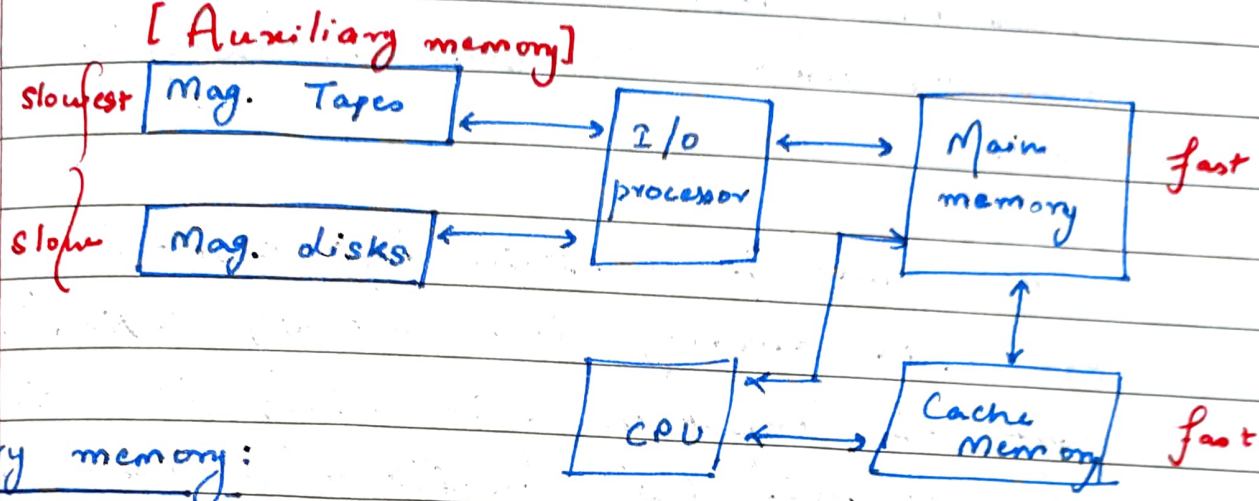
DATE: __/__/____
PAGE: _____

Ch 12 Mano

Memory Organization

Memory Hierarchy →



Auxiliary memory:
provides backup storage. ← ∴ Programs that are not being used

Main memory:
memory unit that directly connects to the CPU. ← ∴ Programs that are currently needed are stored in main memory.

Cache Memory: > It is very special, v-high speed memory.

- Used to increase the speed of processing by making current programs and data available to the CPU at a rapid rate.

→ Since CPU is faster than main memory, thus limit of processing speed is due to main memory.

→ The cache stores segment of programs currently being executed in the CPU & temprory data required frequently in present calculations.

Good Write

DATE: __/__/____
PAGE: ____

Multi-Programming

refers to existence of 2 or more programs in different parts of memory hierarchy @ the same time. When one program is waiting for input or output transfer, another program is ready to utilize the CPU.

The task of maintain information's part in the main memory i.e. currently active. The part of Computer System that supervises the flow of info b/w^n auxiliary & main memory is "Memory Management System."

MAIN MEMORY

are based on Semiconductor integrated ckts.

RAM chips can be Static or dynamic

- Static RAMs consist of internal flip-flops that store the binary information. This remains static as long as power is applied to it. [Easy to use]
- Dynamic RAMs stores info in form of electric charges applied to the capacitors, this stored charge tends to discharge with time thus it needs to be periodically recharged by refreshing the d-memory by cycling thru the words every few milli secs to restore charge. [Less consumption]

Good Write

DATE: __/__/____
PAGE: ____

Main memory has both:

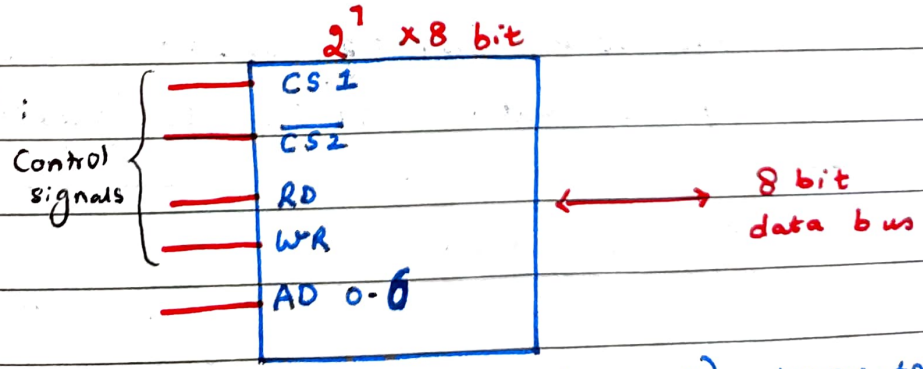
RAM is used to store bulk of programs which are subject to change.

Rom stores the permanenty resident programs, tables of constants which wont change.

Bootstrap Ladder : The initial program is stored in the Rom portion ; it is called Bootstrap loader. Its function is to start the computer system when power is turned ON.

It consist of turning power ON & starting to execute the initial program. Thus when turned on ; PC ← first address of the Bootstrap ladder. It loads a portion of OS from disk to main memory & the control is transferred to the OS. (for general use)

RAM Chip :



A 8-bit data bus (bi-directional) connects it with the memory.

→ Write (WR)
→ Read (RD)

CS1 = 1 & CS2 = 0 for it to work.

DATE : ___/___/_____
PAGE : _____

when WR = 1 CS1 = 1 CS2 = 0

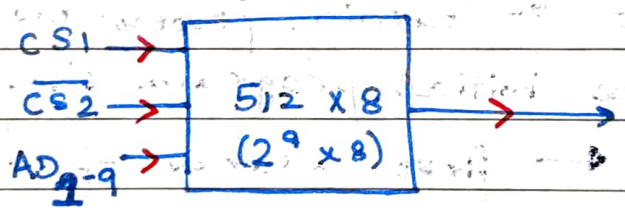
the RAM accepts the data from the 8 bit data bus to the address given by AD_{0-6}

when RD = 1 CS1 = 1 CS2 = 0

the RAM sends the data bust with the data given by the address AD_{0-6}

* Rom chip

for the same size Rom has more bits as its cells occupy less space



Single directional as Rom can only send data. or only (RD) oper^n can be used here.

whenever CS1 = 1 & CS2 = 0 then data bus is given data stored by address given by AD_{1-9}

