9345 1001 0011 0100 0101

16 bits

Binary Coded Decimal (BCD)

Each digit of a decimal number is replaced by a four digit binary numbers.

Weighted Code.

Decimal digit	8421 code	5421 code
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0010
4	0100	0100
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

0110

0+4+2+0 5+0+0+1

Ex. Representation of $(591)_{10}$ in BCD 8421 code:

BCD: 0101 1001 0001

Ex. Representation of $(804)_{10}$ in BCD 5421 code:

1011 0000 0100

469 $2^9=512$ $2^{10}=1024$

$(513)_{10} = 1000000001$

10 bits

$2^9 = 512$

$$\begin{array}{l|l}
2 & 513 \\ \hline
2 & 256 \quad\quad 1 = a_0 \\ \hline
 & 128 \longrightarrow 0 = a_1 \\
 & \vdots \\
2 & \quad\quad\quad\quad 1 = a_9 \\ \hline
 & 0
\end{array}$$

BCD

$(513)_{10}$

$= 0101 \; 0001 \; \underline{\underline{0011}} \rightarrow$ 12 bits

8421 code

$(a_3 a_2 a_1 a_0)_{BCD-8421}$

$= 8a_3 + 4a_2 + 2a_1 + 1a_0$

$$\left(a_3\, a_2\, a_1\, a_0\right)_{BCD-5421}$$

$$= 5a_3 + 4a_2 + 2a_1 + a_0$$

$$\left(1001\right)_{BCD-5421}$$

$$= 5 + 0 + 0 + 1 = 6$$

Addition in BCD 8421 code:

Ex.   **5:   0101**   **Ex.   4:   0100**

           + 5:   0101   **+7:   0111**

1010   1011

+110   +110

0001 0000   0001 0001

Invalid BCD number	Converted to valid BCD number
1010	0001 0000
1011	0001 0001
1100	0001 0010
1101	0001 0011
1110	0001 0100
1111	0001 0101
10000	0001 0110
10001	0001 0111
10010	0001 1000

+6

Rule: Add 0110 (6) to each resulted invalid BCD number for the addition in BCD.

Ex.

$$\begin{array}{rccc} 314: & 0011 & 0001 & 0100 \\ +\ 256: & 0010 & 0101 & 0110 \\ \hline & 0101 & 0111 & 1010 \\ & & & +\ 110 \\ \hline & & & 1\ \underline{0000} \end{array}$$

$(0101\ 0111\ 0000)_{BCD}$

$= (570)_{10}$

Ex.

$$\begin{array}{rccc} 689: & 0110 & 1000 & 1001 \\ +\ 458: & 0100 & 0101 & 1000 \\ \hline & 1011 & 1110 & 10001 \\ & & +\ 110 & +\ 110 \\ \hline & & 1\ \underline{0100} & 1\ \underline{0111} \end{array}$$

$$\begin{array}{r} 1011 \\ 110 \\ \hline 1\ \underline{0001} \end{array}$$

Result : $(0001\ 0001\ 0100\ 0111)_{BCD}$

$= (1\ 147)_{10}$

Excess-3 Code:

Add 3 to each decimal digit.

Ex. Convert 36 to an excess-3 number.

$$\begin{array}{cc} 3 & 6 \\ +\ 3 & +\ 3 \\ \hline 6 & 9 \end{array}$$

$(36)_{10} = (0110 \quad 1001)_{Ex\text{-}3}$

Addition in Excess-3 Code:

Ex.

$$\begin{array}{lr} 2: & 0101 \\ +9: & +\ 1100 \\ \hline & 0001\ 0001 \\ & 0011 \quad 0011 \\ \hline & 0100 \quad 0100 \end{array}$$

0000 … 0010 X

0011, 0100, 0101, …, 1100 valid

1101 … 1111 X

$(2)_{10} + (9)_{10}$

$= (0100 \quad 0100)_{Ex\text{-}3}$

$= (0001 \quad 0001)_{BCD\text{-}8421}$

Ex.

3: 0110 → 6

+ 4: 0111 7

$$\begin{array}{r} 0110 \\ +\,0111 \\ \hline 1101 \\ +\,\cancel{0011} \\ \hline \cancel{10000} \end{array} = \begin{array}{r} 1101 \\ -\,0011 \\ \hline (1010)_{Ex\text{-}3} \end{array}$$

010

Rule: add 0011 to the groups which produces a carry and subtract 0011 from the groups which did not produced carry.

Ex.

832: 1011 0110 0101

+ 729: 1010 0101 1100

$$\begin{array}{rrr} 1011 & 0110 & 0101 \\ 1010 & 0101 & 1100 \\ \hline 10101 & 1100 & 10001 \\ +\,011 & -\,011 & +\,011 \\ \hline 1000 & 1001 & 0100 \end{array}$$

$$\begin{array}{r} 0001 \\ +\,011 \\ \hline 0100 \end{array}$$

Result: $(0100\ 1000\ 1001\ 0100)_{Ex\text{-}3}$

$= (1561)_{10}$

The Gray Code:

Each Gray Code number differs from the preceding number by a single bit.
It is a reflected code proposed by Frank Gray. This is used in Karnaugh Map (K-map) to simplify Boolean functions. This also used to design error correcting codes.

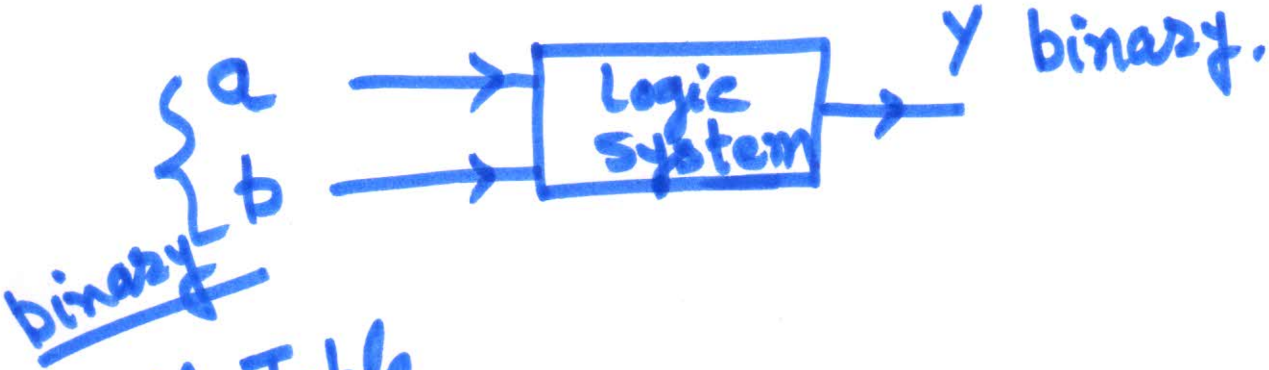
	2bits Gray codes		Binary
0	0	0	00
1	0	1	01
2	1	1	10
3	1	0	11

