Digital Circuits

↳ 1 or 0

1. Digital Codes:
1. Binary Coded Decimal (BCD)
2. Gray
3. Parity
4. ASCII.

① BCD:

MSB ← Most significant bit

Least significant bit LSB

Decimal(D)	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

$2^3 \quad 2^2 \quad 2^1 \quad 2^0 \qquad 2^n$

$5_D \rightarrow 0\ 1\ 0\ 1$

$\hookrightarrow 0\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0$

$= 0 + 4 + 0 + 1 = 5$

$55_D \rightarrow 0101 \quad 0101$

Application: Numeric displays.

② Gray:

→ 1 bit changes for subsequent no.

Decimal	Gray code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

③ Parity: Error detection code.

In binary data transmission (long distance) unusual data change may occur.

Detects odd combination of change.

Parity → Odd
↳ Even } Types

First 4 Binary Data/bits or Message	Last/ 5th bit Odd Parity	Last/ 5th bit Even Parity
0000	1	0
0001	0	1
0010	0	1
0011	1	0
0100	0	1
0101	1	0
0110	1	0
0111	0	1
1000	0	1
1001	1	0
1010	1	0
1011	0	1
1100	1	0
1101	0	1
1110	0	1
1111	1	0

④ ASCII: American Standard Code for Information Interchange.

(Application: Keyboards for PCs)

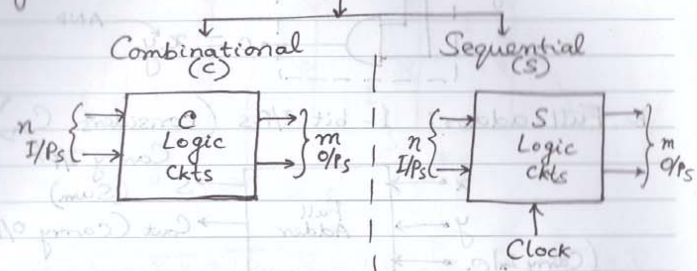
Special/extended binary code → Alpha-numeric code.

1 byte: 8-bits : $2^8 = 256$ possible codes/values

Refer to ASCII table from internet.

eg.	'a'	means	96_D	or	61_H
	'A'	"	65_D	or	41_H
	'1'	"	49_D	or	31_H
	','	"	44_D	or	$2C_H$

2. Digital circuits: Types

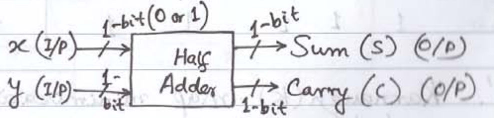


e.g. 1. Adders (Logic/ binary adders)
2. Subtractor
3. Decoder/encoder
4. Multiplexer/de-mux (mux)

~~1~~1. Flip-flops/latches
2. Counters/timers
3. Memory
4. Processors (micro-processors/controllers)

3. Digital adder (binary adder): Not a direct/AC V-adder

a) Half adder: For 1-bit I/Ps

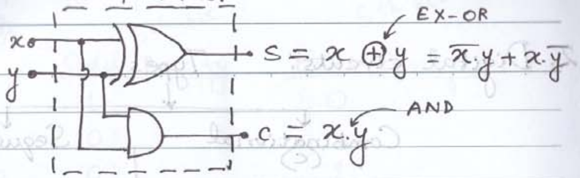


Truth table:

x	y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1
		↗ EX-OR O/P	↖ AND O/P

x y s c
0 + 0 = 0, 0
0 + 1 = 1, 0
1 + 0 = 1, 0
1 + 1 = 0, 1

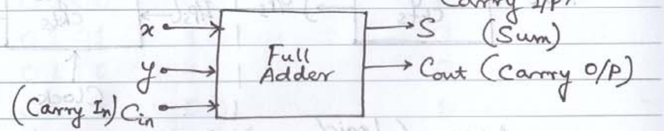
Half Adder



$S = x \oplus y = \overline{x}.y + x.\overline{y}$

$c = x.y$

b. Full adder: 1-bit I/Ps (considers C_{in}) Carry I/P



Truth Table:

	C_{in}	x	y	S	Cout
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

S & Cout are not matching with any standard logic gate.
↓
K-map minimization is reqd'.

