CHAPTER - 4
INPUT - OUTPUT ORGANIZATION

Peripheral Devices :-

Input & Output devices attached to the computer are also known as "Peripherals".

→ Among the most common peripherals are Keyboards, displays units, and printers.

→ Peripherals that provide auxiliary storage for the system are magnetic disks and tapes.

→ Peripherals are electromechanical and electromagnetic devices of some complexity. Only a very brief discussion of their function will be given here without going into detail of their internal construction.

Monitor and Keyboard

→ Video monitors are the most commonly used peripherals. They consist of a Keyboard as the input device and a display unit as the output device.

→ There are different types of video monitors, but the most popular use a cathode ray tube (CRT).

Printers :- Printers provide a permanent record on papers of computer output data & text.

→ There are three basic types of character printers. daisywheel, dot matrix, and laser printers.

Magnetic Tape :- Magnetic tapes are used mostly for storing files of data: for example; a company's payroll record.

→ It is one of the cheapest and slowest methods for storage and has the advantage that tapes can be removed when not in use.

Magnetic disk :-

→ Magnetic disk have high speed rotational surfaces coated with magnetic material.

→ Disks are used mostly for bulk storage of programs and data.

ASCII Alphanumeric Characters:-

Input and Output devices that communicate with people and the computer are usually involved in the transfer of alphanumeric information to and from the device and the computer. The standard binary code for the alphanumeric characters is ASCII (American Standard Code for Information Interchange). It uses 7 bits to code 128 characteristics.

* Input-Output Interface:- *

→ Input-Output provides a method for transferring information between internal storage and external I/O devices.

→ The differences that exist between the central computer and each peripheral. the major differences are.

1) Peripherals are electromagnetic and electromechanical devices and their manner of operation is different from the operation of the CPU.

2) The data transfer rate of peripherals is usually slower than the transfer rate of the CPU.

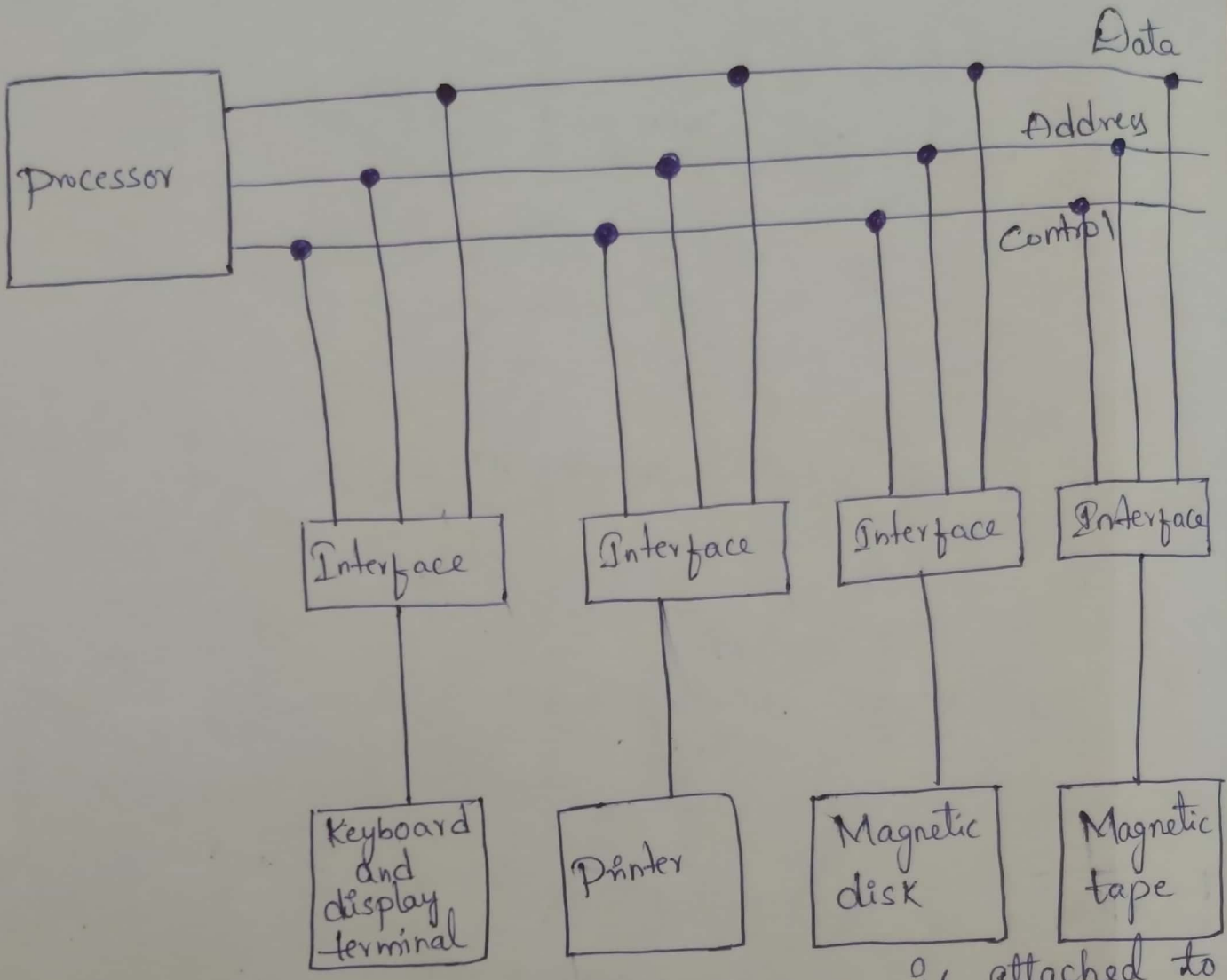
3. Data codes and formats in peripherals differ from the word format in the CPU and memory.

