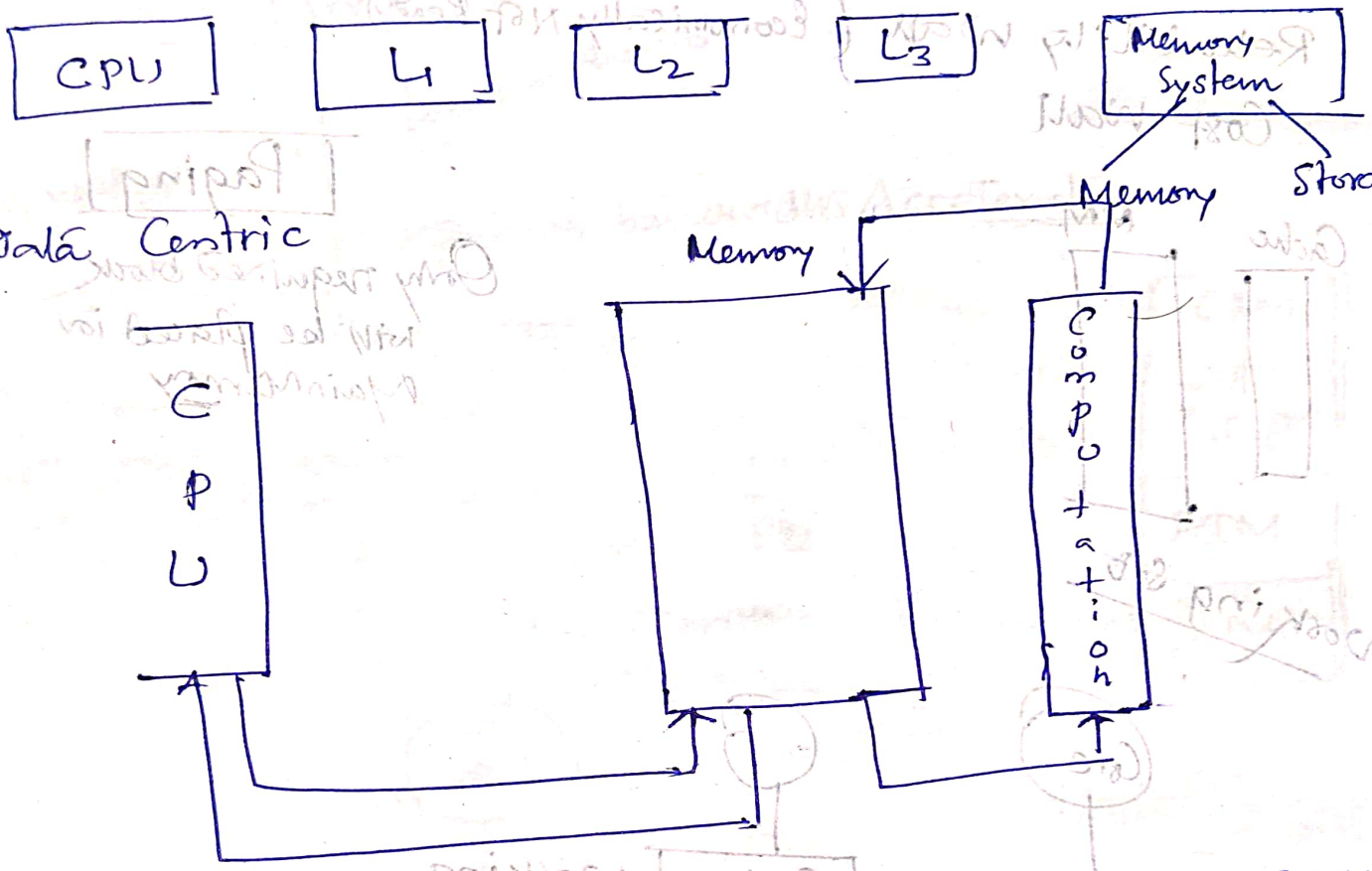
input size → [ ]

Computing (In-Memory) Architecture

07/01/25

Von Neumann → CPU Centric → Data Centric

Walls



CMOS

1. Power Wall

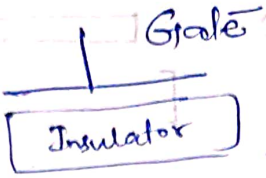
- Static power — stand by mode (power consumption)
- dynamic power — when will be in function

$P = CV^2f$

No. of

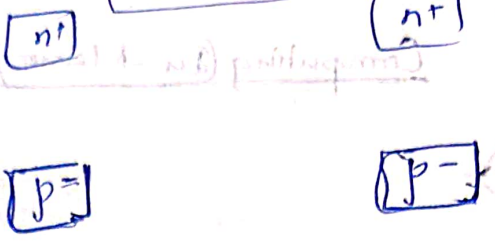
No. of power

MOS Scaling



SiO_2

High Dielectric Constant



2. Memory Wall | 3. (ILP) Wall

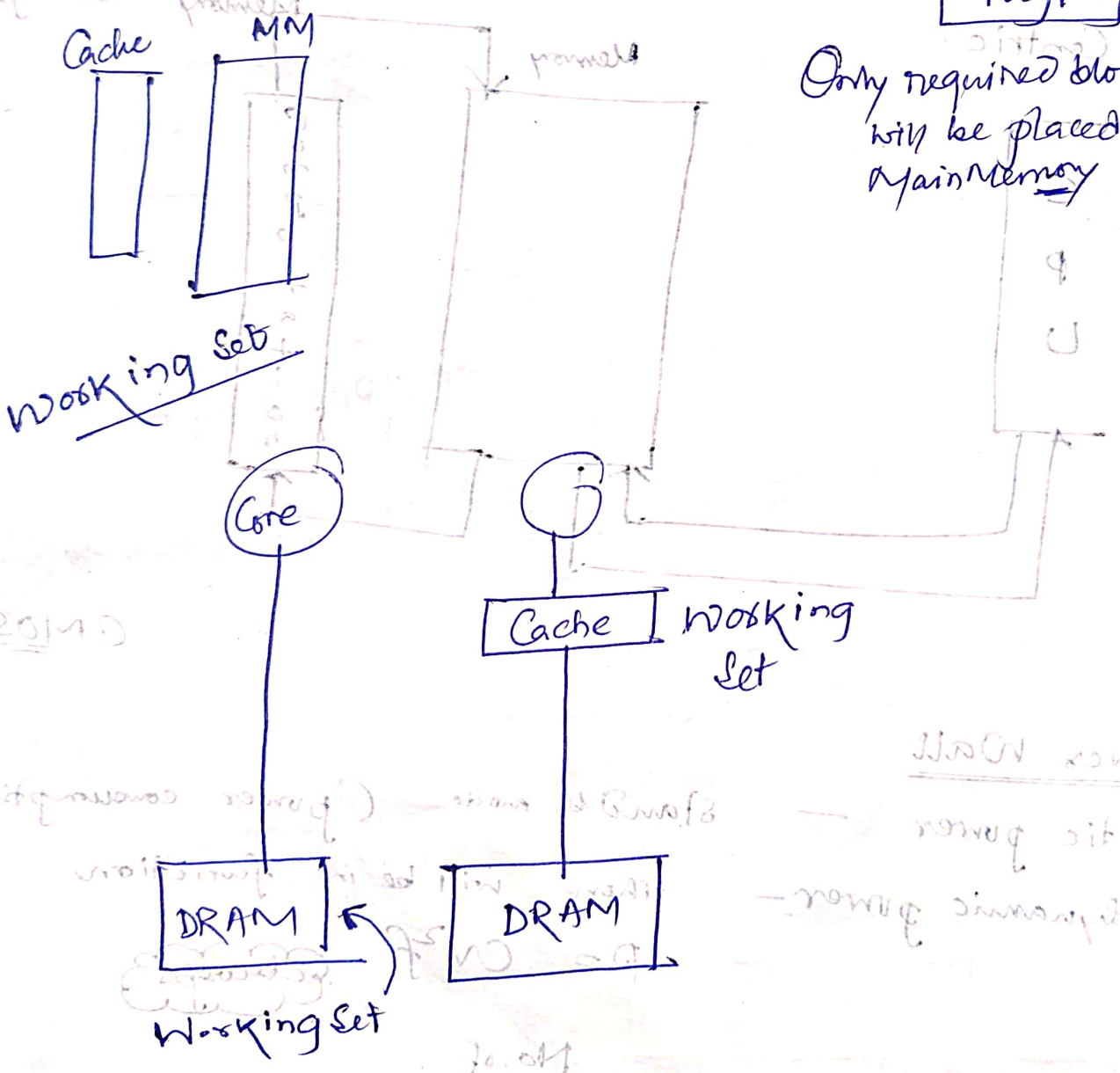
4. Technology Wall

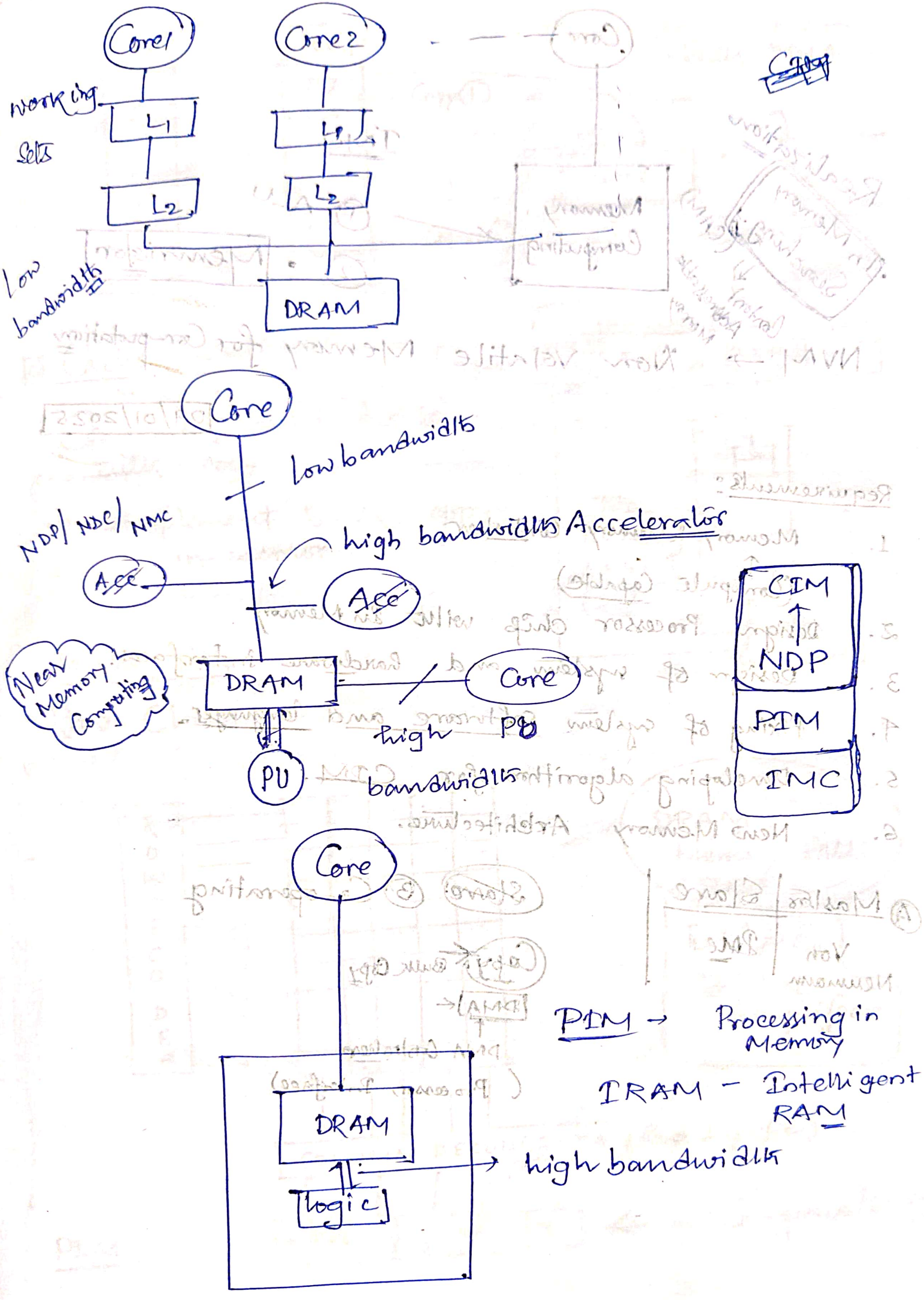
5. Reliability Wall (Economically NOT Ready)

