Input - Output Interface

→ The Input - Output interface allows transferring information between external input / output devices (i.e., peripherals) and processors.

Interface

Why Input - Output Interface

→ We require the input - output interface b/w many differences exist b/w each peripheral and the central computer while transfering data.

→ Some significant differences b/w the peripheral and CPU are:-

(1) The nature of the CPU is electronic, and that of the peripheral devices is electromechnical and electromagnetic. So, There are many differences in the mode of operation of both CPU and peripheral devices.

(2) We have a synchronization mechanism because the data transfer rate is slower in peripheral devices than CPU.

(3) In peripheral devices, data code and formats differ from the format in the CPU.

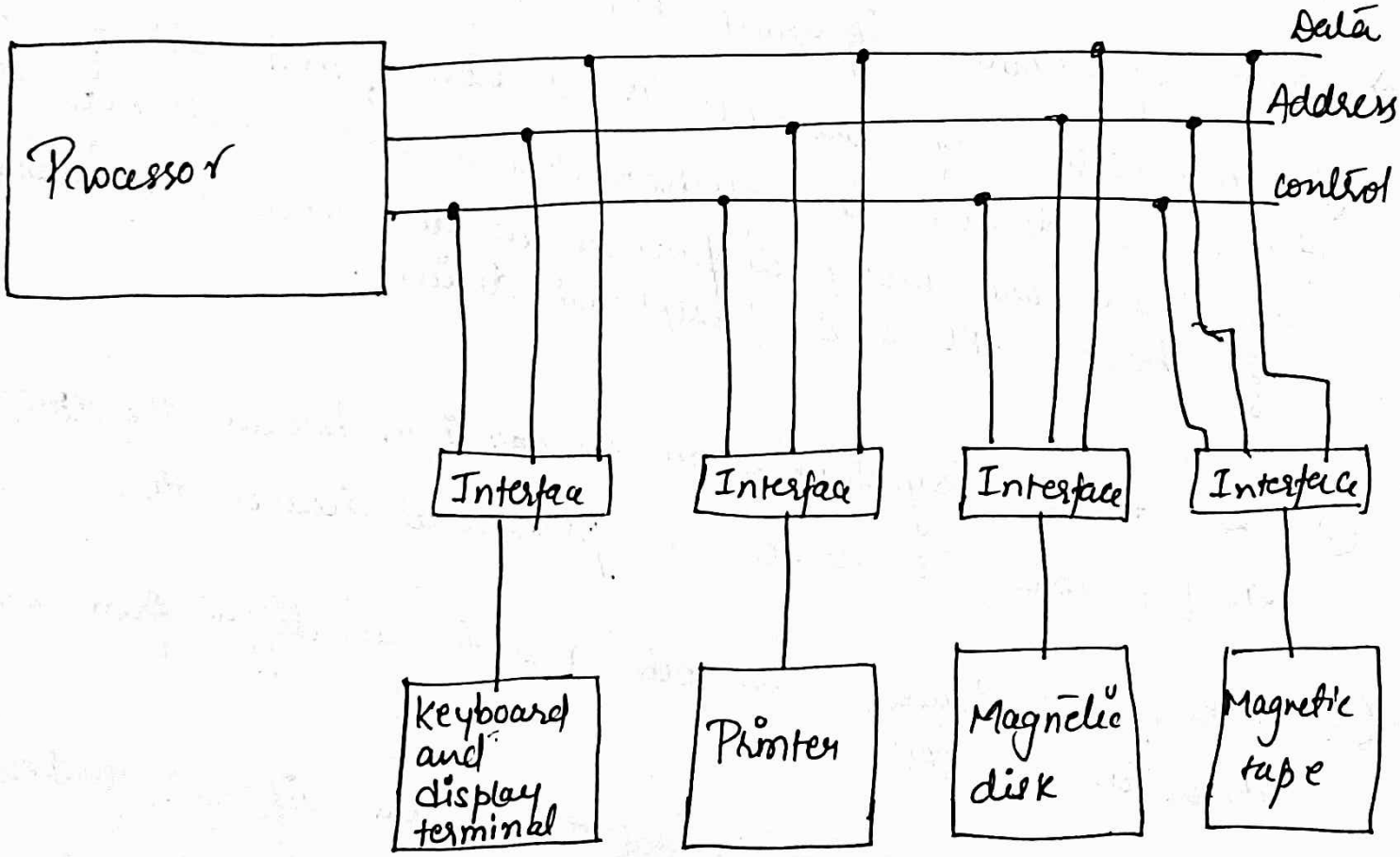
(4) The operating modes of peripheral devices are different, and each can be controlled donot to disturb the operations of any other peripheral devices connected to the CPU.

Note :-

→ To resolve these differences, computer systems include special hardware components between the CPU and peripherals to supervise and synchronize all the input and output transfers. These components are called interface units because they interface b/w the processor bus and the peripheral devices

→ In addition, each device may have its own controller that supervise the operations of the particular mechanism in the peripheral.

I/O BUS AND INTERFACE MODULES



Connection of I/O bus to input output devices

→ A typical communication line blw the processor and several peripherals is shown in figure.

→ The I/O bus consists of data lines, address lines and control lines.

→ The magnetic disk, printer, and terminal are employed in practically any general-purpose computer.

→ Each peripheral device has associated with it an interface unit.

→ Each interface unit decodes the address and control received from the I/O bus, interprets them for the peripheral, and provide signals for the peripheral controller. It also synchronizes the data flow and supervises the transfer between the peripheral and processor.

→ Each Peripheral has it own the controller that operates the particular electromechanical devices.

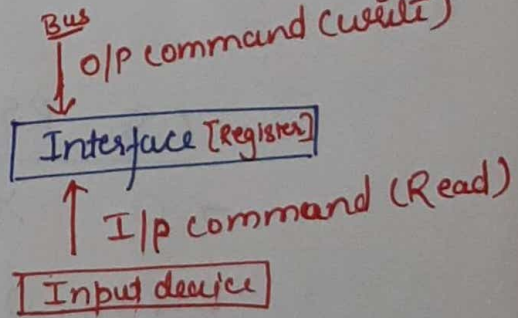
I/O COMMANDS

I/O ~~comman~~ commands are the instructions executed ~~on~~ in the interface & its attached to peripheral devices.

① Control Command: used to activate the peripherals & to tell it what to do.

② Status Command: used to test various status conditions in the interface & the peripherals. like whether the peripherals are busy, or free or error.

③ Output data (I/O (cmd write)): It causes the interface to respond by transfering data from bus into one of its registers. or cause the I/O module to take an item of data from the data bus & subsequently transmit that data item to the peripheral.

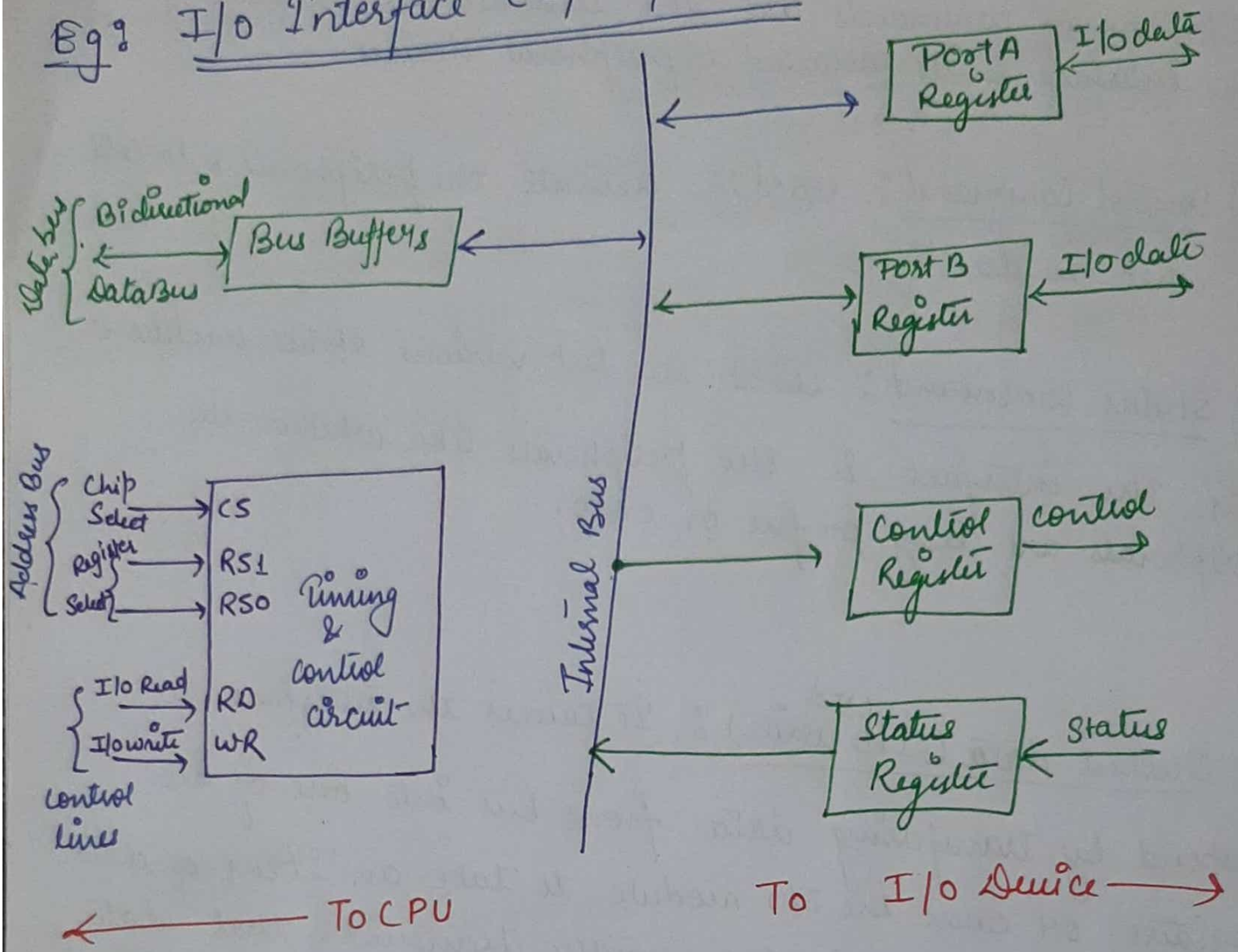


④ Input data (I/O read):

Interface receives an item of data ~~base~~ from the peripheral & places it in its buffer register. or causes the I/O module to obtain an item of data from the peripheral & place it in an internal buffer.

(internal buffer register / data register)

Eg: I/O Interface (I/O Module)



CS	RS1	RS0	Register selected
0	X	X	None: data bus in high-impedance
1	0	0	Port A register
1	0	1	Port B register
1	1	0	control register
1	1	1	Status register

→ I/O device transfer data or communicate with CPU with the help of databus. And it consist of it's own Bus Buffers.

→ Here, CPU has another unit called Timing & control unit

Timing and control circuit consist of

① Chip select (CS):

Address Bus :- Selects the interface unit through the chip Select (CS) [enable] and the two registers (RS_1, RS_0) select input. [among four register from I/O device port]

→ I/O Data :- Transfer of data to the I/O device or from the I/O device is happens either with the help of port A & port B

→ Interface unit may be work as a output device or Input device depends of on the situation

Ex: I/O device means like magnetic tape. which works as a I/P as well as O/P
In this case Both the port registers are use one for I/P & another for O/P.

If it uses only for I/P means it these ports are used as I/P data
If it uses only for O/P means these ports are used as O/P data

Control Register :- It receives control information from the CPU By loading the appropriate bits 1 or 0 to the control register.

Status Register: It stores status conditions like error related I/O devices

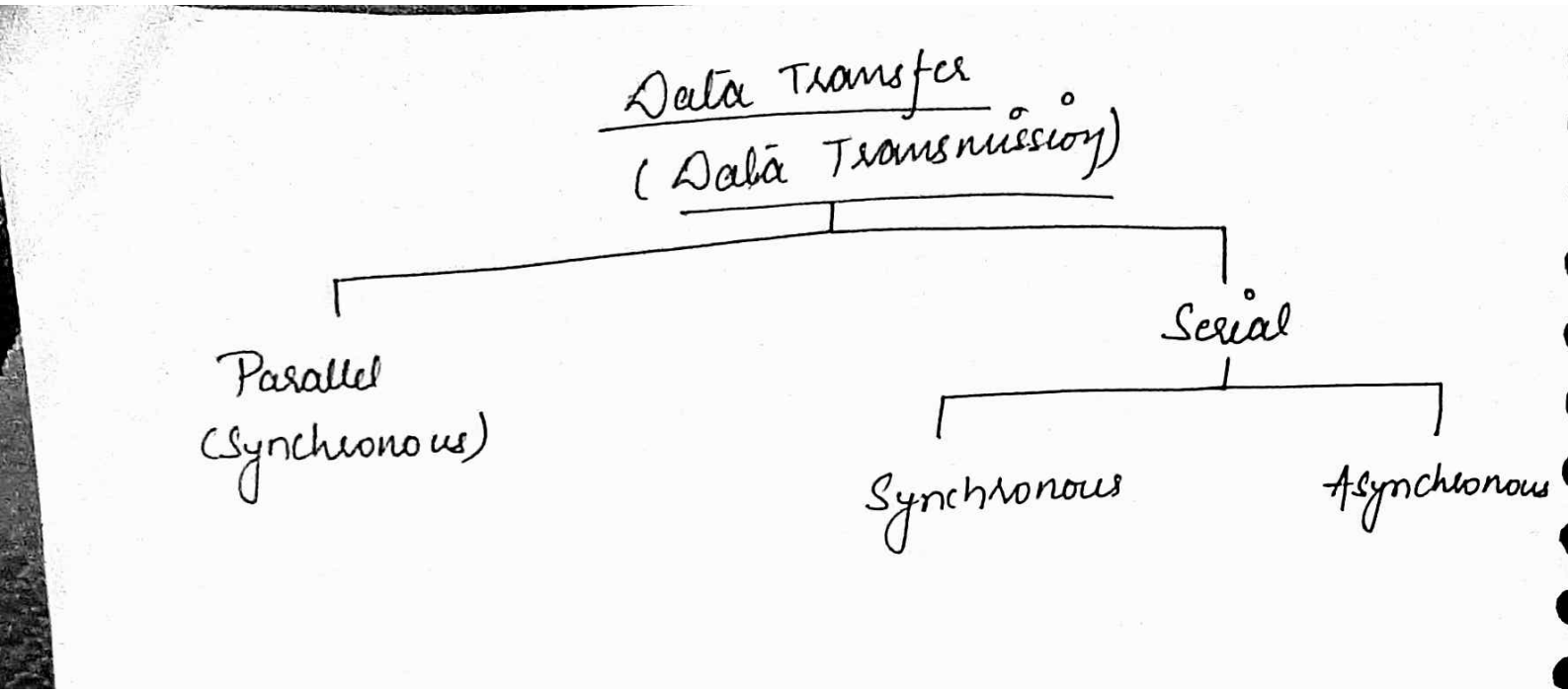
I/O Mapping
(I/O Addressing / I/O Interfacing Technique)

→ When computers use one common-bus to transfer information between Memory or I/O & CPU. Then

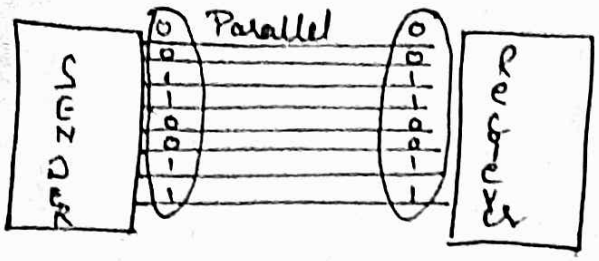
→ There are 2 ways that computer programs communicate with I/o device. or 2 techniques for addressing the I/O device by the CPU.