4. The operating modes of peripherals are different from each other.

→ To resolve differences, computer systems include special hardware components between the CPU and peripherals to supervise and synchronize all input and output transfers.

I/O Bus and Interface modules :-

→ The I/O bus consists of data lines, address lines and control lines. The magnetic disks, printer and terminal are employed in practically any general-purpose computer. The magnetic tape is used in some computers for backup storage.



→ The I/O bus from the processor is attached to all peripheral interfaces to communicate with a particular device, the processor places a device address on the address lines.

Control Command :- A control command is issued to activate the peripheral and to inform it what to do.

A status command :- Is used to test various status conditions in the interface and the periph eral.

A data output command :- It causes the Interface to respond by transferring data from the bus into one of its registers.

The data input command :- The data input comman is the opposite of the data output.

I/O Versus Memory bus :-

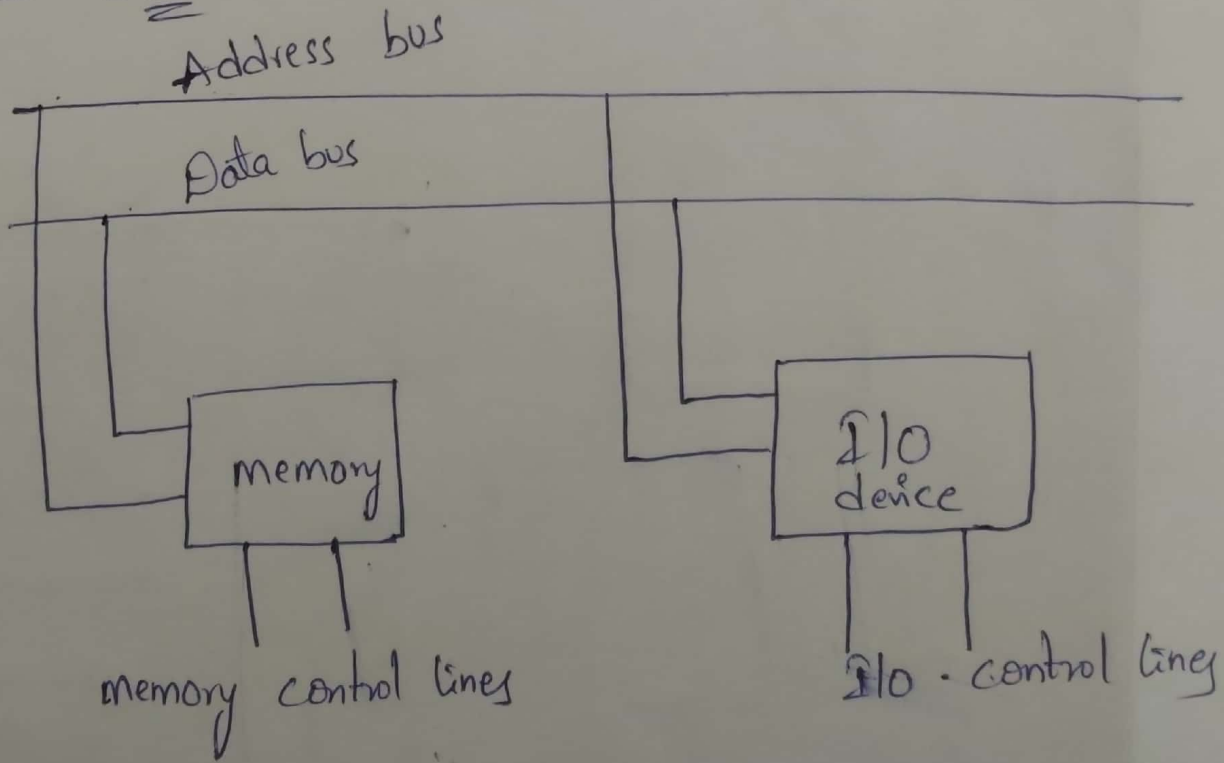
Like I/O Bus, Memory bus also contains data, Address & control lines.

→ Three ways to communicate
- → use two seperate buses
- → use one common bus but seperate control lines
- → use common bus with common control lines.

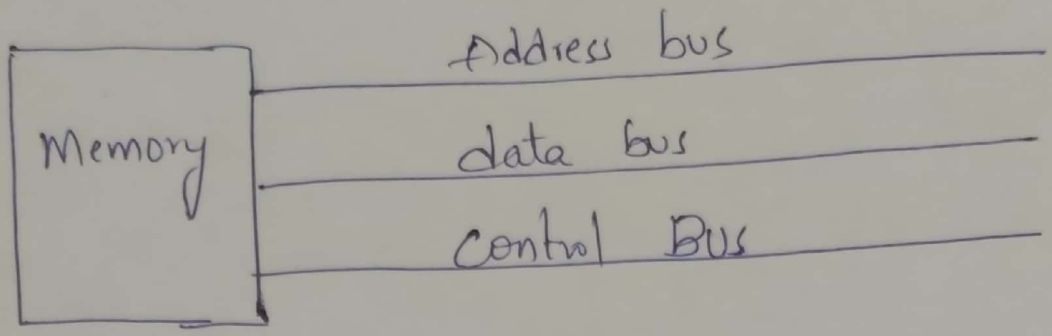
Isolated versus - Memory mapped I/O

Isolated I/O	Memory mapped I/O
• Seperate read/write control lines	A single set of read/write control lines
• Seperated memory and I/O address spaces	Common address space
• Distinct input & Output instructions	No specific input & output instructions
	considerable flexibility in handling I/O operations