6. Cost Wall

Paging

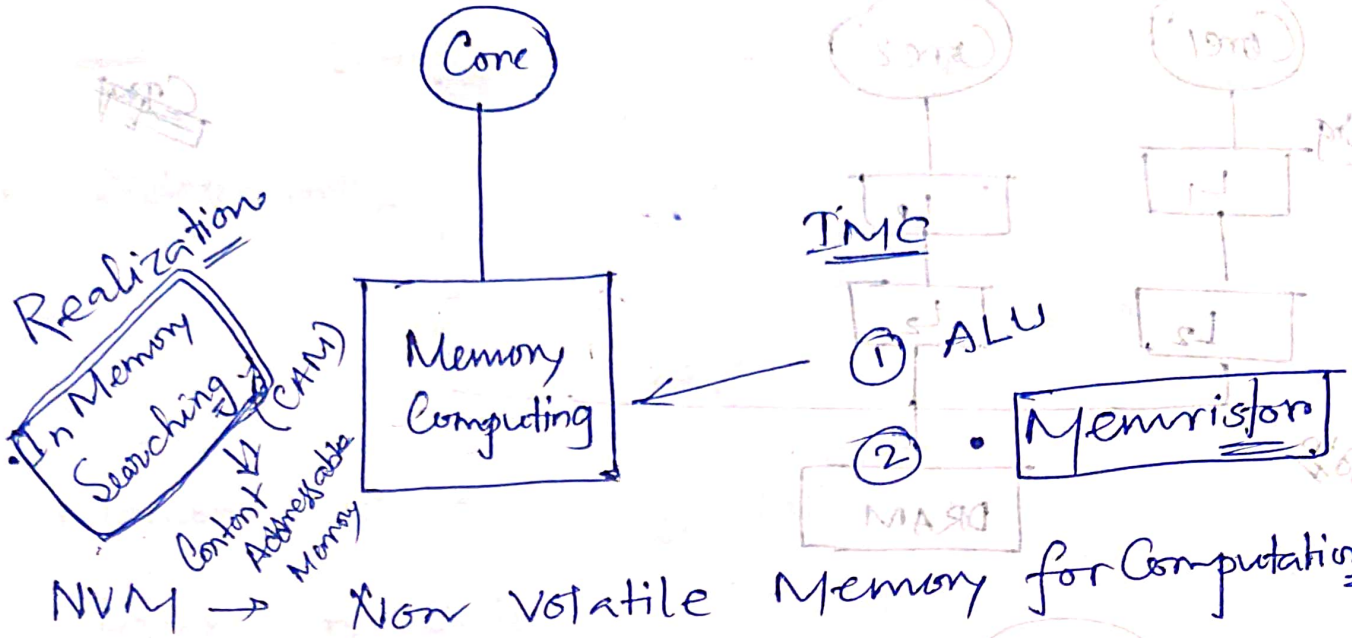
Only required block will be placed in Main Memory





PIM → Processing in Memory

IRAM – Intelligent RAM



09/01/2025

Requirements:

1. Memory & Memory Controller
   (Compute Capable)
2. Design Processor chip with In Memory
3. Design of system and hardware interfaces.
4. Fixing of system software and languages.
5. Developing algorithm for CIM.
6. New Memory Architecture.

(A) Master	Slave
Von Neumann	DMA

(Slave) (B) Co-operating

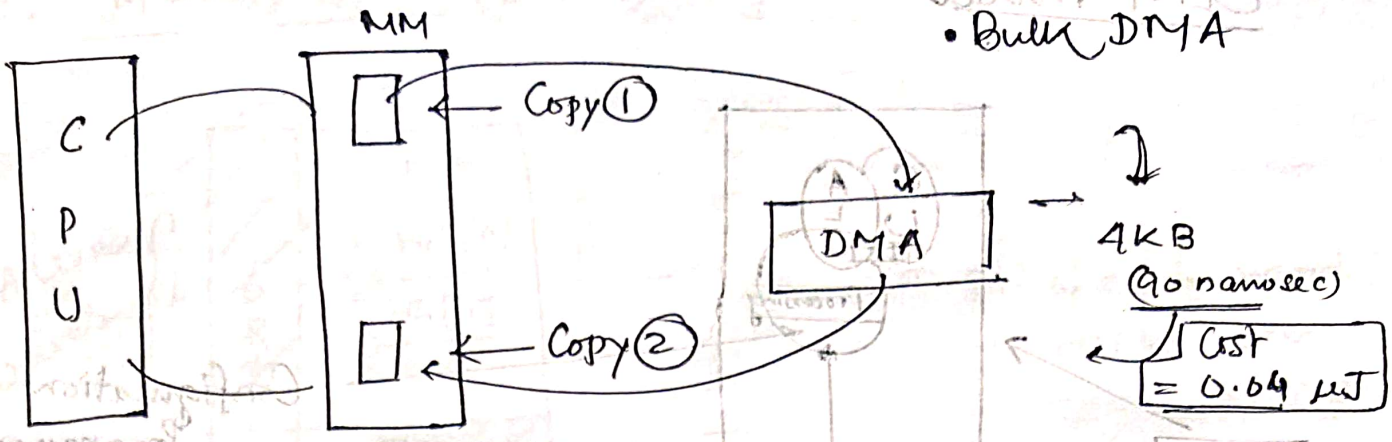
Copy ← Bulk Copy

DMA ←

DMA Controllers

(Processor, Interface)

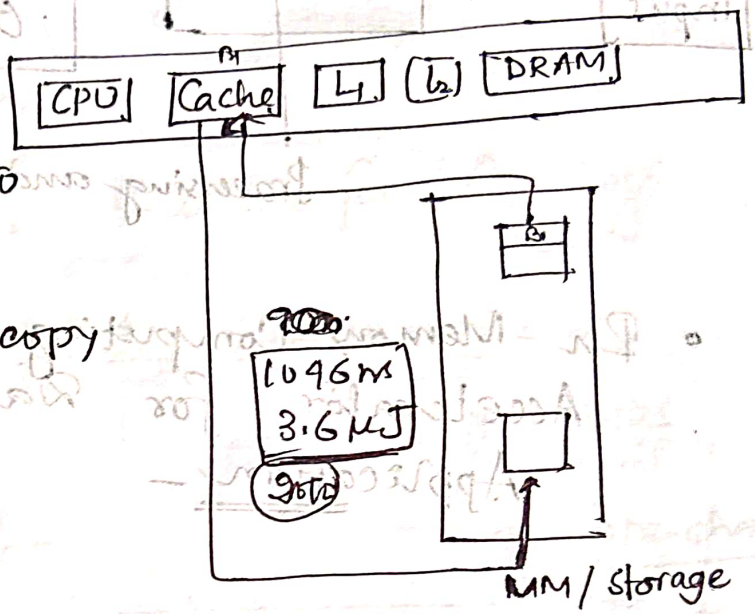
• Bulk DMA



Cache Pollution

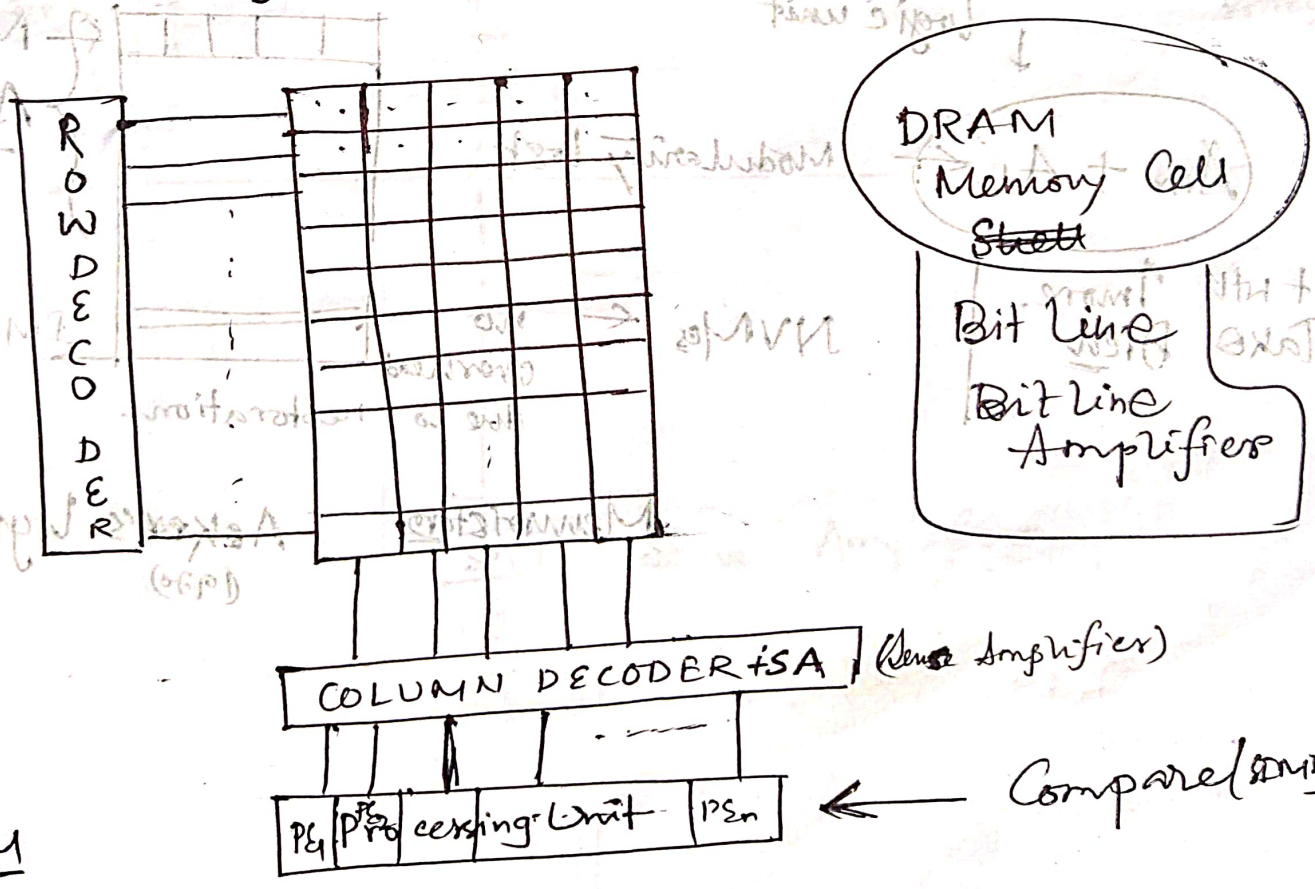
Cache is polluted due to bulk copy.

