Sequential logic circuit.

No: Date: / /

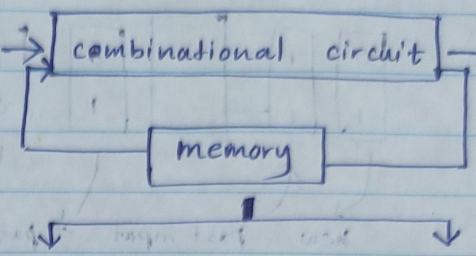
logic circuits

Combinational circuit

input → combinational c.. → output

- generally consist several inputs
- several output signals
- and an Interconnetion of gates.
- when we apply same inputs. The combinational circuit always gives the same output
- every input has a unique single output.

Sequential circuit



Synchronous. Asyncronus.

Asynchronous

- Sequential circuits are built of memory elements and combinational logic devices.
- The output of the circuit not only depends the inputs.
- bat also with the current content on its memory.
- so these circuits produce different outputs for the same input depends on what's stored in the memory.

මෙහි කොත් store වෙලා තියෙන previous output Affect වෙනවා new output හදන.

new input + previous output = new o/p

Sequential circuit

Synchronous S.L.S

- various block of the logic circuit are triggered by the same clock signal.
- The clock signal called, "master clock".
- Therefor the output of these blocks change at the same time.

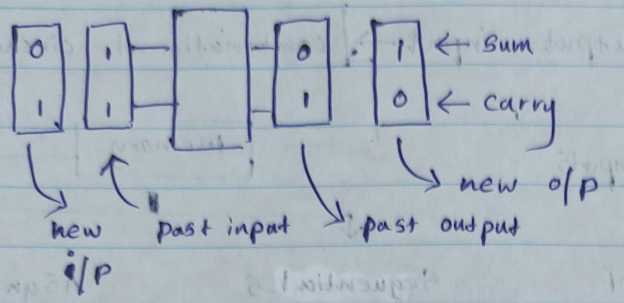
Asynchronous S.L.S

- In this type, various parts of the circuit, triggered by different signals.
- So, the outputs of different blocks of the circuit can change at different instances.

Atlas

No : _____ Date: __/__/____

- in sequential circuit the present output depend on the present input. as well as past ouput / outputs.

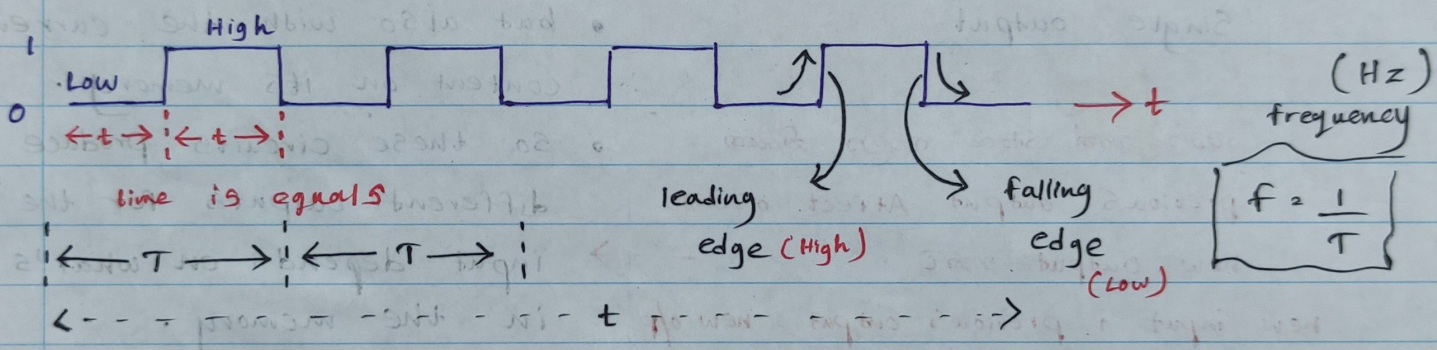


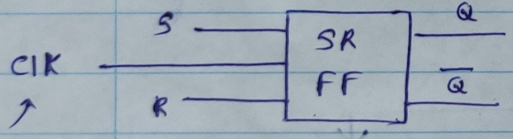
{ flip-flops
counters
registers.

Sequential → mainly ഉപയോഗിക്കുന്ന signals, clock ഉം, triggering methods.

What is clock (CLK) (important)

- Clock is a signal. / it decide the time of input.





(clock pulse is the control i/p)

- The circait (flip-flop) works when the clock is in High pulses.
- The flip-flop is functiona in the leading edge. and falling edge.
  leading → Low to High pulse
  falling → High to low pulse.

clock cycle or period - The it takes the clock to change from 1 to 0 and back 0 to 1 is clock cycle or period.

Extra

Duty cycle → Ratio of time the signal High↑ out of total time.

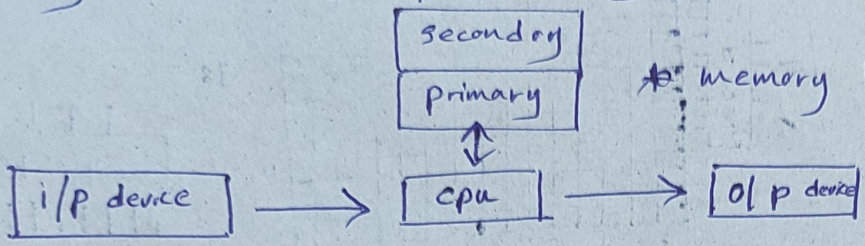
$= \frac{t/2}{t}$ ← total time.

Atlas

$= 1/2 = 50\%$ of ↑High signals.

memory

memory → primary / secondary / registers.



* memory basics.

* memory hierachies.

- speed is high
- capacity is low.
- access time is low
- near to cpu
- cost is high / expensive.

KB Register fastest
MB Cache L1, L2 very fast
GB RAM fast
TB Disk slow

- speed is low
- increase capacity
- increase access time
- slow and cheap
- Away from cpu

memory → Registers
→ primary (cache, main memory (RAM)) } electronic and valatile memory.
→ Secondary (disks / hdd)

➤ Primary memory

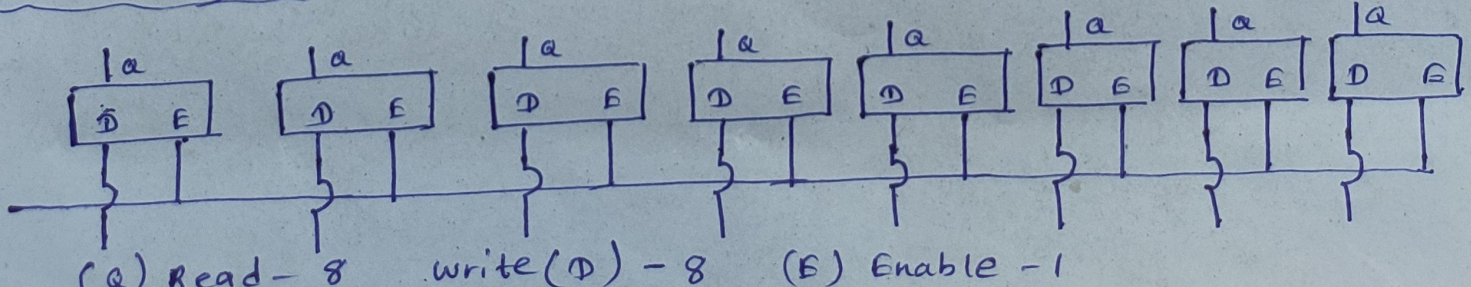
- primary memory works in random with the cpu to store data, Programms and proccessed information.
- primary memory determined the size and number of programs that can be run simultaneously.