Binary

0000
0001
0010
0011
0100

3bits Gray Codes		
0	0	0
0	0	1
0	1	1
0	1	0
1	1	0
1	1	1
1	0	1
1	0	0

4bits Gray Codes			
0	0	0	0
	0	0	1
	0	1	1
	0	1	0
	1	1	0
	1	1	1
	1	0	1
0	1	0	0
1	1	0	0
1	1	0	1
	1	1	1
	1	1	0
	0	1	0
	0	1	1
	0	0	1
1	0	0	1

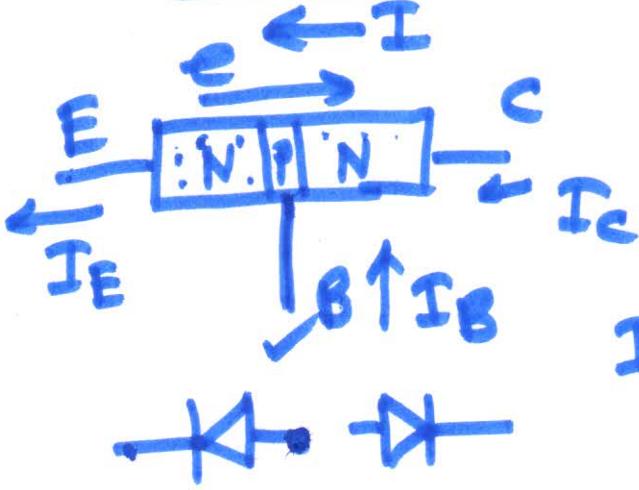


Truth Table

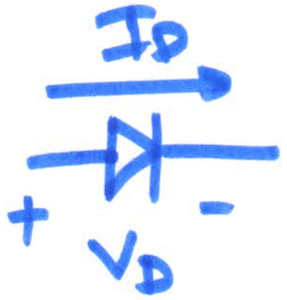
a	b	Y
0	0	0
0	1	1
1	0	1
1	1	1

- NOT
- OR
- AND
- NOR
- NAND
- XOR
- XNOR

BJT

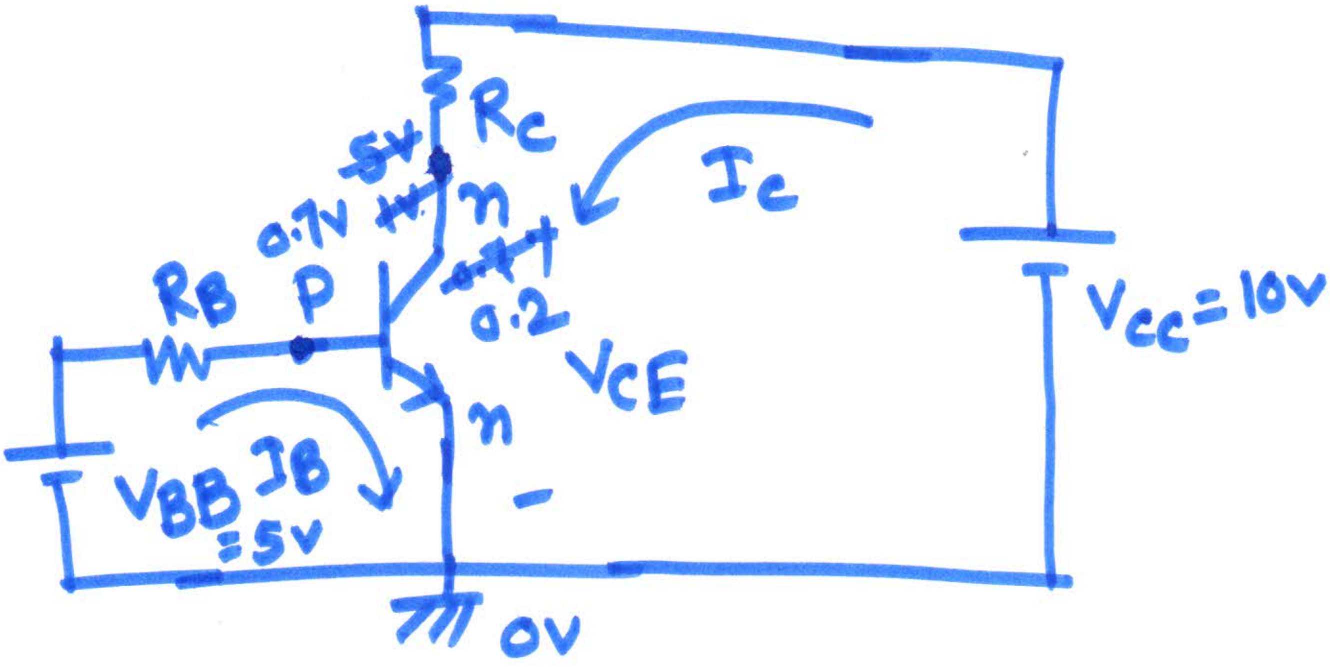


$I_B + I_C = I_E$



$$\left.\begin{array}{l} V_D > 0.5 \\ V_D = 0.7 \end{array}\right\} Si$$

BE Jun	BC Jun	Para	Operating Region	App.
✓ RB	RB	$I_B = I_C = I_E = 0$	Cut-OFF	'OFF'
FB	RB	$I_C = \beta I_B$	Active	Amplification
RB	FB	$I_E = \beta_F I_B$	Inverse Active	NAND
✓ FB	FB	$V_{CE} = 0.2V$ $I_C < \beta I_B$	Saturation	'ON'