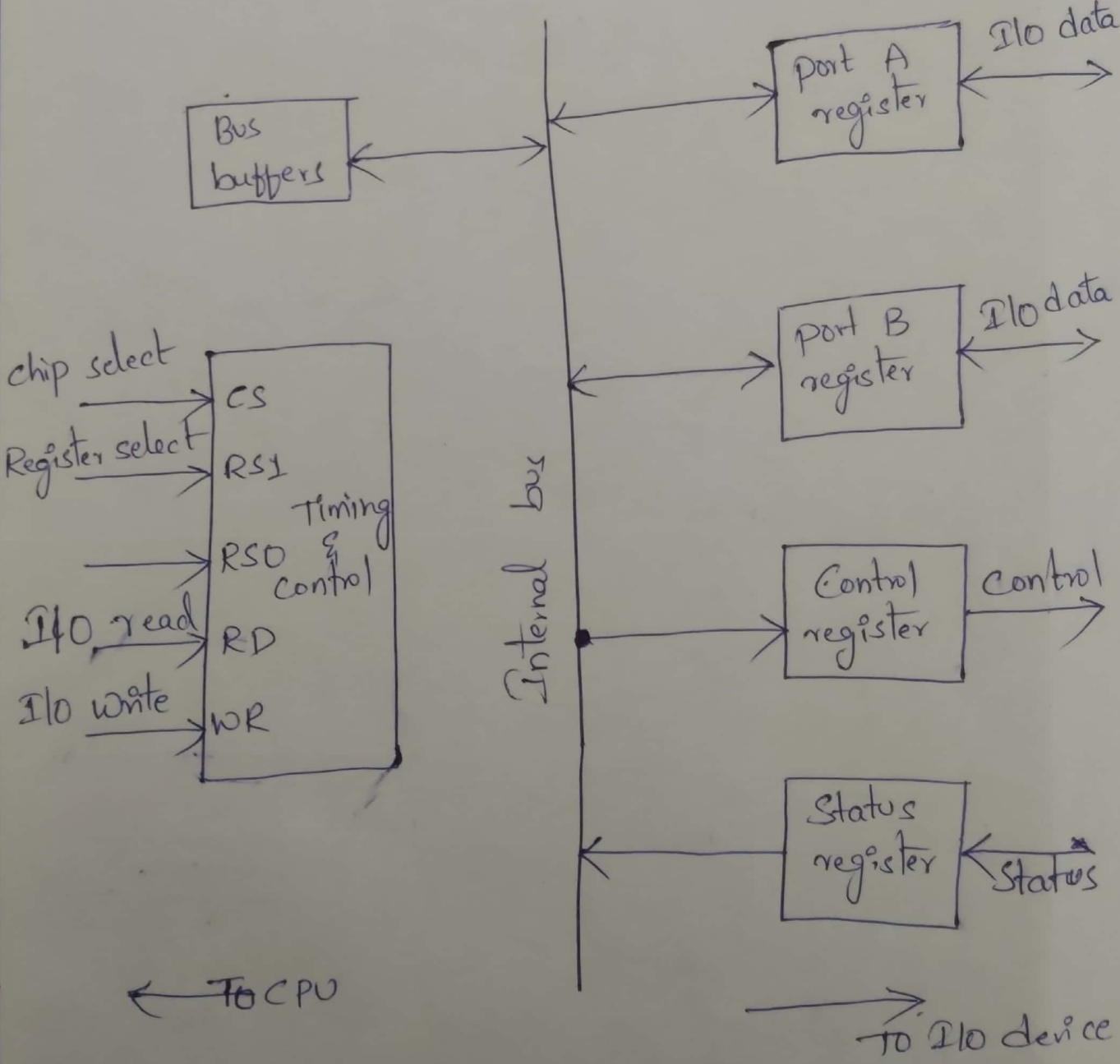
Isolated I/O :-



Memory mapped I/O



Example of I/O Interface :-



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

→ An example of an I/O interface unit is shown in block diagram. It consists of two data registers called ports, A control registers, status register, bus buffers, and timing and control circuits.

→ The interface communicates with the CPU through the data bus. The chip select and register select inputs determine the address assigned to the interface.

→ The I/O read and write are two control lines that specify an input & output.

→ The I/O data to and from the device can be transferred into either port A & port B.

→ Status information is received from the status register; and the data are transfered to and from ports A & B registers.

Asynchronous data transfer :-

→ The internal operations in a digital system are synchronized by means of clock pulses supplied by a common pulse generator.

→ But two units such as CPU and I/O interface, are designed independently of each other.

→ In most cases the internal timing in each unit is independent from the other in that each uses its own private clock for internal registers.

→ In that case, the two units are said to be asynchronous to each other. This approach widely used in most computer systems.

→ Asynchronous data transfer between two independent units require that control signals to be transmitted blw communicating units.

→ There are two ways to achieving asynchronous data transfer.

