Mod 1

1.1 Intro. Binary : number systems
Octal
Hexadecimal

Binary code : Codes.
BCD code
excess - 3 code
gray code
ASCII code

1.2 Binary arithmetic,
Binary addition & subtraction (1's complement & 2's complement).

1.3 Gates: NOT, AND, OR, NAND, NOR,
XOR, XNOR.

1.4 Theorems & properties of Boolean Algebra.
SOP & POS form
K maps (2, 3, 4) — simplification.
NAND - NOR realization.

Binary → 2

Decimal → 10

octal → 8

Hexadecimal → 16

(÷)   (x^n 4 +)

CONVERSION OF DECIMAL & TO DECIMAL.

To convert anything to Decimal   (n, ---- , 2, 1, 0) ← powers

→ for eg: we are given 101_2, now we have to multiply the powers of 2 to each digit of 101. & add them.

→ for eg: we are given 64_8, now we have to multiply the power of 8 to each digit of 64 & add then
↳ (descending order)

→ ① $101_2 = 1\times 2^2 + 0\times 2^1 + 1\times 2^0$
$= 4+0+1 = 5 = 101$

→ ② $6\cancel{8}8_8 = 6\times 8^1 + 8\times 8^0$
$= 48 + 1 = \cancel{49}\ 51$ (in octal 63_8 = 51 in decimal)

→ ③ $(\cancel{1A})_{16} = 1\cancel{A}\times 16^2 + 1\times 16^1 + 0\times 16^0$ X $1\times 16^1 + 10\times 16^0$
$(110)_{16} = 256 + 16 = (272)_{10}$ $= 16 + 10 = 26 = (1A)$.

To convert decimal to binary (÷ by 2) → (remainder = Binary code)

→ for going decimal to binary → divide the given decimal numbers by 2 & (remainder is the binary code)

→ for eg: we are given 16   eg: we are given 17.

2	16	
2	8	0
2	4	0
2	2	0
	1	0

$= (10000)_2 = 16$

2	17	
2	8	1
2	4	0
2	2	0
	1	0

$(10001)_2 = 17$

decimal → octal ÷ by 8
→ binary ÷ by 2
→ hexa ÷ by 16.
} remainder = ans

PAGE No.
DATE / /

To convert decimal to octal (÷ by 8) (remainder = octal code)

→ For going from decimal to octal, divide the decimal value by 8, the remainder we get is the octal code.

for eg: we are given $(26)_{10}$.

∴ 8 | 26
  3 | 2 ↑

$(32)_8 = (26)_{10}$

eg: we are given $(44)_{10}$

8 | 44
5 | 4 ↑

$(44)_8 = (54)_{10}$.

To convert decimal to hexa decimal (÷ by 16) (remainder = hexa code)

→ For going from decimal to hexa decimal, divide decimal by 16, the remainder we get is hexadecimal value.

For eg: we are given $(26)_{10}$.

∴ 16 | 26
   1 | 10 → A = 1A

$= (110)_{16}$

* now if we get value of in remainder betn (10-15), we write them as (A-F)

for eg: 16 | 46
        2 | 14 → E

2E

Binary → octal } seperate binary no. into parts of 3 digits (octal) & 4 digits (Hexa)
→ Hexa

PAGE No.
DATE / /

convert Binary to octal : (3 digits)

① Take the binary code

② Divide the binary code into parts of 3 digits

③ convert the parts of 3 digits to numbers

④ combine numbers.

for example : 10|011|110|001

① seperating binary into part of 3 digits

→ 10 , 011 , 110 , 001   4 2 1.

② conver 3 digit parts to numbers.

→ 2 , 3 , 6 , 1

③ combine numbers → $(2361)_8$

Convert binary to Hexadecimal (4 digits)

* similar to octal conversion

① Take binary code.

② Divide binary code into parts of 4digits

③ Convert parts of 4 digits to numbers.

④ combine.

For eg |100|1111|0001|

→ seperating into parts of 4 digits

→ 100 , 1111 , 0001

→ converting to number → 4 , F , 1

= 4F1 //

① seperate into 4 digits.