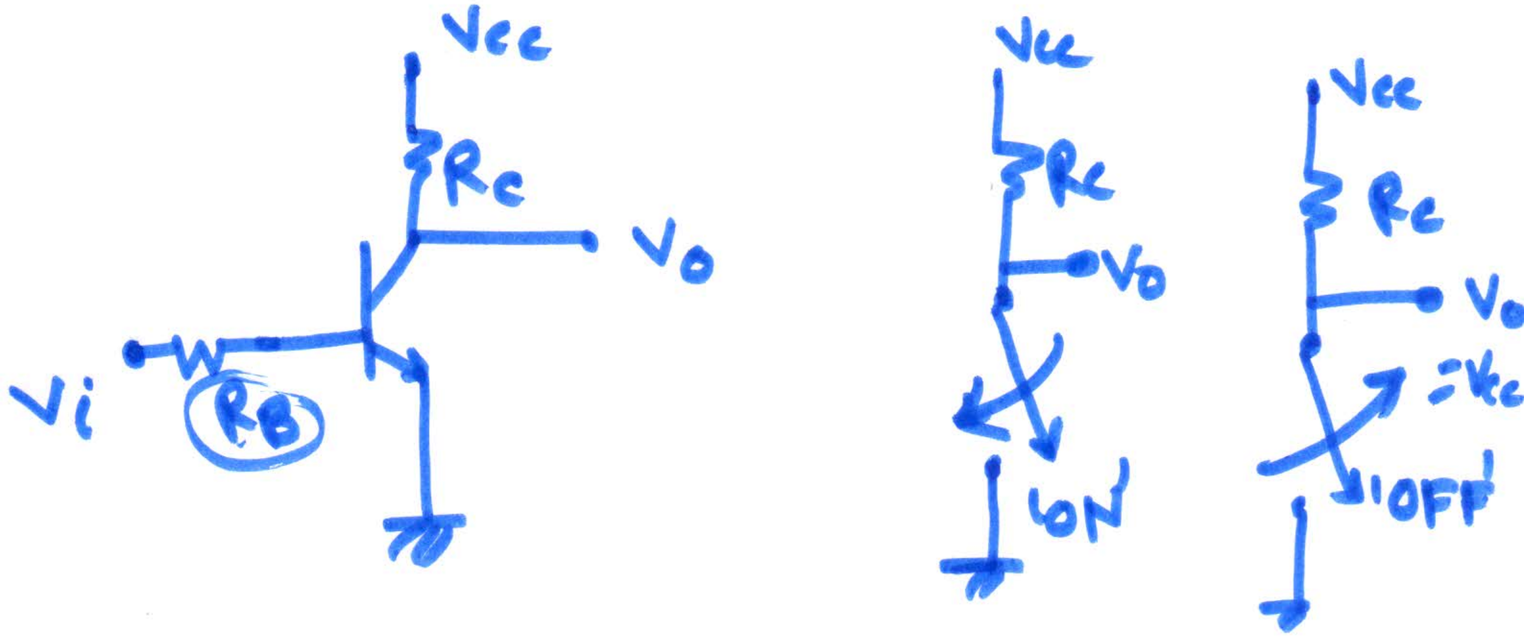
$$V_{CE} = V_{CC} - I_C R_C$$

$$I_C = \beta I_B$$

$$R_B\downarrow \rightarrow I_B\uparrow \longrightarrow I_C\uparrow \rightarrow V_{CE}\downarrow$$

$$I_B = \frac{V_{BB} - 0.7}{R_B}$$

$$V_{CE,sat} = 0.2V$$



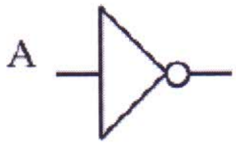
$V_i = 5V$, $V_o = 0.2V \approx 0V$

$V_i = 0V$ $V_o = V_{cc} = 5V$

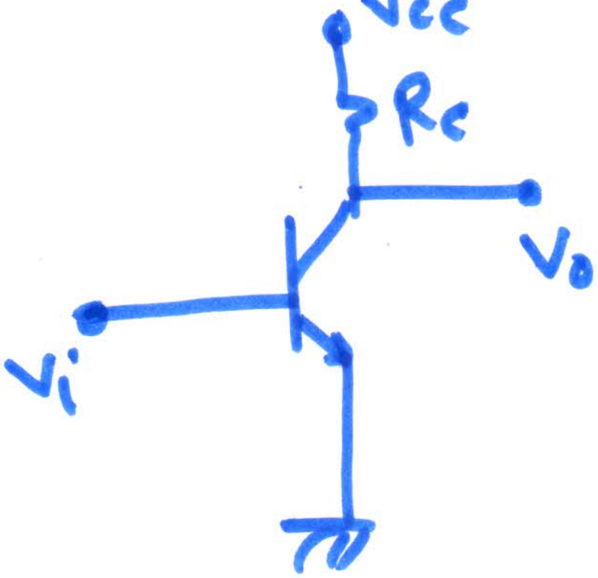
Logic Gates

1. NOT gate (an inverting gate)

A	Y
0	1
1	0



$Y = \overline{A}$

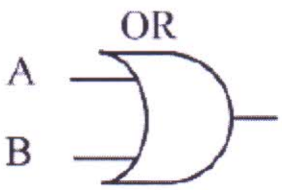


V_i	A	V_o	Y
Low	'0'	High	'1'
High	'1'	Low	'0'