→ Implement bulk copy in-memory.



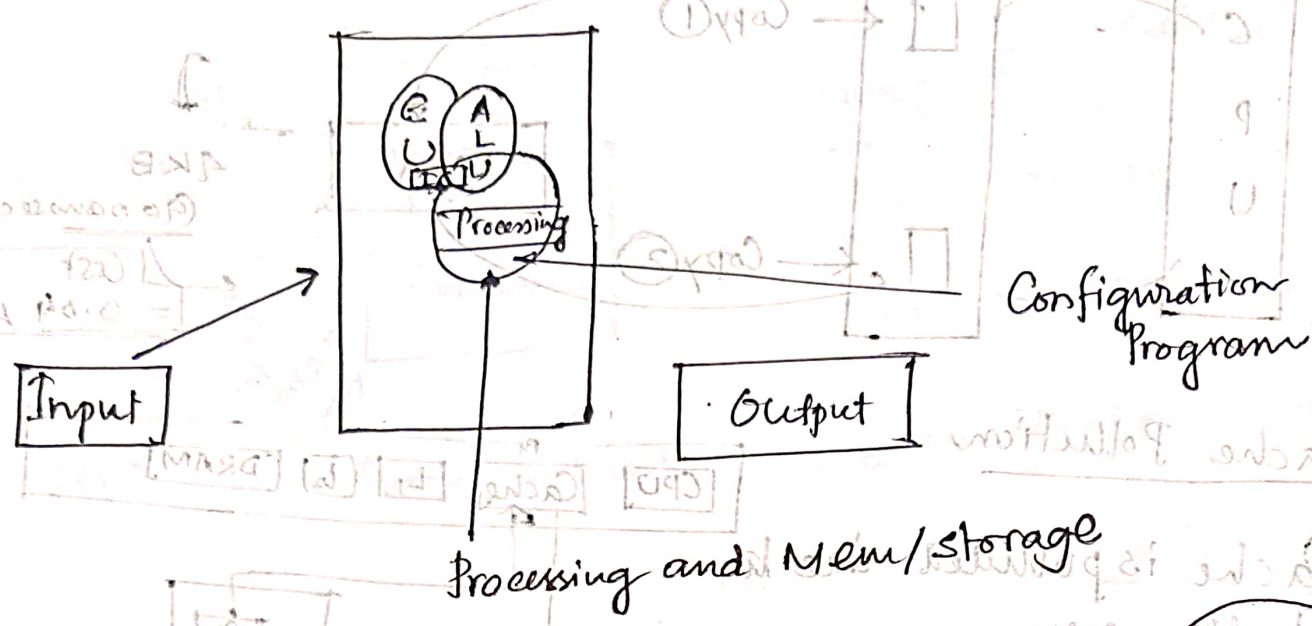
Bulk Copy

• Row Cloning



PIM

CIM Model



Springer

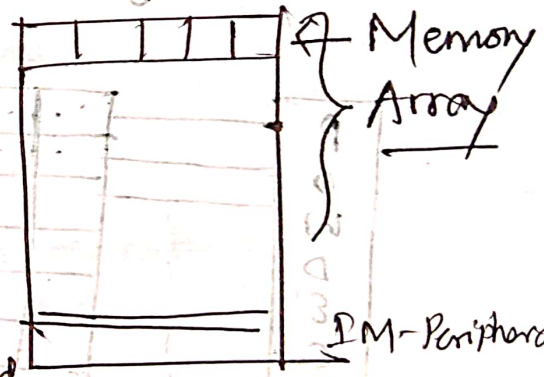
- Ru - Memory Computing Hardware Accelerator for Data Intensive Application — Halwani.

ALU → in Memory

Logic unit

Am + AL ← Modularity lost

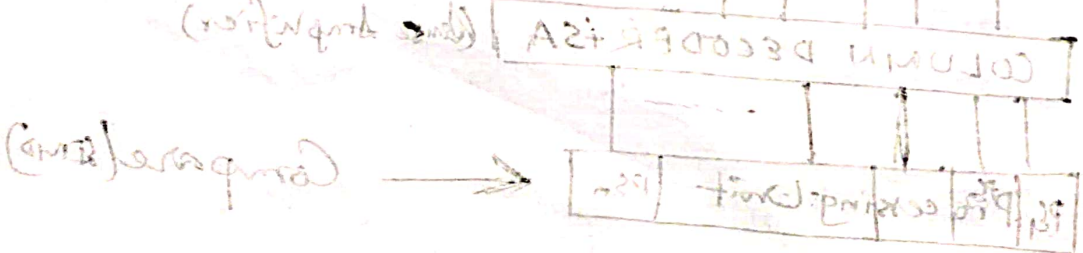
It will more. Take Drea



NVMes ← no overhead due to restoration.

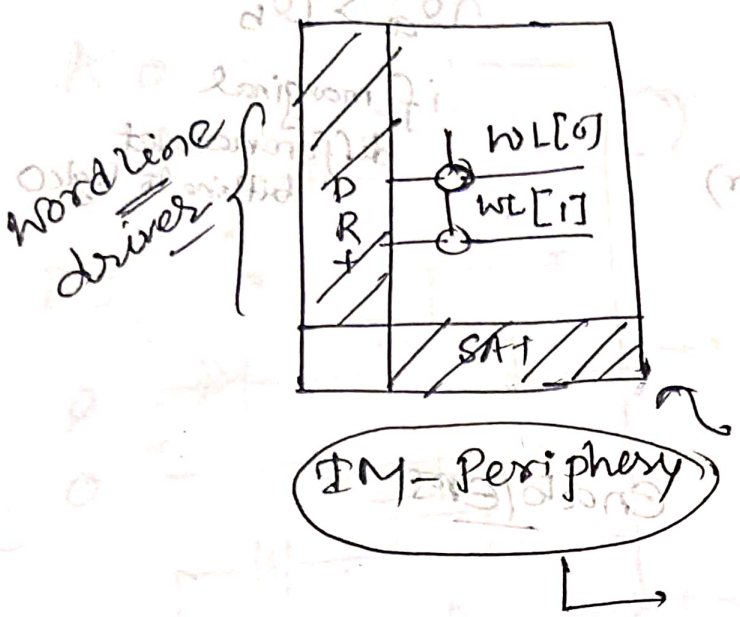
Memristors.

Aaker's logic (1970)



PIM

In-Memory Array (IM-A)