1. Strobe control method
2. Handshaking method.

Strobe Control :-

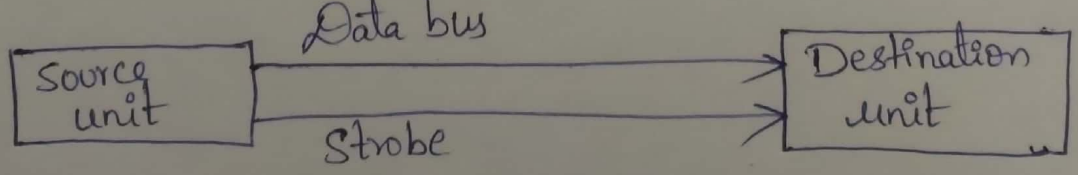
→ The strobe control method of asynchronous data transfer employs a single control line to time each transfer.

→ The strobe may be activated by either the source or the destination unit.

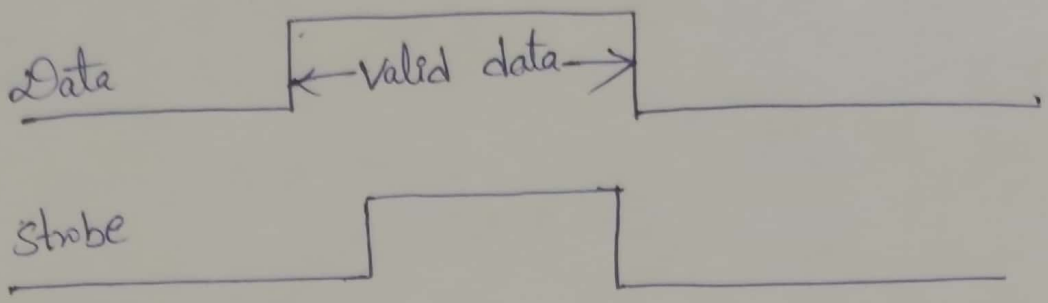
→ the below figure shows a source-initiated transfer

→ The data bus carries the binary information from source unit to the destination unit.

→ The bus has multiple lines to transfer an entire byte a word. The strobe is a single line that informs the destination unit when a valid data word is available in the bus.



(a) Block diagram.



(b) Timing diagram.

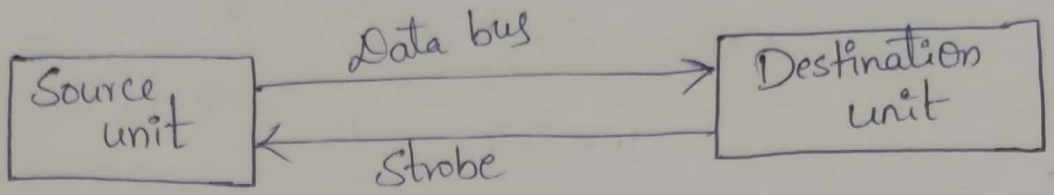
Source-initiated strobe for data transfer

→ As shown in the timing diagram, the source unit first places the data on the data bus.

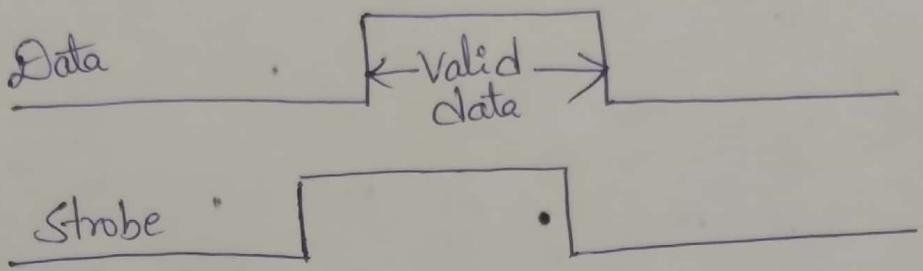
→ After a brief delay to ensure that the data settle to a steady value, the source activates the strobe pulse.

→ The source removes the data from the bus a brief period after it disables its strobe pulse. New valid data will be available only after the strobe is enabled again.