1) Isolated I/o (I/o - mapped I/o or Port mapped I/O)

2) Memory Mapped I/o


Data Transfer
(Data Transmission)
Parallel
(Synchronous)
Serial
Synchronous
Asynchronous

Parallel Transmission



→ Each bit has its own path & Total message is transmitted at the same time.

→ n-bit must be transmitted at the same time.

→ n-bit must be transmitted through n-separate wires.

→ Faster but requires many wires

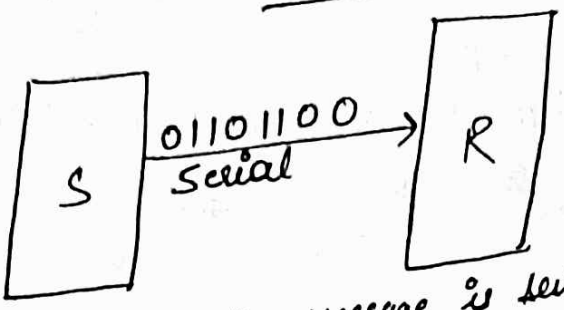
→ Used for short distance where speed is important

Eg: CPU & memory (Bus)

CPU & Printer

→ Parallel Transmissions are always synchronous because Sender and receiver clock is Synchronized

Serial Transmission



→ Each bit in the message is sent in sequence one at a time.

→ n-bit is transmitted through one wire.

→ Slower but less expensive since requires only pair of wire.

→ Used for long distance

Eg: CPU & I/O device

computer to computer.

→ Serial Transmission has 2 types

① Synchronous

② Asynchronous

Serial Transmission

Synchronous

→ Two units share a common clock

→ Data Transfer blw Sender & Reciever is synchronized with same clock pulse

→ Used blw devices that matches in speed.

→ Bits are transmitted continously to keep the clock frequency synchronise in both units.

→ Synchronous means "at the same time" (due to common clock)

→ Fast

→ costly

Asynchronous

→ Two units are independent & each have it own private clock.

→ Data transfer blw sender & Reciever is not synchronized with same clock pulse.

→ Used blw devices that do not matches in speed.

→ Bits are sent only when it is available & line remains idle when there is no information to be transmitted.

→ Asynchronous means "at the regular interval" (due to no common clock)

→ Slow

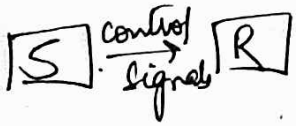
→ Economical

Asynchronous Data transfer

→ It is used when speed of I/O devices don't match with processor & Timing characteristics of I/O devices is not predictable.

Defination

Asynchronous data transfer between two independent units. requires Control Signals to be transmitted between communicating units to indicate the time at which data is being transmitted

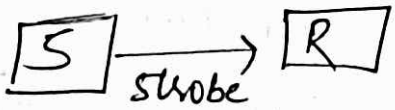


Asynchronous Data transfer has 2 types of methods

① Strobe control

→ a strobe pulse is supplied by one of the units to indicate to the other unit when the transfer has to occur.

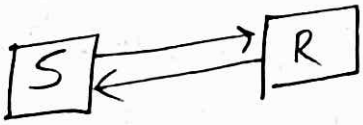
→ After receiving the data at receiver end there is no acknowledgement is given back to sender that data is recieved to resolve this another method is introduced which is called Handshaking method.



② Handshaking method :

→ A control signal is accompanied with each ~~other~~ data item being transmitted to indicate the presence of data.

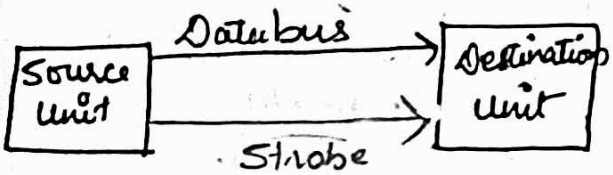
→ The Recieving unit responds with another control signals to acknowledge the reciept of data.



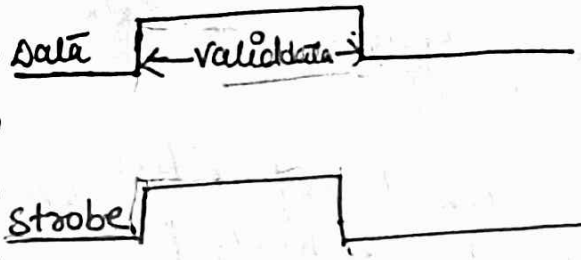
① "Strobe Control" Asynchronous Data Transfer

→ it employs a single control line to time each transfer.
→ the strobe may be activated by either source or Destination unit

Source-initiated strobe for data transfer



ⓐ Block diagram

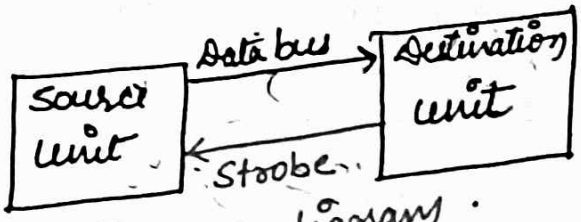


ⓑ Timing Diagram

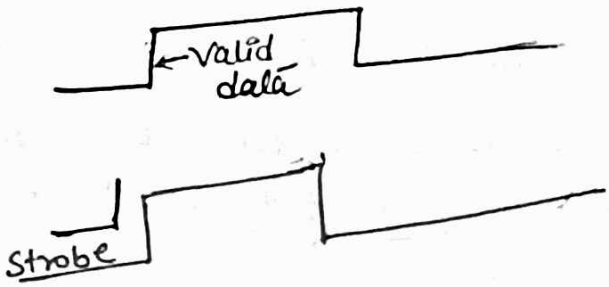
Strobe is a single line that informs a destination unit when a valid data is available in the bus.

Here data is store first into data bus
then strobe pulse is send.
this will enable till the data is not send completely.

Destination-initiated strobe for data transfer



ⓐ Block diagram.

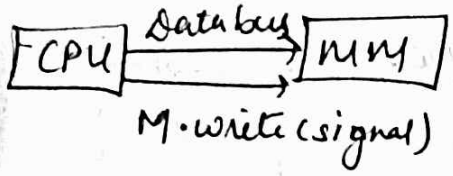


ⓑ Timing diagram

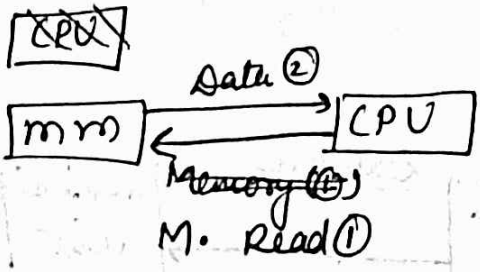
Destination sends a strobe pulse to source for informing that please provide for me

In that case source will store data in into data bus & send to the destination.
after that strobe pulse is disable.

Eg:-
memory write control signal from CPU to memory unit.



Eg:- memory read control signal from CPU to main memory



Disadvantage of Strobe control

→ Source unit that initiates the transfers has no way of knowing whether the destination unit has actually received the data item that was placed in the bus.

→ Similarly, Destination unit that initiates the transfers has no way of knowing whether the source unit has actually placed the data on the bus. or not

i.e., No Acknowlegement.

Note

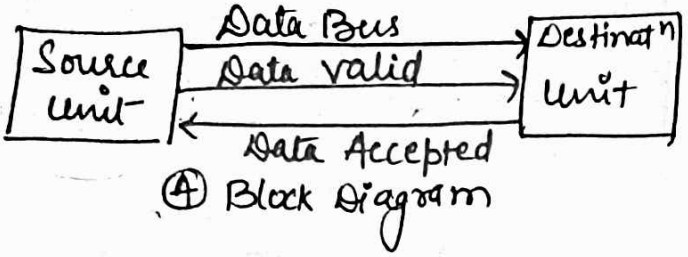
② "Handshaking Method"

for asynchronous data transfer

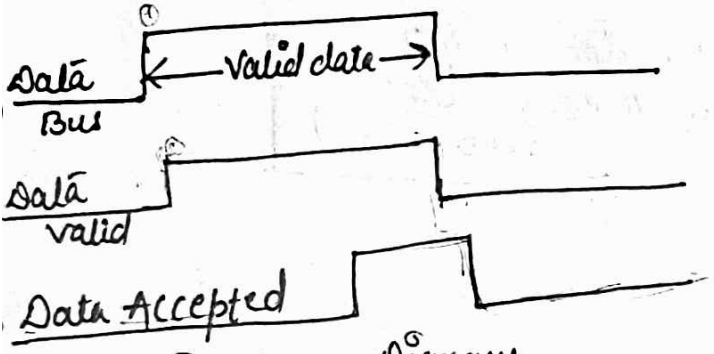
→ Solves the problem strobes method by introducing a second control signal that provides a reply to the unit that initiate the transfer.

→ Strobe control + Acknowlegement signal [Two-wired control]