with bit of each word will be on same bit line

Peripherals

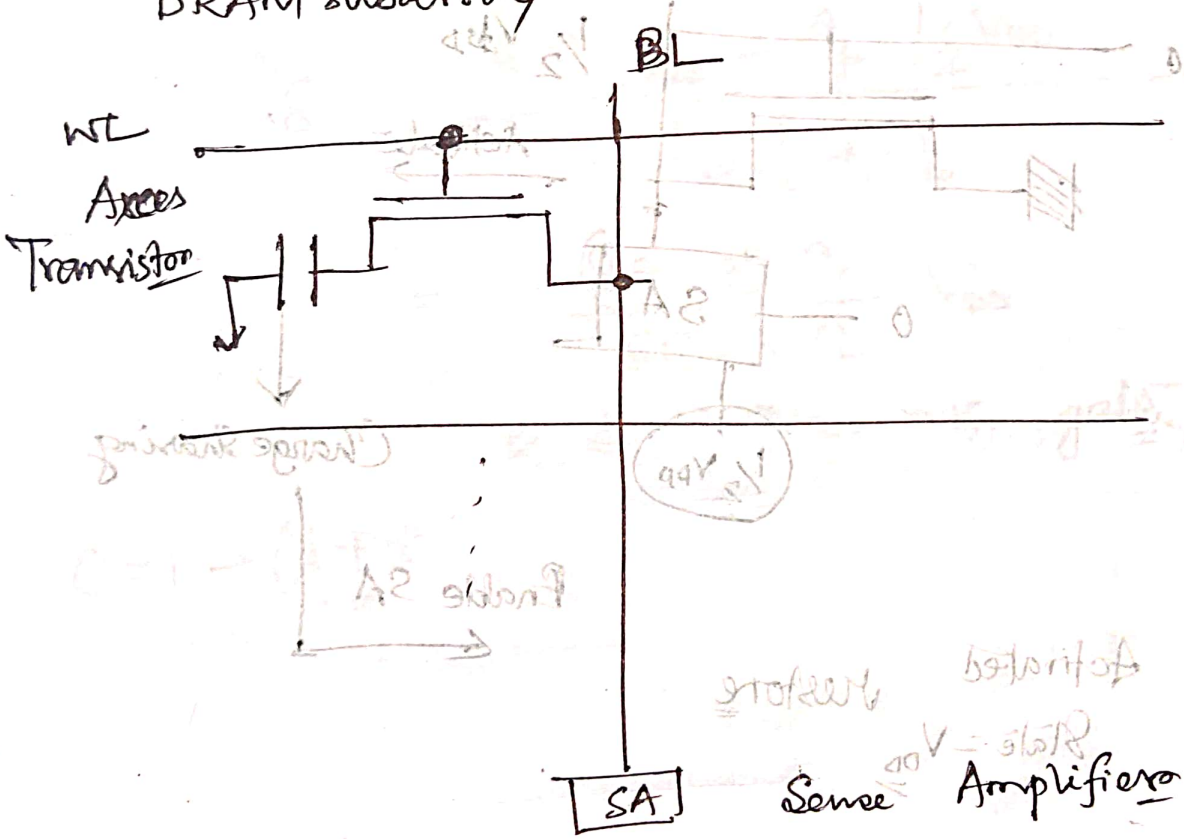
IM-Periphery

Suitable for Analogs

In Memory Computing in DRAM

~~DRAM~~ MOS

DRAM Subarray



pre-charge

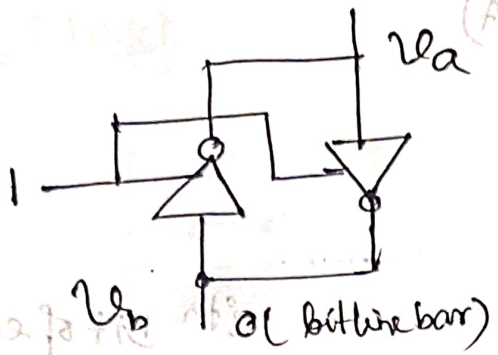
↓

logic1 stored

Sense Amplifiers

↓

• Handling bit line bitline



$v_a > v_b$

if marginal difference set bitline to logic 0

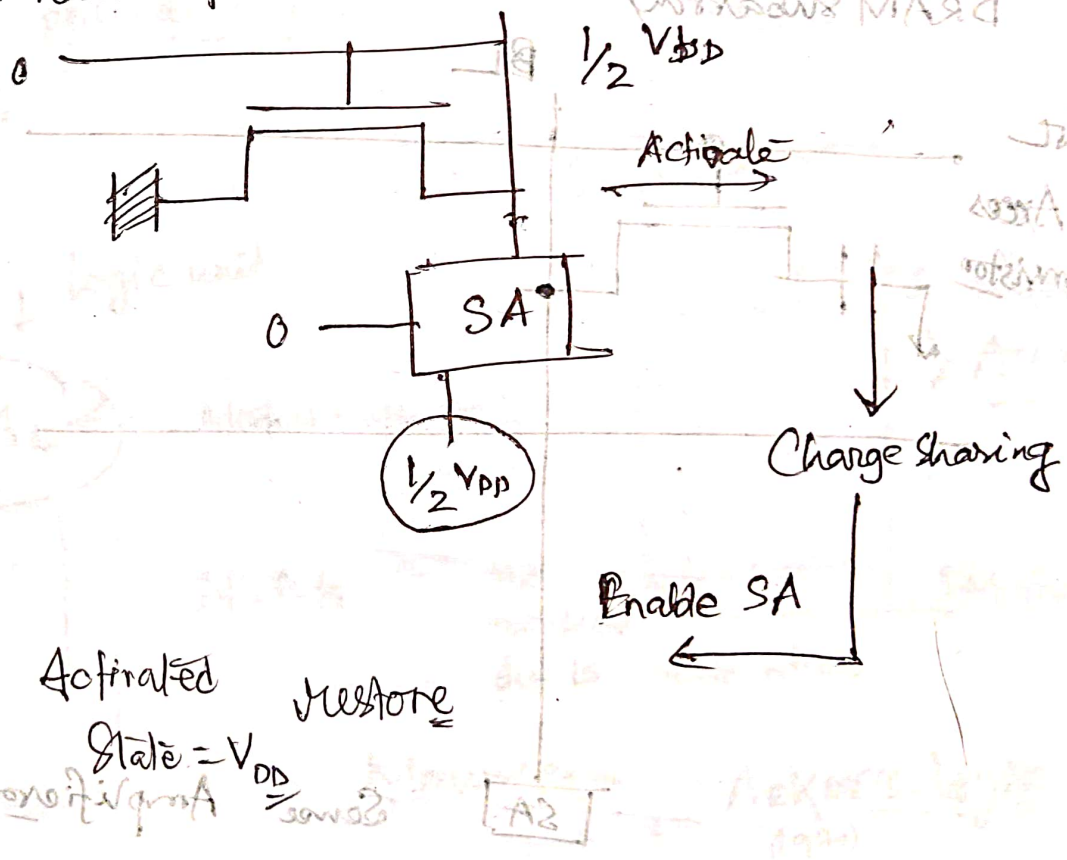
Memory Controllers

- Charge-Sharing

Enablers

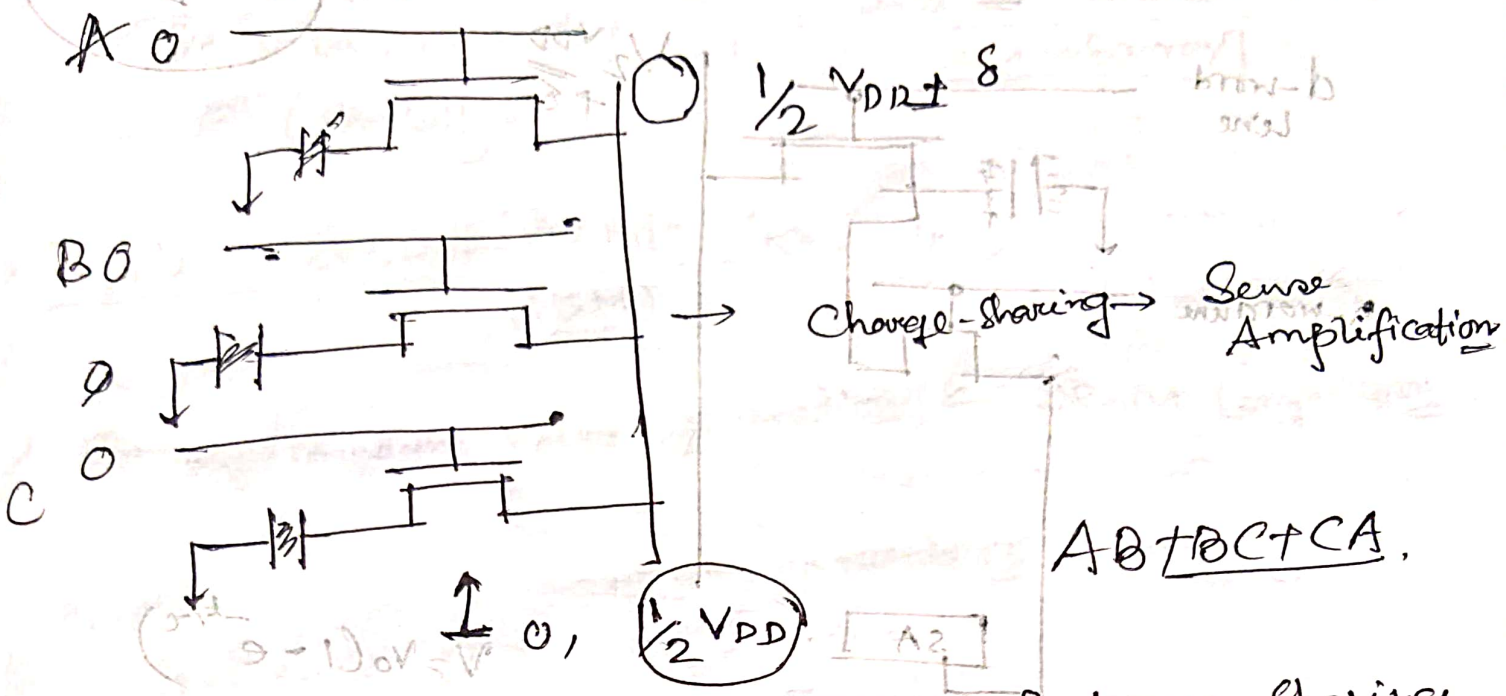
DRAM Read Sequence

- Normal Read Operation



Activated State = V_{DD}

Triple-Row Activation



bitline deviation at the end of charge sharing

$$\delta = \frac{k C_c \cdot V_{DD} + C_b \cdot \frac{1}{2} V_{DD}}{3C_c + C_b} - \frac{1}{2} V_{DD}$$

$$= \frac{(2k-3) C_c}{6C_c + 2C_b} V_{DD}$$

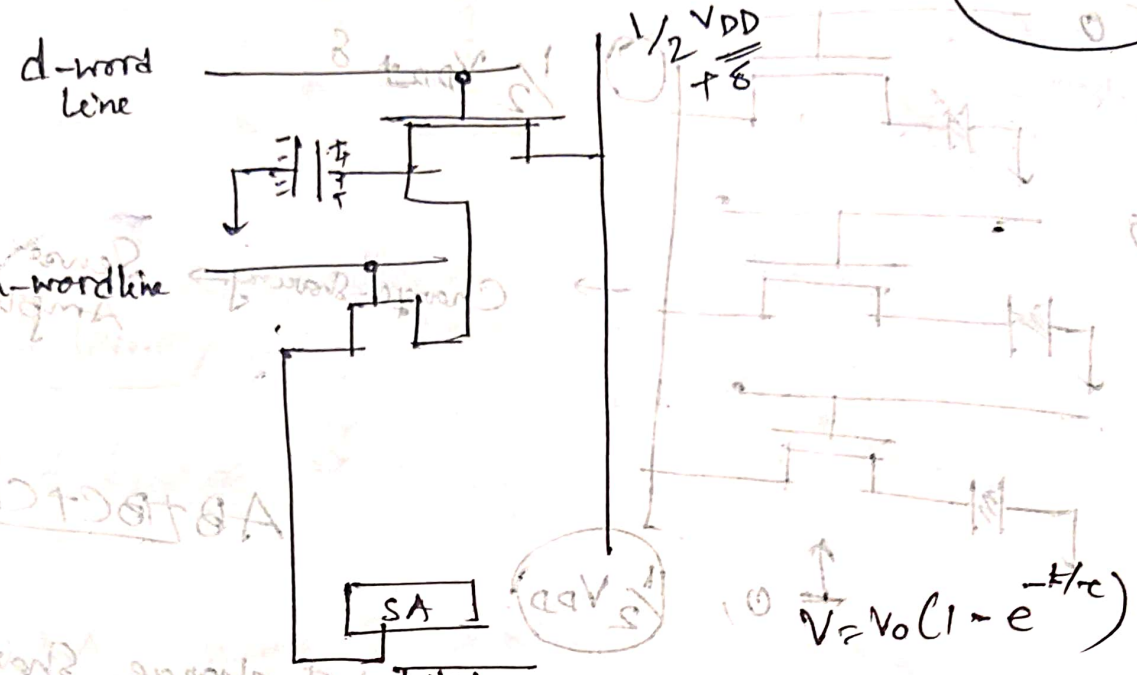
k ← Majority gate.