② convert seperated 4 digits into numbers.

octal
&
hexa decimal → Binary } ① convert to decimal
② Decimal to binary.

Conversion of octal to Binary

① convert octal to Decimal.
i.e. mutiply by powers of 8 in descending order.
& add digits.

② Take the decimal number & make its binary.
→ by ($\div$ by 2) → remainder = binary

eg: $(63)_8$

① convert to decimal. 8^n.
$6 \times 8^1 + 3 \times 8^0 = 51$.

② decimal to binary. ($\div$ by 2)

2	51	
2	25	1
2	12	1
2	6	0
2	3	0
	1	1

$= (110011)_2$

Conversion of hexa decimal to Binary.

① hexa → decimal
② Decimal → binary.

① convert hexa decimal to Decimal.
→ multiply by powers of $(16)^n$ in decending order
& then add.

eg: $(63)_{16}$

① Decimal to Binary.
→ Divide by (2)

→ $6 \times 16^1 + 3 \times 16^0 = (99)_{10}$

2	99	
2	49	1
2	24	1
2	12	0
2	6	0
2	3	0
	1	1

$(1100011)_2$

Octal to Hexa & Hexa to octal

Octal to Hexa

① convert octal to Decimal

→ × by powers of (8) & (+)

② Take decimal & convert to hexa

(÷ by 16)

eg: $(76)_8$

① convert octal to decimal.

$= 7 \times 8^1 + 6 \times 8^0$

$= 56 + 6 = (62)_{10}$

② Decimal to hexadecimal.

(÷ by 16)

16 62

3 14 → E

$(3E)_{16}$

Hexa to octal

① Convert hexa to Decimal

→ × by powers of (16) & (+)

② Take decimal & convert to octal

÷ by 8

eg: $(1A)_{16}$ A = 10

① convert hexa to Decimal

$= 1 \times 16^1 + 10 \times 16^0$

$= 16 + 10 = (26)_{10}$

② Decimal to ~~hexa~~ octal

(÷ by 8)

8 26

3 2

$= (32)_8$

Eg   Decimal to Binary

① $(13.375)_{10} \rightarrow$ Binary.

2	13	
2	6	1
2	3	0
	1	1

(1101)

= (1101.011)

375 → 0.375 × 2 = 0.750 → 0.
→ 0.750 × 2 = 1.500 → 1
→ 0.500 × 2 = 1.00 → 1.
= (011)

* Keep multiplying by 2, till we get 1.

Sign Magnitude Method :

* 1 bit is reserved for sign of the number. i.e. (MSB)
  * 0 = +ve
  * 1 = -ve

* Remaining bits are the number.

Eg (+4) (-ve 4)

= 0100 = 1100
↳ signbit.

Eg					
4	0100	4	0100	∴	0100
+3	0011	-3	+ 1011		- 0011
7	0111	1	≠ 1111		0001

compare & subtract → -4 = 1100

+3 − 0011

-1 1001 = (-1).

#

1s complement Method: (invert all bits of +ve no = -ve no.)

* For +ve same as sign magnitude method.

  i.e. +13 = 0 1101

* For -ve

  ① need to invert all bits of same +ve no.

eg: -13 in 1s complement.

= invert all bits of +13

i.e = 0 1101 = +13

now invert all bits

∴ (-13 = 1 0010)

invert
1 → 0
of +ve
= -ve

2s complement Method (1st complement +1)

* for +ve no. same as sign magnitude method

  i.e. +13 = 0 1101

* for negative numbers

  ① Take 1s complement

  (i.e. invert bits of same +ve no)

  ② Add (+1) to 1s complement.

eg: -13 in 2s complement

① Take 1s complement (invert bits of +13)

= 01101 = +13

∴ -13 in 1s complement = 10010

∴ -13 in 2s complement = (1s complement +1)

= 10010 +1 = (10011) = -13 2's compl.