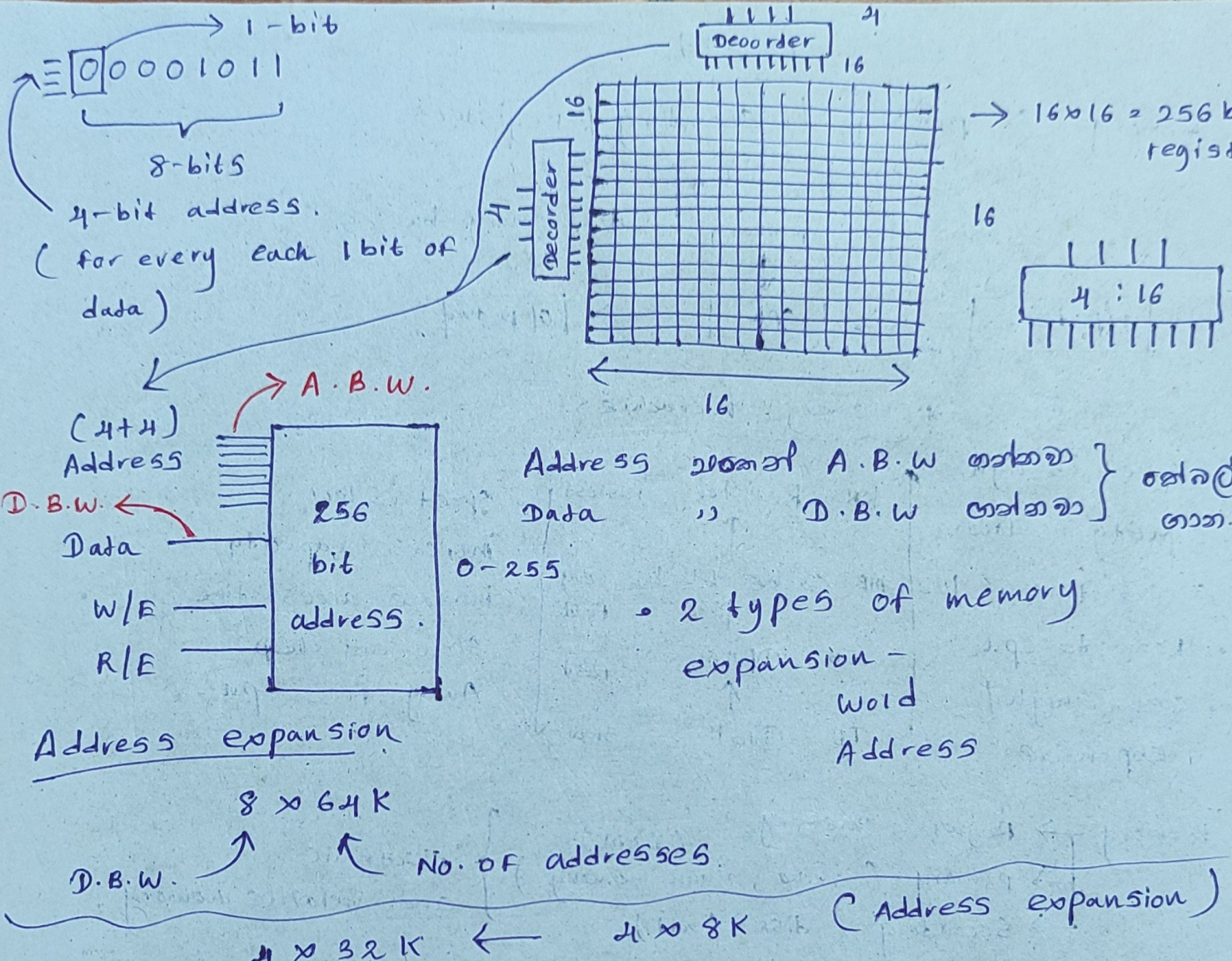
➤ Register

- register is called group of latches combined.
- Can store a single number.
- number of latches is called the width of that register.

8-bit / 16-bit / 32-bit / 64-bit.

8-bit register. D - write (input) Q - Read (output) E - Enable.




1-bit
00001011
8-bits
4-bit address.
(for every each 1 bit of data)
Decoorder
4
16
16
Decorder
4
16
16
16×16 = 256 bit register
4 : 16
A.B.W.
(4+4)
Address
D.B.W.
Data
W/E
R/E
256
bit
address.
0-255
Address යවන්න A.B.W තෝරලා
Data " D.B.W තෝරලා
එන්නල් ගාන.
• 2 types of memory expansion -
Word
Address
Address expansion
8 × 64K
D.B.W.
No. of addresses
4 × 32K
4 × 8K
(Address expansion)

1)

	8 × 64 bit	8 × 16 bit
D.B.W	8 bit	8 bit
A.B.W.	6 bit	4 bit
memory Capacity	64 byte	16 byte

A.B.W.