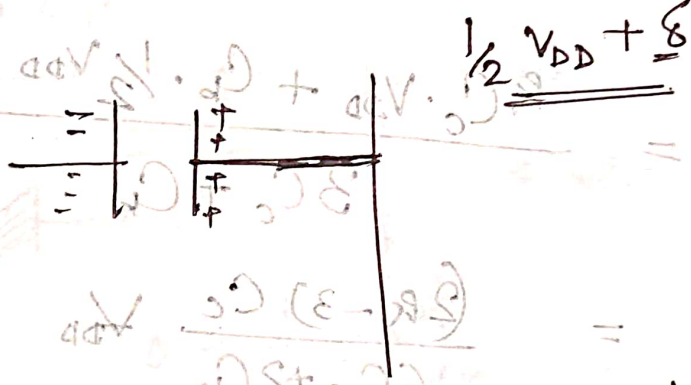
C = 1 → (A+B)

NOT

- Dual-Contact Cell.

$U = \frac{1}{2}CV^2$





Row Clone: Bulk Copy and Initialization

- FPM
- Pipeline

- **Fast Parallel Mode**

  (src $\xrightarrow{\text{bitline}}$ dest)

  src → buffer → dest [src and dest will be in same subarray]

  (SA + bitline)

- **Pipeline Serial Mode** (Row clone) (PSM)

  ~~Trng~~ → random num generation ← DRAM Computation

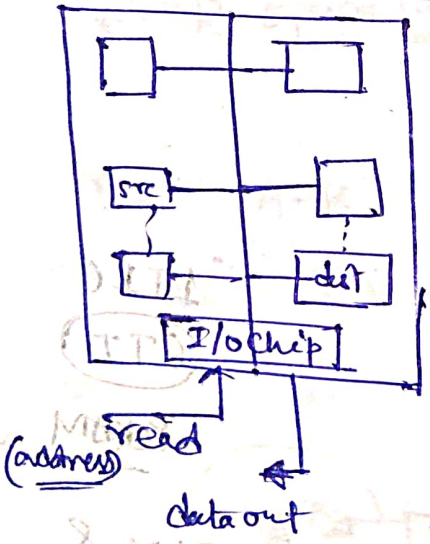
  ⇓

  PseudoRandom Numbers

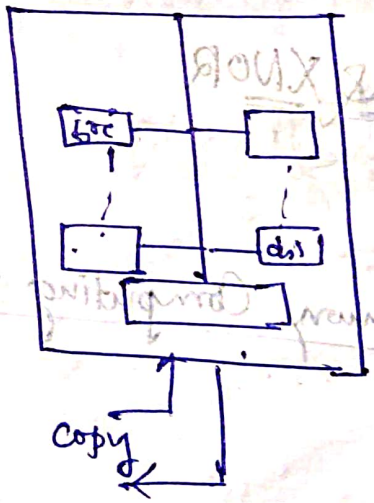
  - following DRAM Computation.
  - ML Applications..

Pipeline Serial Mode (PSM)

16/01/25



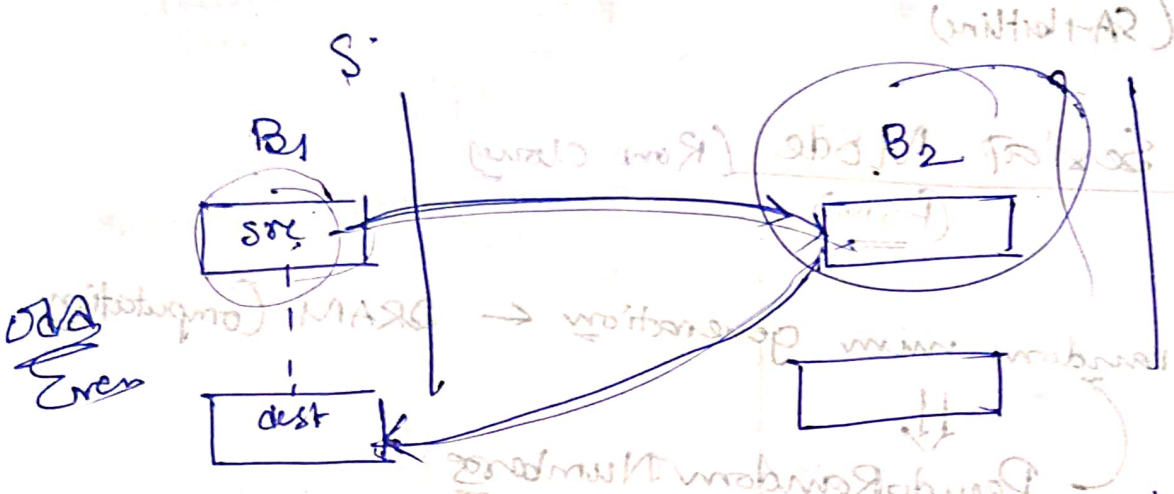
a) Normal Mode



b) Row clone

Bulk Data Copy

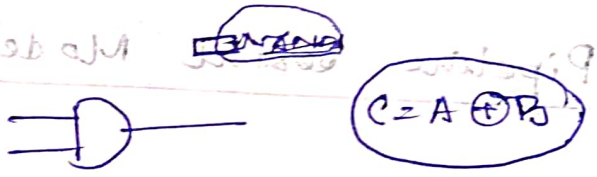
* src & dest are in different subarray but same Bank



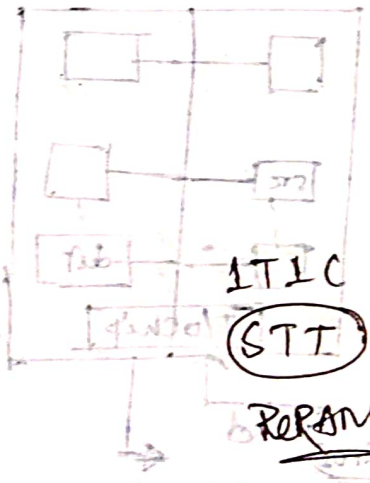
Dynamic Refreshing

- Bulk Initialization
- Bulk Zeroing

- ROC
- XOR & XNOR



In-Memory Computing SRAM

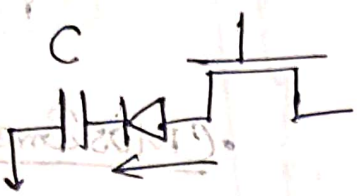


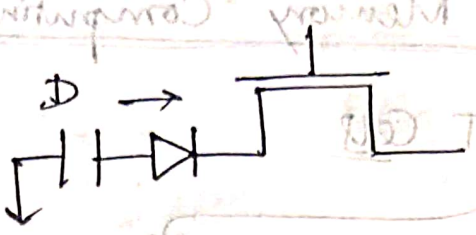
Common Applications of DRAM & SRAM (In Memory) And differences

In Memory Computing SRAM

21/01/25

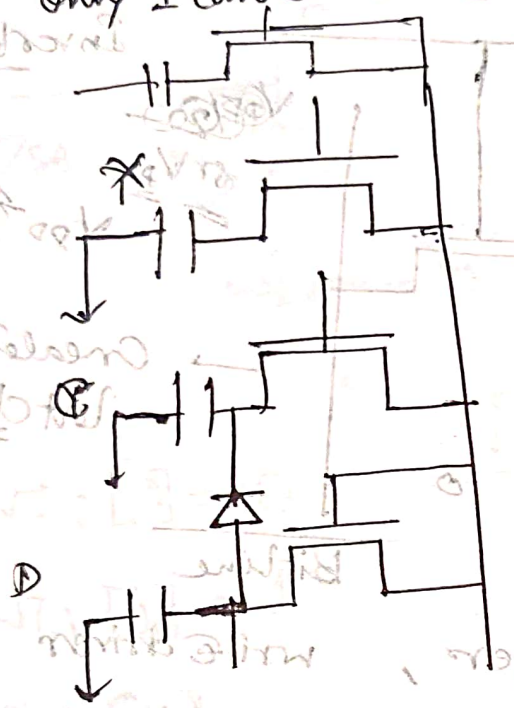
6. ROC (Reduced Operating Cycles)



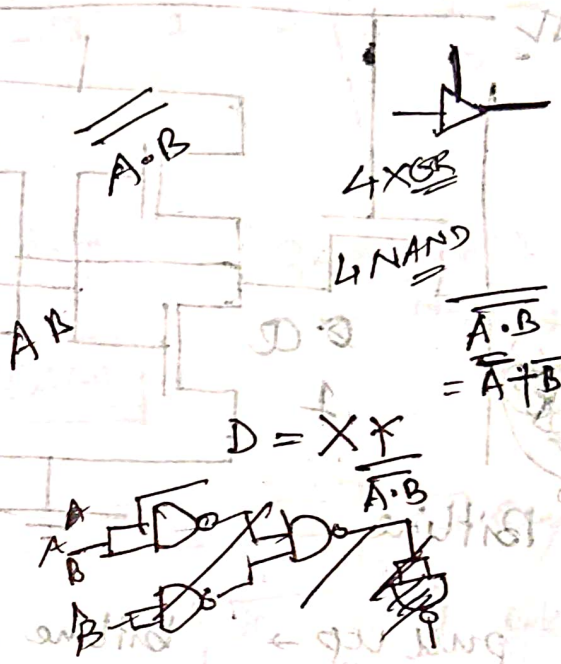


Only 1 can be written in C

Only 0 can be written in D.