→ +13 = 01101

∴ 1s complement

= -13 = 10010

in 2s complement

= 1s complement +1

= 10010 +1 = 10011

Rules for addition

A	+ B	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$1+1=0$ carry ①.

Rules for subtraction

A	- B	diff	burrow carry
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0.

$(0-1)=1$ burrow $=1$.

* $85-28 = 85+(-28)$

-28 = 2s complement or 1s complement.

1s complement addition (add carry to answer)

eg: $85+(-28)$

① take 1s complement of (-28).

② Add 85 + 1s complement of (-28)

③ If carry is generated, add that to answer)

2s complement addition (ignore if carry is generated)

eg: $85+(-28)$

① take 2s complement of 28.

② Add 85 + 2s complement of (28).

③ (If carry is generated, ignore)

PAGE No. DATE / /

Classification of codes

* Weighted codes → Binary, 8421, 2421, etc.
  → each position represent some value.

* Non weighted codes → XS-3 code, Gray code.

* Alphanumeric code → ASCII code.

Binary coded Decimal : (binary code for each digit) (4bit)

* ① Each decimal digit is represented by a 4 bit binary number.

Basically, write binary code for each digit.

Decimal	BCD (same as binary just 4 bits).
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Decimal to BCD.

* $(10)_{10}$ = 1 0
  = 0001 0000
  = $(00010000)_{BCD}$.

* $(13)_{10}$ = 1 3
  = 0001 0011
  = $(00010011)_{BCD}$

* $(17)_{10}$ = 1 7
  = 0001 0111
  = $(00010111)_{BCD}$

~~BCD~~ ~~to~~ Decimal to BCD :

① Take decimal no.

② write the binary code for each digit of decimal.

eg: $(227)_{10}$

= 2 = 0010 → (0010 0010 0111)

2 = 0010

7 = 0111

BCD to Decimal : (group BCD into 4 bits) & (then calc. decimal value)

similar to binary to hexa

① Group the ~~BCD~~ BCD into 4 bit group. &

② Take decimal value of 4 bit groups.

eg: |0010|0010|0111|

① ② ③.

① ⇒ 2

② ⇒ 2 $= (227)_{10}$

③ ⇒ 7

Eg 10 in binary 10 in BCD

10 = 1010. 10 = 1 & 0.

= (0001 0000)

PAGE No.
DATE / /

(self complementing)

Excess - 3 code (XS - 3 code) $\boxed{\text{Add 3 to BCD}}$.

Add 3 to BCD

** convert Decimal to XS - 3 code.

① convert Decimal to BCD.
② Add (0011) (3) to BCD.
= XS 3 code

eg: 5 → BCD = 0101 → BCD + 3 = 0101
+ 0011

XS-3 code for 5 = 1000 = 8

Decimal	BCD	BCD + 3
0	0000	0000 + 0011 = 0011
1	0001	0001 + 0011 = 0100
2	0010	= 0101
3	0011	= 0110
4	0100	= 0111
5	0101	= 1000
6	0110	= 1001
7	0111	= 1010
8	10000	= 1011
9	1001	= 1100

∴ XS 3 code for 24.
① convert 24 to BCD.
② Add 0011 to BCD (24).

→ BCD 24 = 00100100
→ BCD 24 + 3 = 00100100
+ 00101

$\boxed{00100111}$ = XS 3 code for 24

PAGE NO.
DATE / /

* XS - 3 code for digits above 9.

① Take the BCD of the number.

② Add 3 to each BCD value of the number.

Eg XS - 3 code for 24.

① BCD of 24 = 2 4.

(0010 0100)

② Add (0011) to 2 BCD & (0011) to 4 BCD.

= 0010 + 0011 = 0101 ; 0100 + 0011 = 0111

= (01010111)

Eg. XS 3 code for 156.

① BCD of 156 = 1 5 6.

0001 0101 0110

② Add 3 to each digit BCD.

