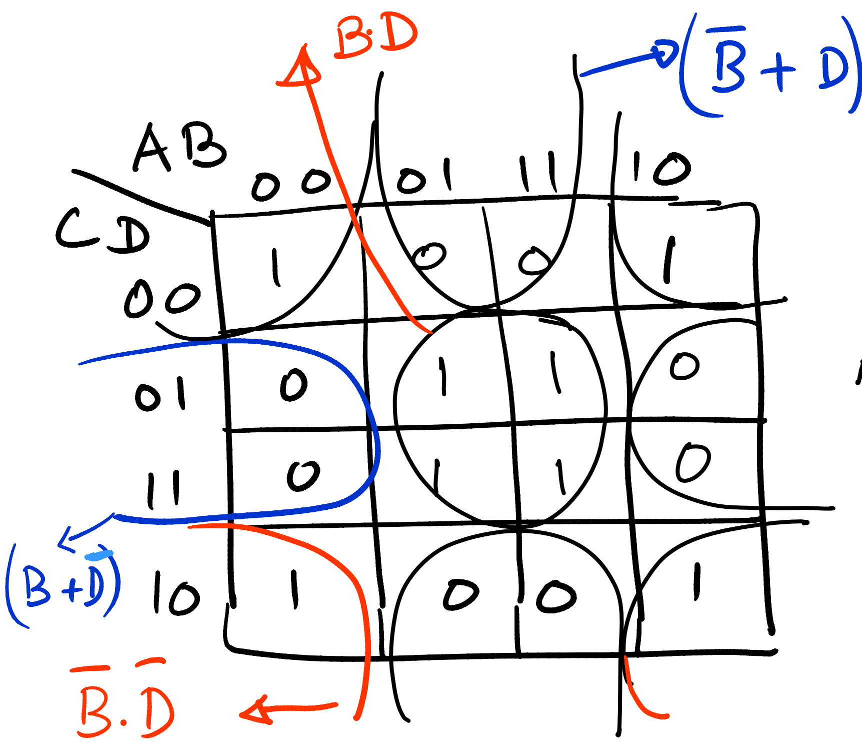
$$\bar{A}B\bar{C}\bar{D} + AB\bar{C}\bar{D} =$$

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

$$= 1 \cdot B\bar{C}\bar{D} = B\bar{C}\bar{D}$$

Variables with both 0 and 1 go.

Adjacent cells



Leftmost column $AB = 00$

Rightmost column $AB = 10$

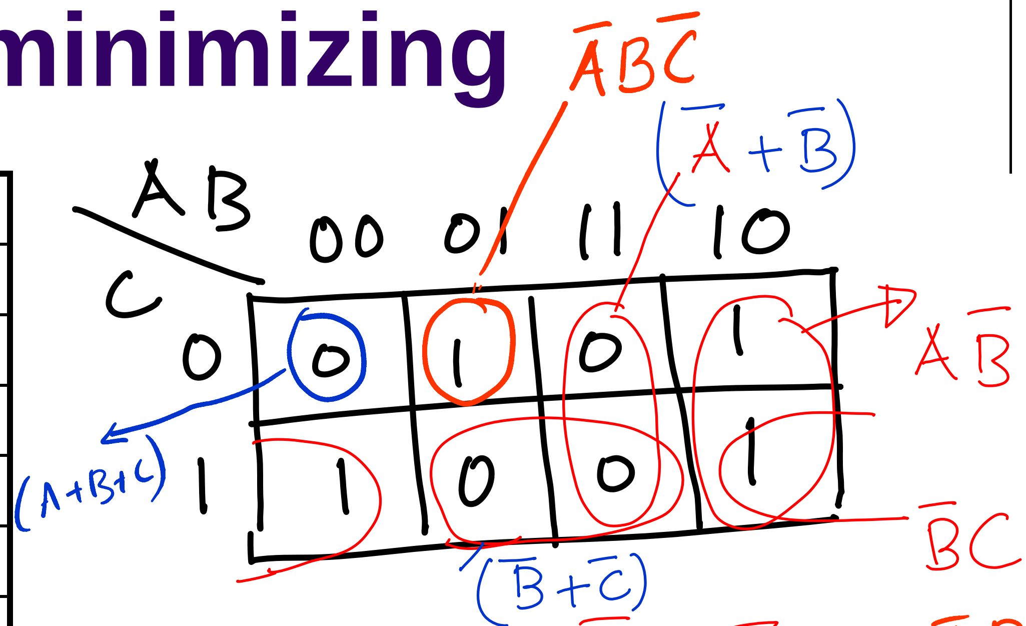
As only A changes from 0 to 1, the left most column and right most column are adjacent