@ OR operation



(b) AND Operation

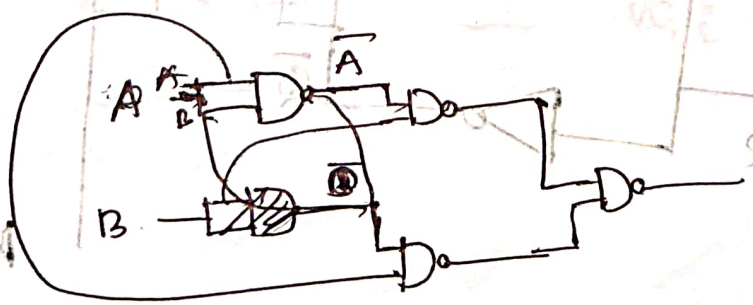
$Y = A \cdot B$

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

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

$= \overline{A \oplus B}$

$= \overline{A} \cdot B$



XNOR

CAM

In Memory Computing in SRAM

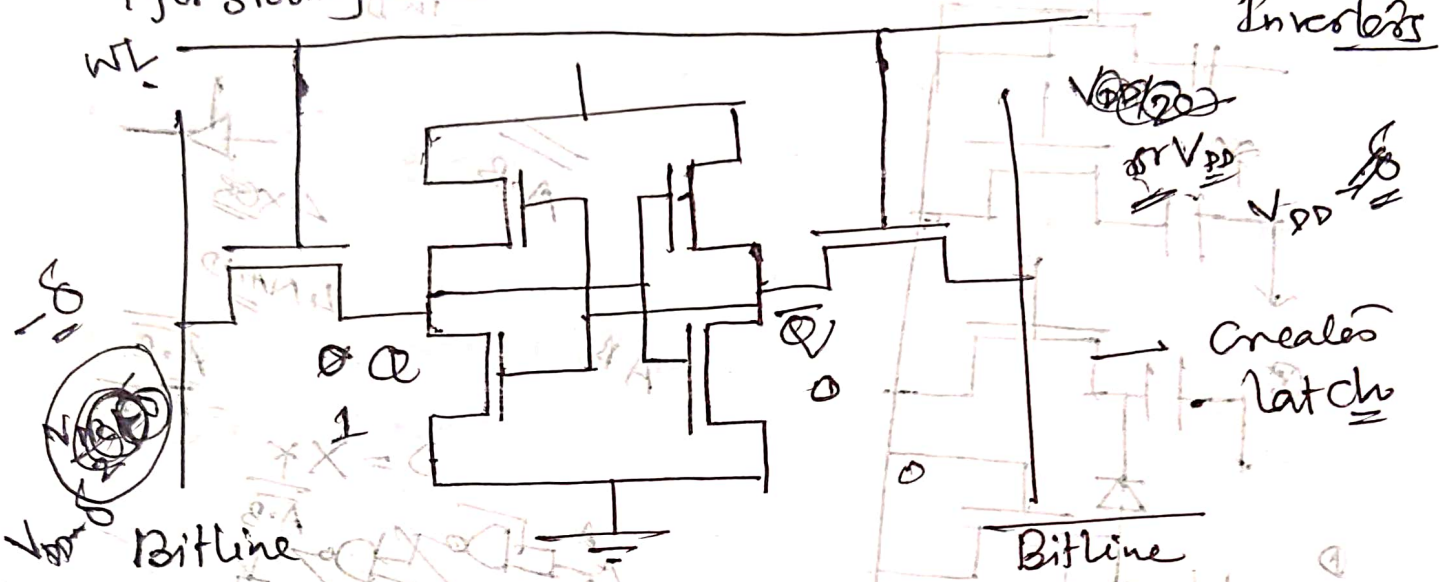
6-T Cell

- CMOS Inverters.

4 for storing data

Access transistor

Cross Coupled Inverters.

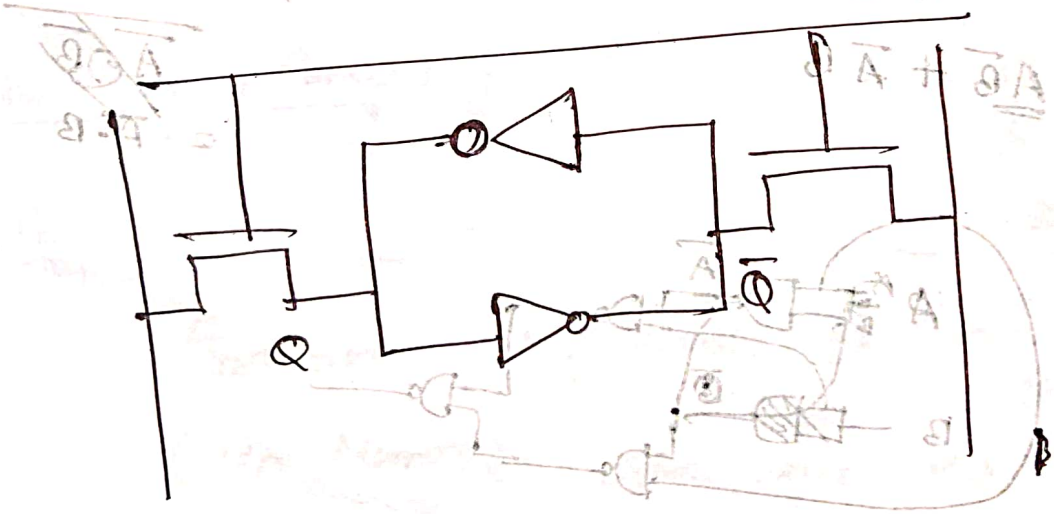


pull up → transistor

bitline prechargers, write drivers

- what is available to bitline.