= 0001 + 0011 = 0100 ; 0101 + 0011 = 1000 ; 0110 + 0011 = 1001

= 010010001001

Binary → Gray, Gray → Binary } $\boxed{\text{MSB remains the same.}}$

Graycode : $Y = A\bar{B} + \bar{A}B$ (XOR)

- Reflected Binary Code (RBC).

* two successive value differs in only one bit.

* Binary number is converted to Graycode to reduce switching operation.

	Binary				Gray			
	B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
3	0	0	1	1	0	0	1	0
4.	0	1	0	0	0	1	1	0

∴ to change from 3 to 4, we need to change 3 bits.

∴ to change from 3 to 4 in Gray code we need to change only one bit

Binary to Graycode Conversion

① Record MSB as it is

* ② Add MSB to next bit, record the sum neglect carry

③ Repeat ②.

* Eg convert 1011 to Gray code.

→ 1 0 1 1

↑ MSB.

1. Record MSB as it is. = 1.

2. Add MSB to next bit = 1 + 0 = 1.

(record sum, neglect carry).

repeate process

= 0 + 1 = 1 = (1110)

= 1 + 1 = 0

~~1 0 1 1~~

~~+1 +0 +1~~

~~1 0 10 0~~

Binary to ~~BCD~~ Gray :

1. Record MSB

2. Add 1st & 2nd pos
   Add 2nd & 3rd pos
   Add 3rd & 4th pos

Record sum ignore carry.

-- continue till last element

eg : 1110.

MSB = 1.

1 + 1 = 0.

1 + 1 = 0 = (1001)

1 + 0 = 1

1 1 1 0

+1 +1 +1

1 0 0 1

1 0 1 1

+1 +0 +1

$\boxed{1\ 1\ 1\ 0}$ ✓

Binary to Gray code → add 1st & 2nd, 2nd & 3rd, 3rd & 4th

Gray to Binary → add sum to next posinon.

* add previous sum to next posinon.

PAGE No.
DATE / /

Graycode to Binary Conversion: [~~same as Binary to Gray~~]

① Record MSB as it is.

②. Add MSB to next bit of graycode.

Basically

~~① Record MSB~~

~~② Add 1st & 2nd position~~

~~Add 2nd & 3rd position~~

~~Add 3rd & 4th position~~

① record sum

② Ignore carry

1 1 1 0
↓ +1 +0 +1
1 0 1 1

~~Eg. 1110 ← Gray code.~~

~~= 1, 1+1 = 0, 1+1 = 0, 1+0 = 1~~

~~= 1001~~

eg: 1110 ← Gracode.

= 1 1 1 0

+ 0 +1 +0 +1

1 0 1 1

$B_1 = G_1$

$B_2 = G_2 \oplus B_1$

$B_3 = G_3 \oplus B_2$

$B_4 = G_4 \oplus B_3$

Basically

① Record MSB + 0 = MSB.

② Add MSB + 0 = MSB to next bit. = sum ①.

Add sum ① to next bit = sum ②.

Add sum ② to next bit = sum ③.

MSB	x	y	z.
+ 0	+MSB	+sum1	+sum 2
MSB	sum1	sum2	sum 3.

Basically add result of previous sum to next bit

eg. 1 0 0 1 (Gray code)

↓ +1 +1 +1

1 1 1 0

if 1001 = binary to Gray

∴ 1, 1+0, 0+0, 0+1

= (1 1 0 1)

ASCII - codes (Group 7 bits in binary).