$2^n = 64$

$n = 2^6$

$n = 6$

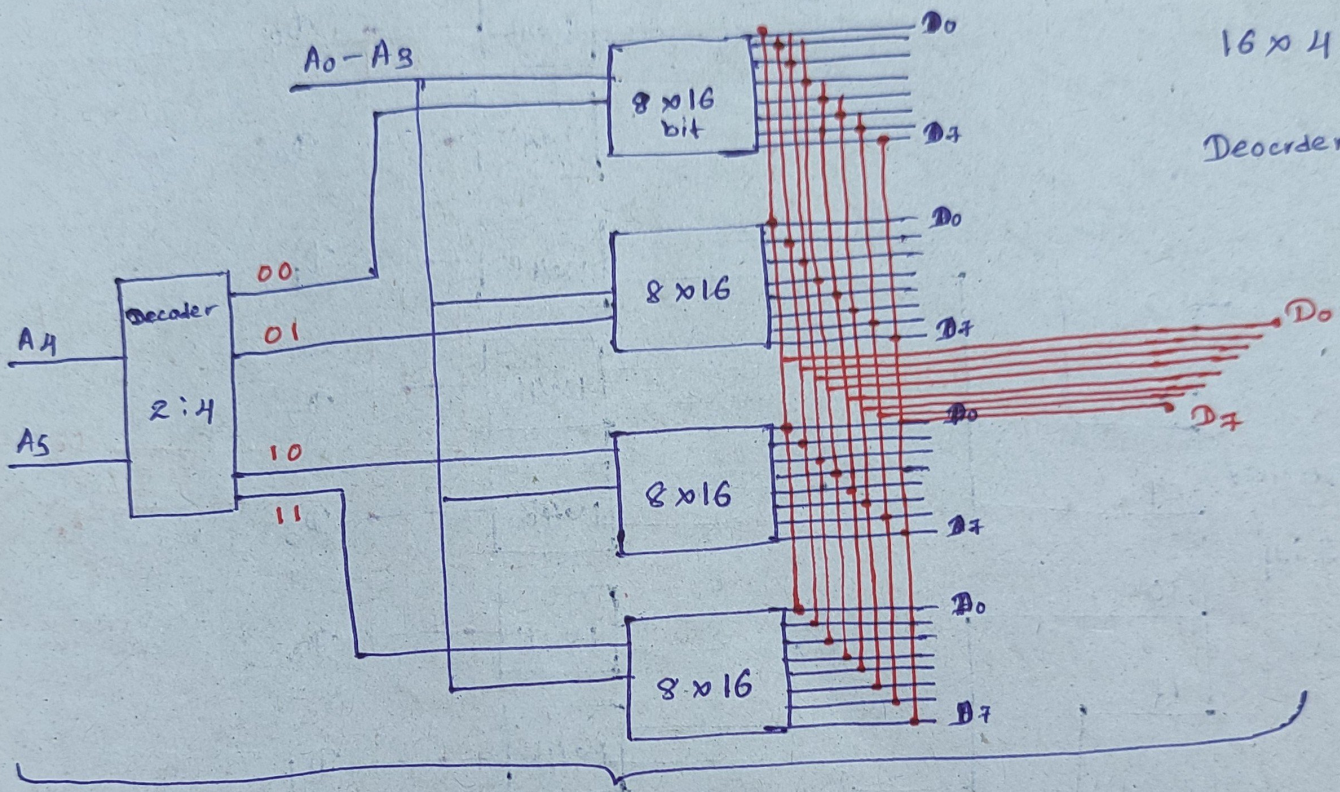
$2^n = 16$

$n = 4$

$16 \times 4 = 64$

Deocrder = 6 − 4 = 2

$2^2 = 4$



8 × 64 bit.

2)

	4 × 16 K	1 × 4 K
D.B.W.	4 bit	1 bit
A.B.W.	14 bit	12 bit
memory capacity	8KB	0.5 KB