b'. Karnaugh (K)-map minimization:
↳ Logical box pattern for arranging required values (8 possibilities for S & another 8 for Cout)

Sum (S)

C_{in} \ xy	00	01	11	10
0	^{0}	1^{1}	^{3}	1^{2}
1	1^{4}	^{5}	1^{7}	^{6}

Cout

Cin \ xy	00	01	11	10
0			1	
1		1	1	1

For Sum

C_{in} \ xy	00	01	11	10
0	^{0}	1^{1}	^{3}	1^{2}
1	1^{4}	^{5}	1^{7}	^{6}

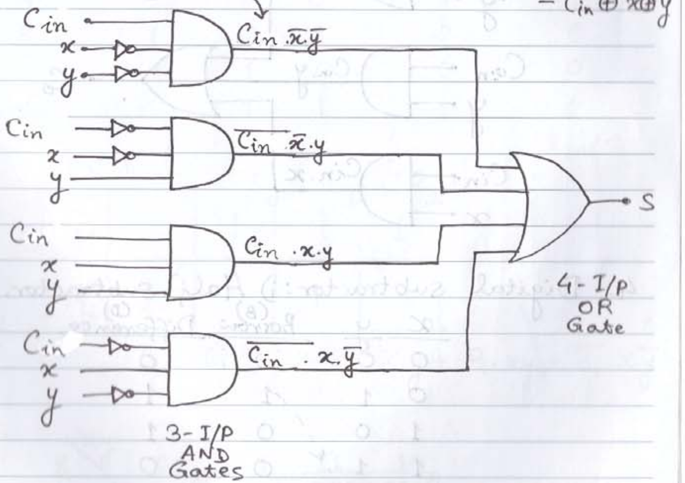
$C_{in}.\overline{x}.\overline{y}$

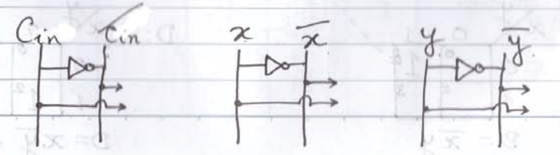
$\overline{C_{in}}.\overline{x}.y$

$C_{in}.x.y$

$\overline{C_{in}}.x.\overline{y}$

$$S = C_{in}.\overline{x}.\overline{y} + \overline{C_{in}}.\overline{x}.y + C_{in}.x.y + \overline{C_{in}}.x.\overline{y}$$
$$= C_{in} \oplus x \oplus y$$

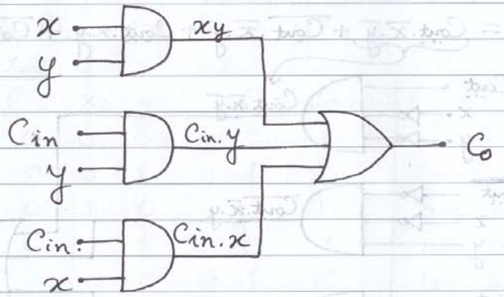

C_{in}
x
y
$C_{in}.\overline{x}.\overline{y}$
C_{in}
x
y
$\overline{C_{in}}.\overline{x}.y$
C_{in}
x
y
$C_{in}.x.y$
C_{in}
x
y
$\overline{C_{in}}.x.\overline{y}$
S
4-I/P OR Gate
3-I/P AND Gates



For Carry out

$C_{in} \backslash xy$	00	01	11	10
0	^{0}	^{1}	1 ^{3}	^{2}
1	^{4}	1 ^{5}	1 ^{7}	1 ^{6}

$$C_o = xy + C_{in}.y + C_{in}.x$$



4. Digital subtractor: i) Half subtractor.

x	y	Borrow (B)	Difference (D)
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

B:

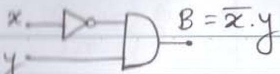
$x \backslash y$	0	1
0	^{0}	1 ^{1}
1	^{2}	^{3}

$B = \bar{x}y$

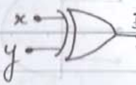
D:

$x \backslash y$	0	1
0	^{0}	1 ^{1}
1	1 ^{2}	^{3}

$D = x\bar{y} + \bar{x}.y$



$B = \bar{x}.y$



$D = x\bar{y} + \bar{x}y$
$= x \oplus y$

ii) Full subtractor:

	Borrow In (Bi)	x	y	Borrow Out (Bo)	Difference (D)
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	1

D:

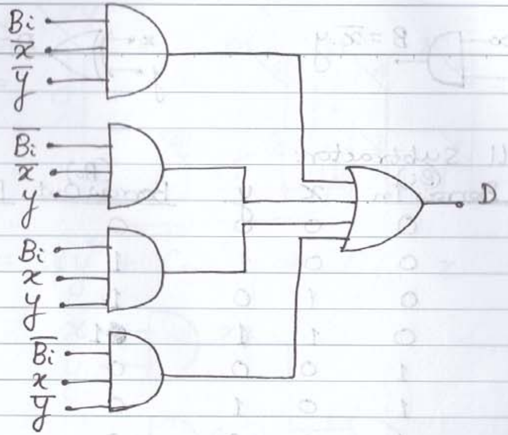
$B_i \backslash xy$	00	01	11	10
0	^{0}	1 ^{1}	^{3}	1 ^{2}
1	1 ^{4}	^{5}	1 ^{7}	^{6}

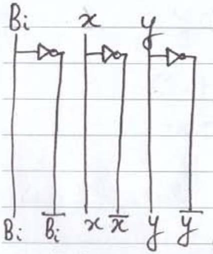
$D = B_i.\bar{x}.\bar{y} + \bar{B_i}.\bar{x}.y + B_i.xy + \bar{B_i}.x.\bar{y}$

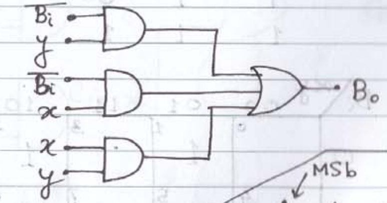
B_o:

$B_i \backslash xy$	00	01	11	10
0	^{0}	1 ^{1}	1 ^{3}	1 ^{2}
1	^{4}	^{5}	1 ^{7}	^{6}

$B_o = \bar{B_i}.y$
$+ \bar{B_i}.x$
$+ xy$



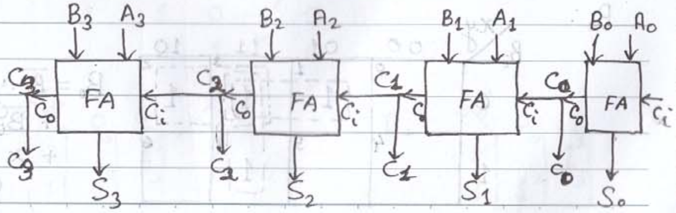




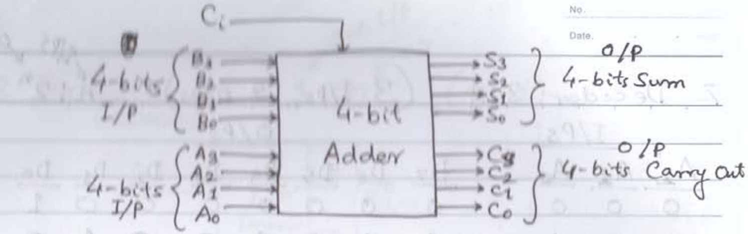
MSb ↙ ↙ LSb

$$\begin{array}{r} A_3\ A_2\ A_1\ A_0 \\ +\ B_3\ B_2\ B_1\ B_0 \\ \hline C_3\ C_2\ C_1\ C_0,\ S_3\ S_2\ S_1\ S_0, \end{array}$$