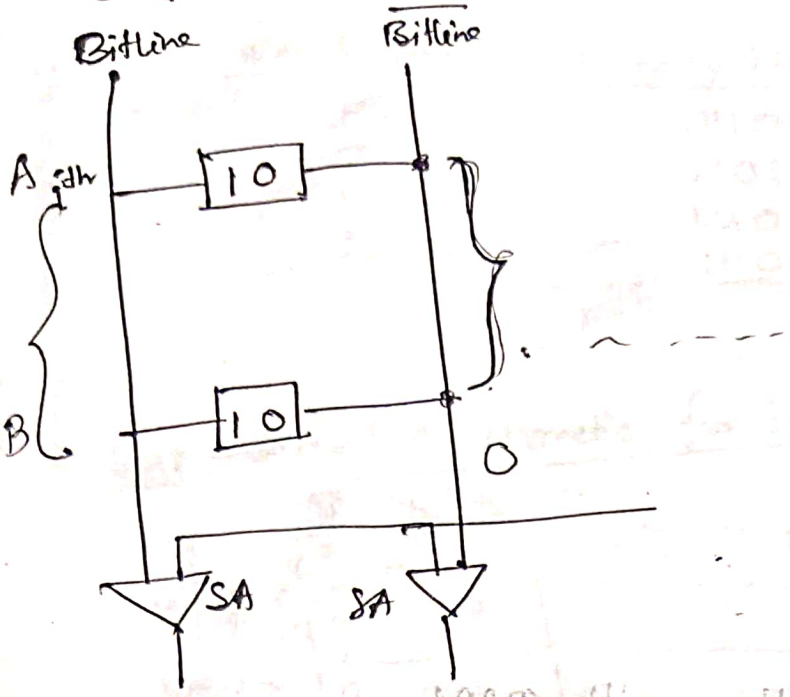
written on the cell will be



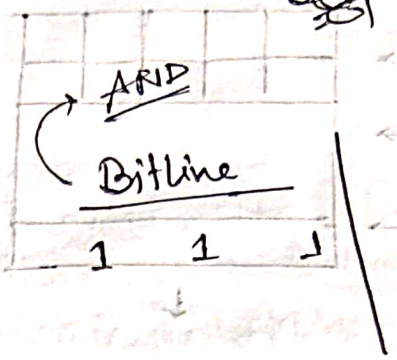
flip Q & $\bar{Q}$

↑ pulled low by write drivers

Digital Computing



Single Ended SA
Differential



NOR

Bitline

1 1 0

0 1

WL_i NOR WL_j

3-ip NAND

NOR

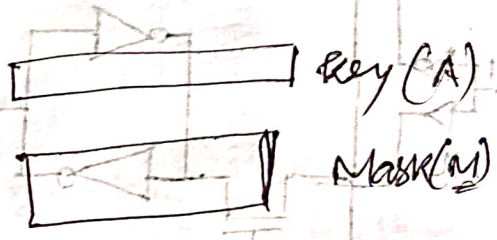
AND

SRAM-CAM

$WL_i[i] = 0$

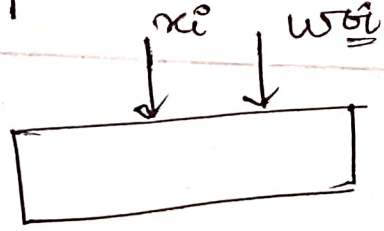
$WL_j[j] = 1$

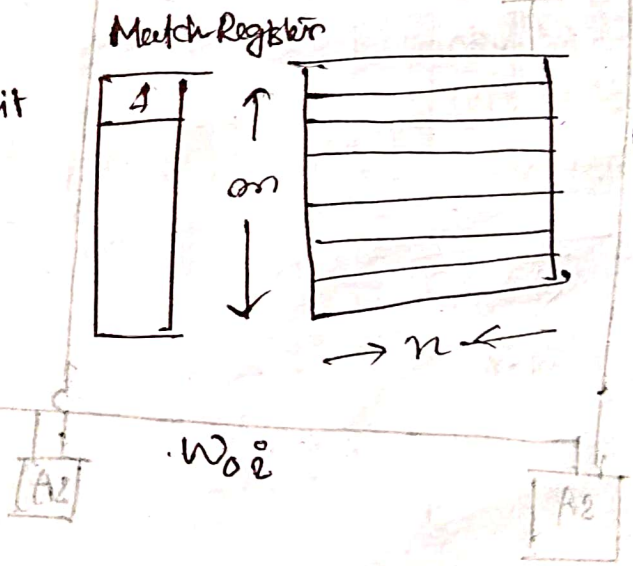
$WL_k[k] =$



Search Operation

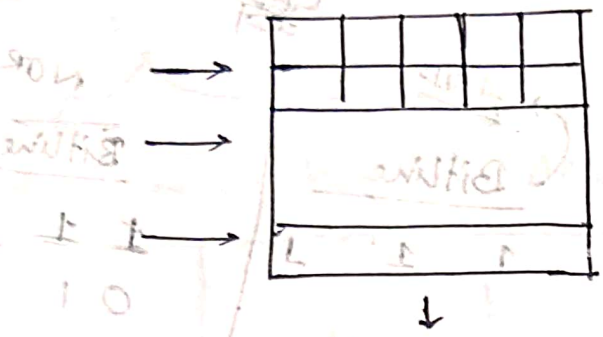
- Comparelogic

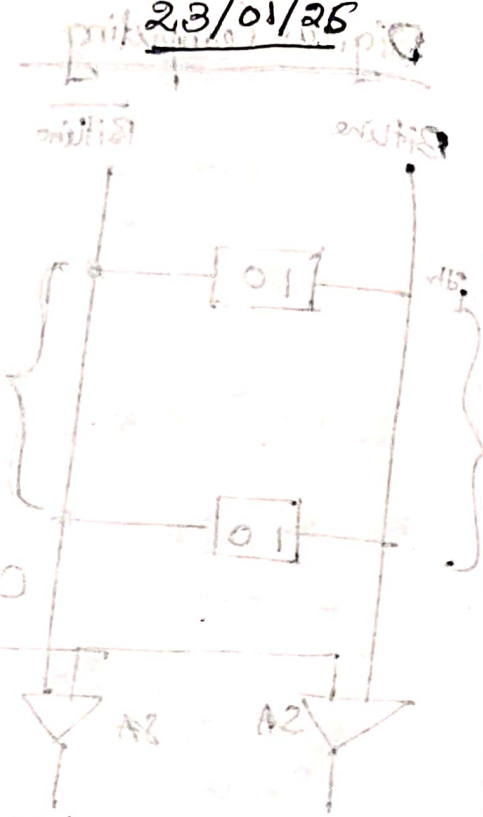




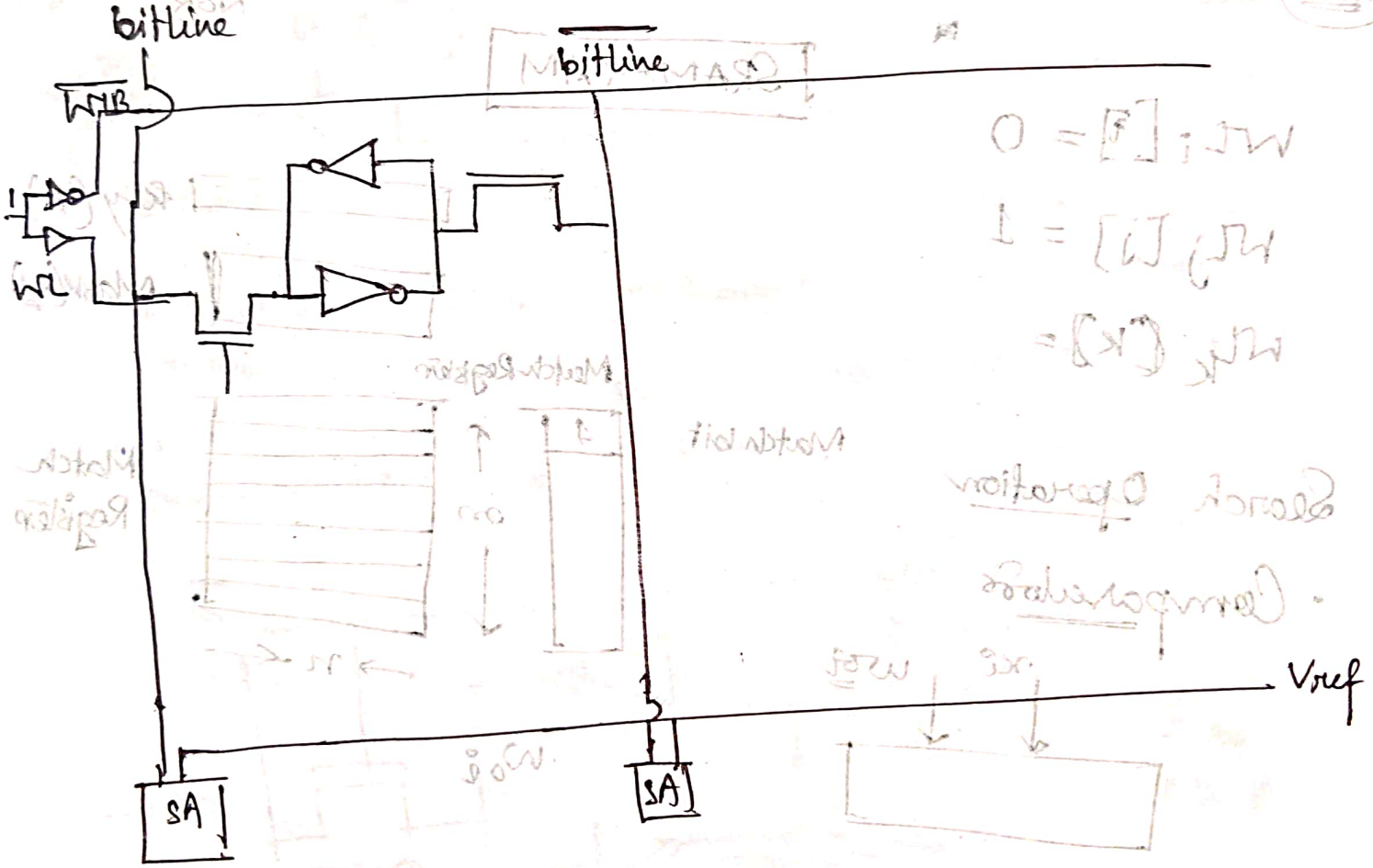
23/01/26

Search



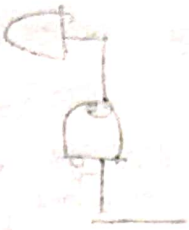


CAM Circuit Implementation with SRAM

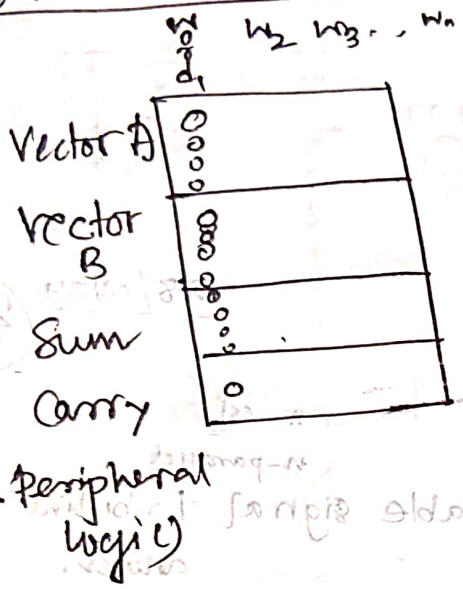


4X4 array of SRAM

Search key	Memory word
1	0110
0	0101
1	1110
1	1110



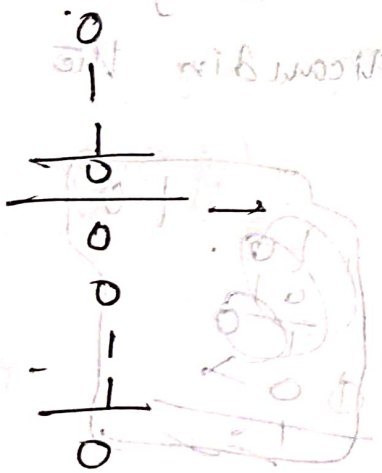
Bit-Serial Arithmetic for Integers



1. Two operands mapped to same bit line
   → Ripple Carry Adders.

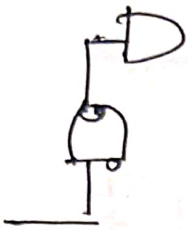
① Read word Lines (RWL) ② 2nd half cycle WWL is activated to store back

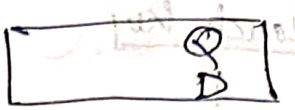
(n+1) cycles.



Array Peripherals

SRAM Peripherals

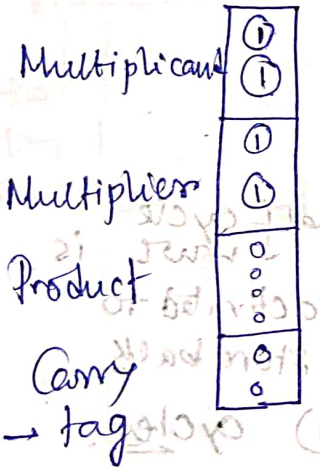




- Sum ($A \oplus B \oplus C_{in}$)
- Carry ($(A \& B) + (A \oplus B \& C_{in})$)

28/01/24

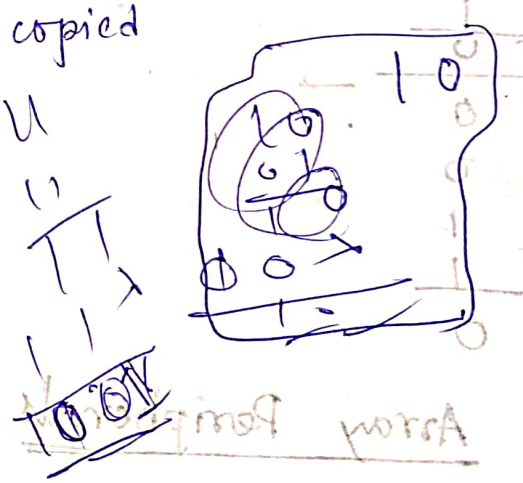
- In-SRAM Multiplication



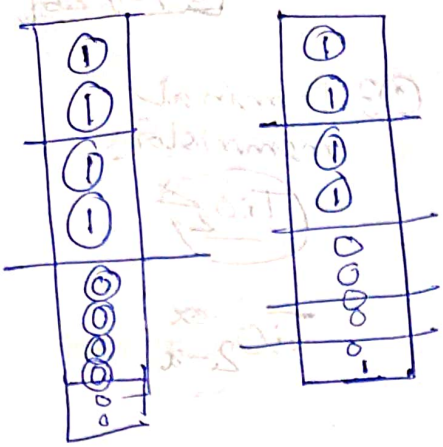
- Tag is an enable signal to bitline driver.
- When tag is 1, Sum is written back to product array.

If tag is 1, multiplicand in the bitline is copied

n^2+5n-2 cycles for n bits.