$\frac{4 \times 16}{1 \times 4} = 16$

4 × 16 K

$2^2 \times 2^4 \times 2^{10}$

2^8

$\frac{2^3 \times 2^{10}}{2^{10}}$

8 KB

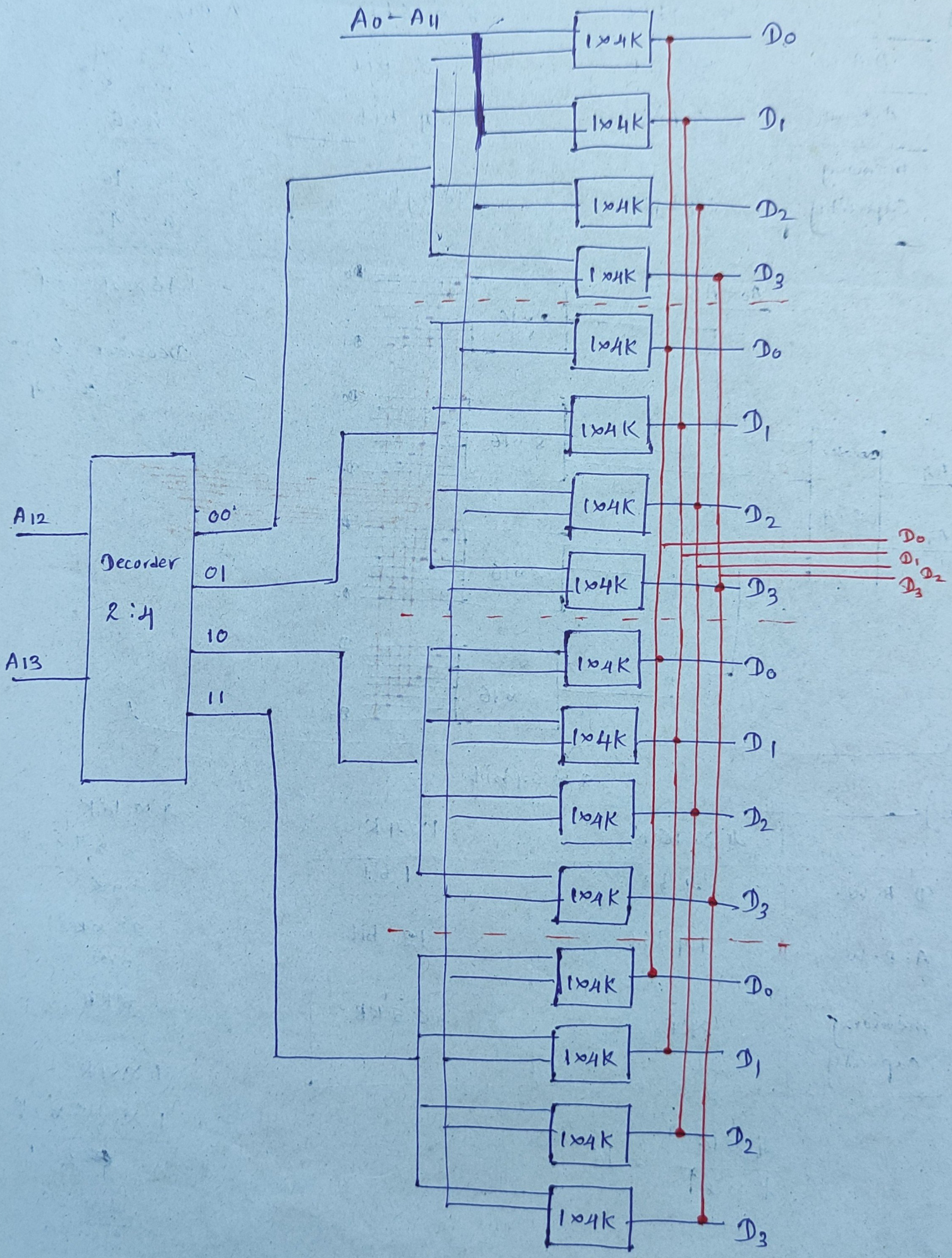
1 × 4 K

$1 \times 2^2 \times 2^{10}$

2^{11}

$\frac{1 \times 2^{10}}{2 \times 2^{10}}$

0.5 KB


A0 - A11
1×4K
D0
D1
D2
D3
A12
A13
Decorder
2 : 4
00
01
10
11

No: Date: / /

ALU

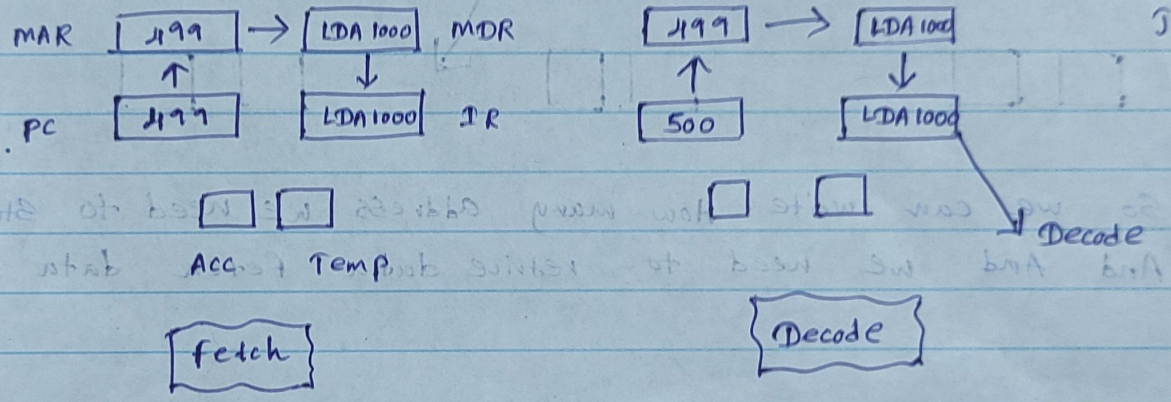
main Registers :- PC - programm counter
MAR - memory address register
MDR - " data "
IR - Instruction register.

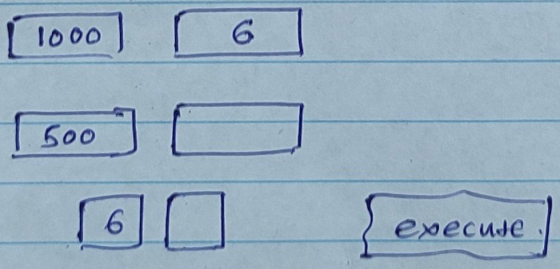
other Register- ACC, Temp

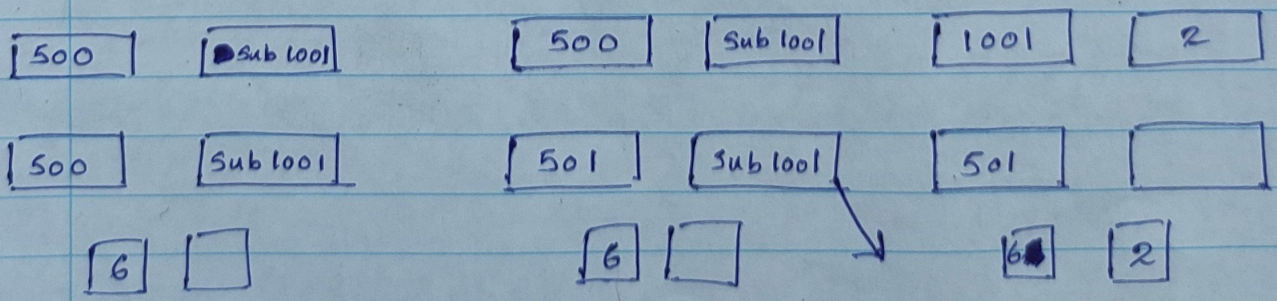
(1) cycle (3) → fetch, decode, execute

Steps

LDA - load
SBA - Substra
STp - Store
Jmp - Jump

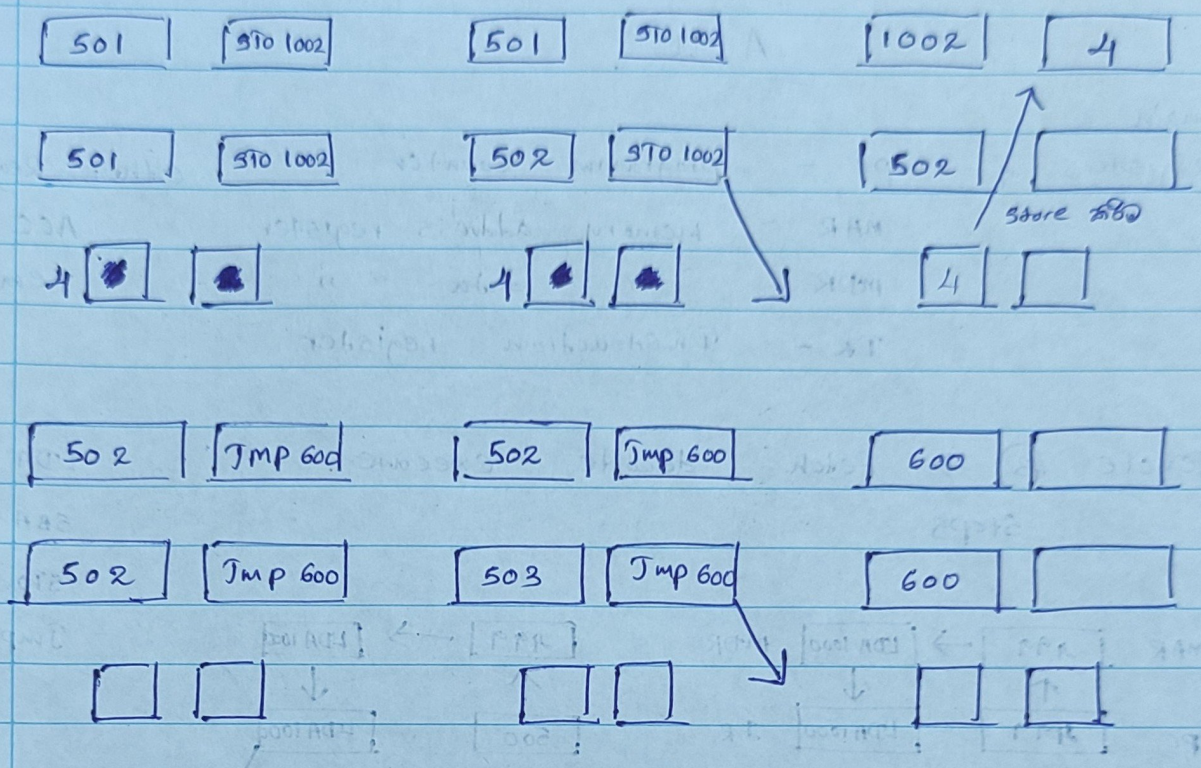






(1) ← value ආවා කියා තාවකාලික temporary

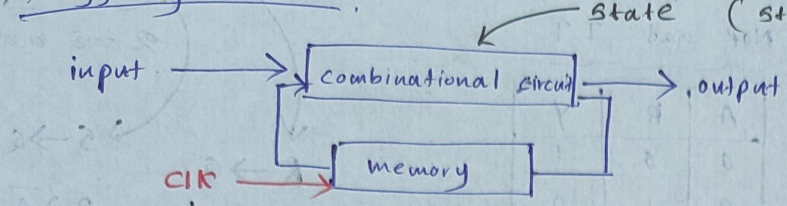
No: _____ Date: ___/___/_____



So we can write How many address we need to store And And we need to retrive data fetch data from.

CA

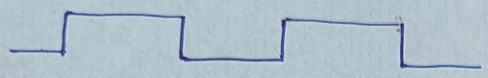
triggering methods



memory means a latch or a flip flop.