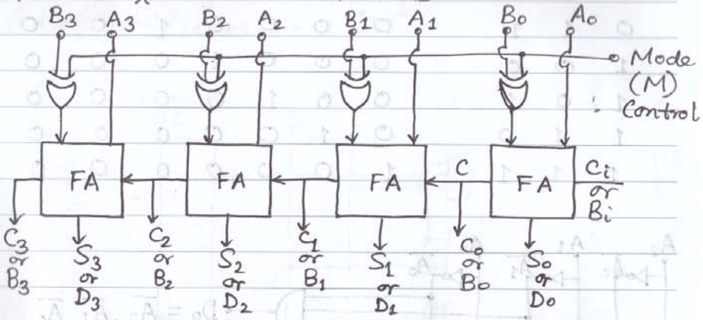
5. Parallel adder : 4-bits



FA: Full adder



6. Parallel adder/subtractor: 4-bits



M=0 ⇒ Adder } Mode control.
M=1 ⇒ Subtractor

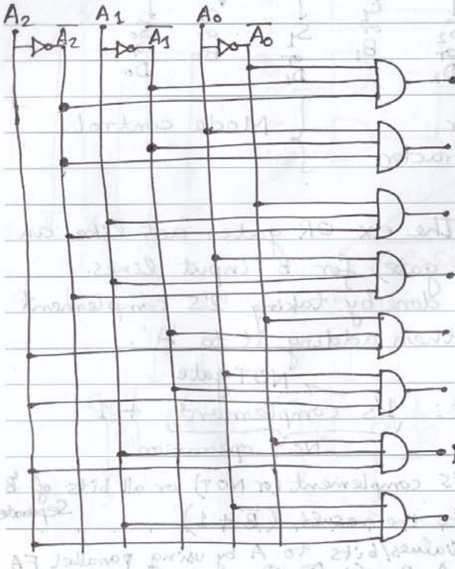
When, M=1, the ex-OR gates act like an inverter (NOT gate) for B input lines.
∴ A-B could be done by taking 2'S complement of 'B' and then adding it to 'A'.

$$2\text{'S complement}: \underbrace{\overset{\nearrow \text{NOT gate}}{1\text{'s complement}}}_{\text{NOT operation}} + 1$$

Steps: 1. Compute 1'S complement (or NOT) or all bits of 'B' Separately
2. Add '1' to the result. ($\overline{B}+1$)
3. Add the values/bits to A by using parallel FA chain.
$A-B = (A+(\overline{B}+1))$

7. Decoder: 3:8   (3 I/Ps, 8 O/Ps) $\overset{I/Ps}{n} : \overset{O/Ps}{2^n}$

I/Ps			O/Ps							
A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0



$D_0 = \overline{A_2}.\overline{A_1}.\overline{A_0}$

$D_1 = \overline{A_2}.\overline{A_1}.A_0$

$D_2 = \overline{A_2}\,A_1\,\overline{A_0}$

$D_3 = \overline{A_2}\,A_1\,A_0$

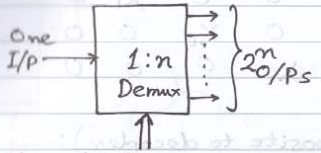
$D_4 = A_2\,\overline{A_1}\,\overline{A_0}$

$D_5 = A_2\,\overline{A_1}\,A_0$

$D_6 = A_2\,A_1\,\overline{A_0}$

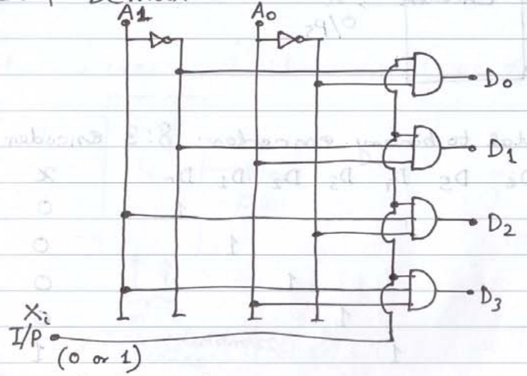
$D_7 = A_2\,A_1\,A_0$

8. Demultiplexer: (demux) $\overset{\text{One I/P}}{1} : \overset{O/P}{2^n}$ $(n>1)$



Address/Control/Select lines (I/P)

1:4 Demux:



A_1, A_0 : Address line

X_i : I/P line

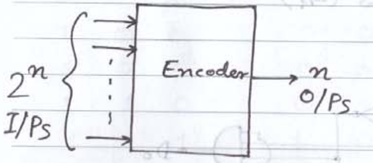
$D_3 - D_0$: Data O/P line

While, $A_1 = 0$, $A_0 = 0$ $\Rightarrow$ $D_0 = \overset{\text{I/P logic signal}}{X_i}$, $D_3 - D_1 = 0$
$\left(\begin{array}{l} X_i = 0 \Rightarrow D_0 = 0 \\ X_i = 1 \Rightarrow D_0 = 1 \end{array}\right)$