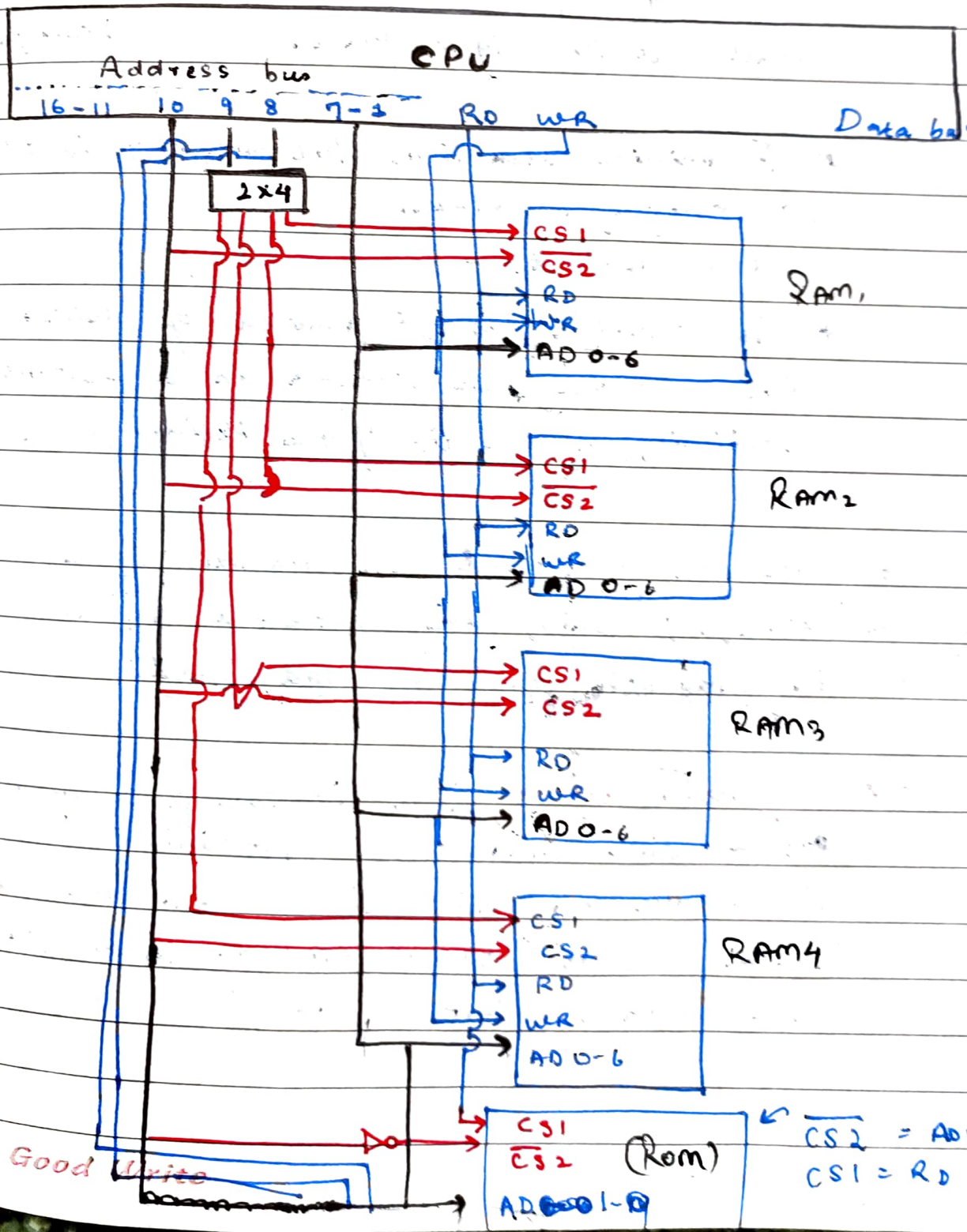
Memory Address Map

Component	Hex. address	Address Bus [AD_{16-1}] 10 9 8 7 6 5 4 3 2 1
128 B RAM 1	0000 – 007F	0 0 0 x x x x x x x
128 RAM 2	0080 – 00FF	0 0 1 x x x x x x x
128 RAM 3	0100 – 017F	0 1 0 x x x x x x x
128 RAM 4	0180 – 01FF	0 1 1 x x x x x x x
512 Rom	0200 – 03FF	1 x x x x x x x x x

DATE: ___/___/_____
PAGE: _____

Note: AD_{10} → 0 RAM
1 Rom

if $AD_{10} = 0$ → AD_{9-8} → 00 RAM1
01 RAM 2
10 RAM 3
11 RAM 4



Auxiliary Memory →

DATE: ___/___/___
PAGE: ___

[These are magnetic devices]

The avg time req to reach the storage loc" in memory and obtain its content : Access time

Seek time ← → Transfer time

Seek time >> Transfer time

time req to put the read-write head at the right place.

time req to transfer data.

* To minimize Seek time the memory is organized in records or blocks.
  Set of characters
* The transfer rate = number of characters/words the device can transfer in a second after positioning at the tip of the beginning of record.

⇒ Bits are recorded as magnetic spots on the surface; which are detected by the change in magnetic feild by read head.

write-head

→ Magnetic Disk → is a circular plate of metal/plastic coated with magnetized material.
There are concentric circles on the disk called 'tracks' where data is stored. These tracks are then broken into 'sectors'

Good Write

DATE: __/__/____
PAGE: _____

They can be of 2 types:

> A single read/write head for each disk surface. The track address bits are used by a mech. assembly to move the head into the specified track position.

> Seperate read/write heads are provided for each track, the address bits can electronically select a particular track via a decoder. (Expensive)

In a sector the track at more distance from center will have more circumference. Thus to balance out the number of bits in 1 track sector the density has to vary. (of bits)

Disks which are permanently attatched to the unit assembly & cant be removed are called 'hard disk' while removable disks are called 'floppy disk'

→ Magnetic Tapes is a strip of plastic coated with magnetic medium. Bits are recorded as magnetic spots on tape. Read/write heads are mounted one in each track.

Adv: It can be stopped, started, move forw./backw.

Dis Adv: Cant be started or stopped fast enough.

> Gaps are left when tape is stopped, at the time of starting again, these gaps provide runner up length to get on the right speed again.

Good Write

DATE: __/__/____
PAGE: _____

Each record on tape has an identification pattern at begⁿ & end.

* The pattern @ begⁿ : tape control identifies the record number.
* The pattern @ end : It realises beginning of a gap"

Associative Memory or Content addressable memory

Concept & Problem: The time req. to find an item stored in memory can be reduced if stored data can be identified for access by the contents of the data itself rather than the address.