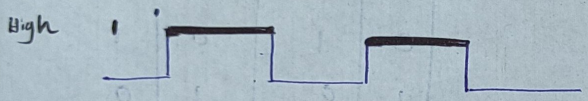
(state of the circuit is stored in the memory)

* 2 methods of triggering
  - Level
  - EDGE — ⊕ve edge, ⊖ve edge.

 → normal clock signal.

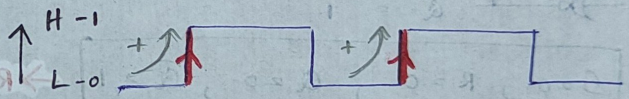
Level triggering

- when ever the clock remains in high their will be a transition in flip-flops. / Latch

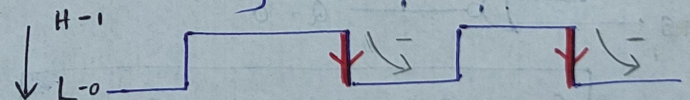


edge triggering

1) ⊕ve edge triggering
   - When the clock signal is in leading edge (low to High pulse) there will be a transition in flip/flops or Latch

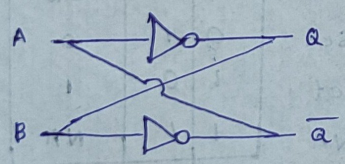


2) ⊖ve edge triggering
   - When the clock signal in the falling edge (High to low pulse) there will be a transition in memory (flip-flop or Latch).



Latch

What is a latch.



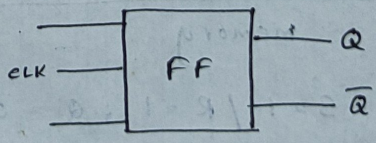
- basic latch created using 2 NAND gates.

* A latch is a data storage device that can store 1 bit of information.

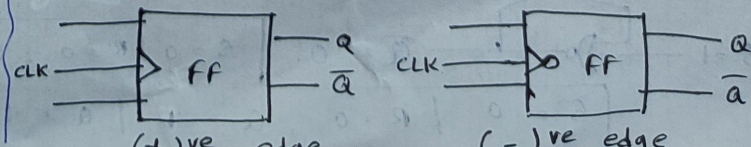
* The basic digital storage (Latch) made by crosscoupling two NAND gates.

- The outputs of each NOT gates are connect to the input of each other. This combination called latch.

Level

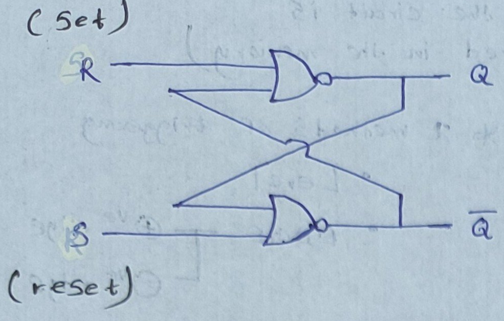


Edge triggering:



(2) SR Latch (set reset latch)

NOR
NAND



- SR Latch using - NOR gate.

NOR gate T.T.

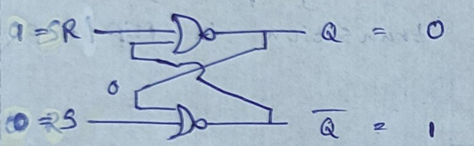
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

R → Q
S → $\overline{Q}$

S → Q
R → $\overline{Q}$

NOR		NAND	
(R) Reset	Q = 0	(R) Reset	Q = 1
(S) Set	Q = 1	(S) Set	Q = 0

1) case (1) S = 0 | R = 1 , Q = 0 | $\overline{Q}$ = 1 → Reset



S = 0 , R = 0 , Q = 0 , $\overline{Q}$ = 1 → Hold State.

memory

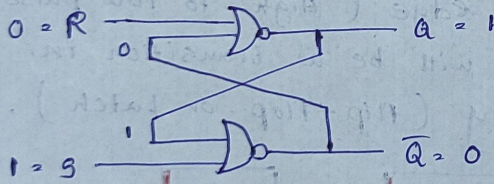
S	R	Q	$\overline{Q}$
0	0	memory (before)	
0	1	0	1
1	0	1	0
1	1	not allowed	

Qn

State	S	R	Qn+1
Hold	0	0	Qn
Reset	0	1	0
Set	1	0	1
NA	1	1	NA

- SR Latch NOR T.T. -

2) case (2) S = 1 | R = 0 , Q = 1 | $\overline{Q}$ = 0 → Set



S = 0 R = 0 , Q = 1 , $\overline{Q}$ = 0 → Hod State

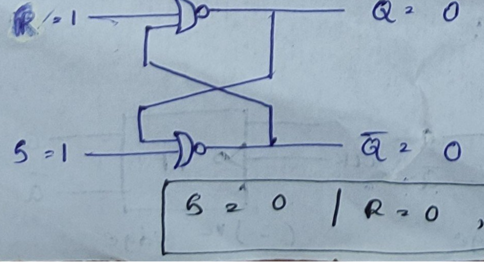
memory

S	R	Q	$\overline{Q}$
0	0	NA	
0	1	1	0
1	0	0	1
1	1	Qn	

S	R	Qn+1
0	0	NA
0	1	1
1	0	0
1	1	Qn

SR Latch NAND

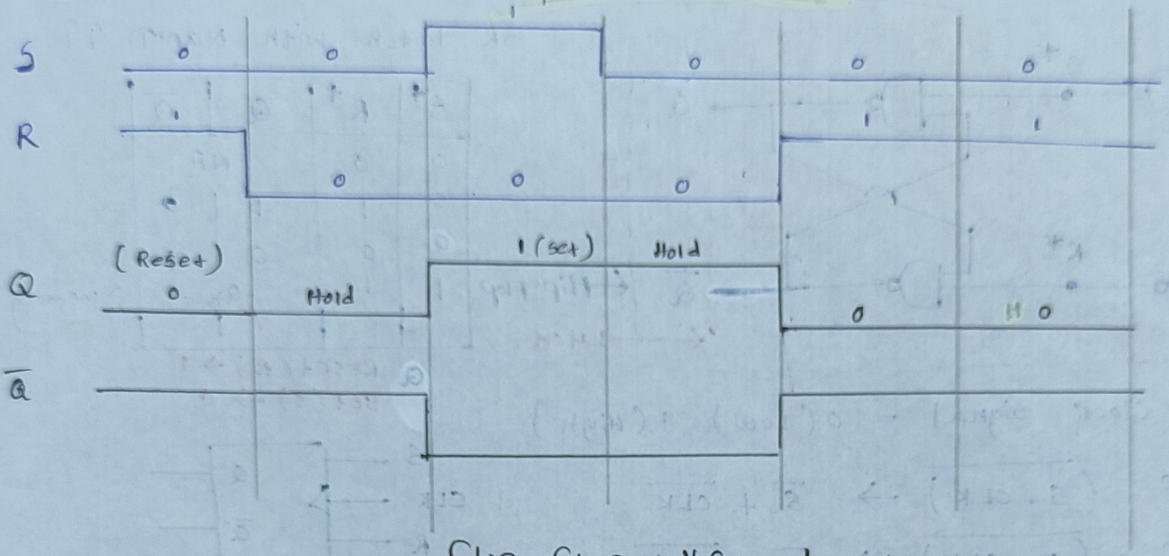
3) Case (3) S = 1 / R = 1 , Q = 0 | $\overline{Q}$ = 0