Destination-initiated strobe for data transfer:-



(a) Block diagram



(b) Timing diagram

→ The above diagram shows a data transfer initiated by the destination unit.
→ In this case the destination unit activates strobe pulse, informing the source to provide the data.
→ The source unit ~~provide~~ responds by placing the requested binary information on data bus.
→ The data must be valid and remain in the bus long enough for the destination unit to accept it.
→ The destination unit then disables the strobe.
→ The source removes the data from the bus after a predetermined time interval.

Handshaking approach :-

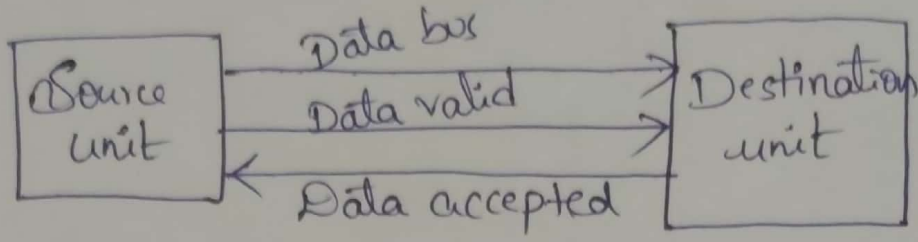
→ The disadvantage of the strobe method is that the source unit that initiates the transfer has no way of knowing whether the destination unit has actually received the data items that was placed in the bus.

→ Similarly the destination unit that initiates the transfer has no way of knowing whether the source unit has actually placed in data on the bus.

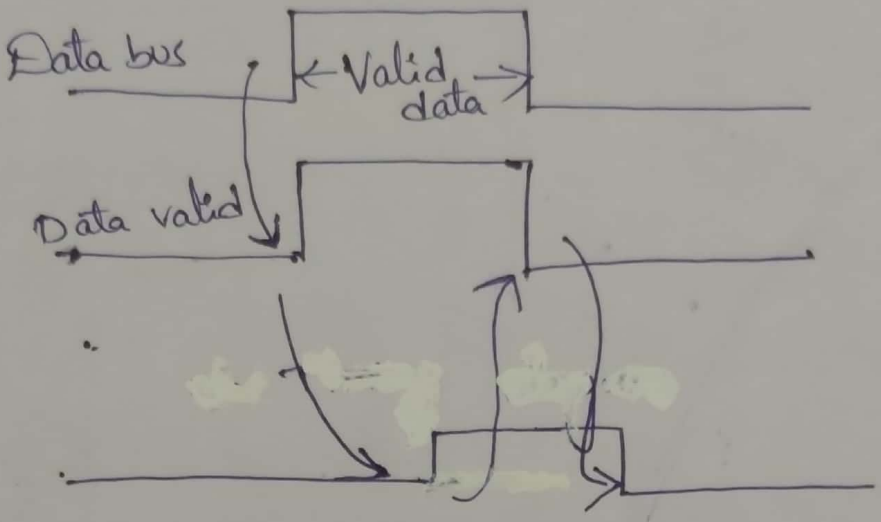
→ The handshake method solves this problem by introducing a second control that provides a reply to the unit that initiates the transfer.

→ The data transfer procedure when initiated by the source. The two handshaking lines are data valid, which is generated by the source unit, and data accepted, generated by the destination unit.