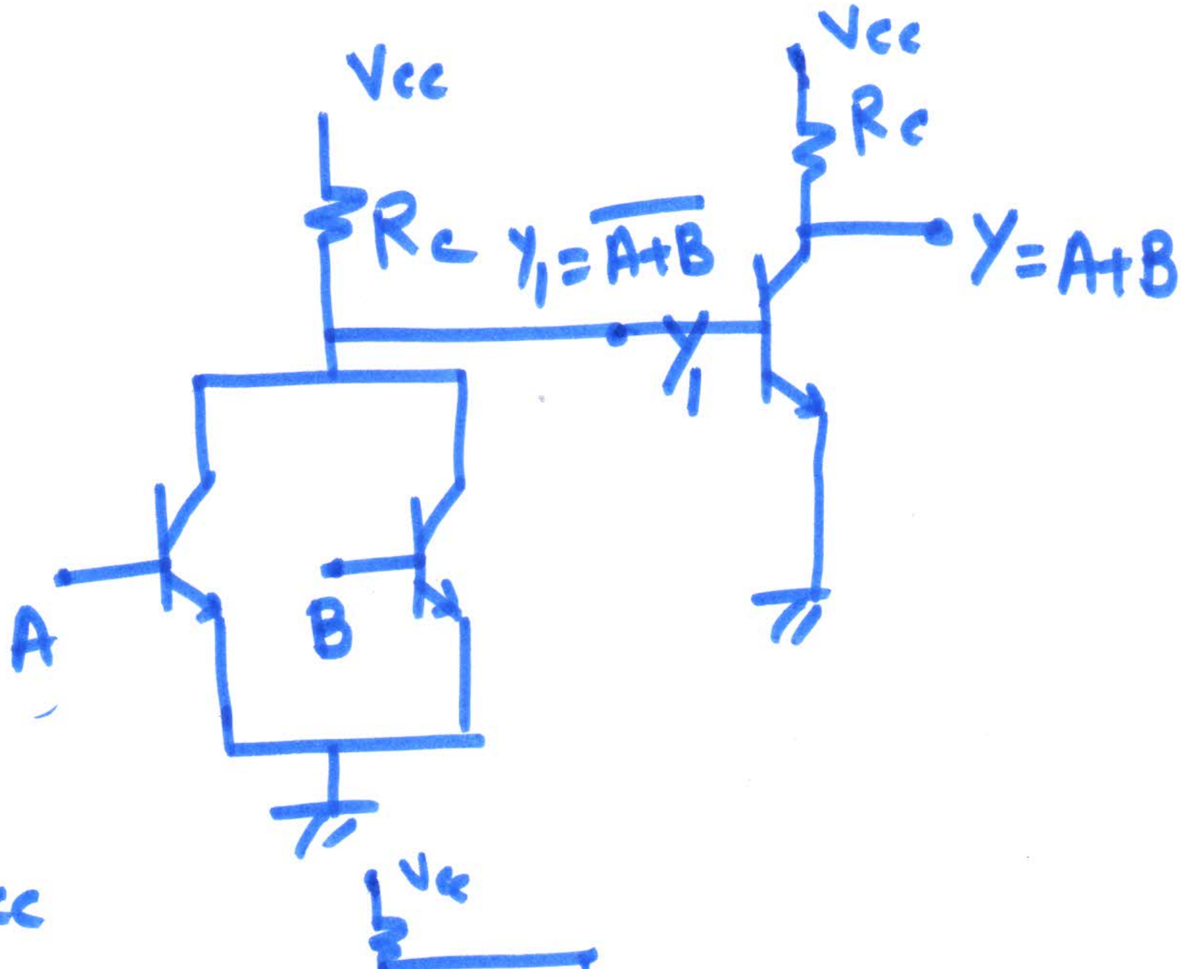
2. OR gate

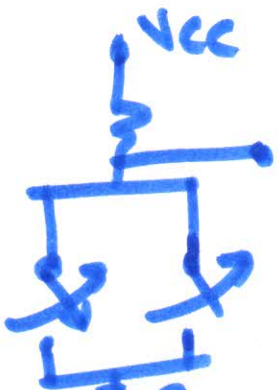
TT

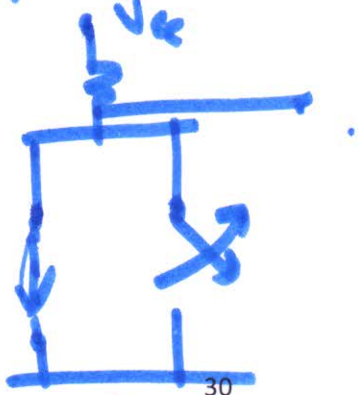
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