* 48 - 57 = 0 to 9.
* 65 - 90 = A - Z capital.
  97 - 122 = a - z small

ASCII

Binary → Decimal → Ascii

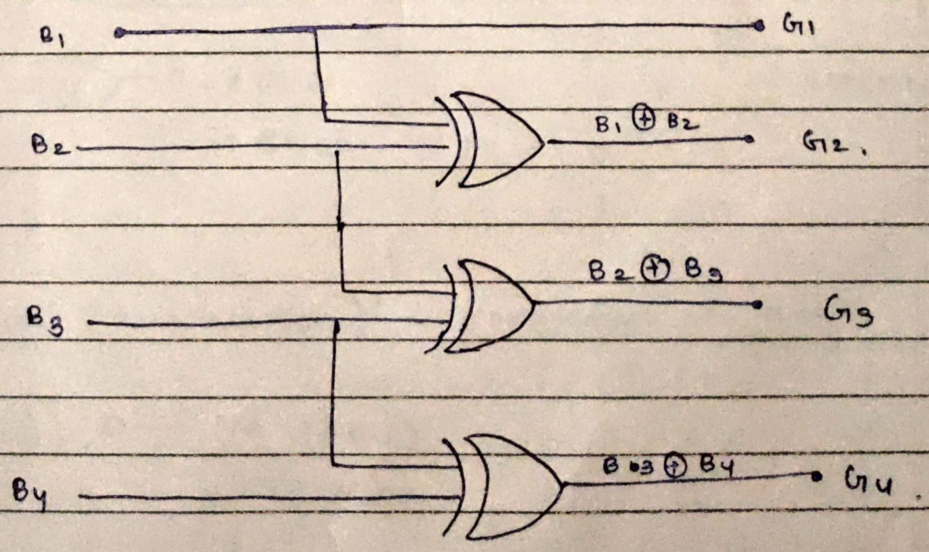
Group 7 bits in binary.
convert each group to decimal.

$\underbrace{1010001}_{81}\underbrace{1001011}_{75}$

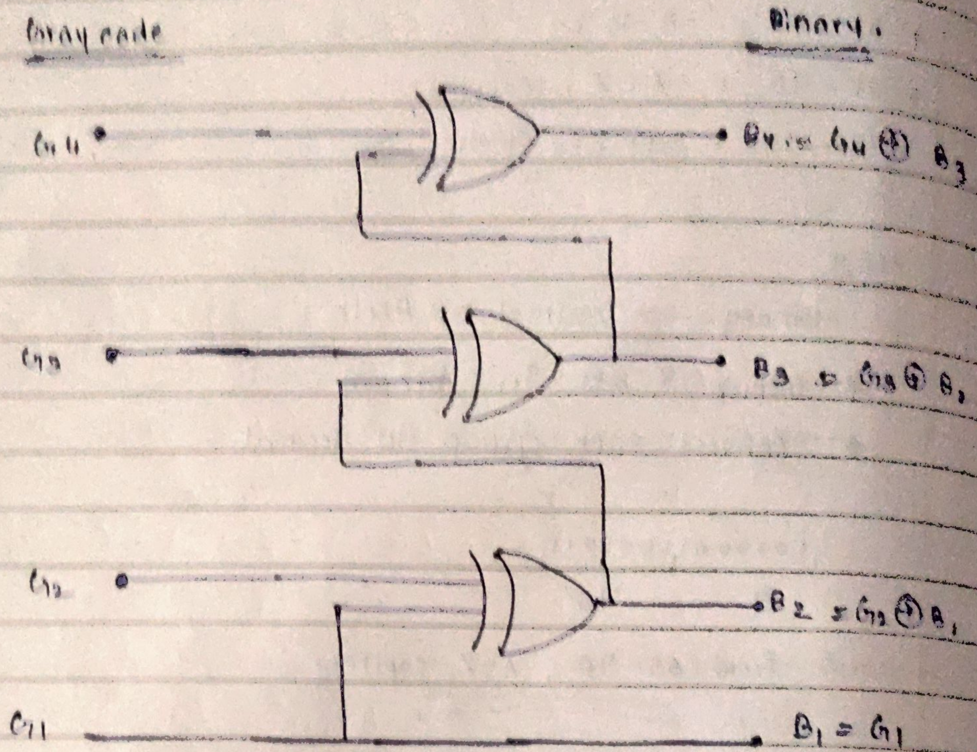
∴ from 65-90, A-Z capital.

Binary to Gray code = XOR. (Just apply XOR)

Binary Bits. Gray bits.



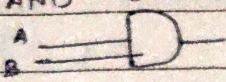
Gray code to Binary (XOR logic)



BCD to XS-3 code

Basic Logic Gates

① AND (Multiply).