S = 0 | R = 0 , Q = 1 | $\overline{Q}$ = 0

S = 0 | R = 0 , Q = 0 | $\overline{Q}$ = 1 → NOT Allowed

SR – Latch timing Diagram (NOR gate) (3)



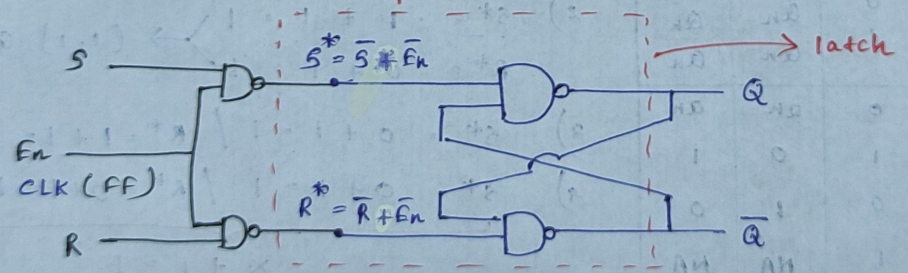
S	R	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	NA

flip-flop VS Latch

flip-flop → Edge-triggering Sensitive

Latch → level triggering »

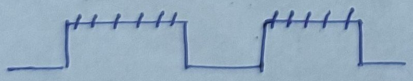
- SR latch controlled input :- (with NAND gate)



NAND T.T.

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

Latch ব্যবহার করে



level triggering

$S^* = \overline{S.En}$

$R^* = \overline{R.En}$

S = 0 / R = 0 / En = 0

$S^* = \overline{0.0} = 1$

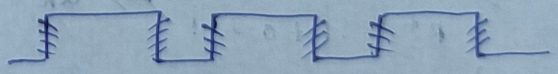
$R^* = \overline{0.0} = 1$

(1, 1) = Qn / memory

S	R	Qn+1
0	0	NA
0	1	1
1	0	0
1	1	Qn

SR Latch NAND T.T. Reduced

Flip-flop ব্যবহার করে

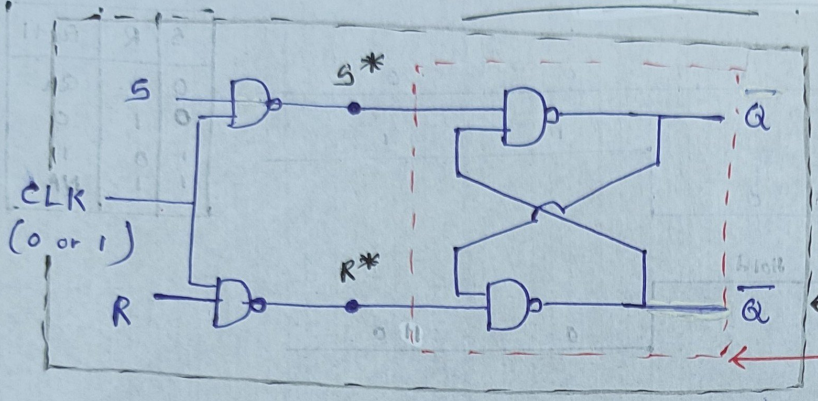


edge triggering

S	R	Q	$\bar{Q}$
0	0	NA	
0	1	1	0
1	0	0	1
1	1	Qn	

SR Latch with NAND T.T.

SR - Flip flop (NAND Gate)



SR latch with NAND T.T.

S*	R*	Q	$\overline{Q}$
0	0	NA	
0	1	1	0
1	0	0	1
1	1	Qn	

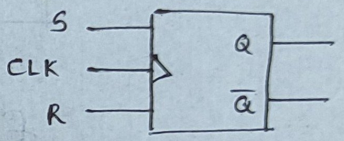
→ memory

Reset (R) → 1
Set (S) → 0

CLK - Clock signal = 0 (Low) 1 (High)

$S^* = \overline{(S \cdot CLK)} \rightarrow \overline{S} + \overline{CLK}$

$R^* = \overline{(R \cdot CLK)} \rightarrow \overline{R} + \overline{CLK}$



SR - Flip flop. (Edge)

	CLK	S	R	Q	$\overline{Q}$
Low	1) 0	0	0	Qn	Qn
	2) 0	0	1	Qn	Qn
	3) 0	1	0	Qn	Qn
	4) 0	1	1	Qn	Qn
High	5) 1	0	0	Qn	Qn
	6) 1	0	1	0	1
	7) 1	1	0	1	0
	8) 1	1	1	NA	NA

Low.

1) $S^* = 1 + 1 = 1$, $R^* = 1 + 1 = 1$ → (1, 1) Qn memory

2) $S^* = 1 + 1 = 1$, $R^* = 0 + 1 = 1$ → (1, 1) Qn

3) $S^* = 0 + 1 = 1$ / $R^* = 1 + 1 = 1$ → Qn

4) $S^* = 0 + 1 = 1$ / $R^* = 0 + 1 = 1$ → Qn

X + 1 = 1

Truth table → next p.g.

5) $S^* = 1 + 0 = 1$, $R^* = 1 + 0 = 1$ → Qn

6) $S^* = 1 + 0 = 1$, $R^* = 0 + 0 = 0$ → 0, 1

7) $S^* = 0 + 0 = 0$, $R^* = 1 + 0 = 1$ → 1, 0

8) $S^* = 0 + 0 = 0$, $R^* = 0 + 0 = 0$ → 0, 0

X + 0 = X

Tables → SR - Flip Flop (Truth table 4)

20.3 / 21.2

Truth table

CLK	S	R	Qn+1
0	0	0	Qn
0	0	1	Qn
0	1	0	Qn
0	1	1	Qn
1	0	0	Qn
1	0	1	0
1	1	0	1
1	1	1	NA

n.s.