& while, $A_1 = 1$, $A_0 = 1$ $\Rightarrow$ $D_3 = X_i$, $D_2 - D_0 = 0$
$\left(\text{If } \begin{array}{l} X_i = 0, D_3 = 0 \\ X_i = 1, D_3 = 1 \end{array}\right)$

I/P	Select I/Ps		O/Ps			
X_i	A_1	A_0	D_3	D_2	D_1	D_0
0/1	0	0	0	0	0	X_i
0/1	0	1	0	0	X_i	0
0/1	1	0	0	X_i	0	0
0/1	1	1	X_i	0	0	0

9. Encoder (opposite to decoder):



I/PS → $2^n : n$ ← O/Ps

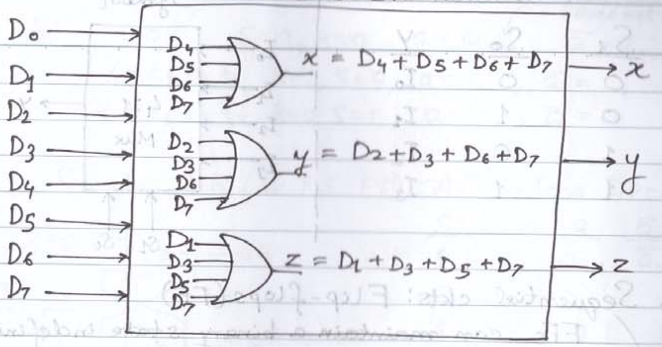
Octal to binary encoder: 8:3 encoder

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	x	y	z
							1	0	0	0
						1		0	0	1
					1			0	1	0
				1				0	1	1
			1					1	0	0
		1						1	0	1
	1							1	1	0
1								1	1	1

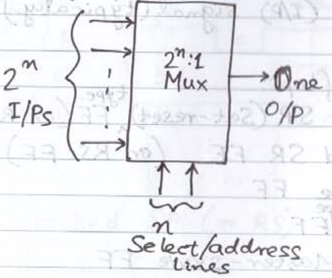
Note: Only one of the I/Ps should be active at a time (mandatory requirement).

Priority encoder → Solves the prob.

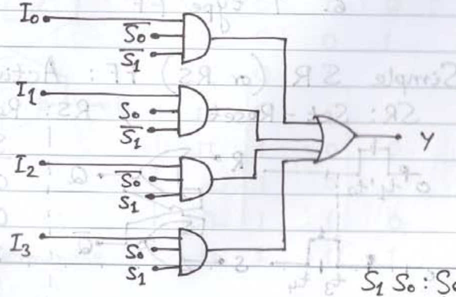
I/PS → 8:3 ← O/PS Encoder



10. Multiplexer: $2^n : 1$ (Opp. of demux)
(Mux)



4:1 Mux

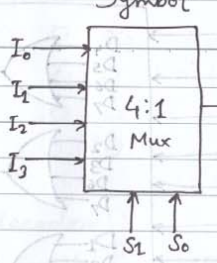


S_1 S_0 : Select lines (I/P)

Characteristic table:

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Symbol



11. Sequential ckts: Flip-flops (FF)

FFs can maintain a binary state indefinitely until directed by an I/P signal to swith states. (Power supply must always be present)

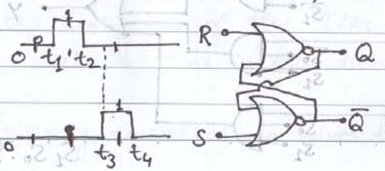
→ Ckts with clock (I/P) signal (typically)

Types of flip-flops:

1. Simple SR (Set-reset) type FF (or RS FF)
2. Clocked SR FF (or RS FF)
3. D-type FF
4. J-K type FF
5. J-K Master-slave FF
6. T type FF

12. i) Simple SR (or RS) FF: Active high I/Ps.