$Y = A+B$



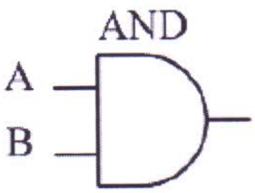




$A=0, B=0$ $\qquad$ $A=1 \vee B=0$

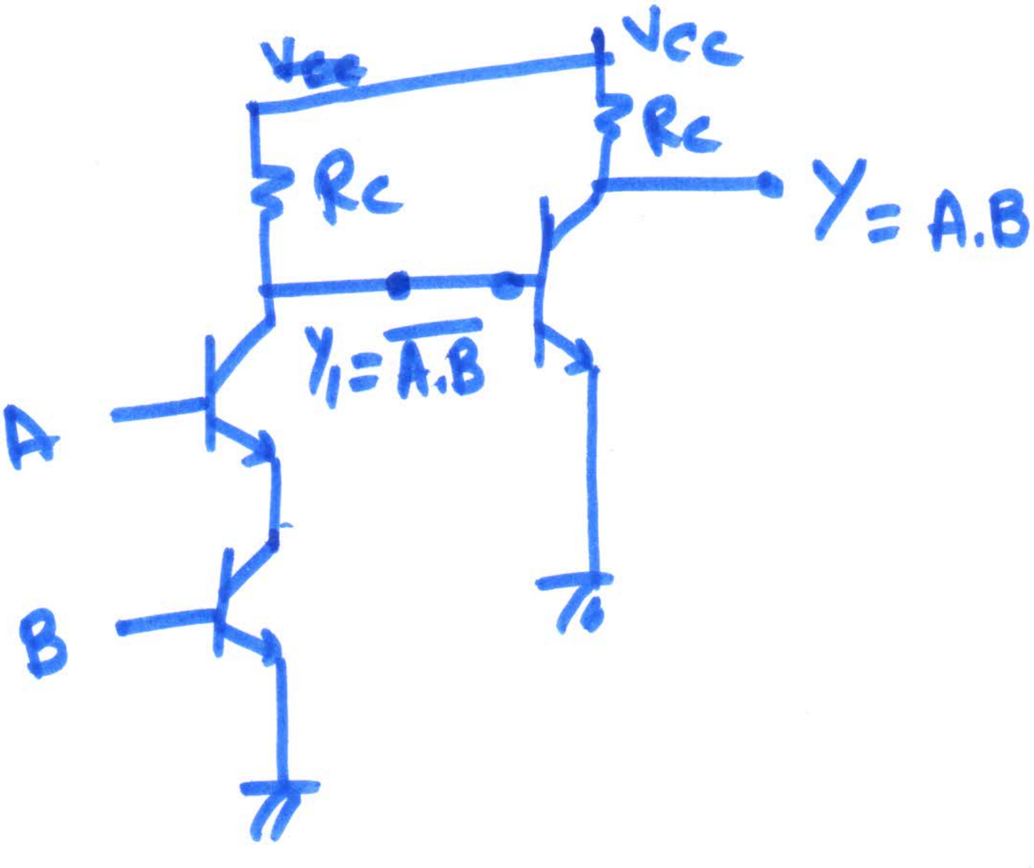
3. AND gate

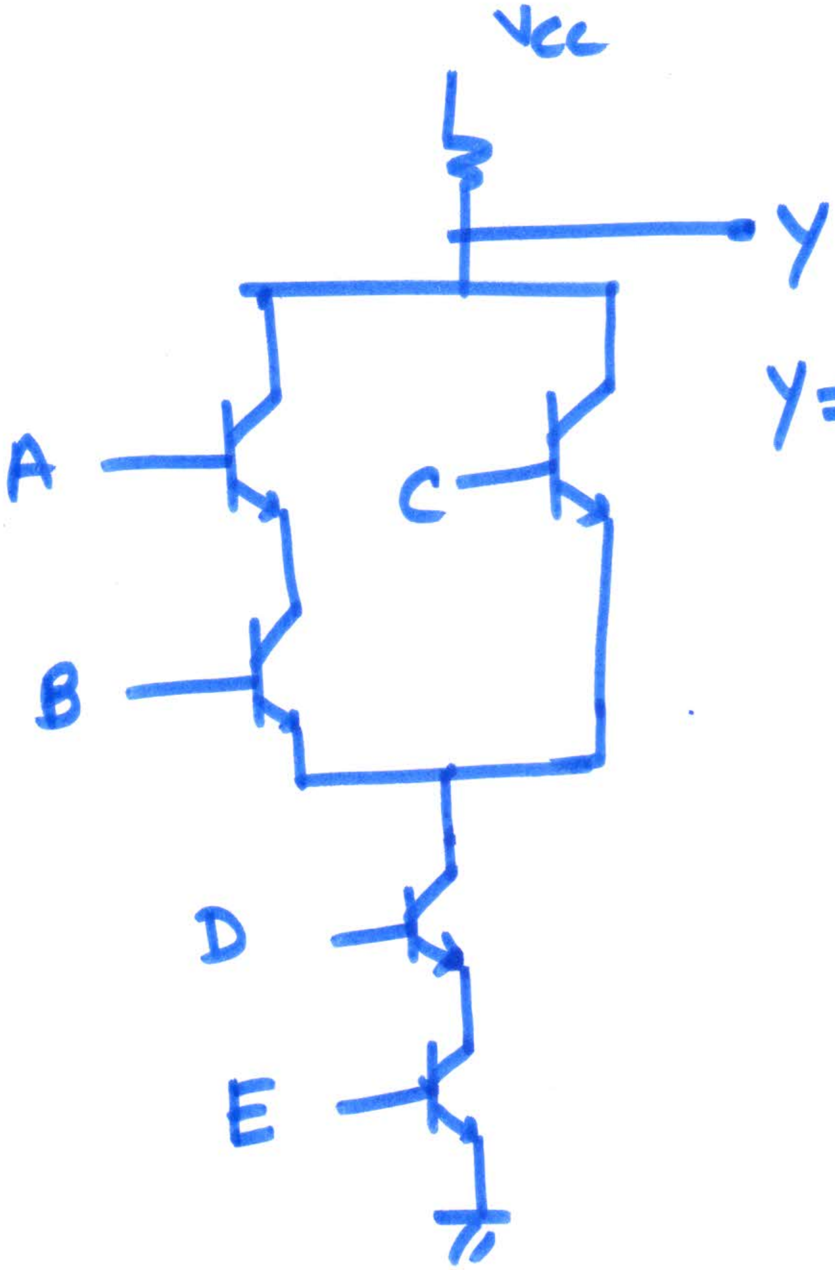
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