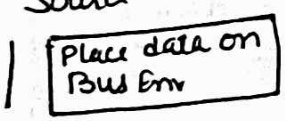
Source-initiated Handshaking



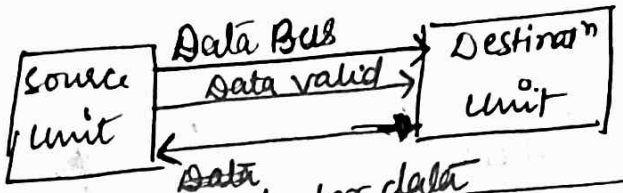
(a) Block Diagram

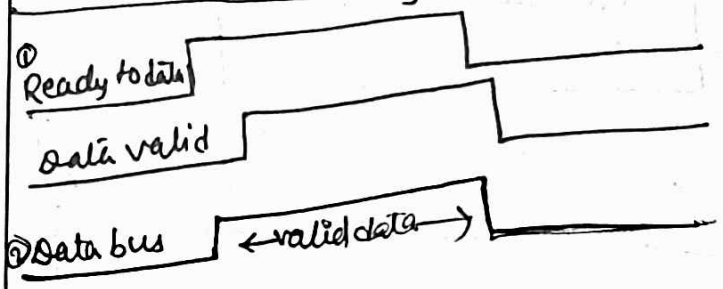


(b) Timing Diagram



Destination-initiated Handshaking





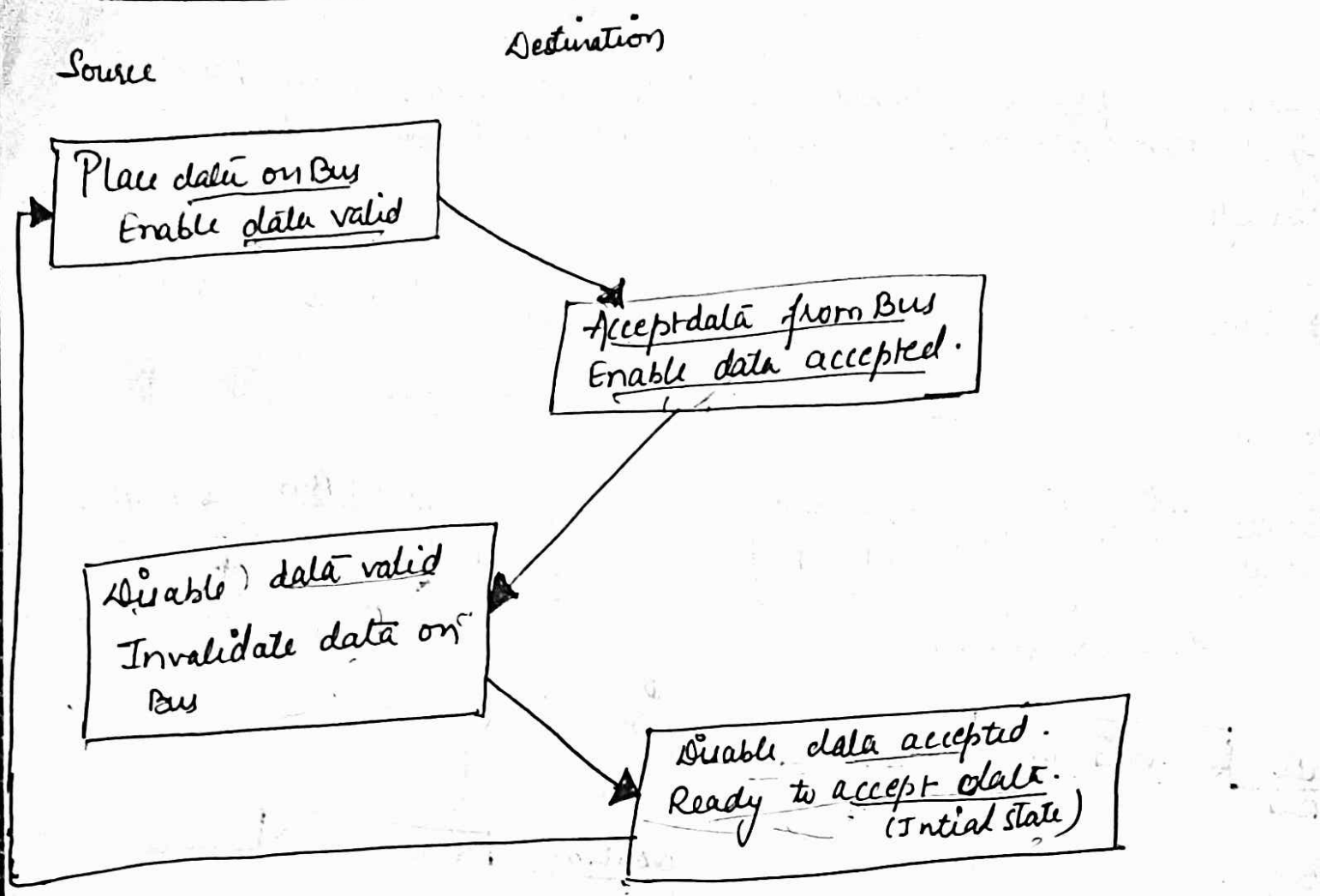
(b) Timing Diagram

Advantages

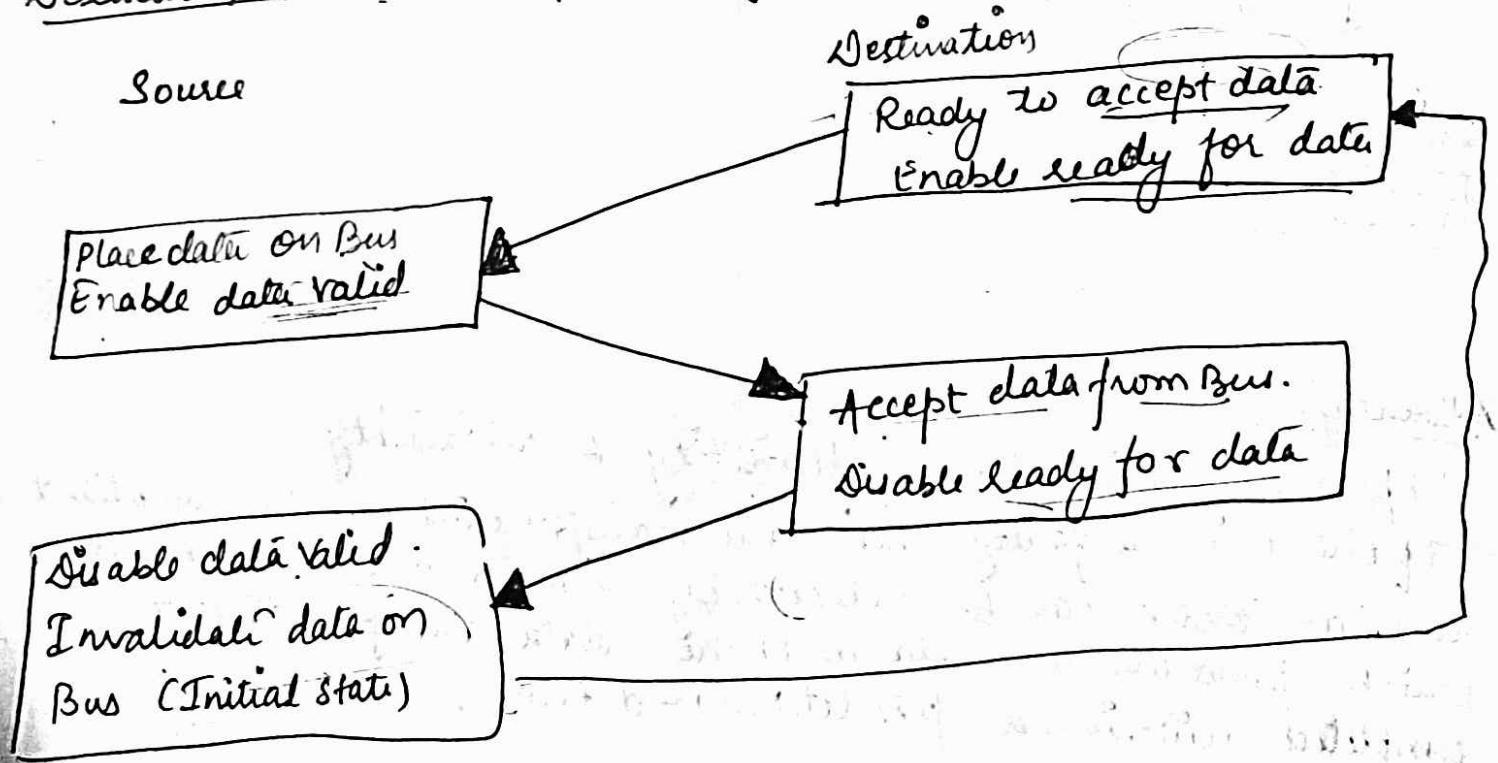
→ It provide a high-degree of flexibility & reliability.

→ If one unit is faulty, the data transfer will not be completed Such an error can be detected by time out ~~net~~ mechanism which produces an alarm if the data tranfer is not completed within a predetermined time.

Source-initiated Handshaking



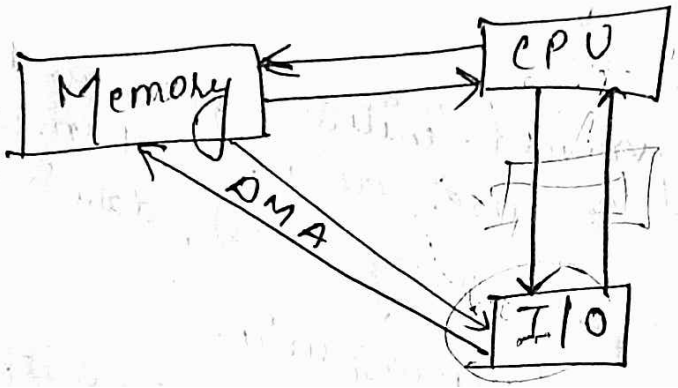
Destination-initiated Handshaking



Modes of transfer

- Binary information received from an external device is usually stored in memory for later processing.
- Information transferred from the central processing computer into an external device originates in the memory unit.

- The CPU merely executes the I/O instructions and may accept the data temporarily, but the ultimate source or destination is the memory unit.



→ Some modes use the CPU as an intermediate path; others transfer the data directly to and from the memory unit.

→ Data transfer to and from peripherals may be handled in one of three possible modes:

(1) Programmed I/O

(2) Interrupt - initiated I/O

(3) Direct memory access ~~(acc.~~ (DMA)

Note: In programmed - initiated I/O and Interrupt-initiated I/O, CPU is use as an intermediate path while in DMAC Direct memory access), data is exchange directly from the memory unit.
In DMA their is a channel blw Input - output devices and Memory through which data is transferred.

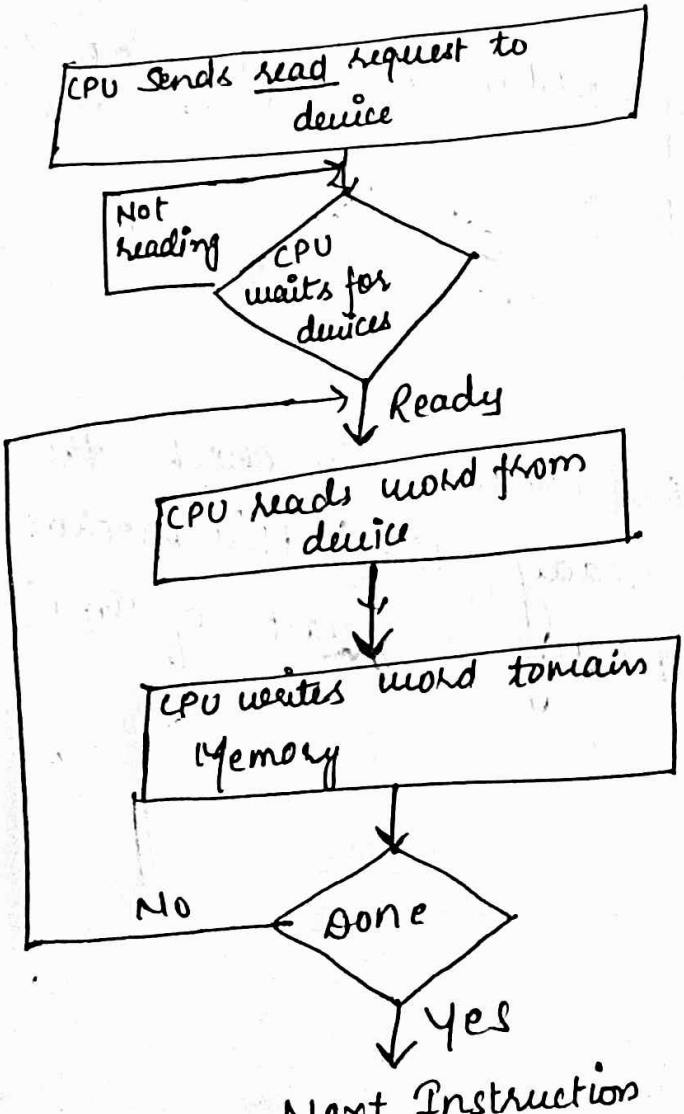
Programmed I/O

→ Programmed I/O operations are the result of I/O instructions written in the computer program.

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

→ Usually, the transfer is to and from a CPU register and peripheral

→ Transferring data under program control requires constant monitoring of the peripheral by the CPU.



PROGRAMMED I/O FOR INPUT OUTPUT DATA TRANSFER

→ In programmed I/O, CPU makes a request & then CPU stays in program loop (Polling) until the I/O device indicates that it is ready for data transfer. [the I/O device takes no further action to alert the CPU (i.e., it doesnot interrupt the CPU]

Disadvantage:

Time consuming process since it keep CPU busy needlessly.

* To avoid this problem interrupt facility can be used

Polling:- When CPU continously check the devices to See if it is ready, then this method is called polling [which is totally a waste of time]

Interrupt-Initiated I/O

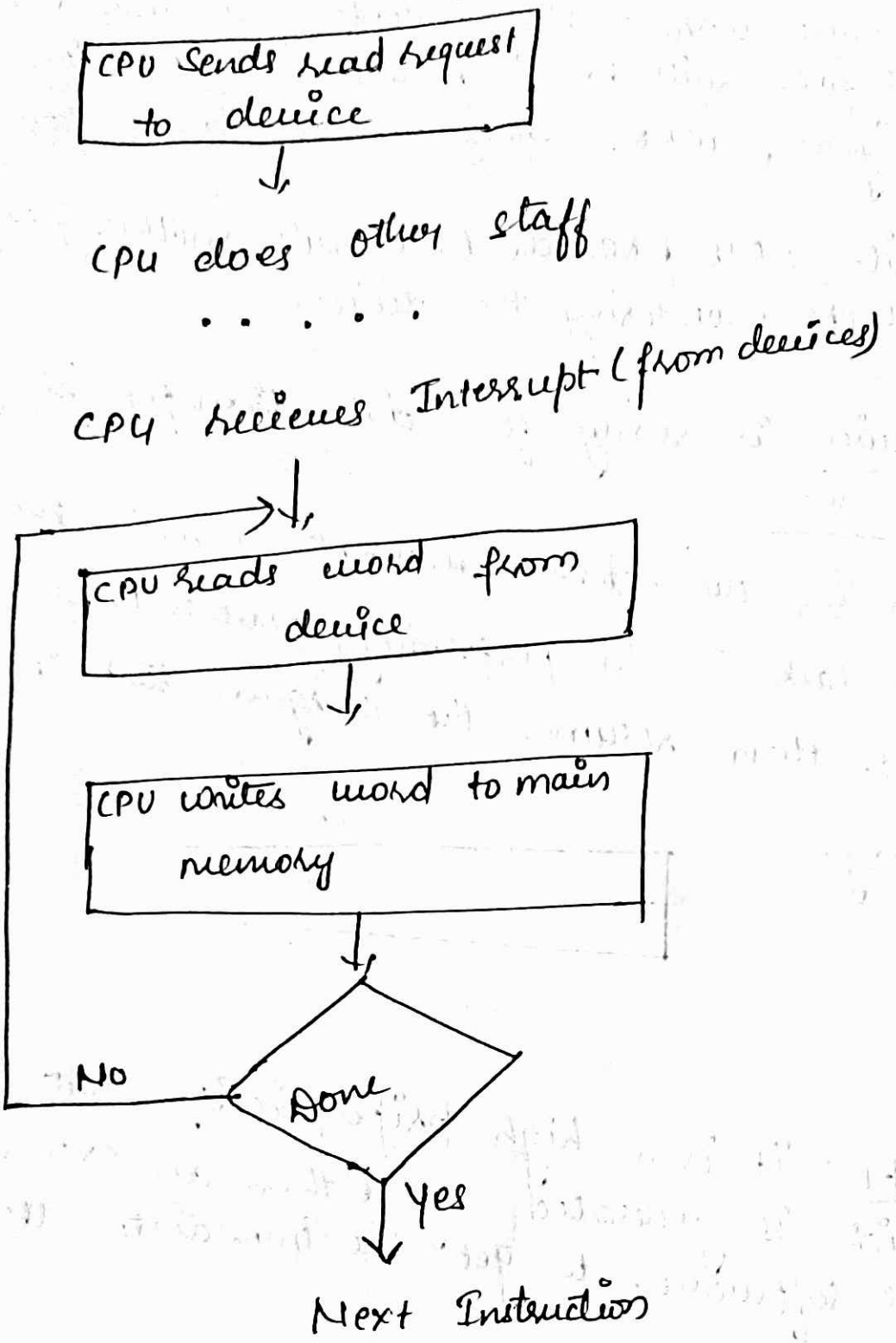
→ In Interrupt-initiated I/O, instead of continuous monitoring of CPU, interface will be informed to issue an Interrupt request signal, when data are available from the device.