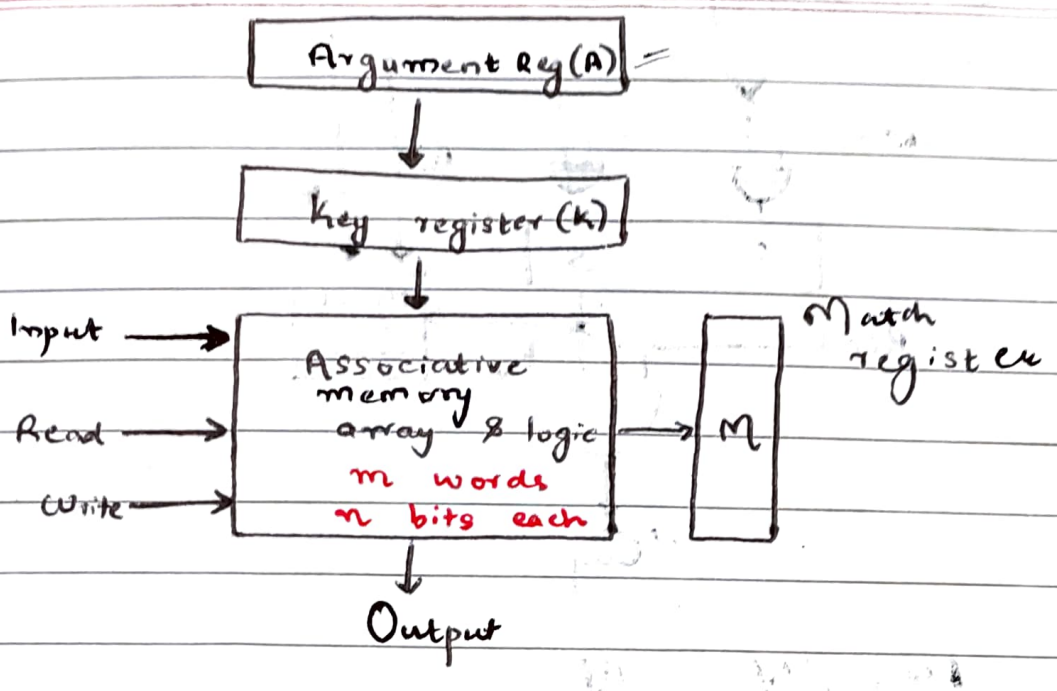
Working: When a word is written in associative memory, no address is given. The memory finds the empty location to store the word.

Write: When a word is to be written in an ass. memory, the content of the word / part of the word is specified. The memory locates all the words matching the specified content & marks them for reading.

Disadvantage: Since every memory location will need logic for Write, it is thus expensive & complex.

DATE: / /
PAGE:

Hardware:



> **Argument register (A)**

Each word in memory is compared in parallel with the content of the argument register.

> **Key register (K)** [Provides a mask for choosing]

The part of the word from associative memory which needs to be compared to the memory is set 1 in K & rest is set to 0.

> **Match register (M)**

The words that match a corresponding bits of argument register sets the match bit to 1.

eg:

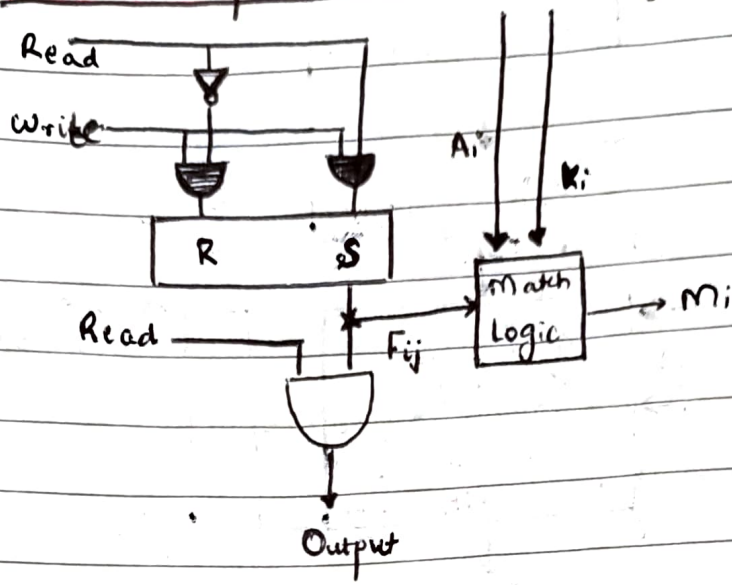
A	101 111100
K	110 100000
req address:	1 0 X 1 X X X X X

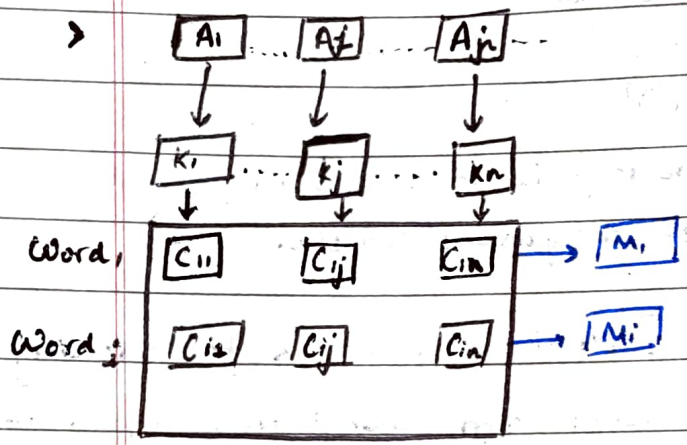
[X : Dont care]

Good Write

DATE: __/__/____
PAGE: ____

One cell of associative memory





Say $C_{ij} = F_{ij}$ & x_j = Match of j^{th} bit

then $x_j = [A_j F_{ij} + A_j' F_{ij}']$

either both 1 or both 0.

$$[M_i = x_1 \cdot x_2 \cdots x_n]$$

Considering Key register

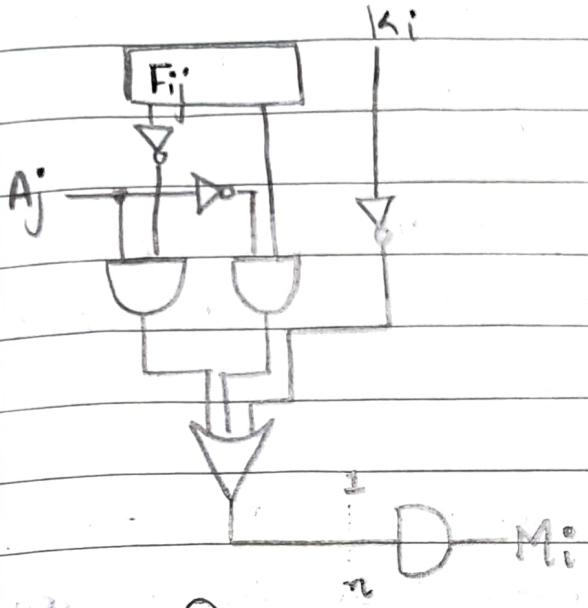
$$(x_j') = x_j + k_j' = \begin{cases} x_j & \text{if } k_j = 1 \\ 1 & \text{if } k_j = 0 \end{cases}$$

$$M_i' = \prod_{j=1}^{n} (x_j + k_j') = \prod_{j=1}^{n} [A_j F_{ij} + A_j' F_{ij}' + k_j']$$

DATE : ___/___/_____
PAGE : _____

~~CACHE MEMORY~~

Match logic



Read Operation :

DATE :
PAGE :

Cache Memory

The reference to memory at a given moment/ intervals of time tend to be combined within a few localized areas in memory. This is "locality of reference"