$Y = A.B$

$= AB$

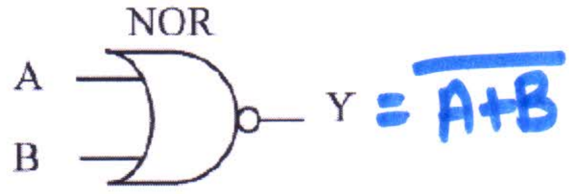


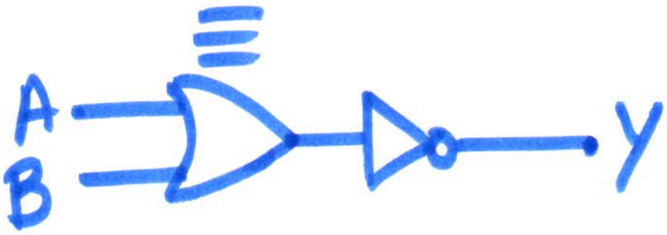


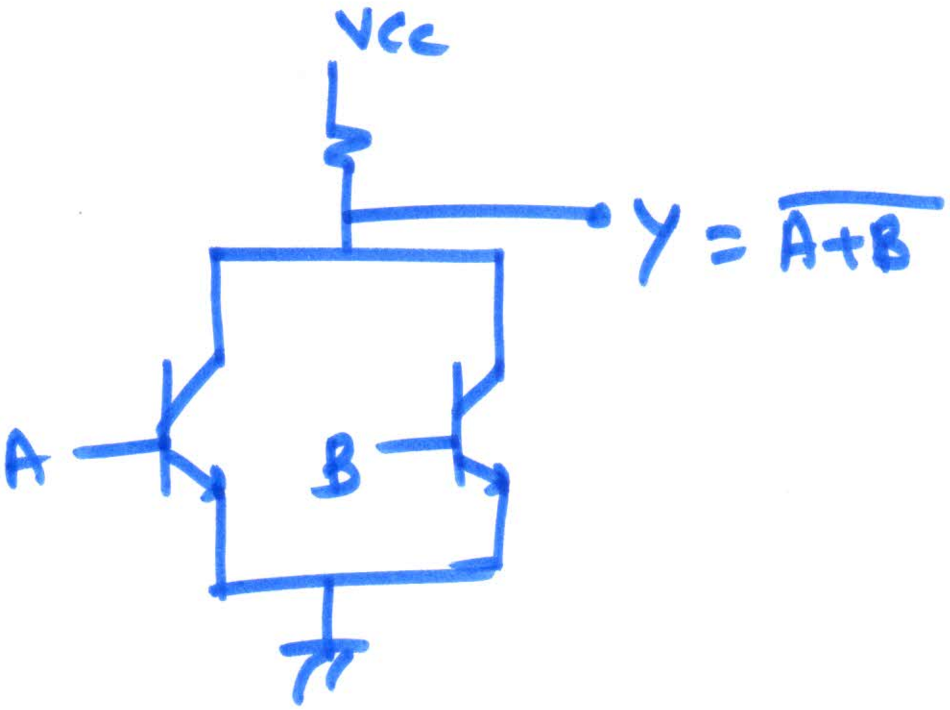
$$Y = \overline{(A.B + C).D.E}$$

4. NOR gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

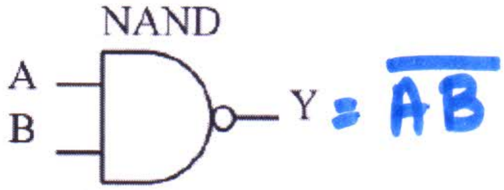


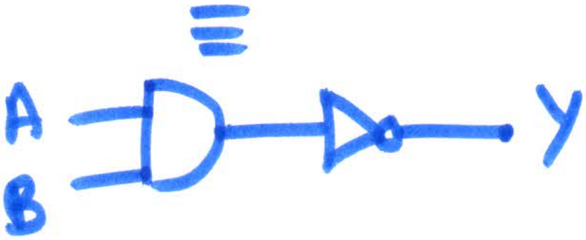


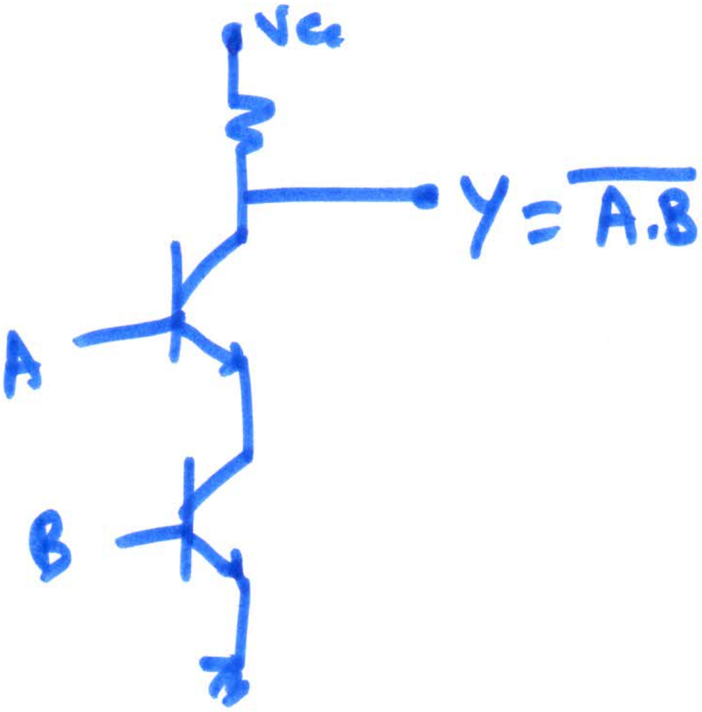


5. NAND gate

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

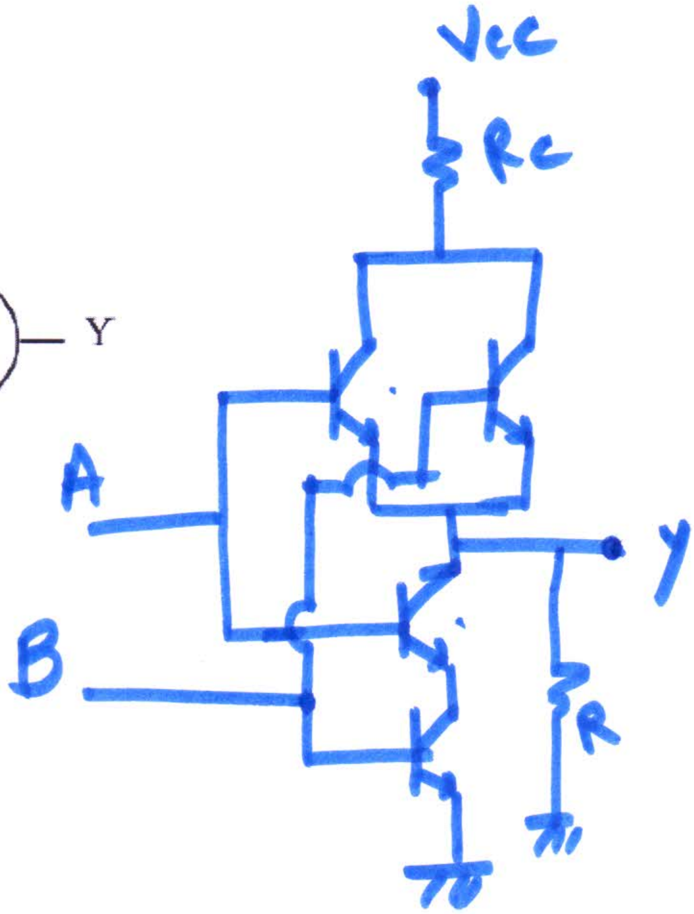






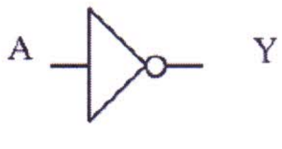
6. XOR gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

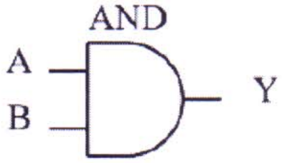


Basic Logic Gates

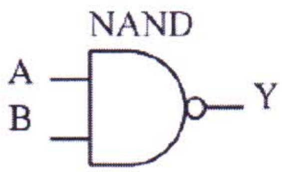
A	Y
0	1
1	0



A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

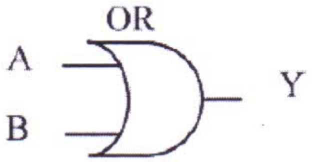


A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

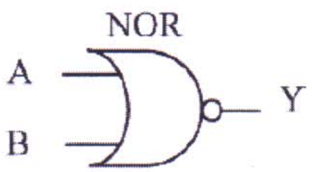


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

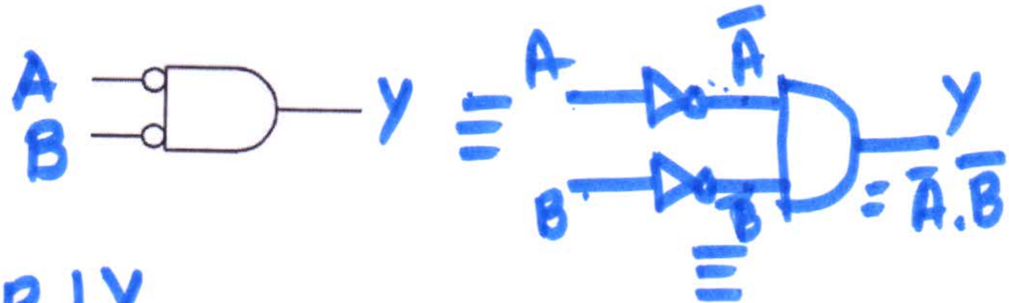
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