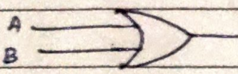


$Y = A \cdot B$

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

AND (×).

② OR (ADD).



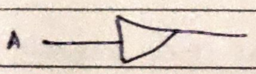
$Y = A + B$

$1 + 1 = 1$.

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

OR (+).

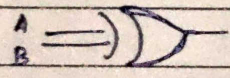
③ NOT (complement).



A	Y
1	0
0	1

NOT (complement).

④ X OR (PURE ADDITON)



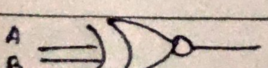
$Y = A \oplus B$

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

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

XOR (pure addition)

⑤ X-NOR (pure addition)' = complement of (XOR)' ⊙.



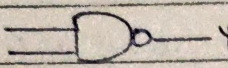
$Y = (A \oplus B)'$

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

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

$\boxed{(XOR)' = XNOR}$

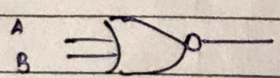
(6) **NAND** $X = (A \cdot B)'$ (complement of AND)


$Y = (AB)'$

(A ·	B)'	Y.
0	0	1
0	1	1
1	0	1
1	1	0.

(7) **NOR** $X = (A + B)'$ (complement of OR)

OR = $\overline{(1+1 = 1.)}$


$Y = (A + B)'$

(A +	B)'	Y
0	0	1
0	1	0
1	0	0
1	1	0

serial → use AND.

parallel → use OR.

PAGE No.
DATE / /

Boolean Algebra

* Commutative law	* Associative law	* Distributive law
(10) • $A+B = B+A$	• (12) $(A+B)+C = A+(B+C)$	(14) • $A\cdot(B+C) = AB + AC$
(11) • $A\cdot B = B\cdot A$	• (13) $(A\cdot B)\cdot C = A\cdot(B\cdot C)$	(15) • $A+(B\cdot C) = (A+B)\cdot(A+C)$

Duality theorem

① change each OR to AND

② change each AND to OR

③ complement each 0 & 1

Fundamental laws.

* AND laws

$x = 1 \quad x' = 0.$

① $x \cdot x = x.$

② $x \cdot 1 = x$

③ $\boxed{x \cdot 0 = 0}$ AND

④ $\boxed{x \cdot \bar{x} = 0}$ Gate

because $x = 1$

$\bar{x} = 0.$

* OR laws.

$A = 01, \ \bar{A}$ & $A' = 0$

⑤ $A + 0 = A$

⑥ $\boxed{A + 1 = 1}$ $1+1 = 1$ OR

⑦ $A + A = A$ (Gate.)

⑧ $\boxed{A + \bar{A} = 1}$ $1+0 = 1$ OR

$0+1 = 1$

⑨ $\bar{\bar{X}} = X$

⑩ Top of page.

Dual of Distributive law

⑮ $(X + (YZ)) = (X+Y)\cdot(X+Z)$

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

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

PAGE No

DATE / /

DeMorgans Theorem

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

converting AND to OR or OR to AND

Identity Theorem :

* $x + (\bar{x}y) = (x+y)$
* $x\cdot(\bar{x}+y) = (x\cdot y)$

→ $x + (\bar{x}y)$

$= (x+\bar{x})\cdot\cancel{(x+y)} = \cancel{(x+y)}$.

$x=1$, $\therefore \bar{x}=0$.

$= \cancel{0 \cdot (x+y)} = 0$. $x+\bar{x}=1$ $x=0$, $\bar{x}=1$.

$\therefore (x+\bar{x}) = 1$.

→ $x(\bar{x}+y) = x\cdot y$.

$(x\cdot\bar{x}) + \cancel{(y\cdot x)} = \cancel{(x\cdot y)}$.

$0 + (y\cdot x) = (x\cdot y)$

1 + any input = 1.

PAGE No.