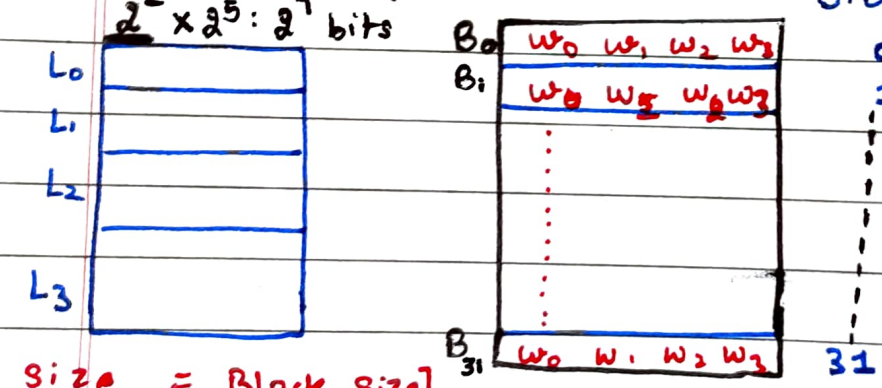
Thus we tend to place those frequently used data/ memory in the fast small memory called cache memory.

Basic working: If CPU needs to access a memory, first the cache is examined. If found in the cache it is read from the fast memory. If not, then the main memory is accessed, this block is then transferred to the cache memory

Hit ratio: The ratio of hits (match)/total computer ref. is called hit-ratio.

Transforming data from main-mem to cache : mapping process

Associative Mapping



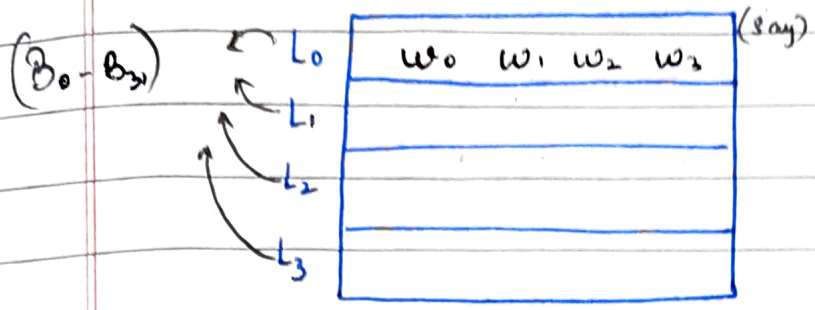
[Line size = Block size]

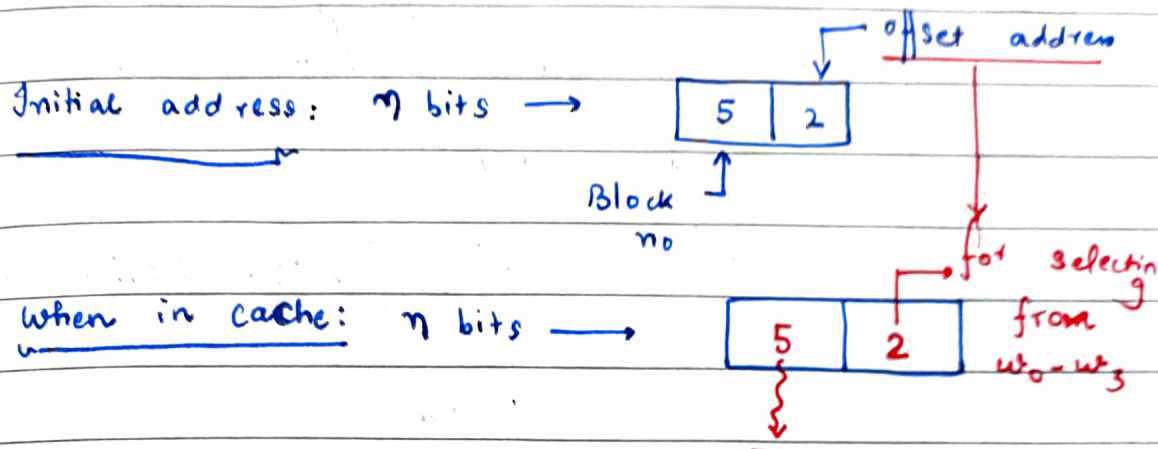
2^5 bit or [2^3 words] ← 2^5 → 2^{10} bit or [128 word] → 128 byte

2^5 block

$2^2 \times 2^5$ byte

Any block can go anywhere in any line, a full block goes in a line.





Disadv: Comparisons increased.

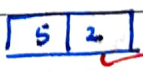
Adv: Hit ratio increased.

Direct Mapping

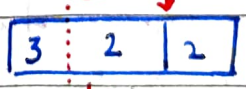
8 options

L_0 can only have B_0 / B_4 / B_8 ....
L_1 " " " B_1 / B_5 / B_9 ....
L_2 " " " B_2 / B_6 / B_{10} ....
L_3 " " " B_3 / B_7 / B_{11} ...

Initial address: n bits :



When in Cache: n bits :



Good Write Say $[B_0, B_4 \cdots B_{28}]$

DATE: __/__/____
PAGE: ____

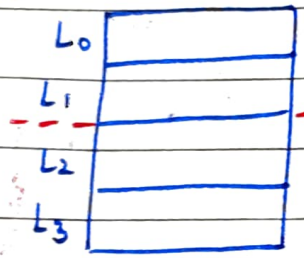
Disadvantages of Direct Mapping:

2 words with same line no but with different tag value cant reside in cache memory at the same time. then we will have to change the block as the same line has to accessed.

Set - Associative Mapping

k- way set associative

⟹ 2- way set associative



$S_0 \not\Rightarrow$ Associative inside the set

$$\frac{\text{Number of lines}}{k} = \text{No of sets}$$

you can put block (0, 1)(2, 3)

in either L_0 or L_1 (S_0)

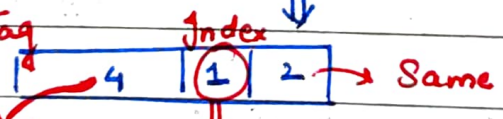
in either L_2, L_3 (S_1)

Before: 7 bits:

Block no

Block offset

After:



to choose b/w^n To choose b/w^n S_0 & S_1

[$B_0, B_1, B_4, B_5, B_8, B_9 \ldots$]

Good Write

DATE: / /

Replacement Algo → (Only for Associatives)

When cache is full then if there is a fail. then we do replacement to replace the tag-data.

These Algos are: Random replacement @ rando

FIFO: first in first out

LRU: Replaces the item least recently used by CPU.

Writing to the cache:

(WRITE THROUGH) **Method 1**: The cache memory when ever changes → parallely make changes in the main memory also.

Adv: The main memory is always updated in sync with the cache memory. & main-memory data is always valid.

(WRITE BACK) **Method 2**: The cache locⁿ is updated only during write oprⁿ. The locⁿ is the marked by a flag so that when word is removed from cache it is copied to main memory.

As long as the word is in a cache, memory doesnt need to be accessed. & it may be changed several times

Good Write

DATE : __/__/____
PAGE : _____

Cache Initialisation

The cache is initialised when power is applied or when main memory is loaded with a complete set of programs from auxiliary memory. It is initialised with some invalid data.