Reduction



- Memristor
  - MAGIC
  - IMPLY
  - Hybrid CMOS

RRAM

(SIMD Impl.) Carry Lookahead Adders

Akers Logic

Memristor based 2×2 binary multiplier

- N-Input NRL
- DRAM
- SRAM
- Memristor
- MRAM
- PCM

Memristor → Searching / Sorting using Akers Logic

QCA

Memristor

NVMe

30/01/25

- two terminal device

Memory + Resistor

I, q, V, ϕ

3-terminal memristor

TiO_2

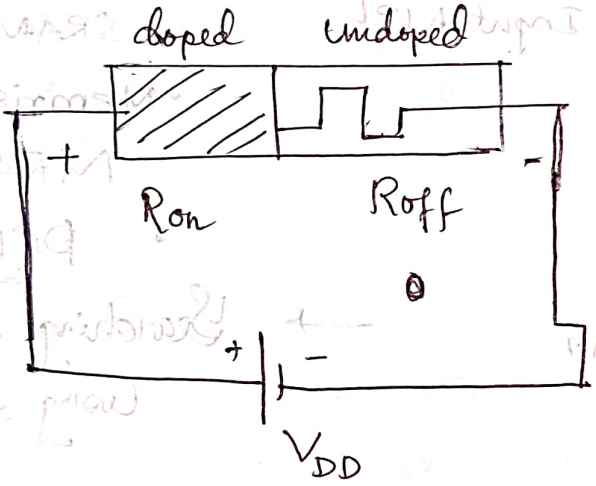
TiO_{2-x}^{-ox}

$$d\phi = M \cdot dq$$

$$\text{Memristance } (M) = \frac{d\phi}{dt} \cdot \frac{dq}{dt} = \frac{v}{i}$$

Memristive device

$R_{off} >> R_{on}$



R_{off} will be 1 if $V_{DD} > V_{Threshold}$

logic design style

& classes

- RVH
- RRM → Snider, Stateful logic
- MRM

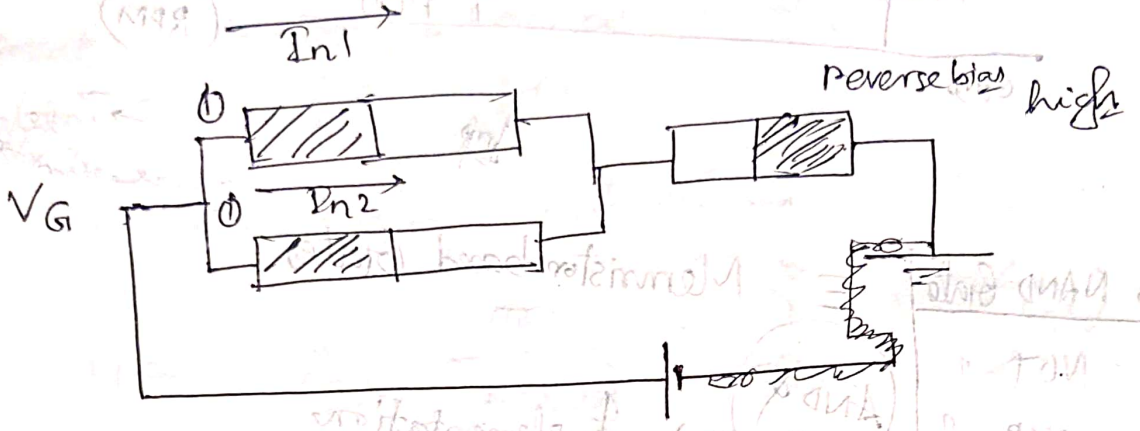
Stateful logic design

- IMPLY (Memory based material Implication)
- MAGIC (Memory aided logic)
- Memristive Array

$0 \cdot 0 = 1$

$1 \cdot 1 = 0$

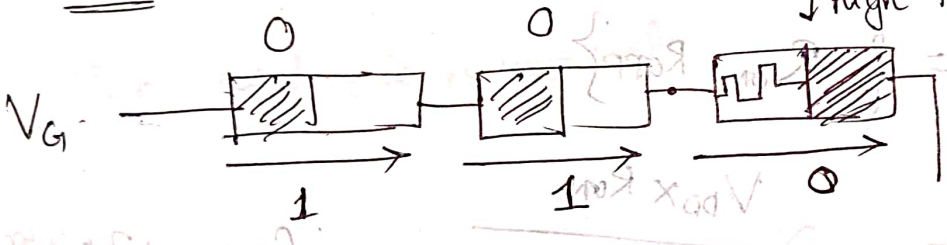
MAGIC NOR



Memristive Crossbar Structure

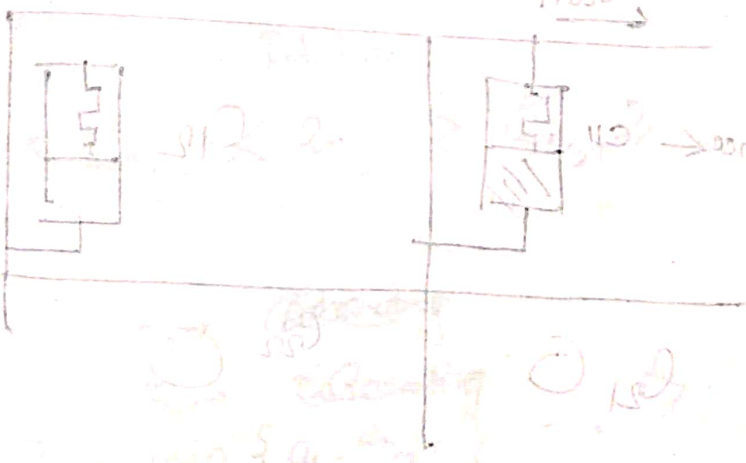
n input NAND

NAND



MRL

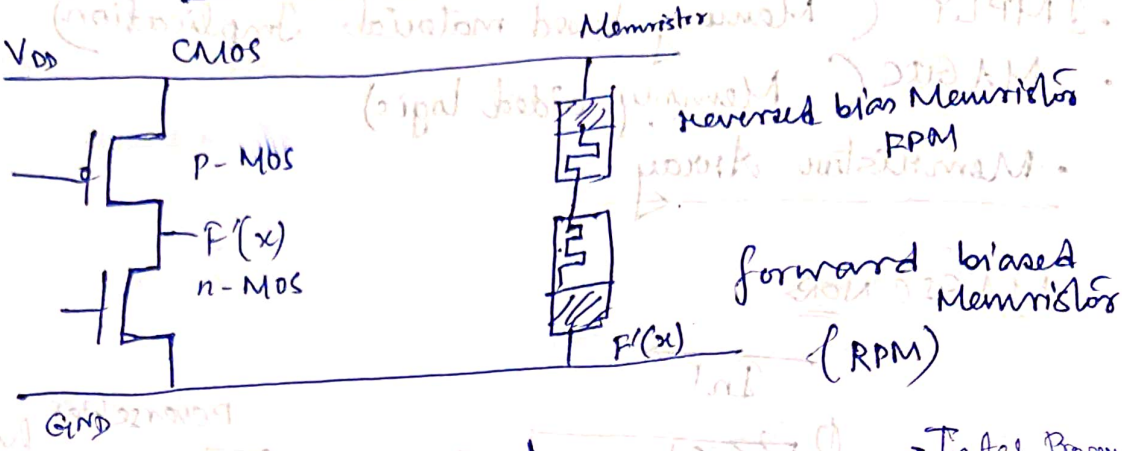
$dq = M \cdot dq$



11/02/25 AND/OR.

• N-Input MRL | • Integration with CMOS Inverters

AND/OR



→ Inter Processor
CMOS → Memristor

• CMOS NAND Gate ≡ Memristor based Gates

NOT "

NOR "

AND & OR → Implementation

$$\{FPM, RPM\} = \{R_{ON}, R_{OFF}\}$$

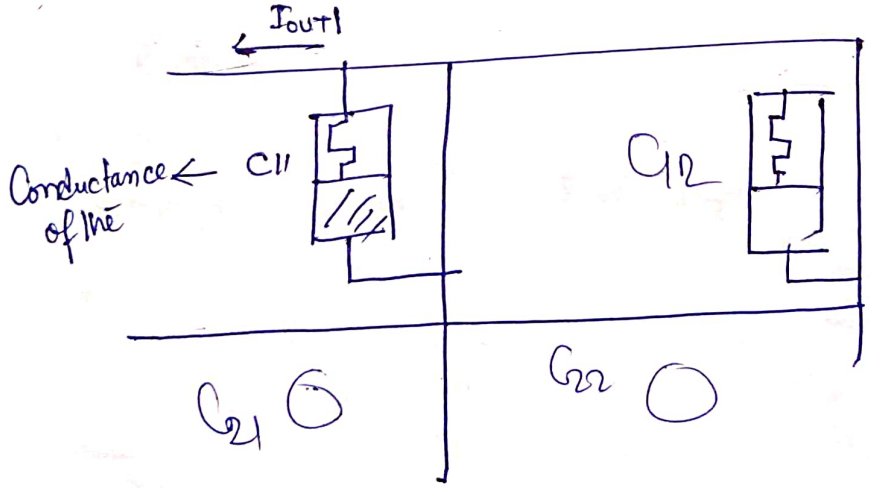
$$V_{OUT} = \frac{V_{DD} \times R_{ON}}{R_{ON} + R_{OFF}}$$

{ VMM → Vector Matrix
MVM → Matrix Vector

In-Memory Matrix Multiplication

$Ax = b$

$x = A^{-1}b$



$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv$$

$x \leftarrow$ In terms of voltage

↓

mapped linearly to amplitudes or duration of read voltage.

Result b of computation $\propto$ resulting current measured along columns of the array.

Current is summed across each column.

$$i_j = \sum_{i=0}^{n} \frac{v_i}{R(i,j)} := \sum_{i=0}^{n} v_i C(i,j)$$

$R(i,j)$ ↓ Cell resistance

$C(i,j)$ ↑ Conductance

$$I = V_I \times C$$

One read cycle is required.

$$C_{m \times n} = A_{m \times p} \; B_{p \times n}$$

In-Memory Sorting

Compare & Swap — Batcher = (Bitonic) 1, 5, 7, 8, 3, 2, 0

Cyclic Rotation

$$a_0 \leq a_1 \leq a_2 \leq a_{n/2-1} \leq a_{n/2} \geq a_{n/2+1} \geq$$

Bitonic Sort

$S_1 \rightarrow$ ~~Ascending~~

$S_2 \rightarrow$ ~~Descending~~

$$S_1 = \langle \min\{a_0, a_{n/2}\}, \min\{a_1, a_{n/2+1}\} \ldots \rangle$$

$$S_2 = \langle \min\{a_0, a_{n/2}\}, \ldots \rangle$$

1, 5, 6, 7, 11, 13, 10, 8, 4, 2, 0

$S_1 = \{1, 5, 7, 4, 2\}$ ⟶ 1

$S_2 = \{13, 10, 8, 9, 11\}$

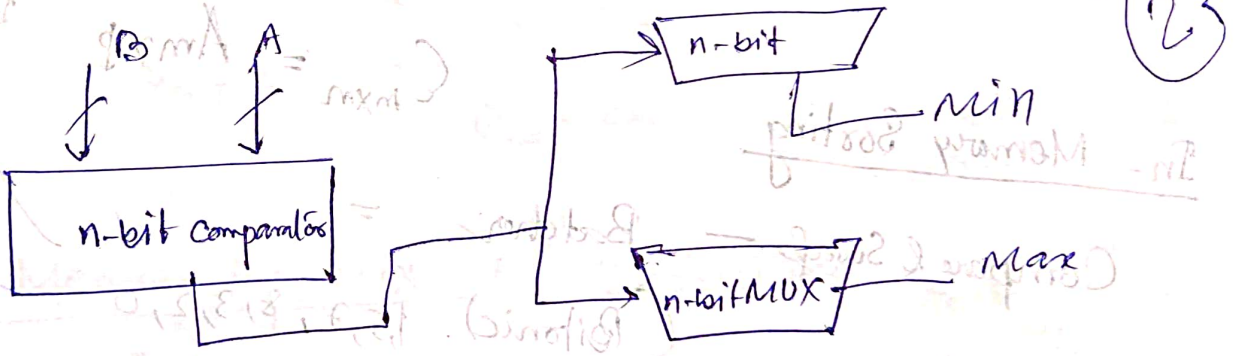
Comparing & Swapping ⟹ bitonic sequence generation

$n = \{1, 0, -1\}$

$n/2 =$

$Y = 19, 2, 72, 3, 18, 57, 600, 150$

left ... key



2^{n+1}

Sorting in Memristive Memory