P.S

Qn+1 → next state

Qn → present state.

characteristic table

when CLK = 1

S	R	Qn	Qn+1
0	0	0	0 (Qn)
		1	1 (Qn)
0	1	0	0
		1	0
1	0	0	1
		1	1
1	1	0	NA
		1	

Qn+1 → This is depend on your inputs and previous state. (S, R, Qn)

Excitation table

Qn	Qn+1	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

when CLK = 1 (High)

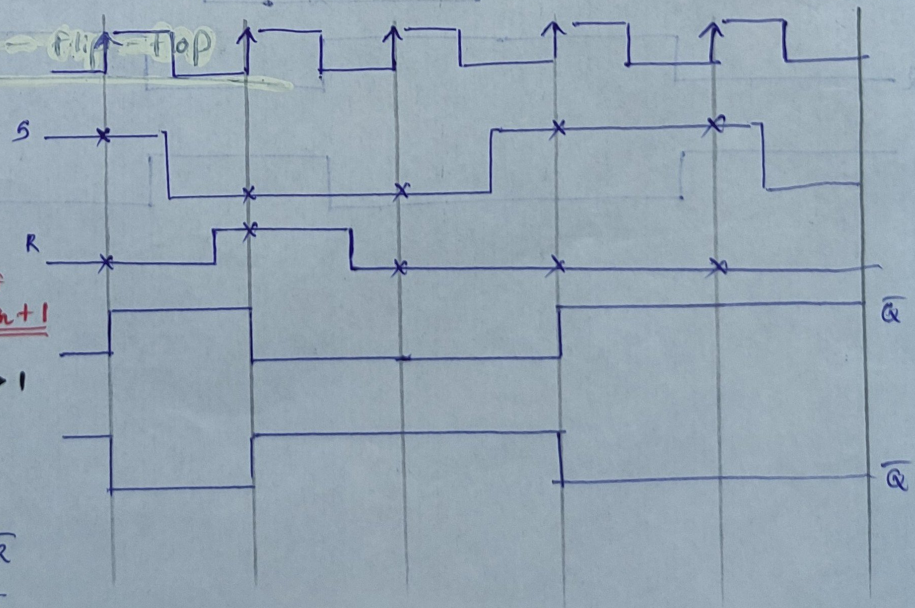
Qn	S	R	Qn+1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

3 inputs

$2^3 = 8$ combinations.

inputs → Qn, Qn+1

outputs → S, R



$2^2 = 4$

find the value of Qn+1

Qn \ SR	00	01	11	10
0	0	0	X	1
1	1	0	X	1

$Q_{n+1} = 1 + 2$

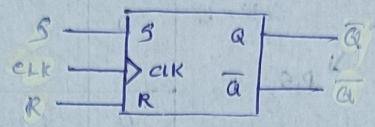
$= S + Q_n\bar{R}$

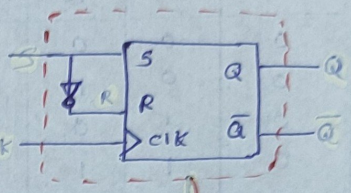
S R

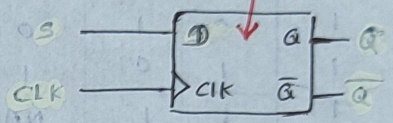
D - Flip-flop (Data flip-flop)

D - flip flop using SR - flip flop

CLK	S	R	Qn+1
0	0	0	Qn
0	0	1	Qn
0	1	0	Qn
0	1	1	Qn
1	0	0	Qn
1	0	1	0
1	1	0	1
1	1	1	NA







	CLK	D	Qn+1
low	0	0	Qn
low	0	1	Qn
high	1	0	0
high	1	1	1

Reduced T.T.

D = 0 , S = 0 / R = 1

D = 1 , S = 1 / R = 0

Q	Qn+1	D
0	0	0
0	1	1
1	0	0
1	1	1

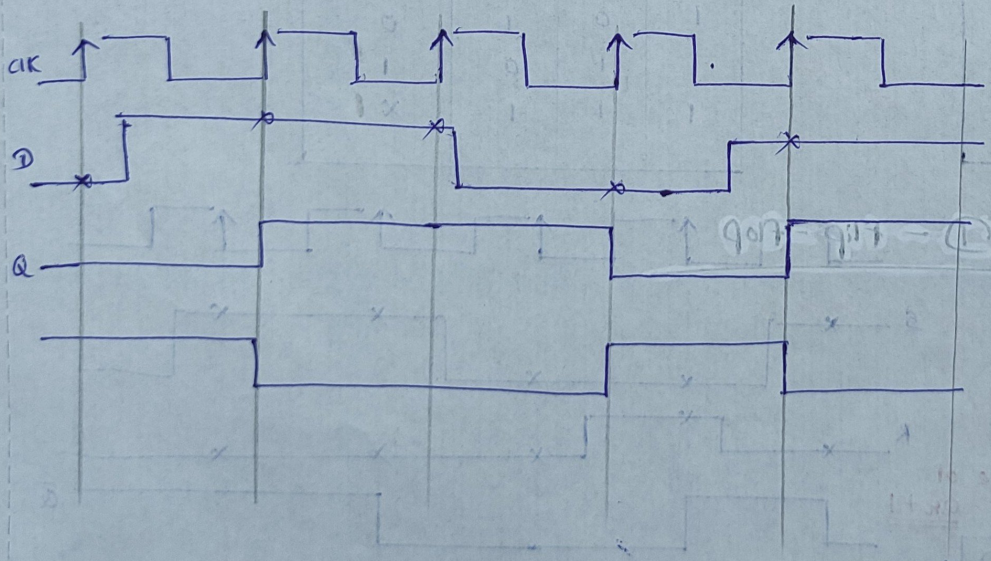
excitation table.

D	Qn	Qn+1
0	0	0
	1	0
1	0	1
	1	1

complete T.T. character

Diagram

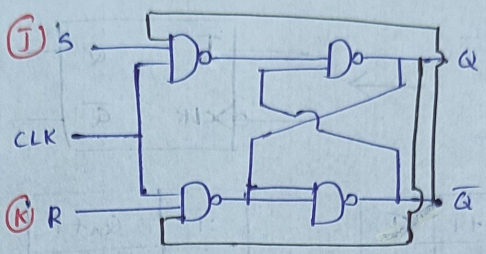
$Q_{n+1} = D$

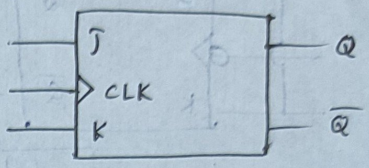


JK - flip-flop (5) මම බලන්න.

- Using SR - Flip flop.

CLK	S	R	Qn+1
0	0	0	Qn
0	0	1	Qn
0	1	0	Qn
0	1	1	Qn
1	0	0	Qn
1	0	1	0
1	1	0	1
1	1	1	Not used





1) CLK = 0 Qn (memory)
2) CLK = 1 J = 0, K = 1, Q = 0, $\bar{Q}$ = 1
3) CLK = 1 J = 1, K = 0, Q = 1, $\bar{Q}$ = 0
4) CLK = 1 J = 0, K = 0, Q = Qn, $\bar{Q}$ = $\bar{Qn}$

2) CLK = 1 J = 1, K = 1
Assume Q = 0 & $\bar{Q}$ = 1

Outputs, Q = 0, 1, 0, 1, ... → Qn+1 = 0 → 1, 1 → 0
$\bar{Q}$ = 1, 0, 1, 0, ... Qn = $\bar{Qn}$
output = $\bar{Qn}$

CLK	J	K	Qn+1
0	0	0	Qn
0	0	1	Qn
0	1	0	Qn
0	1	1	Qn
1	0	0	Qn
1	0	1	0
1	1	0	1
1	1	1	$\bar{Qn}$ ← toggle

T.T for JK-flip flop.