A valid bit indicates if the data is valid or not. When cache is initialised all valid bits are cleared which is set 1 when when first word is loaded to it.

If valid bit = 0 ⟸ replace directly as data is invalid.

Good Write

DATE: ___/___/_____
PAGE: _____

Unit II

Peripheral Devices →

Devices that are under the direct control of the computers are said to be online. They are designed to read info in or out of the memory unit. Input - output devices attached to the computer are called peripherals.

ASCII I/o devices communicate with user & computer using ASCII codes for normal characters.

total 128 → 94 (printable)
- 26 UC
- 26 LC
- 10 0-9
- 32 [%, * etc]

→ 34 (non-printables)

The control chars. are used for routing data and arranging the printed text into a prescribed format.

There are 3 of control chars. types :>

1. **Format effector** → : Control layout of printing
   eg. Backspace, carriage return etc.
2. **Information Seperators**: Seperate data into paras & pages
   eg: Record Seperator (RS), file Seperator (FS)

Good Write

3. Communication Control Characters:

transmission of text b/w^n remote terminals.
eg: STX (Start of text) & ETX (end of text)

→ ASCII is a 7 bit code. thus the 8th bit is free. We use it for other features
eg:
Some printers takes MSB = 0 normal
MSB = 1 to specify greek signal

In communication, 8th bit is used for parity of binary-coded character.

Input Output Interface.

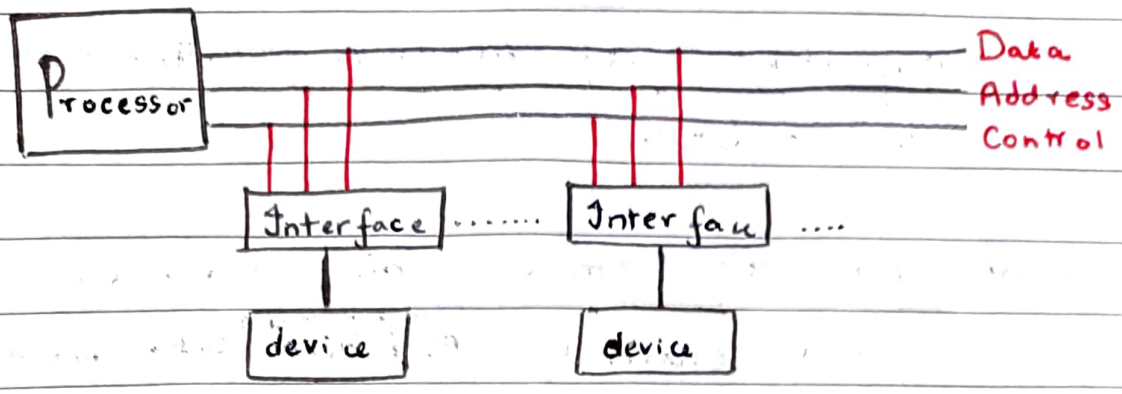
The CPU can't communicate directly with the I/O due to the major differences:

1. Peripherals are EMech. or Em devices and their manner of operation is different from the op^n of the CPU.
2. The rate is v. slow in peripherals than CPU.
3. Data codes in peripheral differ from CPU.
4. Operating modes of peripherals are different & must be controlled so not to disturb others.

Good Write

DATE: __/__/____
PAGE: _____

> Connecting via IO bus.



> Each interface decodes the address & control recieved from the bus, interprets them for the peripheral & provides them signals for controller.

> Each peripheral has its own controller

ms I/O bus is connected to all the I/O interfaces. To communicate with a particular I/O device the device address is placed on the address line.

ms When interface detects its own address on the line it activates the path b/w^n bus lines & the devices.

> I/O Commands : These are commands of an instruction.

a) Control Command : Issued to activate the peripheral and to inform it what to do.

b) Status Command : Used to test various status conditio in the interface and peripheral. The errors may be detected. -s. & bits in status register is set 0.

DATE : __/__/____
PAGE : _____

c) **Output data command**: This command makes the system respond by transferring data from bus into one of its register.

d) **Input data command**: This command makes the system respond by ~~receiving~~ transferring data from register to the data bus.

IO Bus vs Memory Bus

3 ways:

1. "Use 2 seperate bus"

The computer has independent set of data, address, control buses. This is done in computers with separate IOP with the CPU.

> Memory communicates with both CPU & IOP using a memory bus

> The IOP communicates with I/O devices with seperate i/o bus with its own address, data & control lines.

⟹ This provides an independent pathway for info. transfer.

P. T. O.

Good Write

DATE : ___/___/_____
PAGE : _____

2. "Isolated I/O"

CPU has distinct I/O instructions.

⇒ When CPU fetches the operation code of an I/O instruction, it puts the address associated with the instⁿ in the common address bus

⇒ At same time I/O read or I/O write control lines are enabled. [Informing that its an I/O operⁿ]

& Vice-Versa for memory instruction.

Adv: It isolates the memory & I/O addresses thus one not affecting the other.

3. "Memory-mapped"

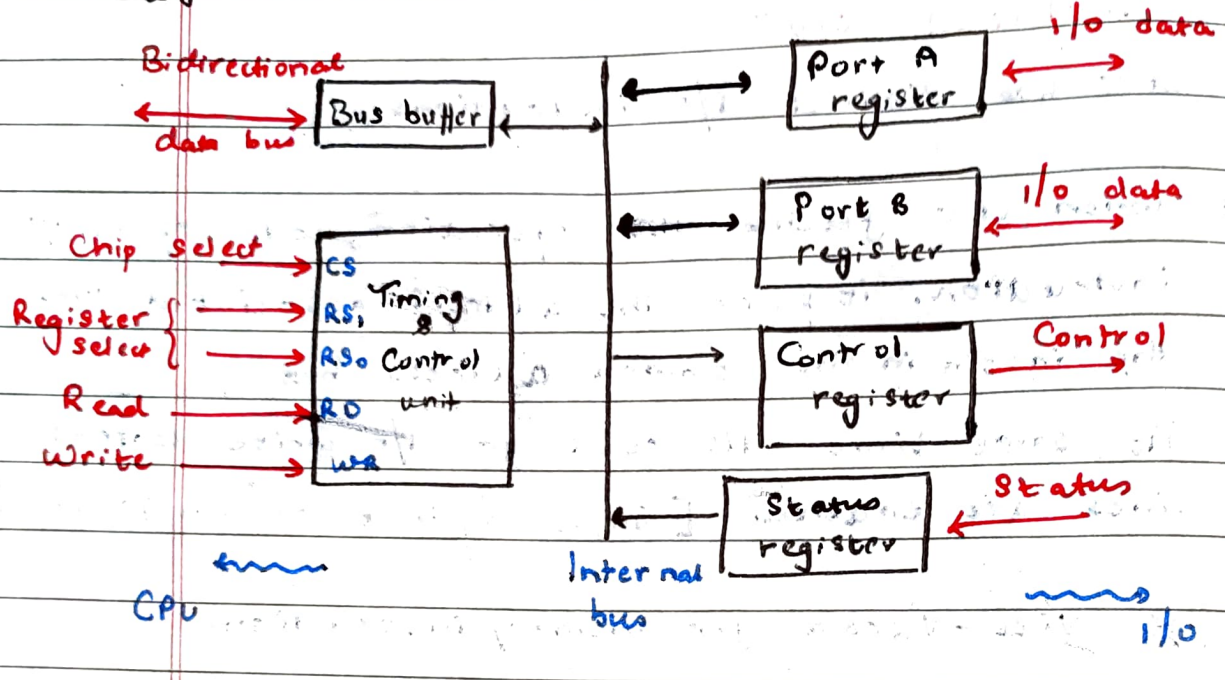
Use the same address space for both the memory and the I/O.