→ Meanwhile, CPU proceeds to execute another programs & interface keeps monitoring the device.

→ when device is ready for data transfer it generates interrupt request.

→ Upon detecting the external interrupt signal, the CPU stops the task it is performing, process the I/O data-transfer & then resumes the original task it was performing.

Interrupt: it is a high ~~prio~~ priority ~~Iead~~ signal which is generated by either an external devic or some software. to get the immediate attention of the CPU.



Interrupt Initiated I/O for Input Output

Data transfer

Advantages

Using interrupts the wait period is not ideally eliminated.

Direct Memory Access (DMA)

→ To transfer large blocks of data at high-speed between external device & main memory. DMA approach is used often.

→ DMA allows data transfer directly between I/O device & main memory with minimal intervention of CPU.

→ DMA means CPU grants I/O Interface authority to read from or write to memory without its involvement.

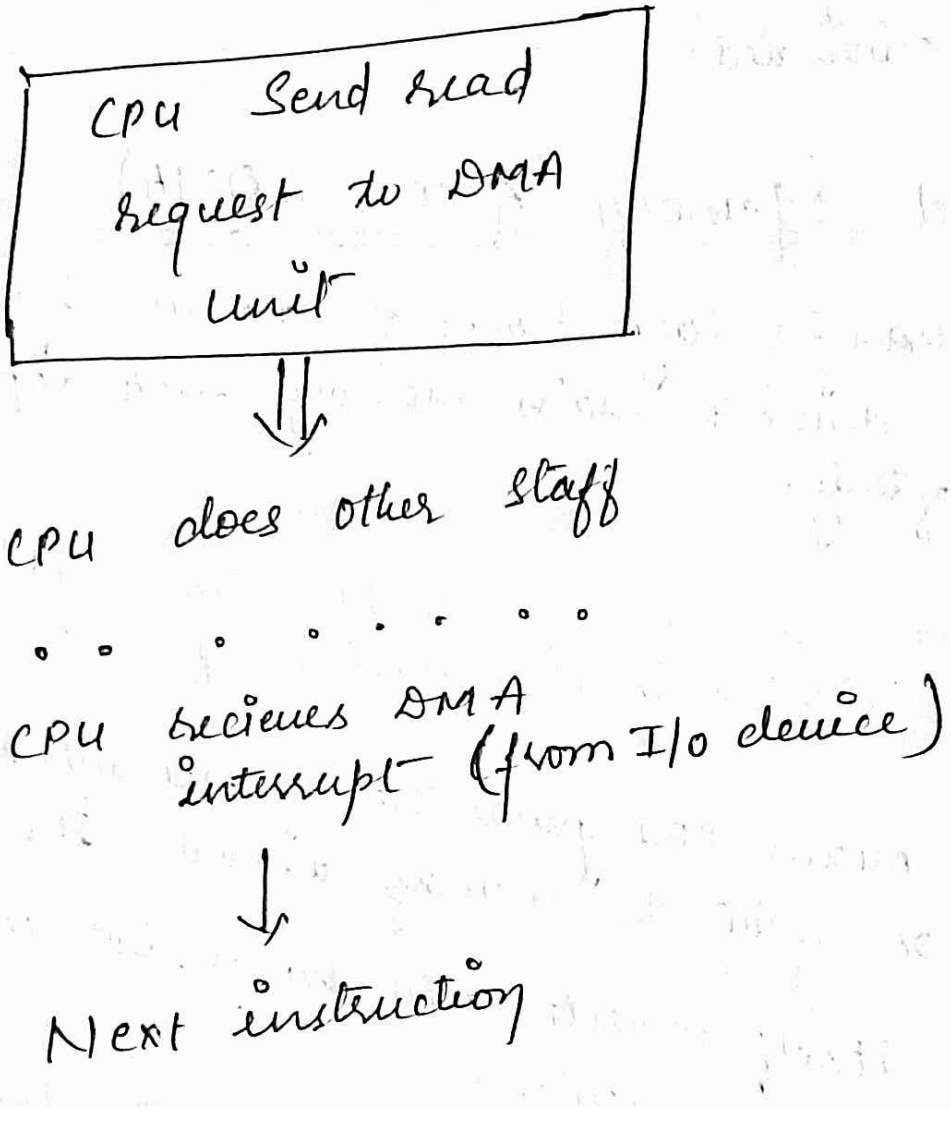
→ DMA itself controls data transfer between main Memory & I/O device.

→ CPU is only involved in beginning & end of the transfer & interrupt only after entire block has been transferred.

* CPU asks the DMA controller to transfer data between a device & main memory & then CPU proceeds to execute other tasks.

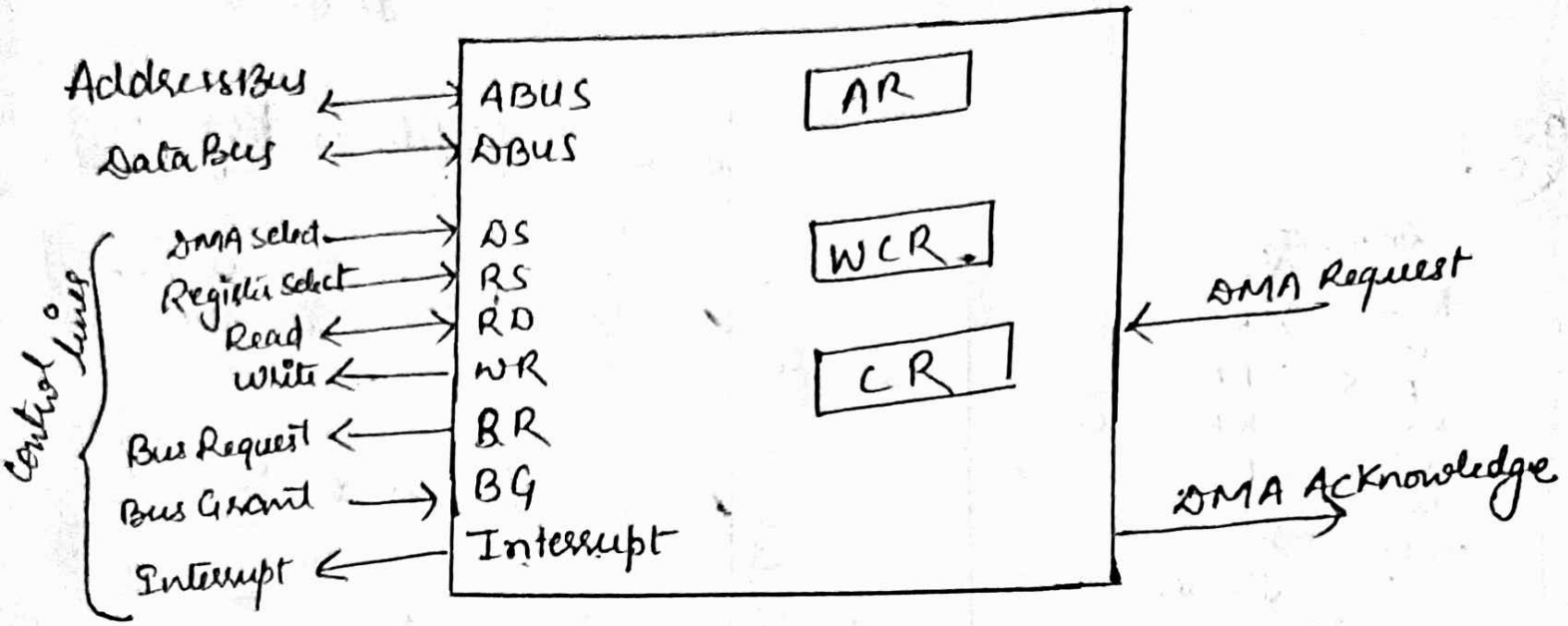
* The DMA controller issues a ~~request~~ requests to the right I/O device, waits & manages data transfer between the device & main memory

* When data transfer is finished the DMA controller interrupts the CPU.



DMA Controller

→ A device which allows Direct Memory Access is called DMA controller.



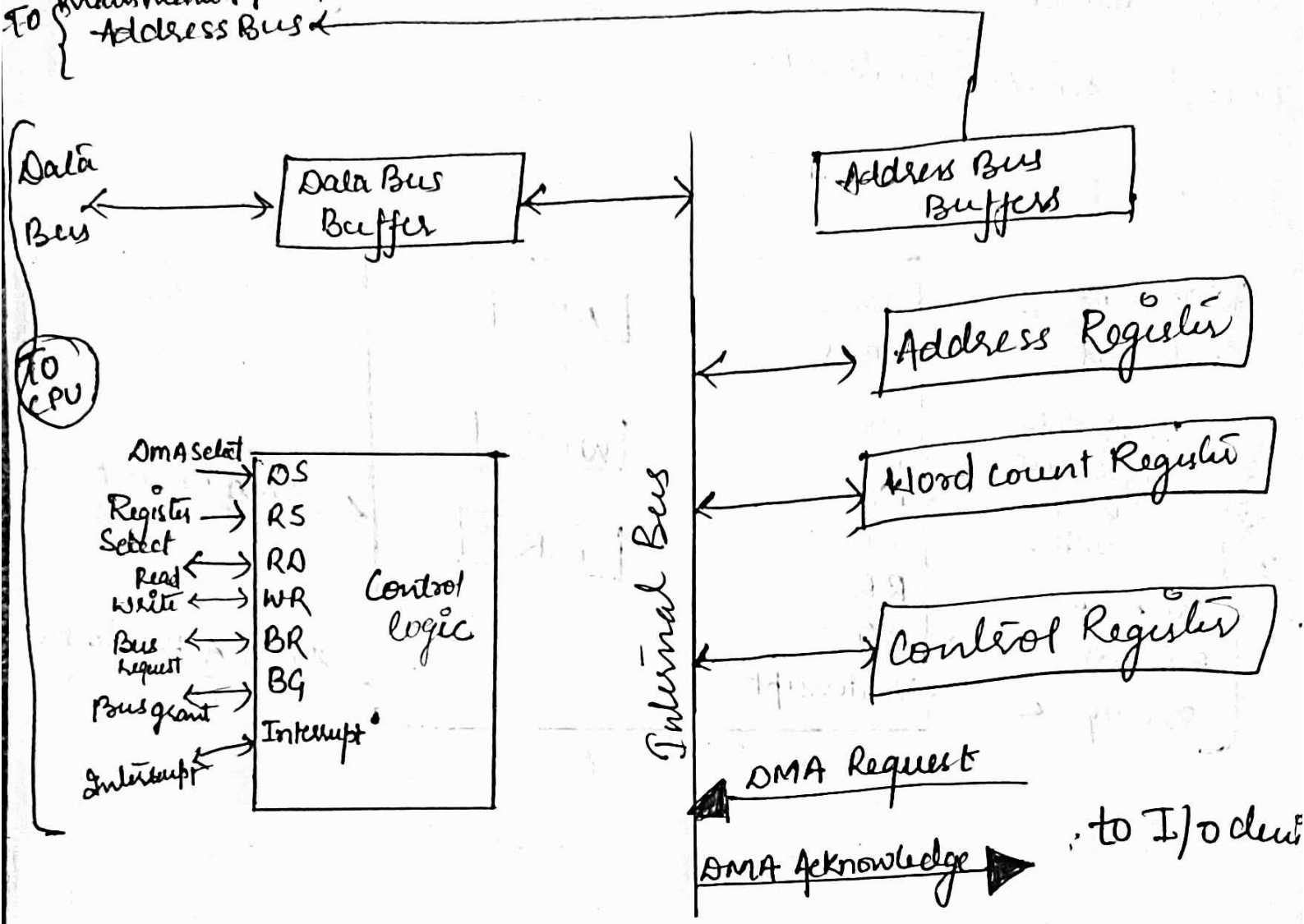
There are 3 register

① AR (Address Register): It holds the Address of data on which memory would be access either for Read or write operation. It communication directly with memory through Address Bus. Here after each transfer (AR) value get inament by 1 (+1)

② Word count Register (WCR): It holds the no of words to be transfer. After each transfer word count value is get decrease by 1 (-1), when it reaches to zero means data transfer is finished completely

③ Control Register (CR):- It specify the mode of transfer [whether read or write in memory]

Internally Detailed Block diagram of DMAC (Direct Memory Access Controller)



Block Diagram of DMA Controller

DMA controller

→ The DMA controller needs the usual circuits of I/O devices. an interface to communicate with the CPU & I/O devices.

→ The DMA controller has three registers:
(i) Address Register (ii) Word count Register, (iii) control Register

(i) Address Register: It contains an address to specify the desired location in memory. It is incremented by 1 after each word is transferred to memory.

(ii) Word count Register:
It holds the number of words to transferred. It is decrement by 1 after each word transferred & internally tested for zero.

(iii) Control Register :-
It specifies the mode of transfer i.e.) Read/write the memory.

→ The DMA controller communicates with the CPU via databus & control lines.

→ The registers in DMA are selected by the CPU through the address bus by enabling DS (DMA select) and RS (Register Select) inputs.