SR: Set-Reset ; RS: Reset-Set



S	R	Q	$\bar{Q}$
0	0	Q_{n-1}	$\bar{Q}_{n-1}$
0	1	0	1
1	0	1	0
1	1	X	X

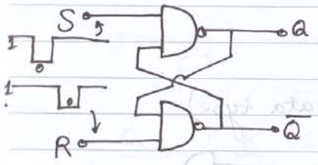
x: Not allowed

At $t=0$, $S=0, R=0$, $Q = Q_{n-1}$, $\bar{Q} = Q_{n-1}$ Previous state (does not change)

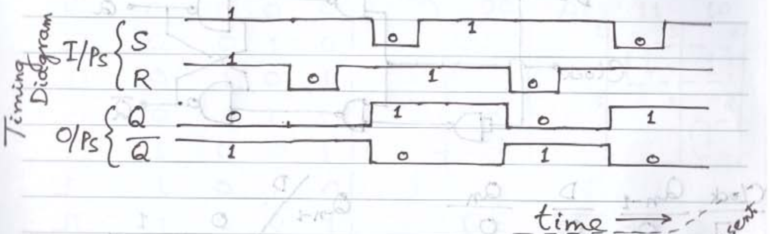
$t = t_1$ to t_2, $R=1, S=0$, $Q=0$, $\bar{Q}=1$

$t = t_3$ to t_4, $R=0, S=1$, $Q=1$, $\bar{Q}=0$

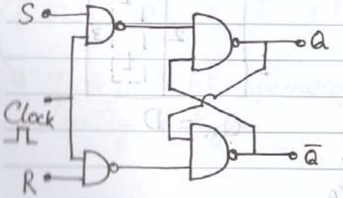
ii) Simple SR (or RS FF): Active low I/Ps.



S	R	Q	$\bar{Q}$
1	1	Q_{n-1}	$\bar{Q}_{n-1}$
0	0	X	X
1	0	0	1
0	1	1	0



13. Clocked SR (or RS) FF:



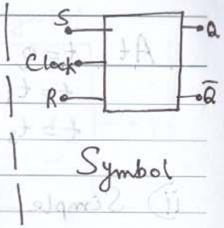
Clock	Q_{n-1}	S	R	Q_n
⊓⊓	0	0	0	0
⊓	0	0	1	0
⊓	0	1	0	1
⊓	0	1	1	X
⊓	1	0	0	1
⊓	1	0	1	0
⊓	1	1	0	1
⊓	1	1	1	X

Previous O/P state

present O/P State

X: Indeterminate

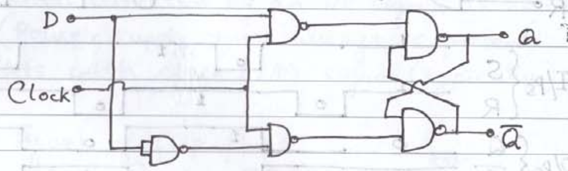
Q_{n-1} \ SR	00	01	11	10
0			X	1
1	1		X	1



Symbol

$$Q_n = S + \bar{R}.Q_{n-1}$$

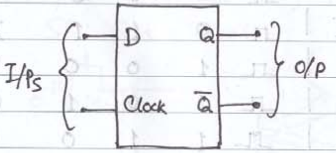
14. D-type FF (Data type)



Clock	Q_{n-1}	D	Q_n
⎍	0	0	0
⎍	0	1	1
⎍	1	0	0
⎍	1	1	1

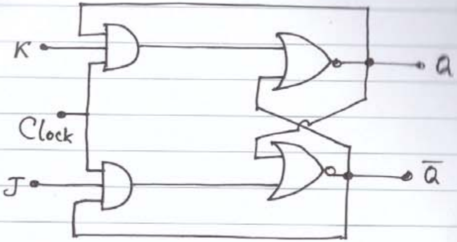
Q_{n-1} \ D	0	1
0		1
1		1

$$Q_n = D$$



Symbol

15. J-K type FF:



Clock	Q_{n-1}	J	K	Q
⎍	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
⎍	0	1	1	1