CPU dont have distinct I/O instⁿ. Thus CPU can manipulate both I/O data & memory words with same instructions.

Adv: To load / store an instruction the codes are same for either I/O reg. or memory words

DATE : ___/___/_____
PAGE : _____

eg of I/O interface



Good Write

Asynchronous Data Transfer

DATE: ___/___/____
PAGE: _____

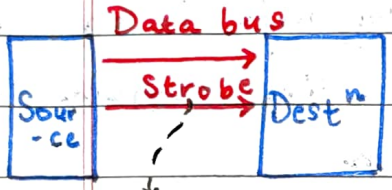
Synchronous: if registers in the interface share a common clock with the CPU registers.

A-synchronous: if internal timing in each unit is independent from the other in that each uses its own pvt. clock for internal registers.

CONTROLS

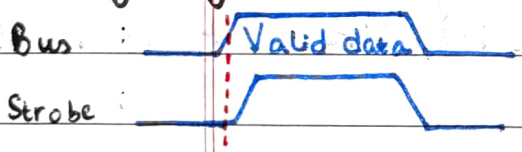
- Strobe Control
- Handshaking

i) originated by source

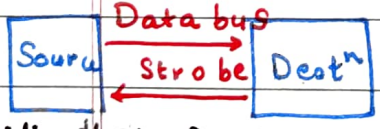


to tell that data in the data bus is valid.

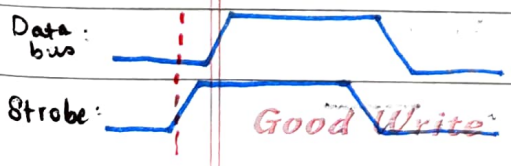
Timing Dig:



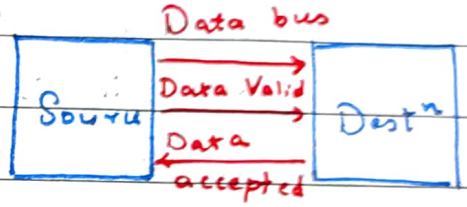
ii) originated by destn



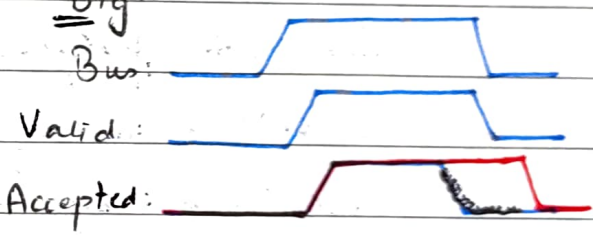
to tell that Destn is ready to accept data.



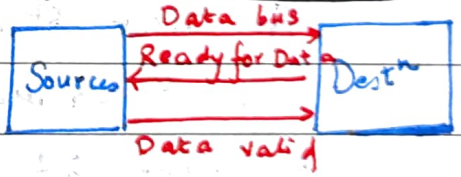
i) originated by source

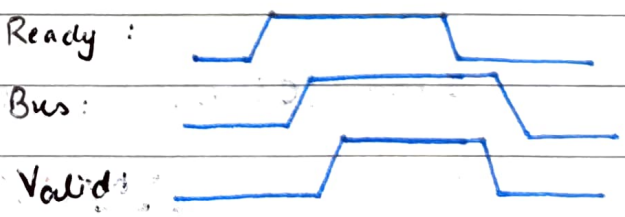


Dig:



ii) originated by destn





Time Out:

In Handshaking if one device is faulty, the data transfer wont be completed. This error is detected by the means of time out as it produces an alarm if the data transfer isnt complete.

It is implemented by a means of an internal clock that starts counting when one of the handshaking controls is enabled, if the return signal doesnt occur in a given time it assumes an error.

→ Async - Serial - Transfer

In async transmission, binary info is only transmitted when it is available & the line remains idle when there is no transmission.

for this purpose we use a data transmission technique used in many ~~interfaced~~ interactive terminals by employing special bits at both ends of character code.

The first start bit = 0
The 8 character bits
& then 1 1 as stop bits.

eg: Send (C5)H. C5 : 11000101

Good Write

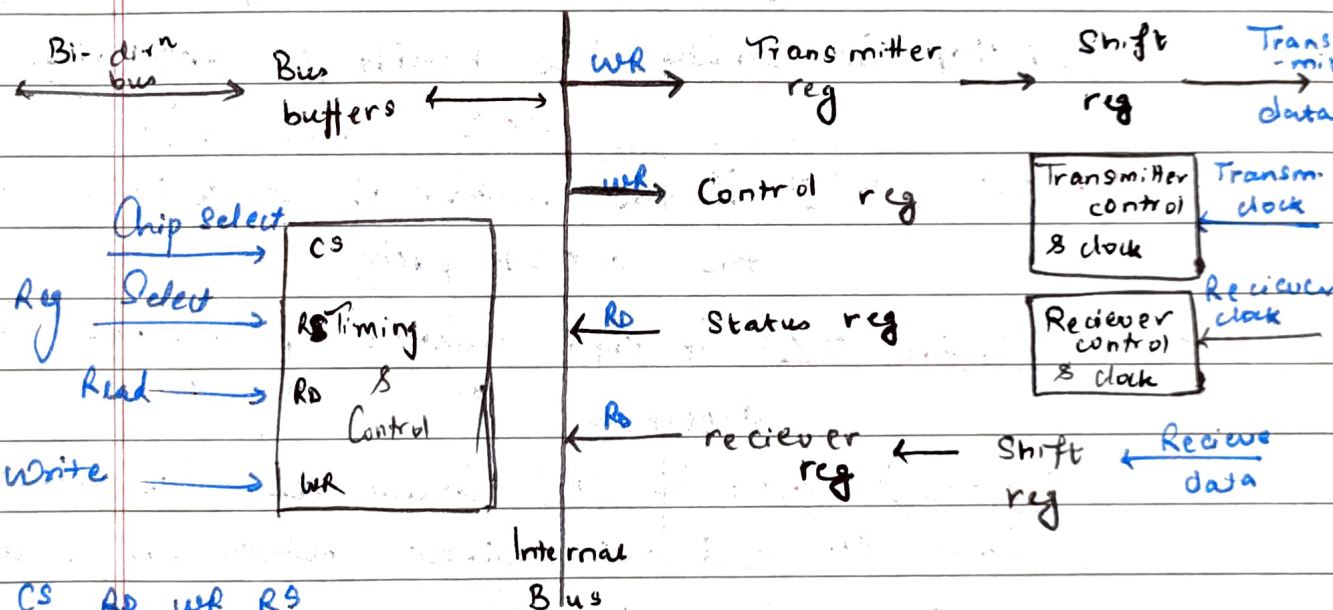
DATE : ___/___/____
PAGE : _____

Baud Rate: is defined as the rate at which serial info is transmitted and is equivalent to the data transfer in bits per second.

eg: 10 characters / sec in a 11-bit format.