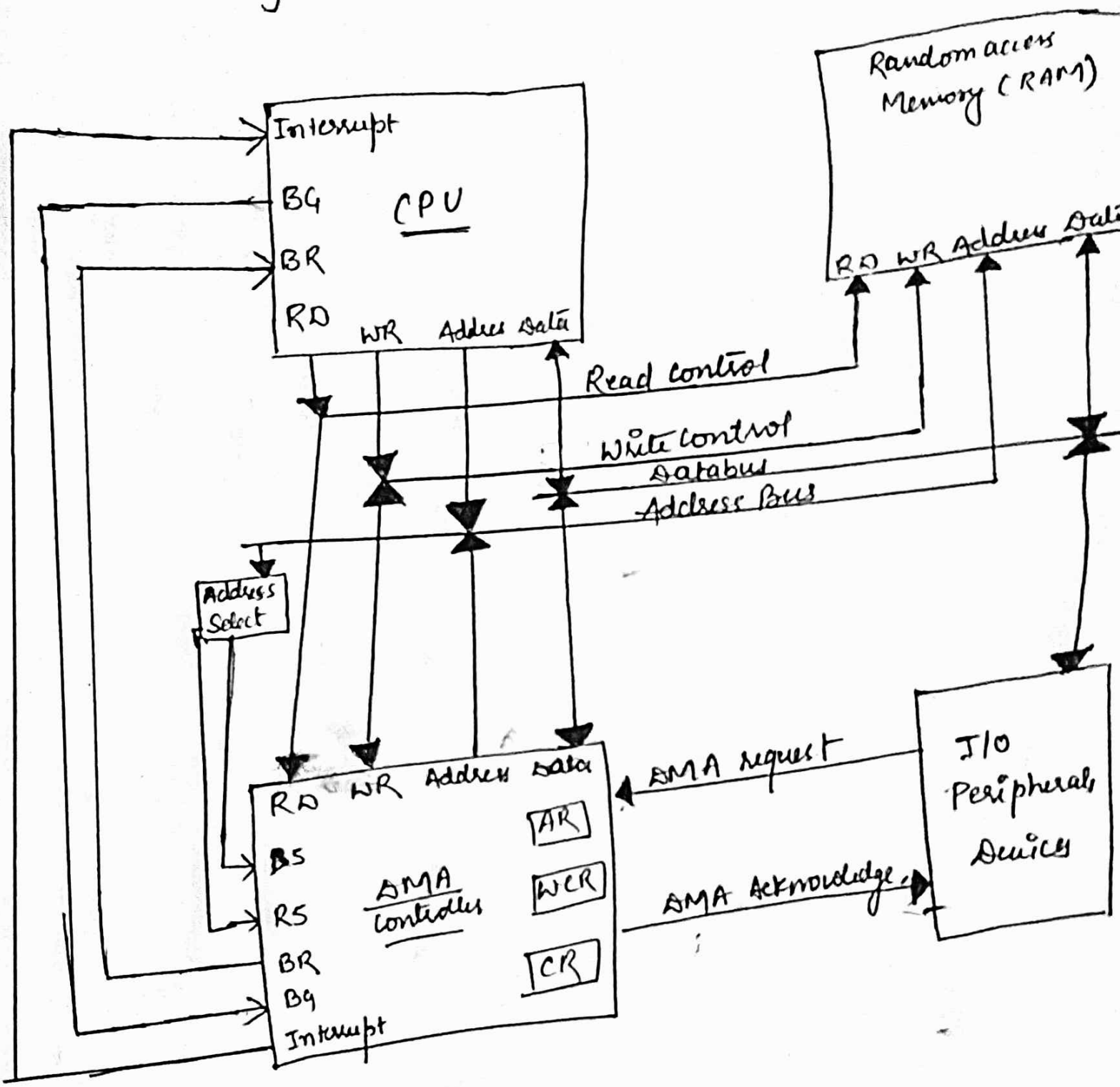
The RD (read) & WR (write) inputs are bidirectional & either of the one is activated at a time (RD or WR)

→ When BG (Bus Grant) i/p is 0 the CPU can communicates with the DMA registers. through the data bus to read from or write to the DMA registers.

→ When BG (Bus Grant) ip is 1, the CPU ~~can~~ ~~Communicates with the DMA registers~~ has relinquished the buses & the DMA can communicate directly with memory by specifying an address in the address bus & activating RD or WR control.

→ DMA communicates with the External input/output devices through the request & acknowledge lines by using Handshaking procedure.

DMA Transfer: (How, DMA transfer data with I/O)



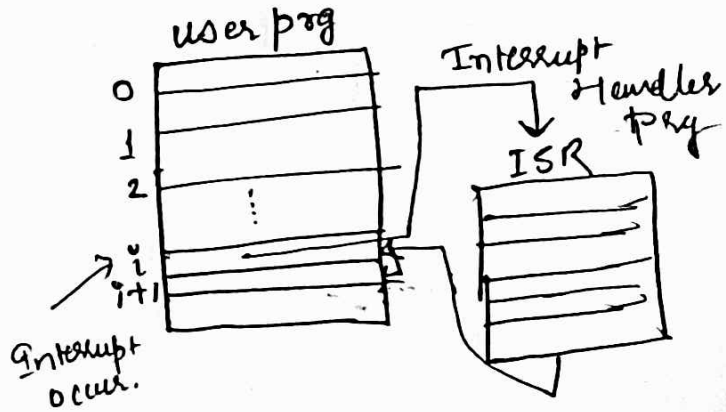
Interrupts

→ Normal execution of programs may be preempted if some device requires urgent servicing to deal with the situation immediately, the normal execution program must be interrupted.

→ Interrupt is a high-priority instruction generated by H/w device or S/w program instruction to get immediate attention of CPU

or

→ An Interrupt is a request from a H/w or S/w program for a ~~Service~~. the immediate Service by the processor.



Interrupt Handler program: Execute the interrupt which is occur.

It will find out which code we execute for this interrupt for executing this code their is an ISR (Interrupt Service Routine)

→ This ISR has some ~~hand~~ address. & this will find out by the Interrupt handler. By checking the address of this ISR in IVT (Interrupt vector Table)

which is store in a memory somewhere.

Ex: for explaination purpose (a interrupt)

① Printer out off paper (all signals are goes to cpu as

② Divide by zero, I/o device request for data transfer,

Power failure, overflow.

These all will generate an interrupt which will have to handle immediatly.

Power failure, overflow.

These all will generate an interrupt which will have to handle immediatly.

Diagram Discription

→ As interrupt generated, the processor provides the requested servie by executing an Interrupt service routine (ISR)

→ The state of processor [content of PC, general register, & PSW (Process Status word)] is first

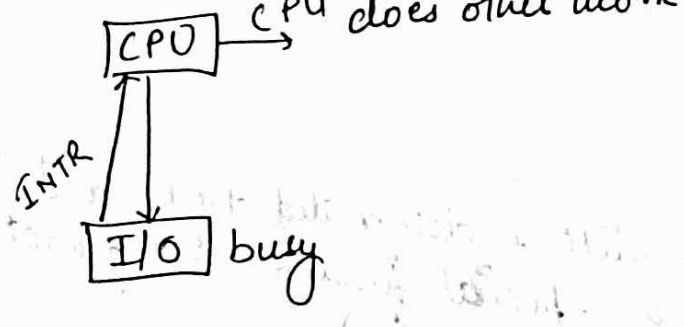
~~PSW~~ ~~PSW~~ ~~contd~~

Saved in memory before servicing the interrupt.

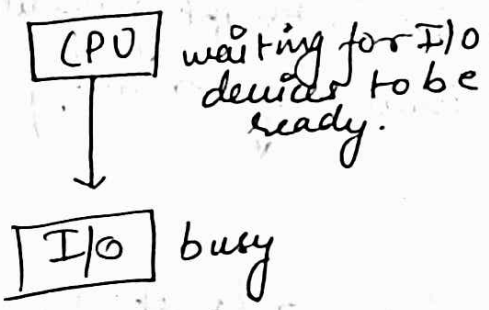
→. When ISR is completed the state of the processor is restored so that interrupted program may continue.

PSW: contain some status flag bits which we will be use to store control information

With interrupt



Without interrupt



Why? Interrupt is used

It is used to improve the performance of CPU.

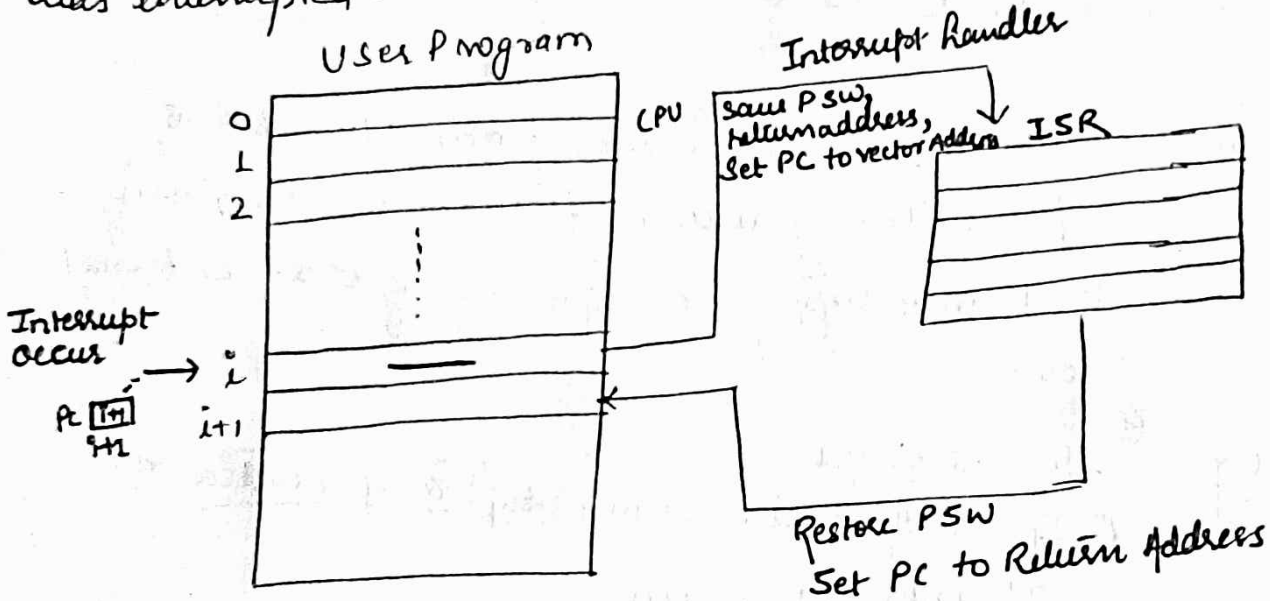
Interrupt Handling Mechanism

① Interrupt is generated by h/w device or s/w program.

② CPU suspends current program execution by storing the return address from the Program Counter (PC), general registers values & values in PSW (Program Status word) into a memory stack by push operation.

③ CPU executes the Interrupt Service Routine (ISR)

④ When ISR finishes, CPU restores (pop) the previous program state & continues the original program that was interrupted.



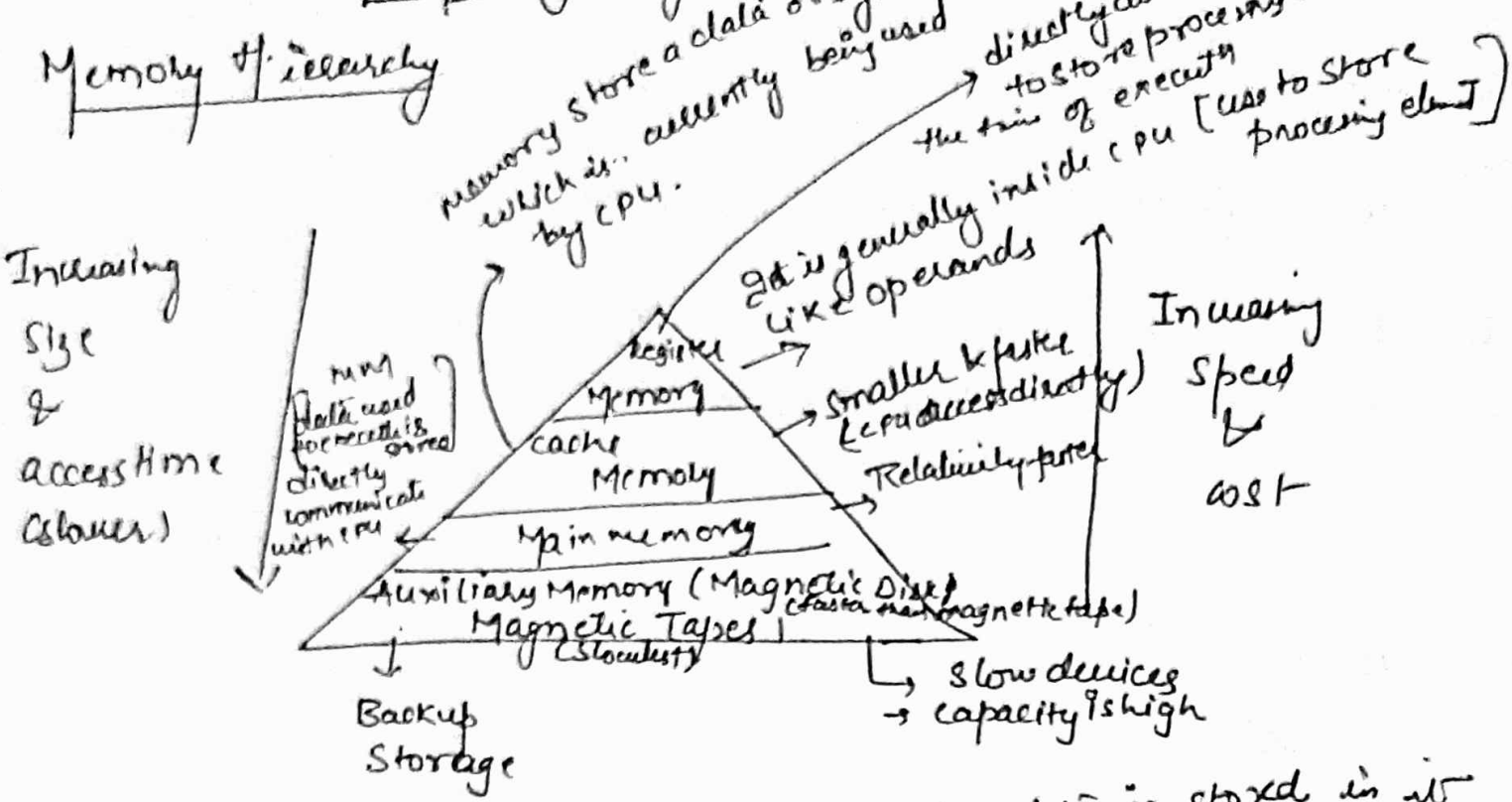
Advantages:-
Efficiency of CPU is improved.

Disadvantages:-
Overhead required to service the interrupt.

[Save status, Restore it] → require time
- ↳ Execution overhead
- ↳ Interrupt latency

Memory Organisation

Memory Hierarchy



Memory:- Essential component because all data is stored in it but it is not possible to store all program or data in a same memory.

→ If we use only Auxiliary or Secondary memory (back up storage) it makes the system slower and not possible to access by CPU same time and very Expensive ~~ots~~ as well as it take time.