Q_{n-1} \ JK	00	01	11	10
0			1	1
1	1			1

$$Q = J.\overline{Q_{n-1}} + \bar{K}.Q_{n-1}$$

Present Logic O/P State

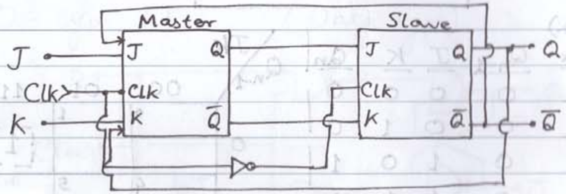
Previous Logic O/P state

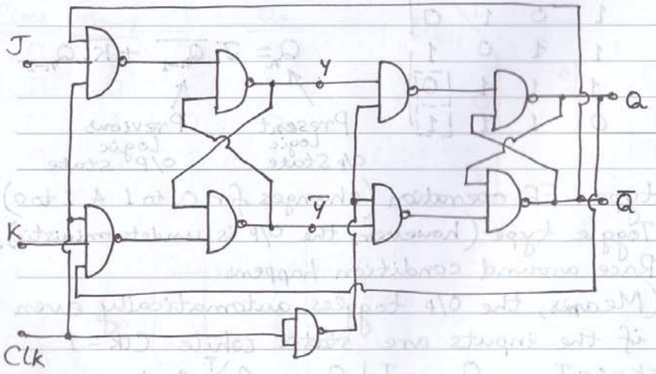
→ T-type FF operation

↑ Toggle type

16. J-K Master-Slave flip-flop:

Clock pulse width is larger than the propagation delay, which causes problem while J=K=1 (toggle mode), as at the end of a clock pulse, O/Ps may/may not toggle to a correct value/logic. Such a problem is solved by using a master-slave FF (Solves the problem of race around)





Clk	Q_{n-1}	J	K	Q_n
	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
	0	1	1	1

} Toggle mode is correctly implemented by using J-K M-S FF.

Components

17. Logic families: Ckt. & Voltage selection in digital systems.

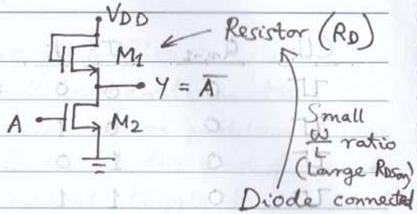
RTL : Resistor Transistor Logic
DTL : Diode Transistor Logic
TTL : Transistor Transistor Logic
ECL : Emitter Coupled Logic
nMOS & pMOS : n- or p-channel MOS
CMOS : Complementary Metal Oxide Semiconductor

a) n-MOS logic: Uses only n-ch MOSFETs.
Basic gates:
1) Inverter (NOT)
2) NAND
3) NOR

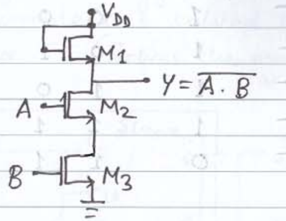
No.

Date.

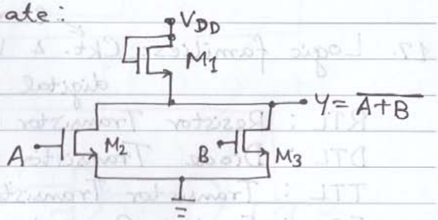
nMOS NOT Gate:



nMOS NAND Gate:



nMOS NOR Gate:



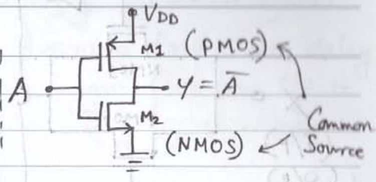
b) CMOS Logic: Both n-channel & p-channel MOSFETs are used (typically same in numbers)

- → Avoids R_D or Diode connected MOSFETs.
- → Offers Low power, as current (static) in between V_{DD} & Gnd is always zero. However (dynamic) current is present during logic transitions only.
- → Smaller area on a Semiconductor integrated circuit (IC).

CMOS NOT Gate

A=0, M_1 (ON), M_2 (OFF), $Y = V_{DD}$

A=1, M_1 (OFF) M_2 (ON), $Y = 0V$

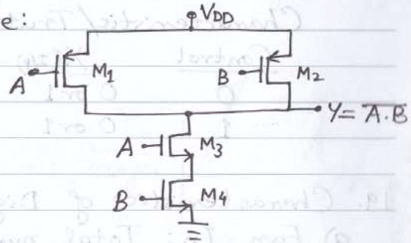


CMOS NAND Gate:

M_1, M_2 : PMOS

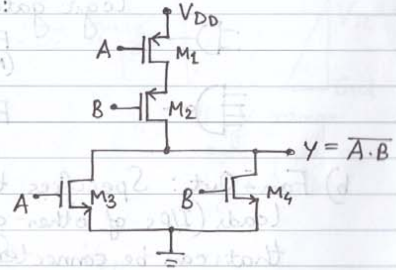
M_3, M_4 : NMOS

A	B	Y
OV	OV	V_{DD}
OV	Vdd	V_{DD}
Vdd	OV	V_{DD}
Vdd	Vdd	OV



CMOS NOR Gate:

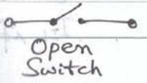
A	B	Y
OV	OV	Vdd
OV	Vdd	OV
Vdd	OV	OV
Vdd	Vdd	OV

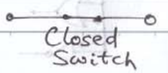


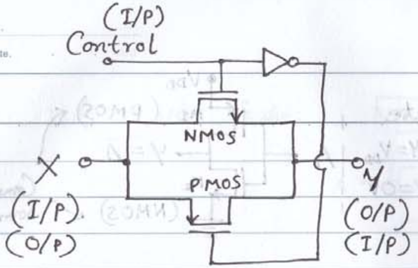
18. CMOS Transmission Gate:

Digitally controlled switch: For Analog or Digital Signal propagation.

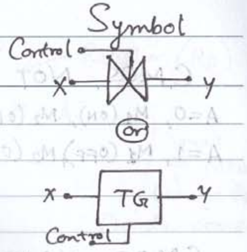
(solid-state ← Semiconductor or no moving parts)







Symbol



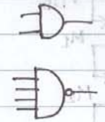
Characteristic/Truth table:

Control	X(I/P)	Y(O/P)
0	0 or 1	Tri-state (or don't care)
1	0 or 1	X(I/P)

Open-ckt

19. Characteristics of Digital ckts:

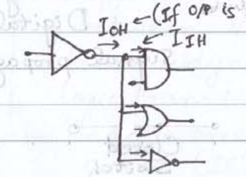
a) Fan-In: Total number of I/Ps in a logic gate. (may be odd or even)



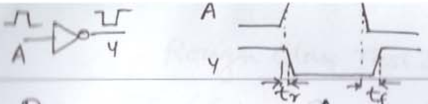
Fan-in = 2
(No. of I/Ps)

Fan-in = 4

b) Fan-Out: Specifies the number of standard loads (I/Ps of other gates/digital logic I/Ps) that can be connected safely to the O/P of a gate without degrading its normal operation (change in O/P logic state due to excessive loading does not occur)



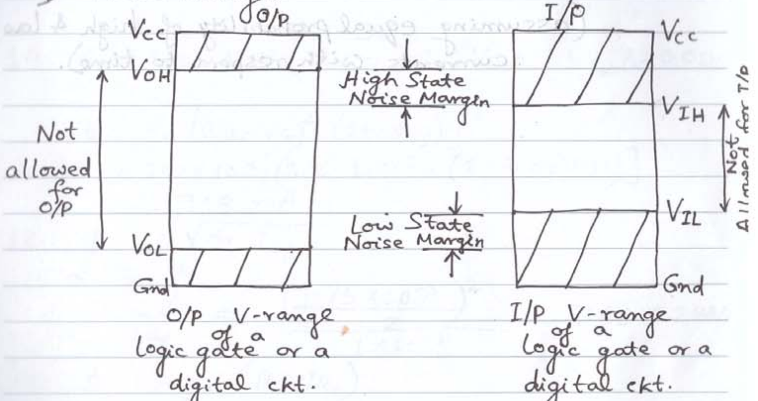
(Same with opp. direction of I_{OL}, & I_{IL} if O/P is 0)



$PD = \frac{t_r + t_f}{2}$

c) Propagation delay: Average transition delay time for signal to propagate from an I/P to its O/P when a binary I/P signal changes a value/logic.
(Time delay of a gate or any digital ckt)

d) Noise margin:



O/P V-range of a logic gate or a digital ckt.

I/P V-range of a logic gate or a digital ckt.

Noise margin (high) = $V_{OH} - V_{IH}$
Noise margin (low) = $V_{IL} - V_{OL}$

where,

V_{OH} :	Voltage	O/P	high	('1' state) for
V_{OL} :	"	"	Low	('0' state)
V_{IH} :	"	I/P	high	('1' state)
V_{IL} :	"	"	Low	('0' state)