→ 11 * 10 bits / sec or 110 baud.

- The receiver receives a 11-bit message from the line & forwards the 8-bit chars to the computer.
- Integrated ckts are available which are specifically designed to provide an interface b/w computers & similar interactive terminals called UART.



CS	AD	WR	RS	
0	x	x	x	—
1	1	x	0	Reciever reg
1	1	x	1	Status reg
1	0	1	0	Transmitter reg
1	0	1	1	Control reg

Good Write

Transmitter :

The CPU reads the status register & checks the flag to see if the transmitter register ~~and the~~ is empty. If yes the CPU transfers a character to the transmitter ~~shift~~ reg. & interface clears the flag to mark the register full. Then data is transferred (in parallel) to the shift register & stop bits are appendende into the shift register.

Reciever :

The recieve data input is in 1-state when line is idle. The reciever control monitors for the line for a 0 signal. Once the start bit is detected, the incoming bits are taken as character bits then they are shifted to "shift register". After recieving it checks the parity & stop bits.

errors may occur

- **Parity error:** if the number of 1s recieved is not the correct parity.
- **Framing error:** if right number of stop bits isn't detected
- **Overrun error:** if CPU doesn't read the character from the reciever register before the next one is available in shift register.

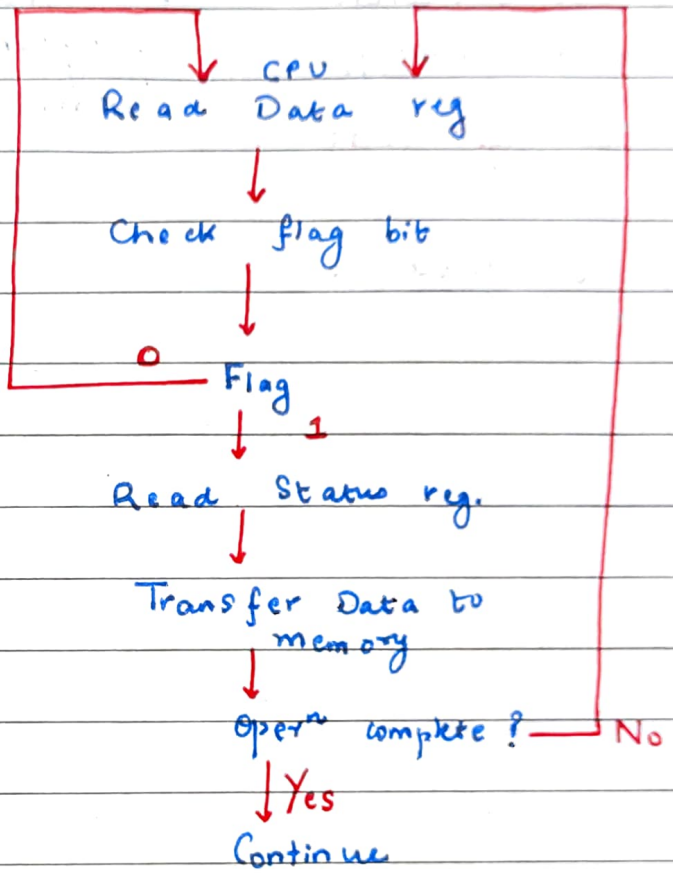
(Read fifo buffer)

Modes of Transfer

1. "Programmed I/O"

Each data transfer is initiated by an instruction in the program.

This requires constant monitoring of the peripheral by CPU; once done there is a need to check if next transfer can be made.



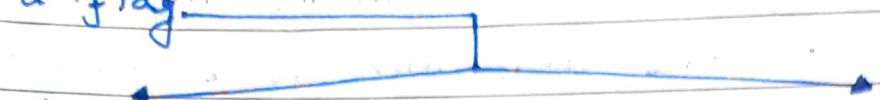
here CPU stays busy if there is no ~~interrupt~~/ input/output thus using the CPU unaffectively.

Good Write

DATE: ___/___/___
PAGE: ___

2. "Interrupt Intiated I/O"

Whenever any I/O actions occur an interrupt is sent to the CPU to tell it that the I/O action exsists by setting a flag.



Vectored

Fixed branch location (already in the memory)

interrupt vector

Non-Vectored

Not fixed & programmable branch loc^n

Software Considerations

Good Write

Priority Interrupted I/O

DATE: ___/___/____
PAGE: _____

Priority interrupt is a system that establishes a priority over the various sources to determine which condition is to be serviced first when 2 or more requests arrive simultaneously.

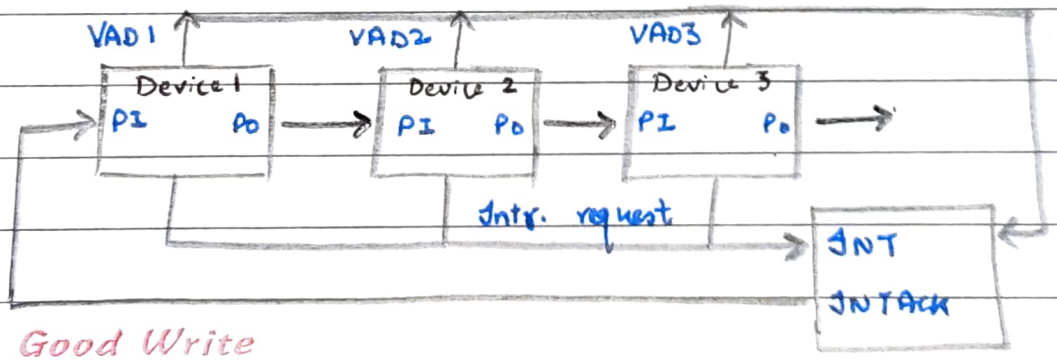
→ **Polling**

We provide one common branch address for all interrupts. The program that takes care of interr. begins at the branch address & polls them interrupt sources in sequence. They are tested from priority (high to Low), if the control signal of the current interrupt is on it gets the branching, otherwise the next is tested.

Disadvantage:

If there are many interrupts, the time req to poll them can exceed the available time to service I/O devices.