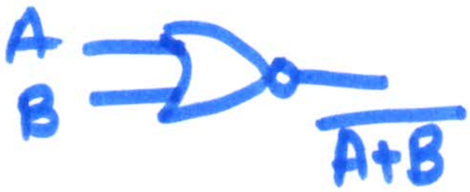


A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



Bubbled (Negated) AND Gate





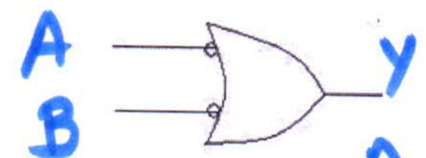
TT

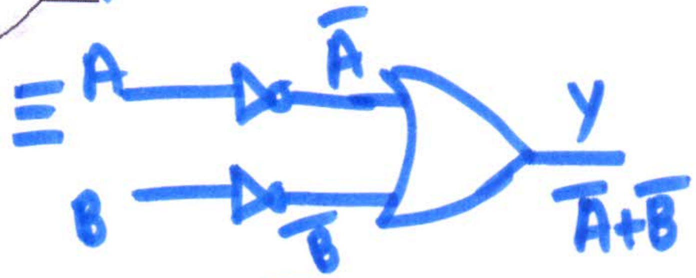
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

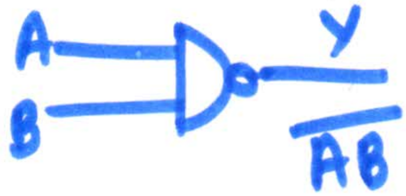
NOR

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

Bubbled (Negated) OR Gate







TT

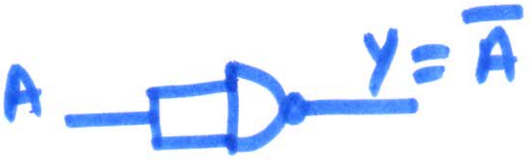
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

NAND

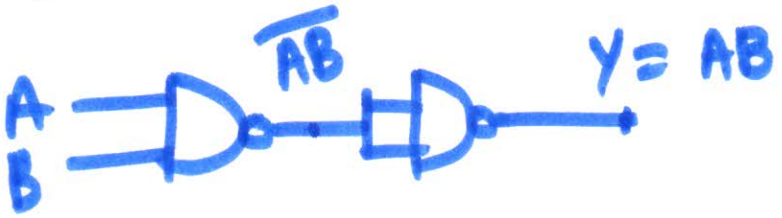
$$\Rightarrow \overline{AB} = \bar{A} + \bar{B}$$