* Topmost row and bottom most row are also adjacent.
* The four corners can be grouped together

$MSOP = B \cdot D + \bar{B} \cdot \bar{D}$, $\quad MPOS = (\bar{B} + D) \cdot (B + \bar{D})$

K-Map for minimizing

A	B	C	W
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0



$$MSOP = A\bar{B} + \bar{B}C + \bar{A}B\bar{C}$$

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

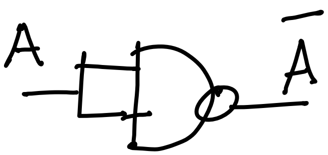
* Try to form as big a group as possible

* One cell can be part of more than one group if necessary.

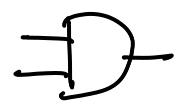
Logic design using only NAND or only NOR gates

- NAND, NOR gates are universal gates
- The AND, OR, NOT can be converted into NAND/NOR gates

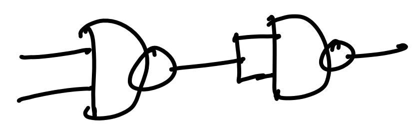
NOT can be made of NAND gate



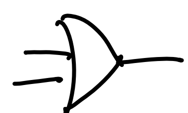
$A \cdot B = \overline{\overline{A \cdot B}}$



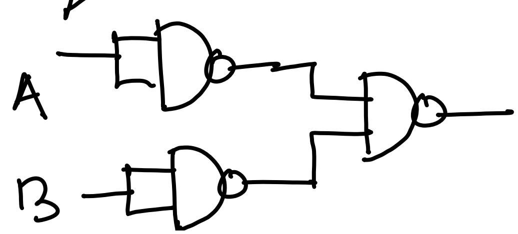
can be replaced by



$A + B = \overline{\overline{A+B}} = \overline{\bar{A} \cdot \bar{B}}$

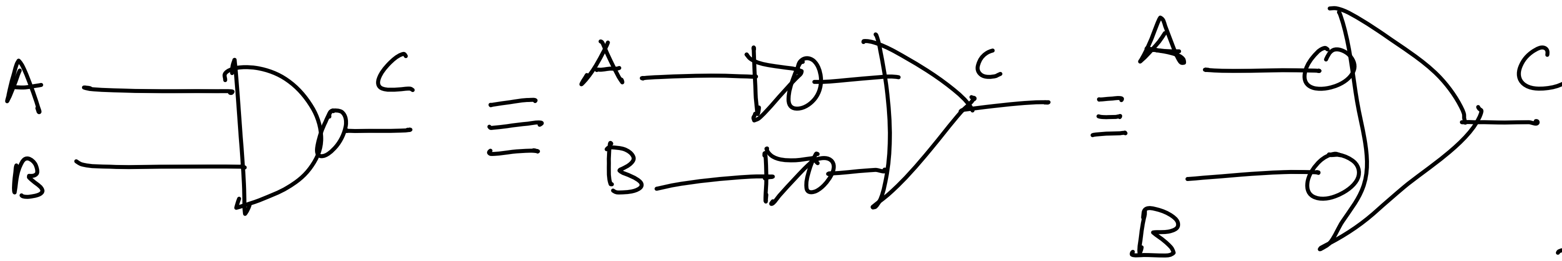


can be replaced by



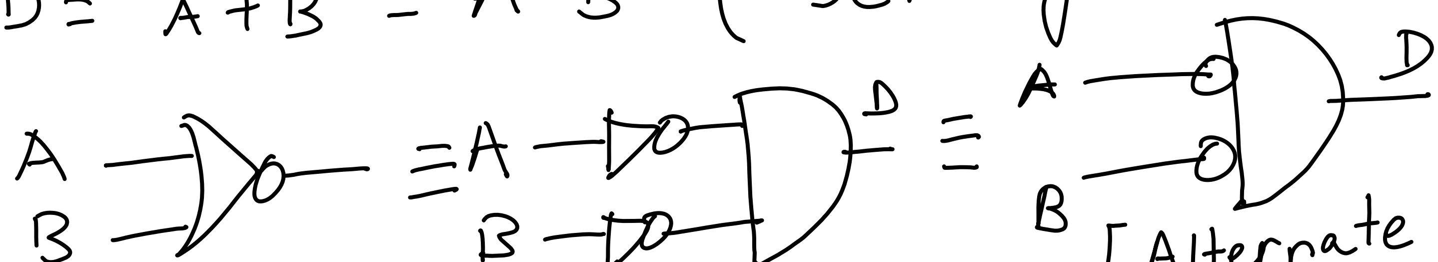
Alternate NAND NOR symbols

$C = \overline{A \cdot B} = \bar{A} + \bar{B}$ (DeMorgan's law).



[Alternate NAND]

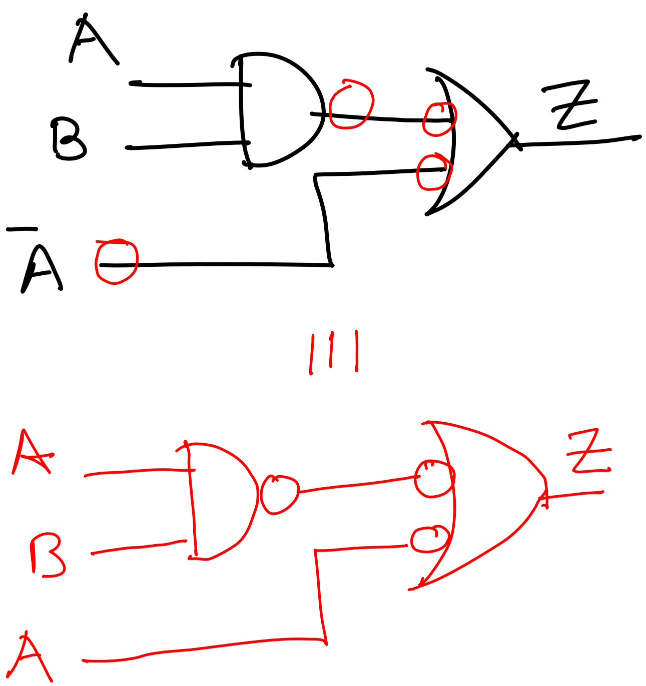
$D = \overline{A + B} = \bar{A} \cdot \bar{B}$ (DeMorgan's law)



[Alternate NOR]