→ Therefore, economically way is to use more than one memory in memory hierarchy. in digital computer.

→ Memory hierarchy contains all the memory storage device

Auxiliary Memory (Magnetic Disk & Magnetic Tapes)

→ Use for Back up storage
→ Slow devices with high capacity
→ In which Magnetic tapes are slowest as compared with Magnetic disk.

Main Memory :

→ Relatively faster than Auxiliary Memory
→ It store data & program which is to be executed
→ It directly communicate with CPU.

Cache memory

→ Smaller & faster
→ CPU access data directly
→ memory store a data or segment which is currently being used by CPU.

Register Memory

→ It is generally inside CPU ~~like operands~~
→ Use to store processing element like operands
→ directly attach with CPU & use to Store processing elements at the time of execution.

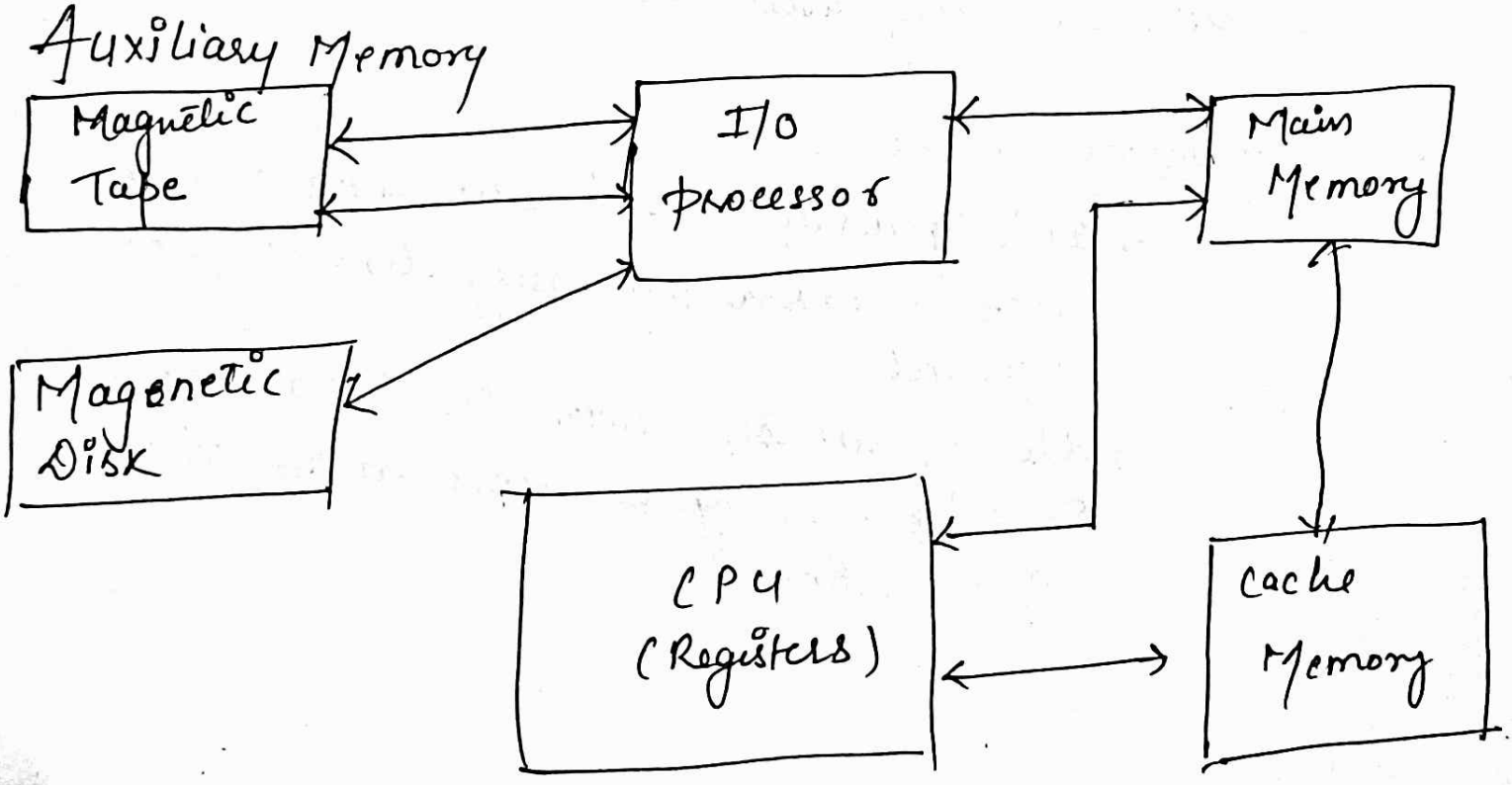
In Memory hierarchy when you move

① Top to bottom

Memory Size Increases &
access Time also Increases
Hence it is slower to access.

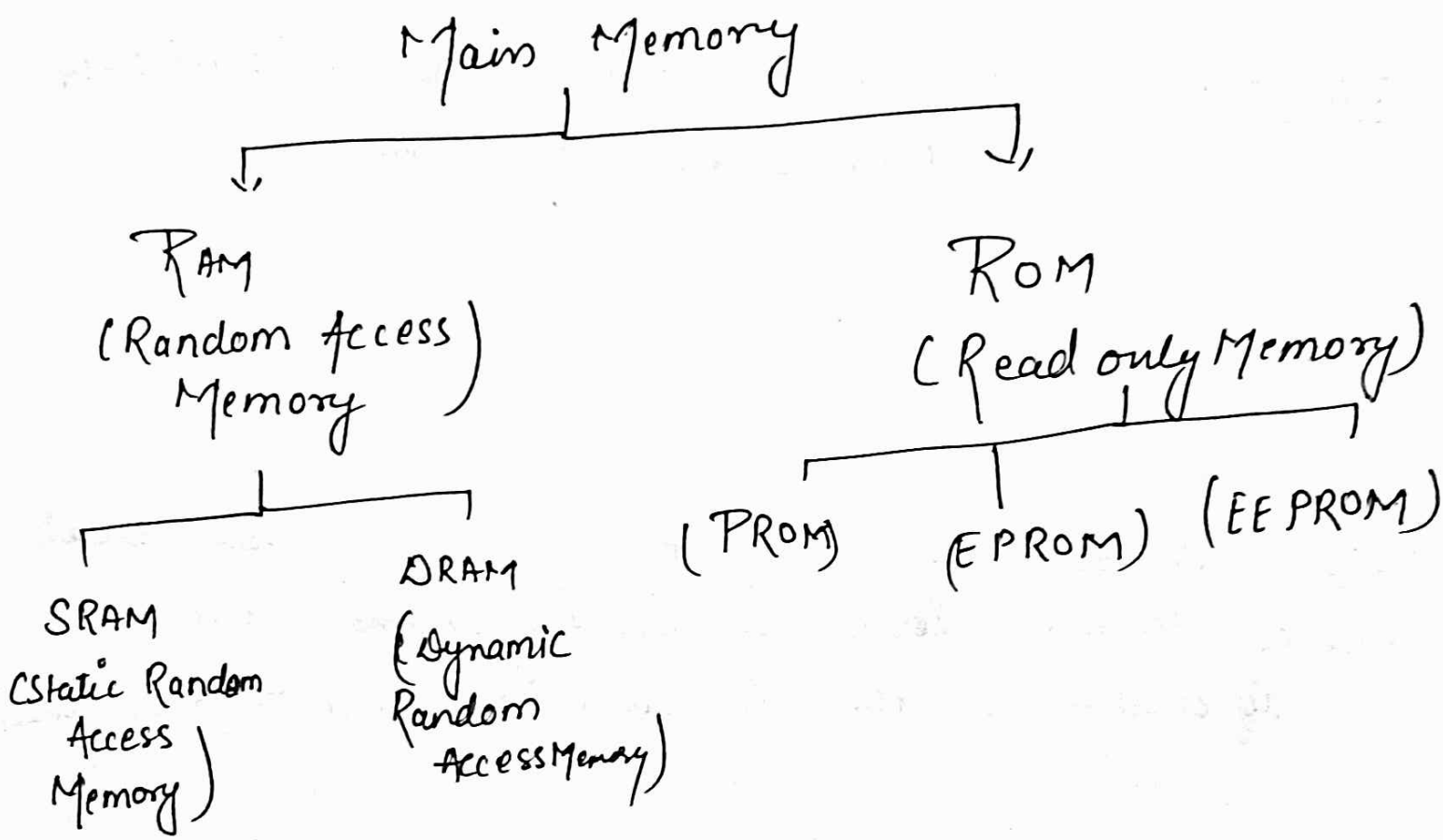
② Bottom to up

Memory speed Increases and cost
is also increases
Hence it is faster to access.

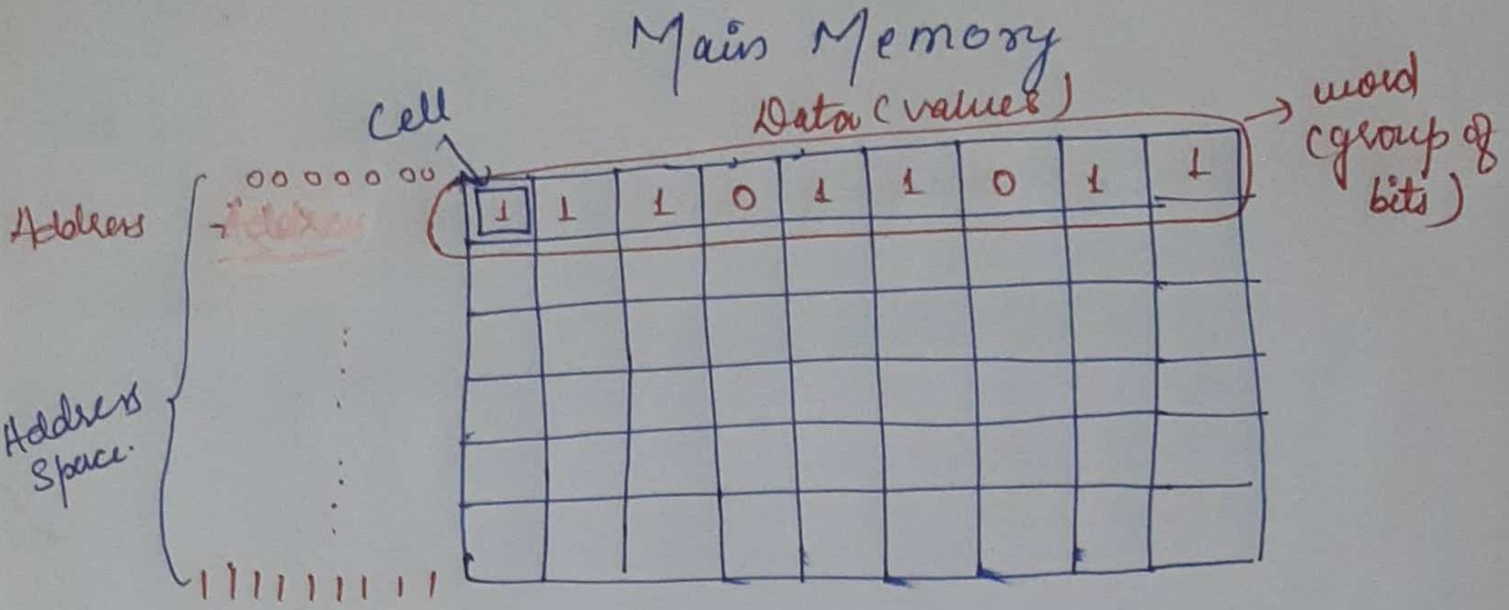


Main Memory

→ Central Storage unit in computer System
→ Fast memory used to store programms & data during execution.
→ Principle technology: Semiconductor IC's.



Internal Structure of Memory (RAM/ROM)



→ MM contains large no of cells. (Storage cells)

Cells: → Each cells is capacable of storing one bit of information (1/0).

→ cells are processed in groups of fixed size words.

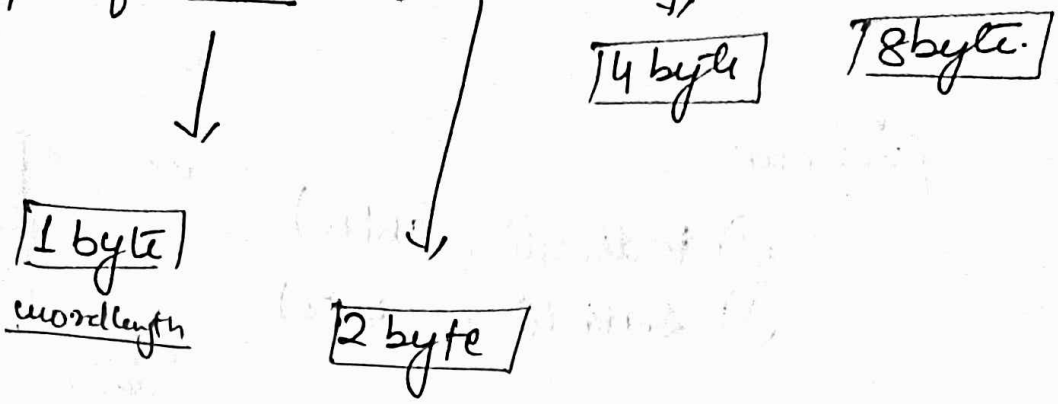
i.e., Data is transferred to & from memory in groups of bits called words

address: For easy access different addresses are associated with each word location in memory.

and we also ~~use~~ need some control cmd that we wanted to store or retrieve from memory.

Word length : No of bits in each word is often referred to as Word length of the computer

A word can group of 8 bits, 16-bits, 32 bits or 64 bits
(growing)



Data is transmitted from the memory or to memory is always in the form of word.

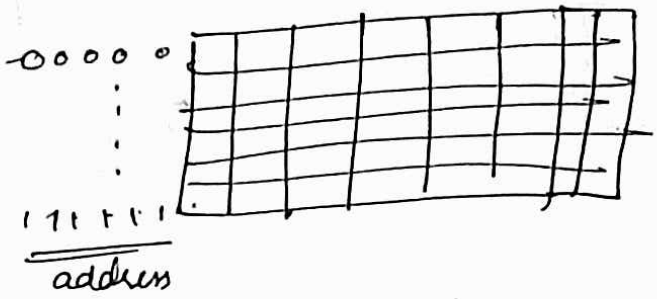
Address space Total no of uniquely identifiable location in memory.