DATE / /

$A = 1$, $\bar{A} = 0$, $B = 1$, $\bar{B} = 0$

(Q1) $Y = \bar{A}B + A\bar{B} + AB$.

take A common.

$\bar{A}B + A(\bar{B} + B)$. ..... $\bar{B} + B = 1$.

$\bar{A}B + A$

$\therefore (C + AD) = (C + A)\cdot(C + D)$

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

$\therefore (A + \bar{A}) = 1$

$\therefore (A + B)$

(Q2). $Y = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$

take $\bar{B}C$ common.

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

$\therefore (\bar{A} + A) = 1$.

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

take $\bar{B}$ common.

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

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

$(C + \bar{C}) = 1$.

$= \bar{A}B\bar{C} + \bar{B}[A + C]$.

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

(Q3) $F = \bar{y}\bar{z} + \bar{w}x\bar{z} + \bar{w}x\bar{y}\bar{z} + y\bar{z}$

take $\bar{y}\bar{z}$ common.

$= \bar{y}\bar{z}(1 + \bar{w}x) + \bar{w}x\bar{z} + y\bar{z})$

$(x + 1) = x$.

$= \bar{y}\bar{z}(1) + \bar{w}x\bar{z} + y\bar{z}$

take $\bar{z}$ common.

$\bar{z}(\bar{y} + y) + \bar{w}x\bar{z} = \bar{z} + \bar{w}x\bar{z}$.

$(\bar{y} + y) = 1$.

$\bar{z} + \bar{w}\bar{x}\bar{z}$

take $\bar{z}$ common.

$\bar{z}(1 + \bar{w}\bar{x})$

$(x+1) = 1$ ∴ $x = 1$

$= \bar{z}$

Solving XNOR.

→ $\bar{A}\bar{B} + AB$

→ $\overline{\bar{A}B + A\bar{B}}$

$= \overline{X+Y} = \bar{x}\cdot\bar{y}$ demorgans.

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

= again demorgans.

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

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

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

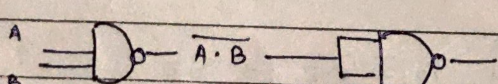
∴ $A\bar{A} + B\bar{B} = 0$.

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

Construct AND using NAND.

∴ $(A\cdot B)' \rightarrow A\cdot B$. (2 Nand Gates)

2 nand gates series



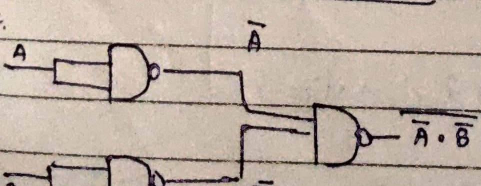
$A\cdot B$ = And Gate.

Construct OR using NAND. $(A+B) \leftarrow (A\cdot B)'$.

short input

2 nand gates parallel

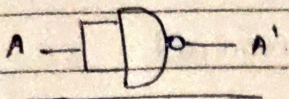
again nand gate



Apply demorgan to $\overline{\bar{A}\cdot\bar{B}}$

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

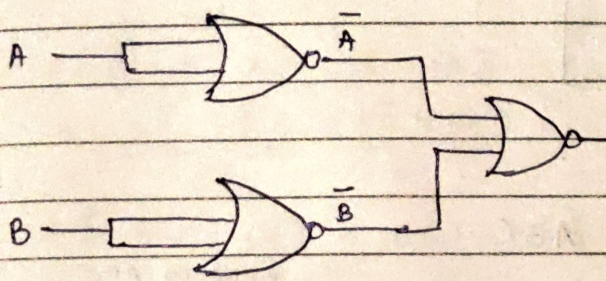
NOT using NAND: $(A \cdot B)' \rightarrow A'$



Construction of AND using NOR

$(A + B)' \rightarrow A \cdot B$.