From AND, OR , NOT to NAND only

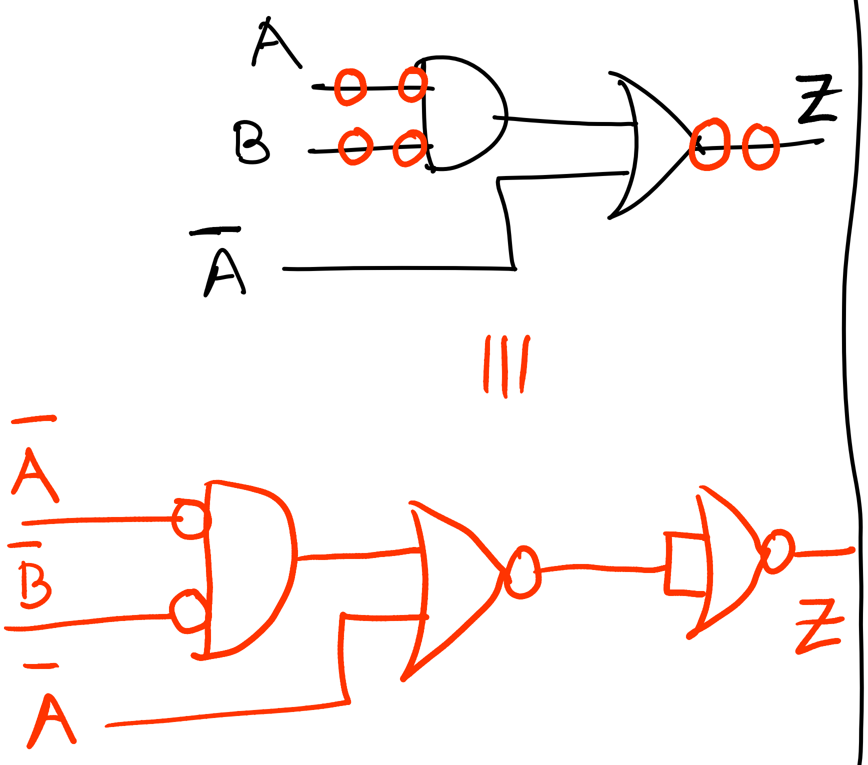
$Z = AB + \bar{A}$



- For AND gate, put an inverter (bubble) at its output.
- For OR gate, put bubbles at the inputs.
- At any point, we can put two bubbles without changing anything.

From AND, OR , NOT to NOR only

$Z = AB + \bar{A}$



- For OR gate, put an inverter (bubble) at its output.
- For AND gate, put bubbles at the inputs.
- At any point, we can put two bubbles without changing anything.

Binary numbers – base 2

- Symbols 0,1
- Positional value

Position	MSB							LSB
Power of 2	7	6	5	4	3	2	1	0
Pos. value	128	64	32	16	8	4	2	1
e.g. Binary	1	0	0	0	1	1	1	1
Decimal value	128+0+0+0+8+4+2+1=143							

Convert decimal to binary

- Keep dividing the decimal number by 2 until you get 1. Arrange the remainders from right to left. Fill the remaining bits with 0.

e.g. to convert 102 to 8 bit binary,

2	102	
2	51	0 — LSB
2	30	1
2	15	0
2	7	1
2	3	1
	1	1

$\Rightarrow$ 0111 1010 (LSB)

Examples – binary number find the decimal value and vice versa

- 00001111
- 10000000
- 11111111
- 00001000

Analog to Digital converter

- An analog-to-digital converter (abbreviated ADC, A/D or A to D) is a device which converts continuous signals to discrete digital numbers. The reverse operation is performed by a digital-to-analog converter (DAC).

Example

- If 0V-5V are converted to an 8-bit digital
  - 0V = bin 0000000 (decimal 0);
  - 5V = bin 1111111 (decimal 255);
  - The range 0-5 is divided into 256 values, i.e each division is equal to 5/255 V.
  - An input value of say 2.5V would be decimal 102 i.e bin 01100110
  - Analog 4.5V would be decimal 4.5/(5/255)=229 i.e. bin 11100101

Relating to the microcontroller experiment (Sample program)

- Connected an analog voltage (0-5V) to the AN0 (chanel 0) of the PIC's ADC port.
- Connect 8 LEDs (each with a 220ohm resistor in series) from pins RD0, RD1… to RD7 to GND.
- Vary the analog input voltage.
- Observe different LEDs light up as the input voltage is varied.