NAND Gate as Universal Gate

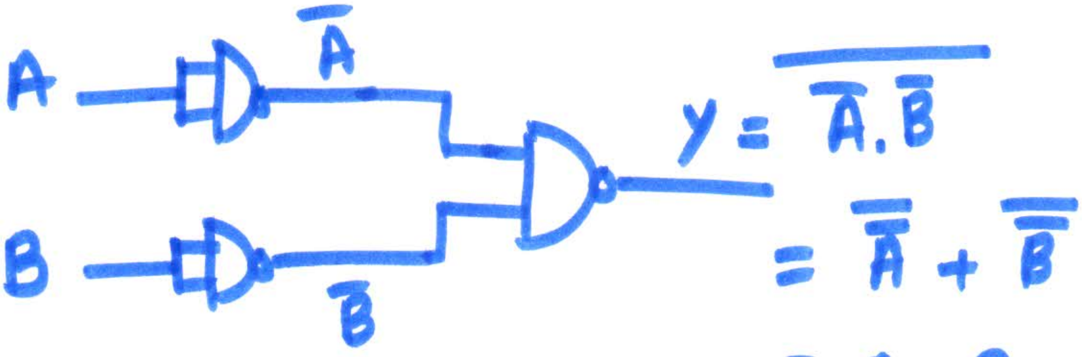
1. NOT



2. AND



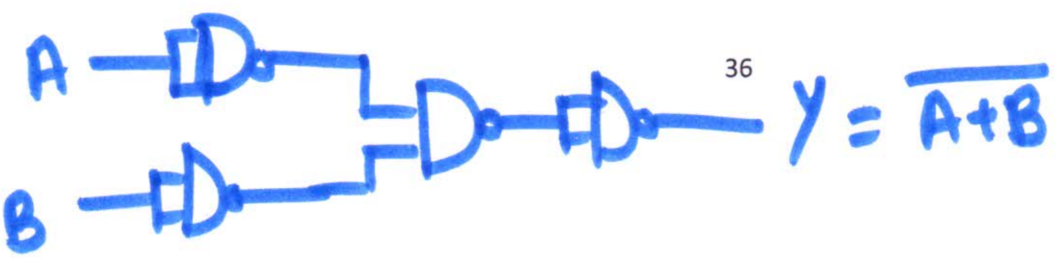
3. OR



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

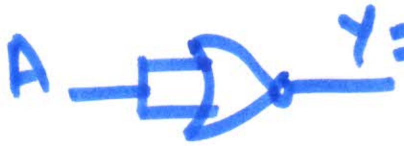
$= A + B$

4. NOR



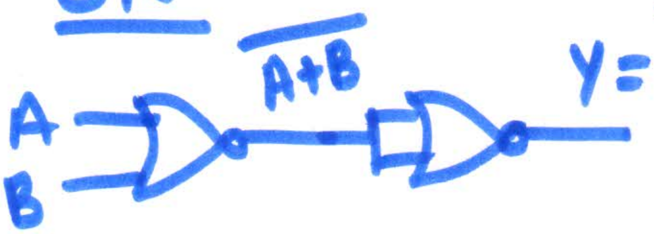
NOR Gate as Universal Gate

1. NOT



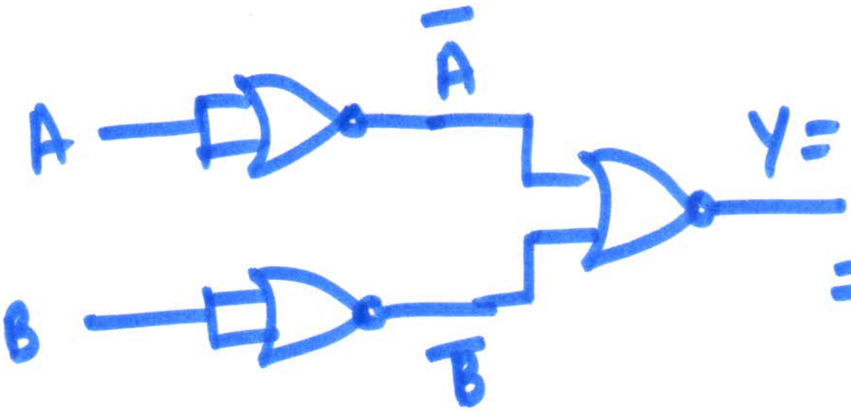
$Y = \overline{A}$

2. OR



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

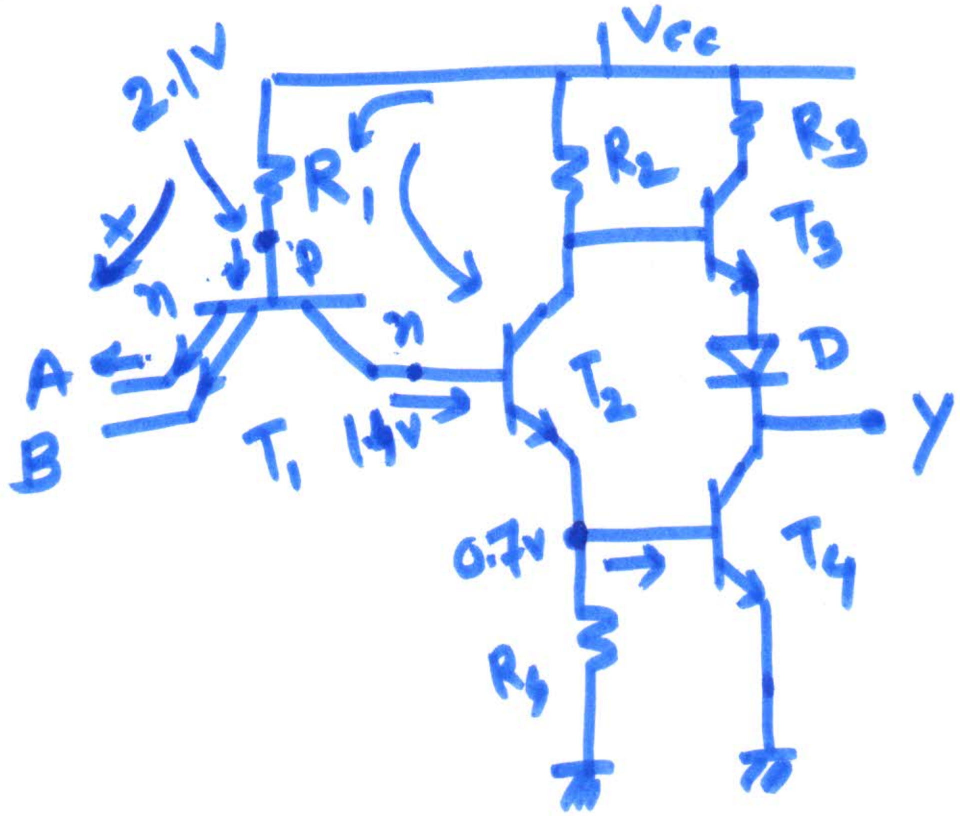
3. AND



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

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

NAND



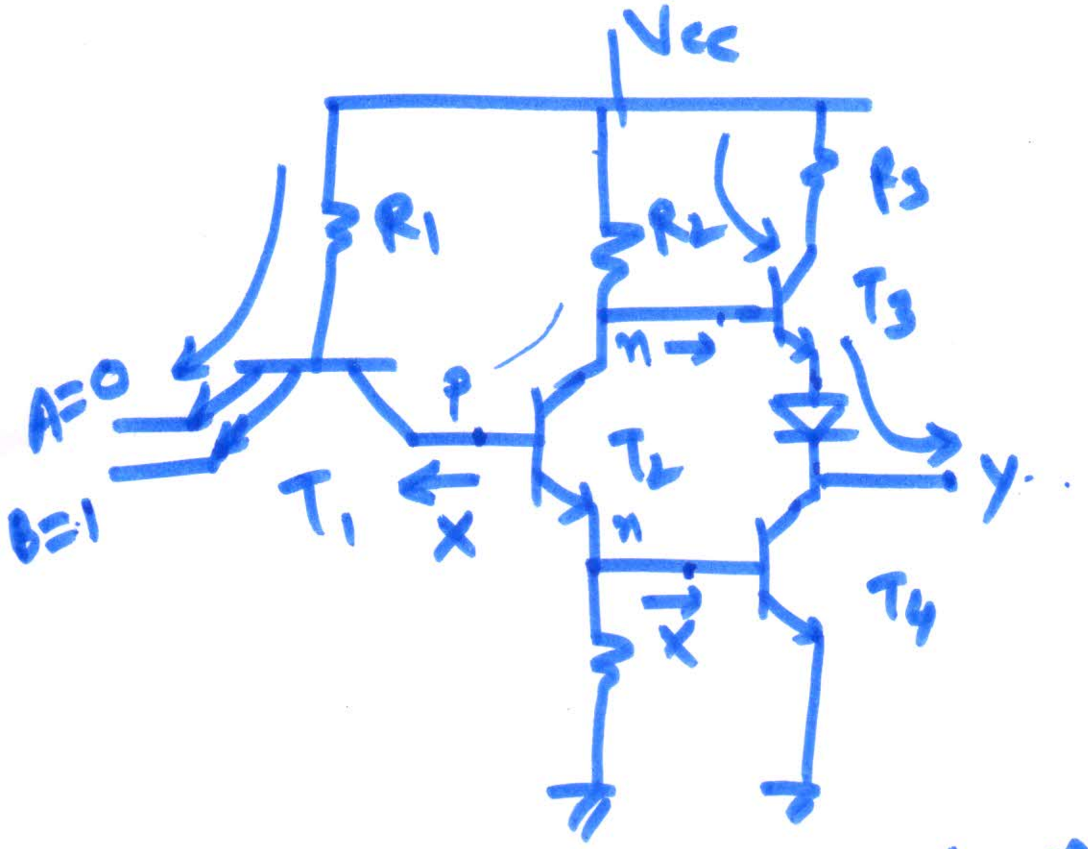
1) $A=1$ & $B=1 \Rightarrow T_1 \rightarrow$ Inverse Active

$T_2 \rightarrow$ ON (Active)

$T_4 \rightarrow$ Saturation

$\Rightarrow Y = 0.2V =$ '0'

2) $A=0$ & $B=1$



'Totem pole'

$$I_E = I_B + I_C$$

$T_1 \rightarrow$ Diode

$T_2 \rightarrow$ 'OFF'

$T_4 \rightarrow$ 'OFF'

$T_3 \rightarrow$ 'Saturation'

$Y = $ '1'