J	K	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	$\bar{Qn}$

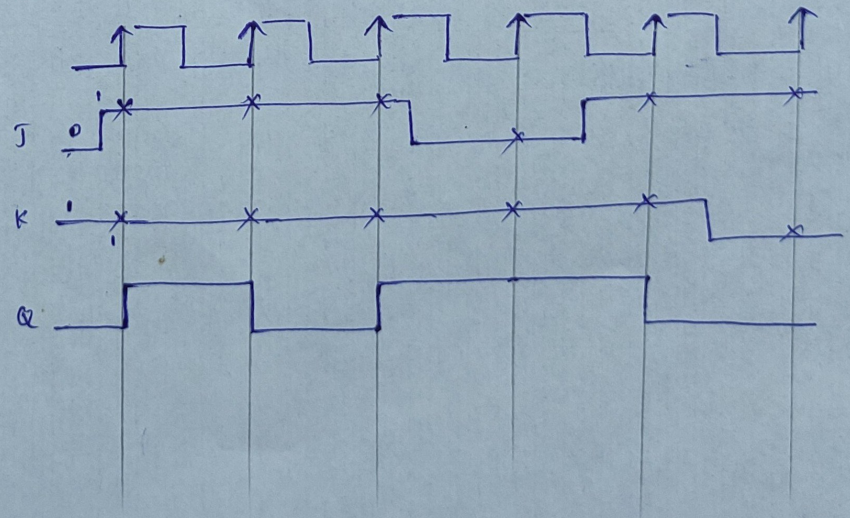
reduced T.T.

Q	Qn+1	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

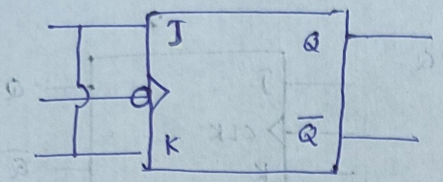
Excitation table.

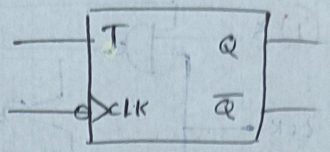
CK=1	J	K	Qn	Qn+1
	0	0	0	0
			1	1
	0	1	0	0
			1	0
	1	0	0	1
			1	1
	1	1	0	1
			1	0

Character / complete . TT.



T-Flip Flop (Toggle in JK-FF)





Q	Qn+1	T
0	0	0
0	1	1
1	0	1
1	1	0

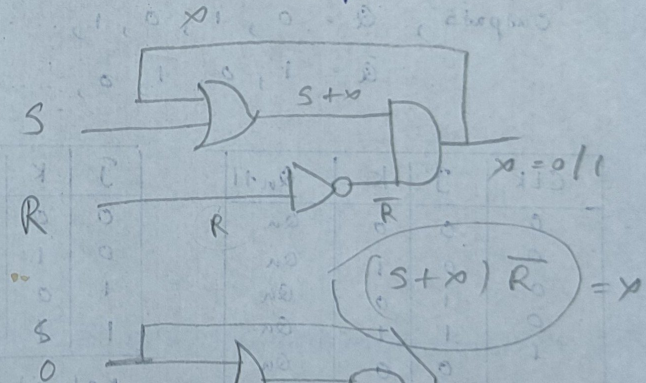
T	Qn+1
0	Qn
1	$\overline{Qn}$

reduced T.T.

CLK	T	Qn+1
low {	0	Qn
	1	Qn
high {	0	Qn
	1	$\overline{Qn}$

T	Qn	Qn+
0	0	0
	1	1
1	0	1
	1	0

Complet T.T.



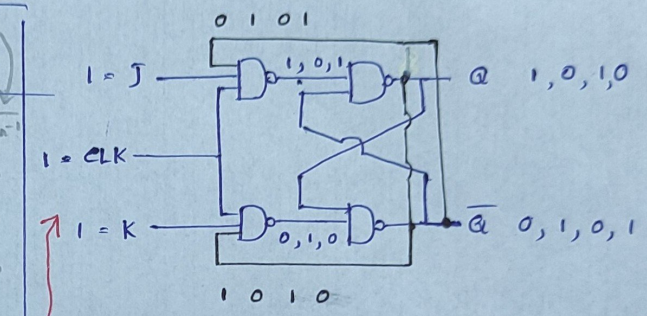
Race-around condition – JK flip flop

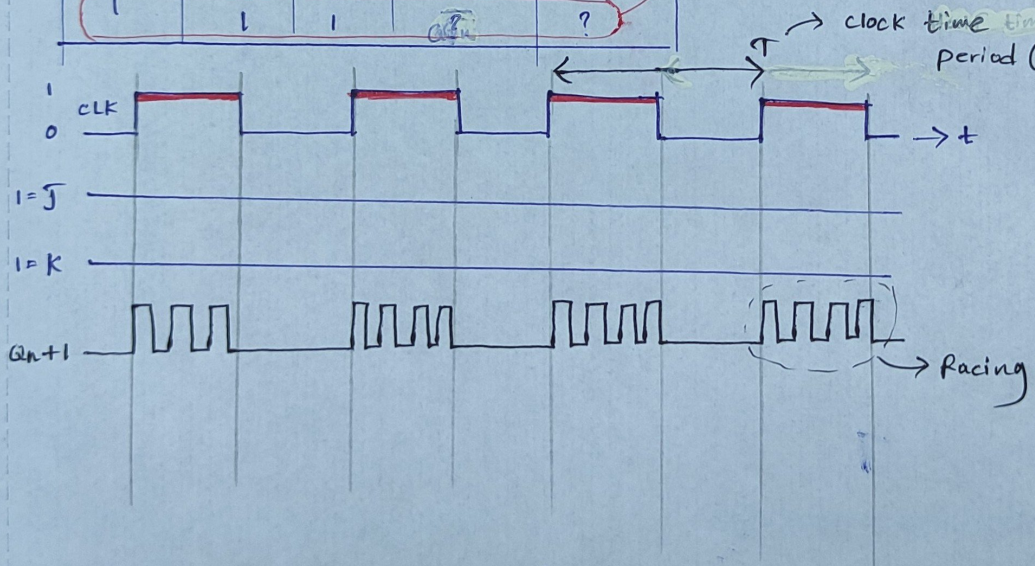
toggle and Race-around are different.

toggle → controlled

Race around – not controlled

CLK	J	K	Q_n / Q_{n+1}	$\overline{Q_n}$ / $\overline{Q_{n+1}}$
0	0	0	Q_n Q_n	$\overline{Q_n}$ $\overline{Q_n}$
0	0	1	Q_n "	$\overline{Q_n}$ "
0	1	0	Q_n "	$\overline{Q_n}$ "
0	1	1	Q_n "	$\overline{Q_n}$ "
1	0	0	Q_n "	$\overline{Q_n}$ "
1	0	1	0	1
1	1	0	1	0
1	1	1	$\overline{Q_n}$	?





T → clock time

period (1→0/0→1) ආවර්;

so, T/2 > delay

Assume T' is the time taken by flip-flop to generate next outputs,

T' > T/2

Conditions to overcome racing,

1) T/2 < propagation delay of the flip-flop
2) edge triggering
3) master slave