Que: Suppose a size of memory is 64KB & a word size of 1 byte (in which one word is of 1 byte)

[It can also be written as $\frac{64KB \times 8bit}{\text{memory size}}$]

find out

1. Address line (bits)
2. Data lines (bits)



Sol ① Address line

Given 64KB x 8bit

For specifying the address how many bits require

~~You can write~~

~~$64 = 2^6$~~

~~$64 \quad 2^6$~~

~~$64 = 2^6$~~

$64\,KB$

$\downarrow$

$2^6 \times 2^{10} = 2^{16}$

Address bit = 16 bit

Data line

from 64kb x 8 bit

we know

one word is of 8 bits

Therefore, data line also become 8 bits

~~Assoc~~

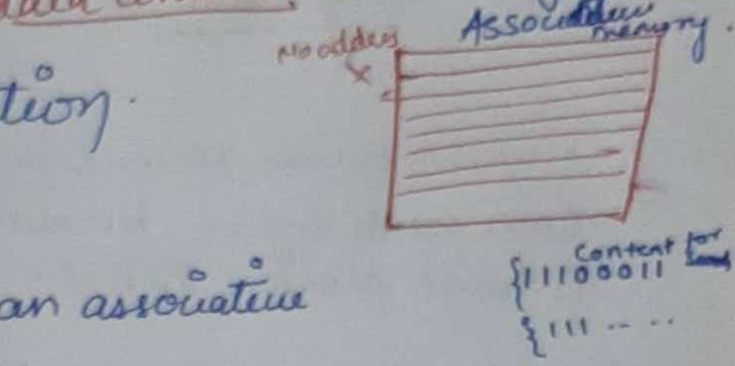
1 KB = 1024 bytes

$= 2^{10}$ B.

Associative Memory

→ A memory unit accessed by content is called an Associative Memory or Content Addressable Memory (CAM).

→ Associative Memory is accessed simultaneously & in parallel on the basis of data content rather by specific address or location.



→ When a word is written in an associative memory, no address is given.

→ This memory is capable of finding an empty unused location to store the word.

→ When a word is ~~capa~~ to be "read" from an associative memory.

the content of word is specified or part of word is specified.

→ The memory locates all words which match the specified content and marks them for reading

→ Because of its organization, the Associative memory is uniquely suited to do parallel searches by data Association

→ An Associative Memory is more expensive than Random Access Memory -

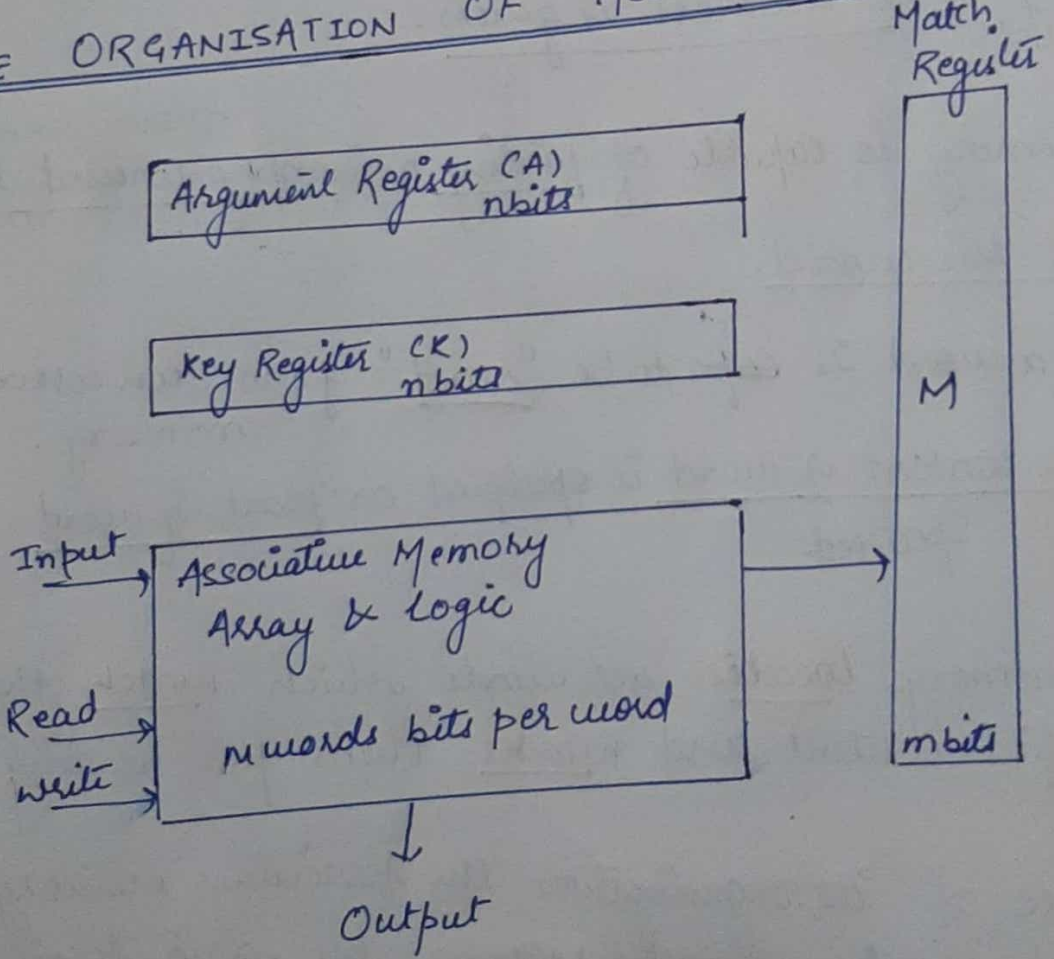
- because each cell must have storage

capacity capability as well as logic circuits for matching its content with an external argument

→ That's why, Associative Memory is useful in application. Where Search Time is very critical & must be very short.

→ Eg:- Neural Networks because we stores data in patterns.
Google search engine, because we provide content on which it will provide result.

HARDWARE ORGANISATION OF ASSOCIATIVE MEMORY



BLOCK DIAGARM OF ASSOCIATIVE MEMORY

ARGUMENT: It contains words to be searched.

Register (A): It has n bits (one for each bit of word).

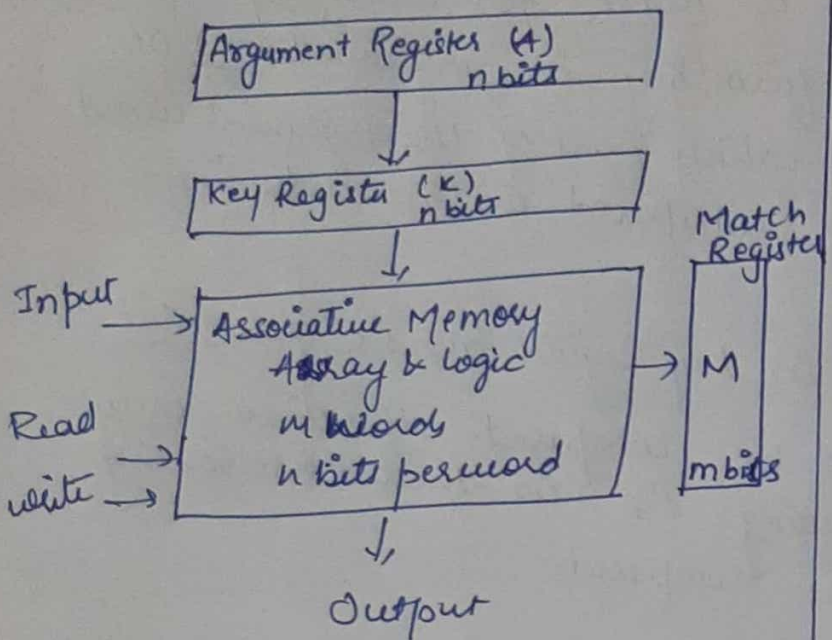
KEY REGISTER: It provides a mask for choosing a practical particular field/key in argument (word) (K) or
→ It specifies which part of the argument word needs to be compared with words in memory.
→ It all bits in key Register are 1's, the entire word should be compared. otherwise, only the bits having 1's in their corresponding position are compared.

Associative Memory Array → It contains the word that are to be compared ~~otherwise, only the bits having 1's in their corresponding position are compared~~. with the argument word in parallel.
→ It consists of m words with n bits per word.

Match Logic (m): → It has m bits, one bit corresponding to each word is the memory array.
→ After the matching process, the bits corresponding to matching words in match register are set to 1.

Reading is accomplished by sequential access in memory for those words whose match bits are set (or 1).

Explaination of Hardware organisation of Associative Memory with Examples



A 101 111100

K 111 000000

			M
Word 1	100	111100	0
Word 2	101	000001	1
Word 3	101	111100	1
Word 4	110	001010	0
Word 5	111	000111	0

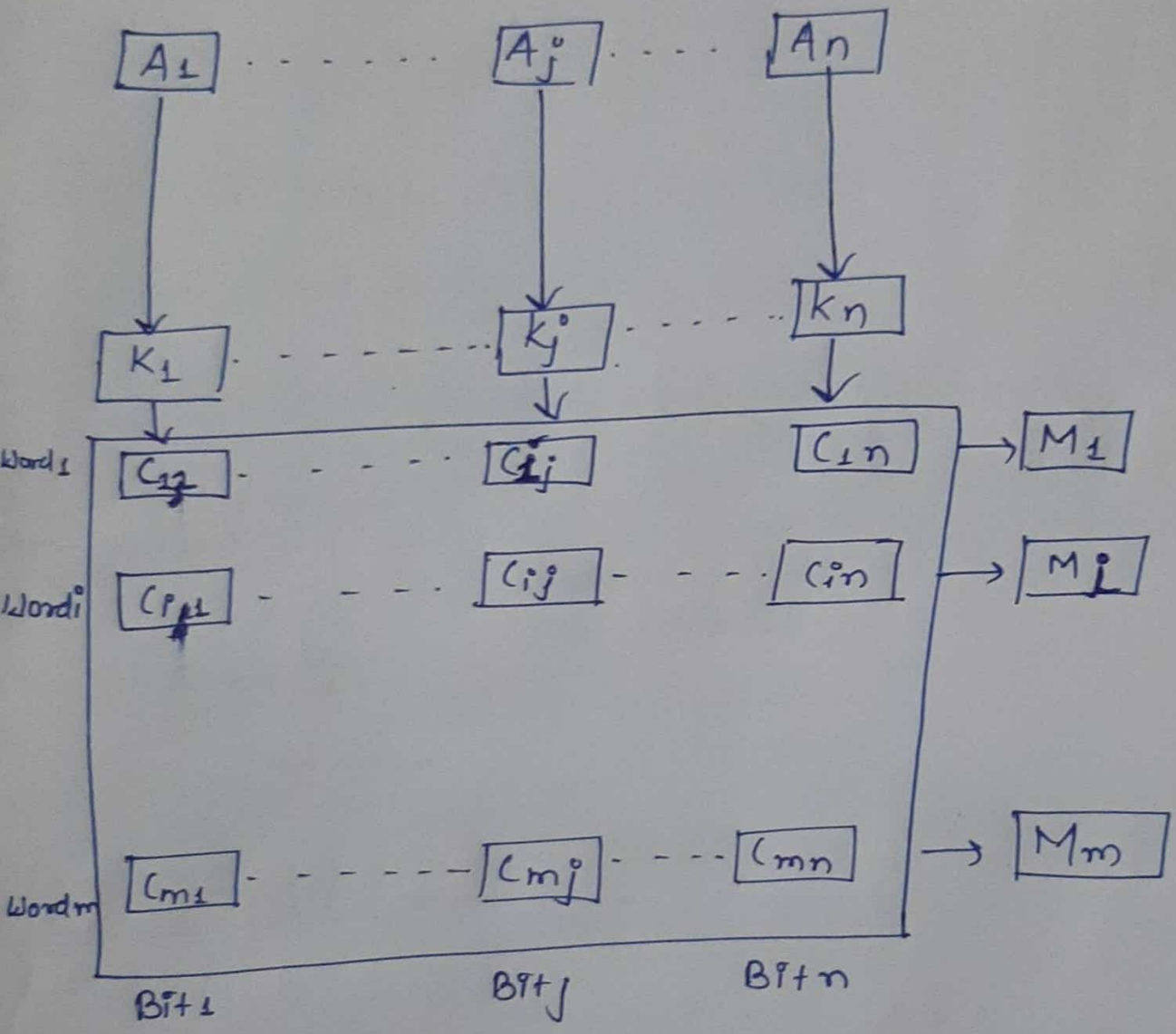
0 → no match

1 → match

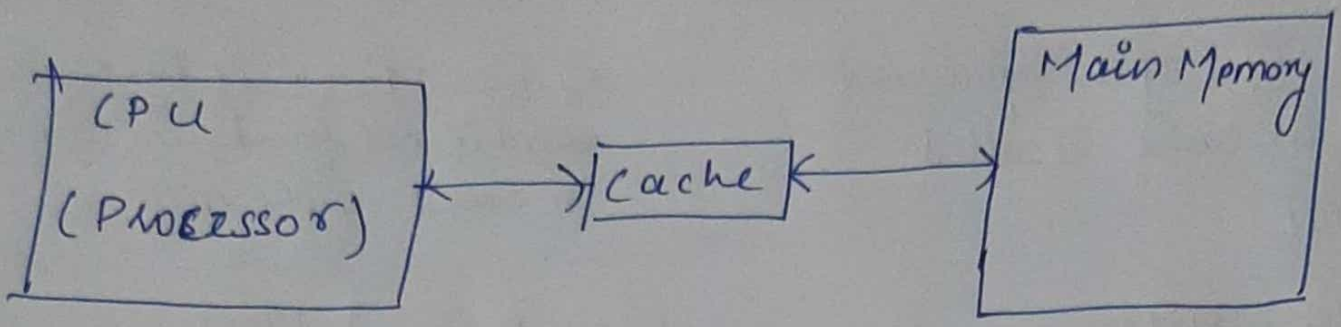
Associative memory of m words, n cells per word

The cells in array is represented by C_{ij} : a cell for bit "j" in word "i"

where i = word number
j = bit position in the word.