2 NOR in parallel & other NOR gate.

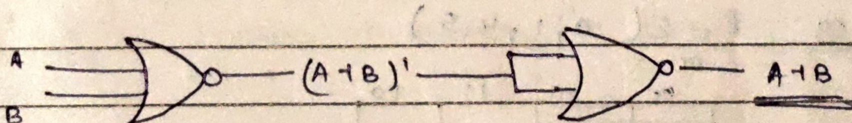


$\bar{A} + \bar{B}$ apply demorgans.

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

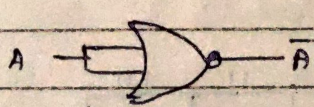
$= AND$

OR Gate using NOR $(A+B)' \rightarrow A+B$.



= OR gate

NOT Gate using NOR



= NOT Gate.

$\bar{A} = 0 \quad A = 1$

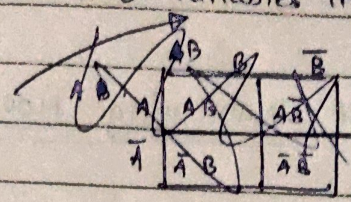
Kmap is always made for ~~Input~~ Output

K-maps

$N = 2^n$, N: no. of cells required

n = no. of variables in boolean exp.

Two variables →



	B	$\bar{B}$
A		
$\bar{A}$		

OR.

	B	$\bar{B}$
$\bar{A}$		
A		

Three variables → ABC.

A \ BC	00	01	11	10
0				
1				

Read as ABC.

eg: $f = \Sigma(0,1,4,5)$

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

quad → $\bar{B}$

4 variable kmap.

AB \ CD	00	01	11	10
00				
01				
11				
10				

| group (1) | (SOP)

Kmap for minterm (1).

PAGE No.	
DATE	/ /

ex. Plot eqn $AB + A\bar{B} + BC$. (Three variable.

BC

A	00	01	11	10
00		1	1	
01	1	1	1	1

* All Spots where A & B are 1
* All spots where A = 1 & B = 0.
* All spots where B & C = 1.

$A + \bar{B}C$

$A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C} + ABC$.

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

$A(\bar{B}(\bar{C}+C) + B(\bar{C}+C))$ $\bar{C}+C = 1$.

$A(\bar{B}+B) = A$ $\bar{B}+B = 1$.

$F = A + \bar{B}C$ Three variables.

$2^3 = 8$ prepare truthtable for $A + \bar{B}C$..

A	B	C	F.
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

$F = (\bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$
$+ ABC)$.
= (for which F = 1)

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

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

$= \bar{A}\bar{C} + AB + ABC$.

$= \bar{A}\bar{C} + AB(1+C)$.

$= \bar{A}\bar{C} + AB$. -- $1+x = 1$.

A ∑ ABC

8) F(A, B, C) = ∑ (1, 3, 5, 7)

= C

ACD + ABCD + ABCD + ACD

CD + ABC + ABD + CD + CAB

(C + D) · (A + B + C) · (C + D + B) · (C + A + B) group 0

[POS]

PAGE No.	
DATE	/ /

kmap for max-term 0.

eg $F(A,B,C,D) = \Sigma m\,(1,3,4,5,9,11,14,15)$.

↳ min terms (1).

$= \Pi m\,(0,2,6,7,8,10,12,13)$.

↳ max terms (0)

: $2^4 = 16$.

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

$(\bar{B}\bar{D}) + AB\bar{C} + \bar{A}BC$

& write it as, invert everything | complement everthing.

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

1. change A to $\bar{A}$

& change ($\cdot$ to $+$) & ($+$ to $\cdot$)

Standard forms

Sum of products (SOP) → minterms → 1 . (Σ) / (Σ)

SOP eqn = ABC + DEF.

Product of sum (POS) → maxterm → 0 . (Π).

POS eqn = $(A+B+C)\cdot(\bar{B}+C+D)$

→ Solved similar to SOP,

1. Plot '0' in the k map instead of 1.
2. Group '0' instead of 1
3. Get eqn in the form of SOP.
4. Convert SOP → POS.
   - → change (·) to (+) & viceversa.
   - → change $\bar{A} \rightarrow A$ & viceversa

SOP (minterms) POS (maxterms)