→ **Daisy-Chaining Priority**



Good Write

DATE: __/__/____
PAGE: _____

> first when an interrupt is encured, then an Interrupt request is sent to the INT.

> if the CPU accepts it, this is acknowledged by sending back INT Ack which goes to PI.

Device working

> if any device has PI = 1 & Interr. request = 1 then VAD is sent to the CPU.
↑ Vector address

∴ PO = $\overline{INT}$. PI

& it sets PO = 0

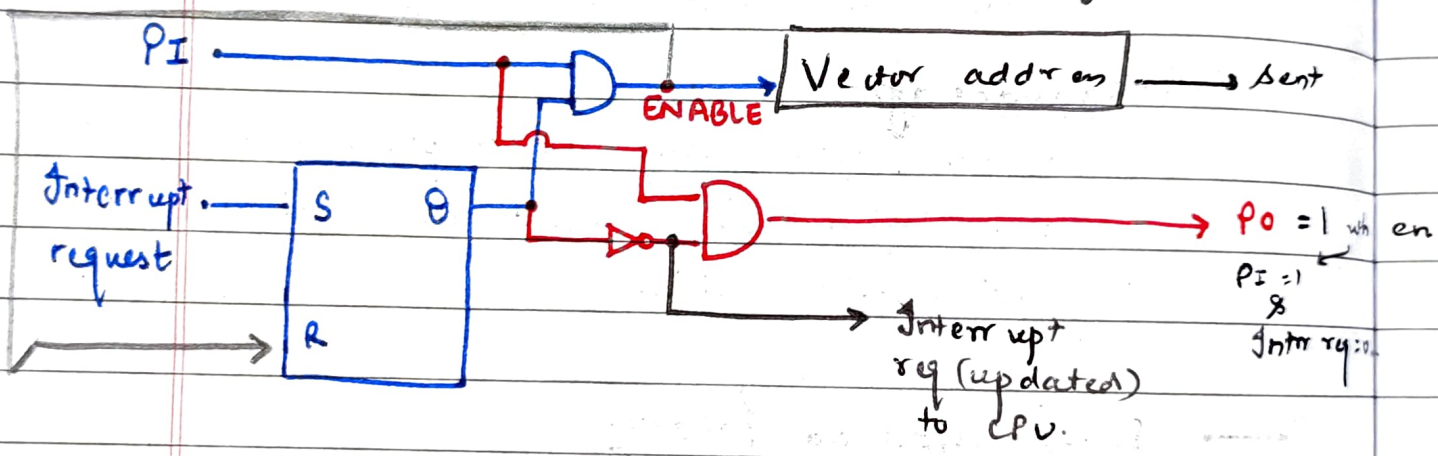
Enable = INT . PI

> if PI = 0 , PO = 0

> if PI = 1 , Interr. request = 0 then PO = 1.

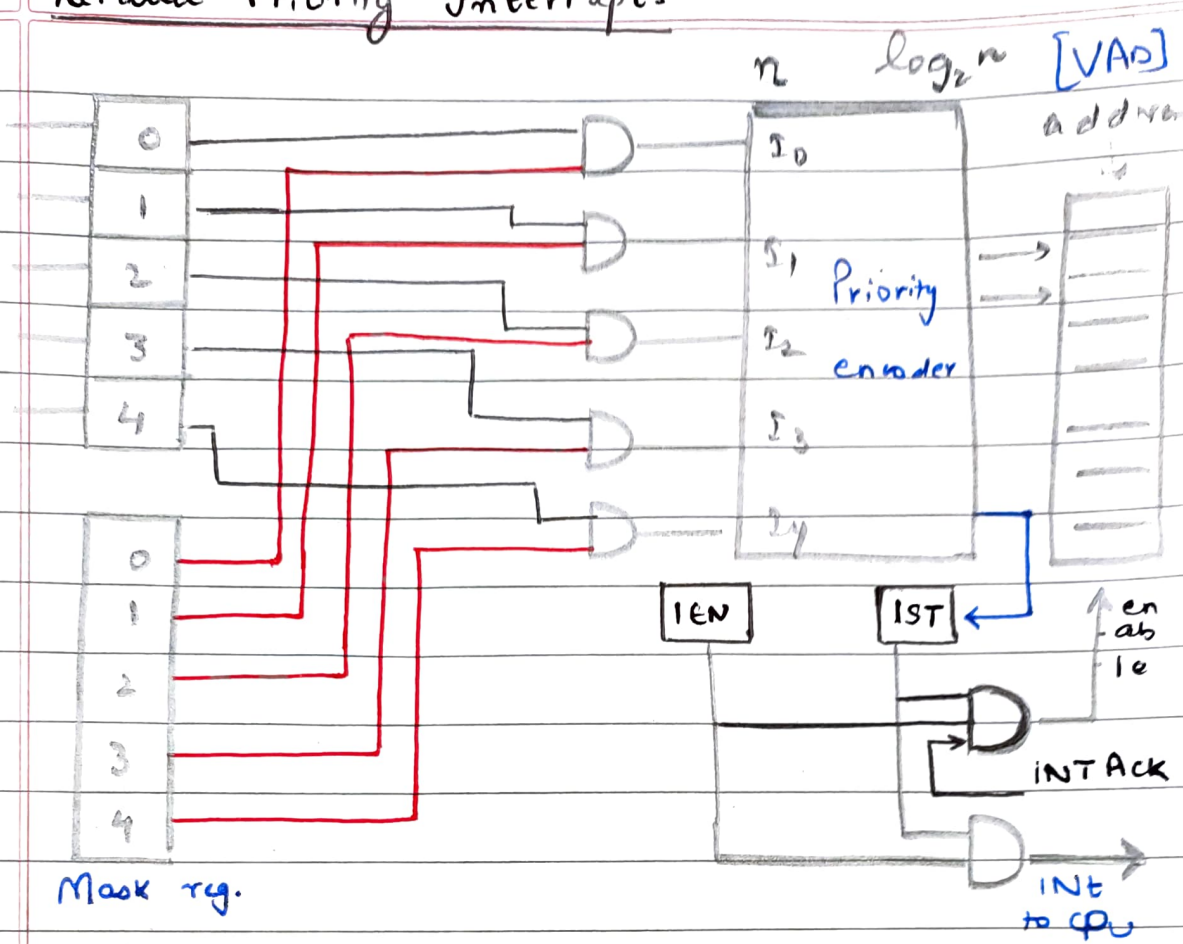
→ finally set Interr req = 0.

> One stage of daisy chain arrangement



Good Write

Parallel Priority Interrupts



- here priority of 0 > 1 > 2 > 3 > 4.
- Maskreg. is used to mask any specific interrupt or will.
- $I_i = [Device_i \cdot Mask_i]$
- This priority encoder thus finally gets the right interrupt

* IST: if any interrupt has occured it will turn 1.

Steps →

* IST turns 1

* if IEN=0 {Nothing works} ; if IEN=1 then Interrupt sent to CPU, which sends back INT Ack.

* Thus finally enabling the VAD.