Cache Memory



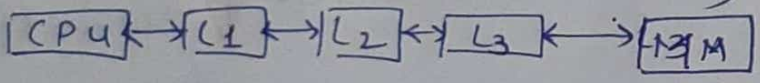
→ Small sized fast memory

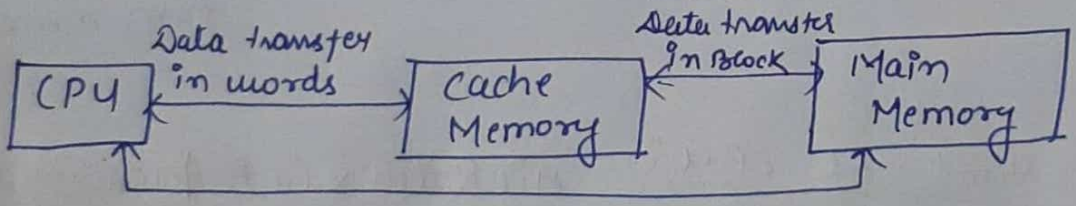
→ Placed between Main Memory & CPU

→ high-Speed volatile Memory & CPU

→ Contains most frequently accessed instructions & data.

→ located inside the CPU chip or motherboard
(External cache - L3)
(internal cache L1 & L2)





Cache Performance

→ It is measured in terms of Hit Ratio

Cach Hit: If the required word is found in cache is called cache hit

Cache Miss: If the required word is not found in cache is called cache miss

Hit Ratio (h) $= \frac{\text{Hits}}{\text{Hits + Miss}}$

$= \frac{\text{no. of hits}}{\text{Total no of CPU References}}$

Miss Ratio (1-h) $= \frac{\text{Miss}}{\text{Hits + Miss}}$

$= \frac{\text{Total no. of miss}}{\text{Total no of CPU references}}$

Cache access Time (Cache Hit Time): Time required to access word from the cache.

Miss Penalty (Cache Miss Time Penalty): Time require to fetch the require block from memory.

Average Access Time of CPU: = ~~Hit Ratio × Cache Access Time + (1 - Hit Ratio)~~

$= \text{Hit Ratio} \times \text{Cache Access Time} + (1 - \text{Hit Ratio}) \times \text{Miss Penalty}$

$= h \times T_c + (1-h) + T_m$

∴ T_m = Memory access time or Miss Penalty

Miss Penalty = Cache access Time + Main Memory Access Time

Que: The access time of a cache memory is 100ns & that of main memory 1000ns. It is estimated that 80 percent of the memory requests are for read & the remaining 20percent for write. The hit ratio for read accesses only is 0.9. A write through procedure is used.

a) what is the average access time of the system considering only memory read cycle.

b) what is the average access taking into consideration the write cycles.

Solution

Average Access Time = Hit Ratio x Cache Access Time + (1 - Hit Ratio) x Miss Penality

Hit Ratio ⇓ Miss Ratio

Miss Penality ⇓ Cache Memory + Main Memory Access Time.

Or

Average Access Time of CPU = $h \times \underbrace{T_c}_{\text{(cache access time)}} + \underbrace{(1-h)}_{\text{(miss ratio)}} \times \underbrace{T_m}_{\text{(cm + mm access time)}}$

Miss Penality = Extra time needed to bring the data (block) from one level to another (memory to cache)

a) Total read access time = $0.9 \times 100 + (1 - 0.9) \times (100 + 1000)$

$= 0.9 \times 100 + 0.1 \times 1100$

$= 90 + 110$

$= 200 ns.$

Total write access time = $1 \times \max(100, 1000)$ (cm, mm)

= 1000 ns

Beoz it is write through cache, largest of the cm & mm access time is considered

(b) Average Access Time for both read & write
= 80% are Read request + 20% are write request.

$= 0.8 \times 200 + 0.2 \times 1000$

$= 160 + 200$

$= 360 ns$

(c) Hit Ratio = $0.8 \times \underbrace{0.9}_{\text{HitRatio for Read access.}} + 0.2 \times \underbrace{0}_{\text{Hit ratio for write access is always 0}}$

$= 0.72$

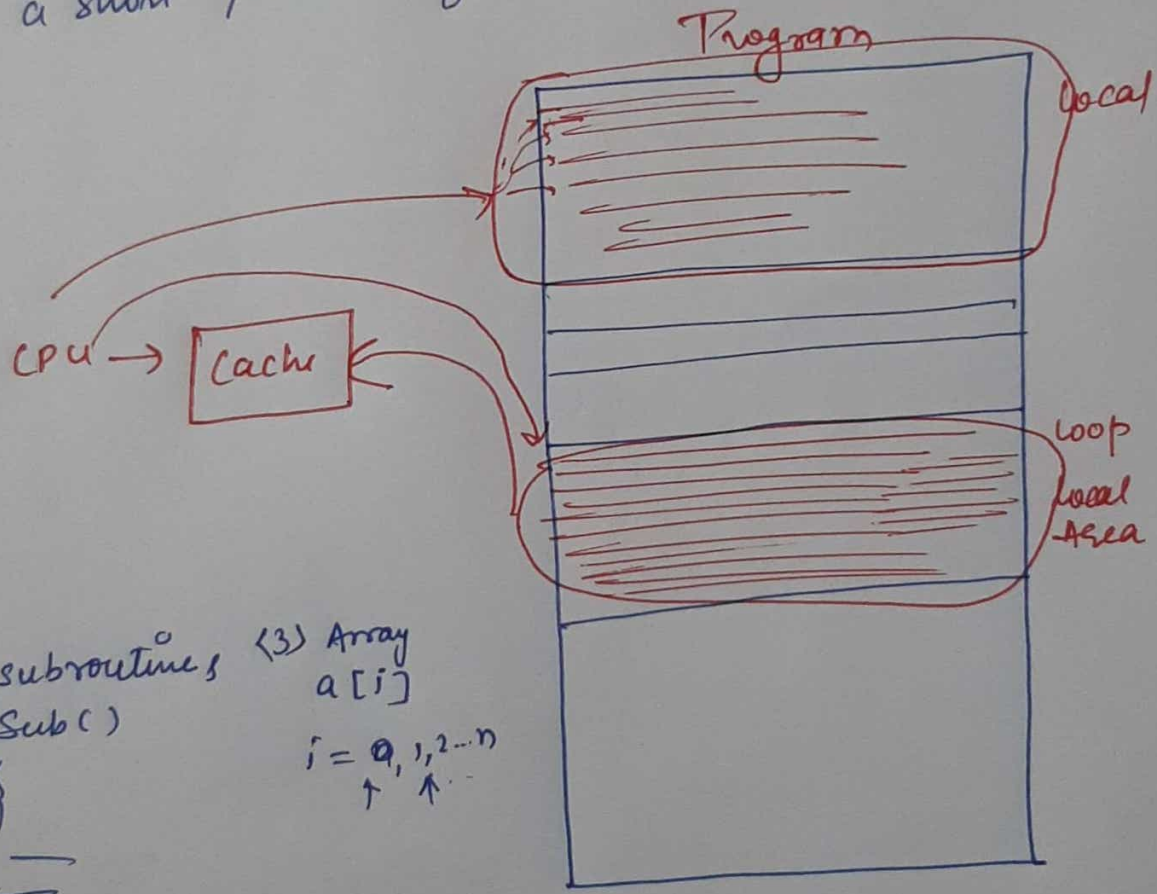
Locality of Reference in Cache Memory

Locality of reference is also known as Principle of locality

→ Locality of reference is a term for the phenomenon, in which the same values or related storage locations are frequently accessed, depending on the memory access pattern.

OR

→ Locality of reference is the tendency of a processor to access the set of memory address locations repetitively over a short period of time.



Eg:

(1) Program
loop
{
}

(2) subroutines
Sub()
{
}

(3) Array
a[i]
i = 0, 1, 2 ... n

(a) Table-lookup Procedure.

2 Types of locality of Reference

① Temporal locality ② Spatial locality

Temporal locality :- It says that if a program accesses one memory address, there is a good chances that it will access the same address

→ It refers to reuse of specific data/resources within a short period of time

Eg: loop

```
for loop (1 to 50)
{
  ==
  ==
}
```

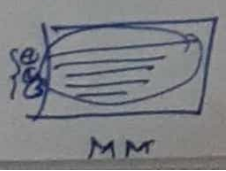
② **Spatial locality**: says that if a program access one memory address, (Data locality) there is a good chance that it will also access other near by addresses.

→ It refers to use of data elements within relatively close storage locations.

Eg: ① Nearly "Every program" exhibits spatial locality
↳ becoz instructn are usually executed in sequences

② In terms of data: Array a[i], i=0, 1, ... 2..

③ Records: Employees Id/Name/Address/Phone.



Levels of Cache Memory

Level 1 (L_1) Cache :

→ L_1 is fastest cache & it usually comes within the processor.
→ Typically it ranges in size from 8KB to 64KB & was the high-speed SRAM (Static RAM) instead of the slower & cheaper DRAM (Dynamic RAM) used for main Memory.
→ It is referred to as Internal cache or Primary cache.

Level 2 (L_2) cache :

→ L2 cache is larger but slower in speed than L1 cache.
→ Stores recently accessed information.
→ Also known as Secondary cache, it is designed to reduce the time needed to access data in cases where data has already been accessed previously.
→ L2 cache comes between L1 and RAM & is bigger than Primary cache.
→ typically ranges from 64KB to 4MB

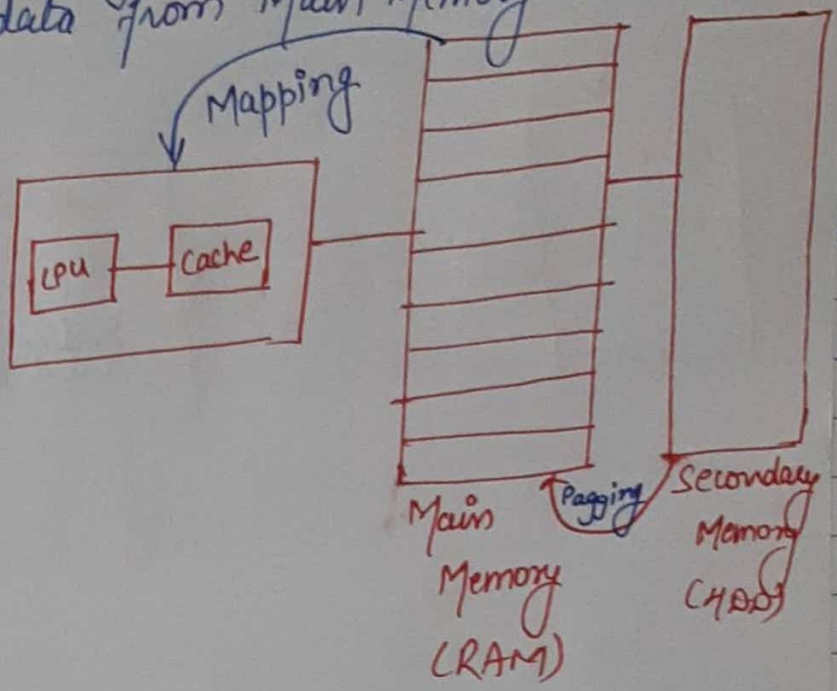
Level 3 (L3) Cache

→ L3 cache memory is an exchanged ~~from~~ form of memory present on the motherboard of the computer.
→ L3 cache is a memory cache that is built into the ~~motherboard~~ motherboard.
→ It is used to feed the L2 cache, and is typically faster than System's main memory but slower than L2 cache.
→ It has more than 3MB of storage ~~unit~~. in it.
→ Not all processors have ~~their~~ this L3 cache, b'coze it is used to enhance the performance of L1 & L2 cache.

Cache Mapping

Cache Mapping is a technique by which content of Main Memory is brought into the Cache Memory

→ It is a transformation of data from Main Memory to Cache Memory.

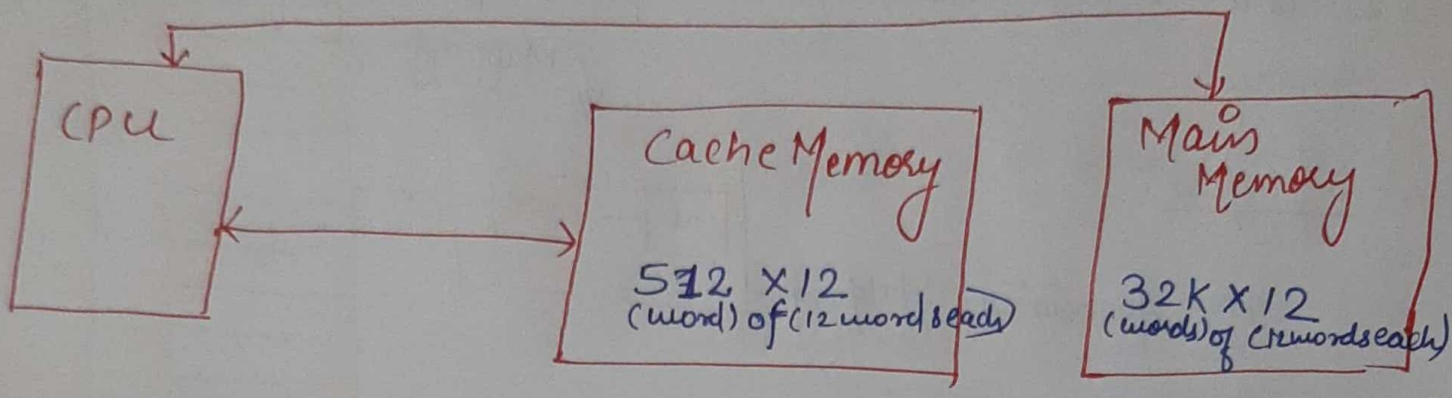


→ 3 Types :-

1. Associative Mapping
2. Direct Mapping
3. Set-Associative Mapping

* ~~Content of~~ Transferring (loading) data (content) of Secondary Memory to Main Memory is called Pagging.

Example :- Common Example for understanding all these mapping.



32K [words]

$2^5 \times 2^{10} = 2^{15}$ words

CPU address = 15 bits

data = 12 bits