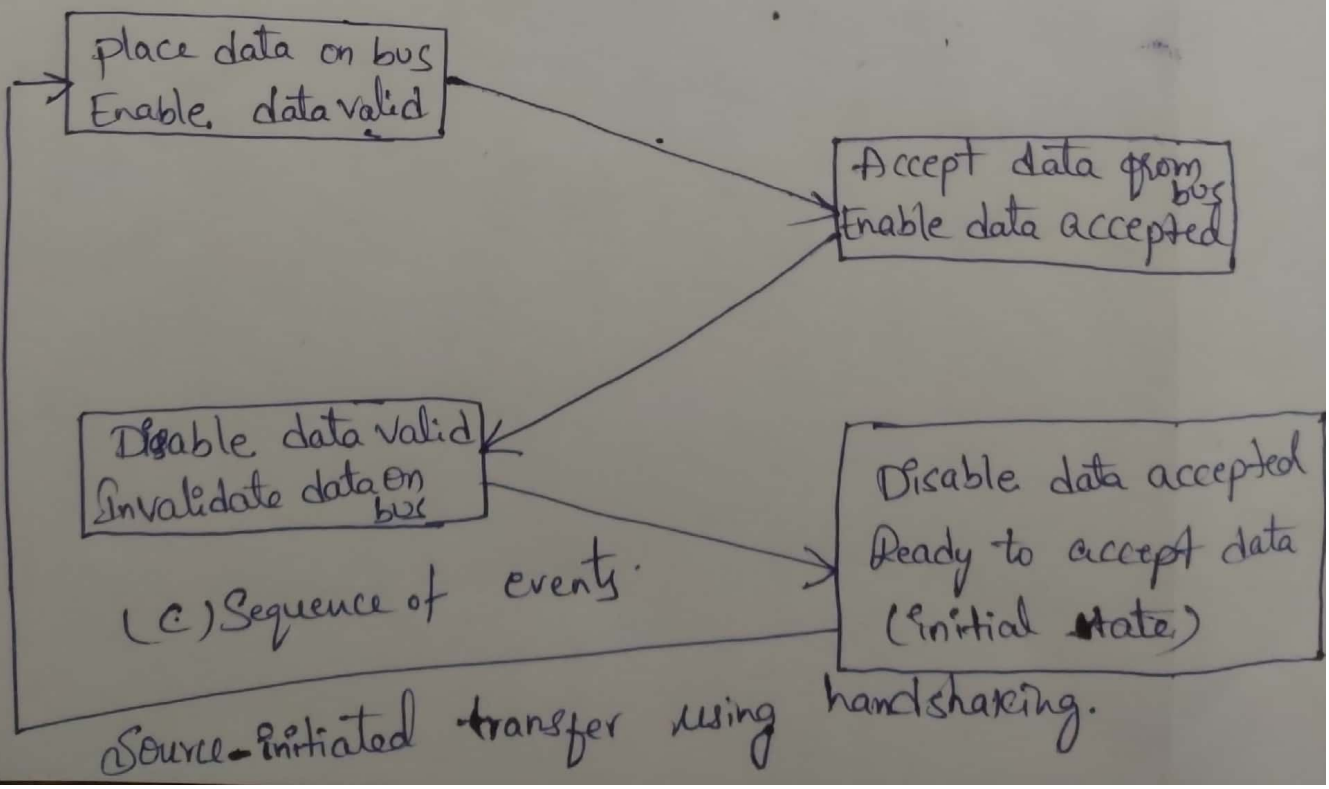
→ The timing diagram shows the exchange of signals between the two units.



(a) Block diagram



(b) Timing diagram.



(c) Sequence of events.

Source-initiated transfer using handshaking.

→ The sequence of events listed in part (c) shows the four possible states that the system can be at any given time.

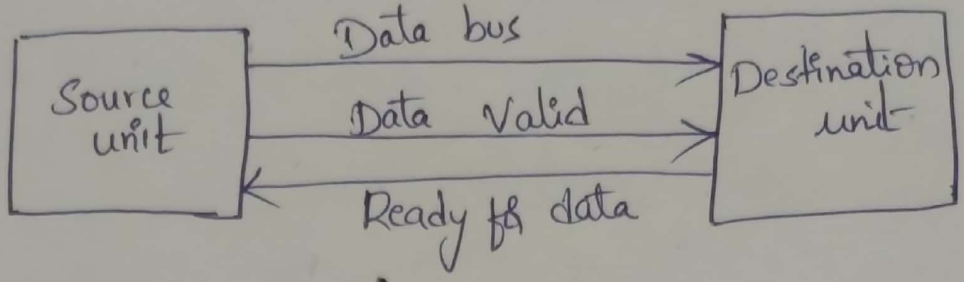
→ The source unit initiates the transfer by placing the data on bus and enabling its data valid signal.

→ The data accepted signal is activated by the destination unit after it accepts the data from the bus.

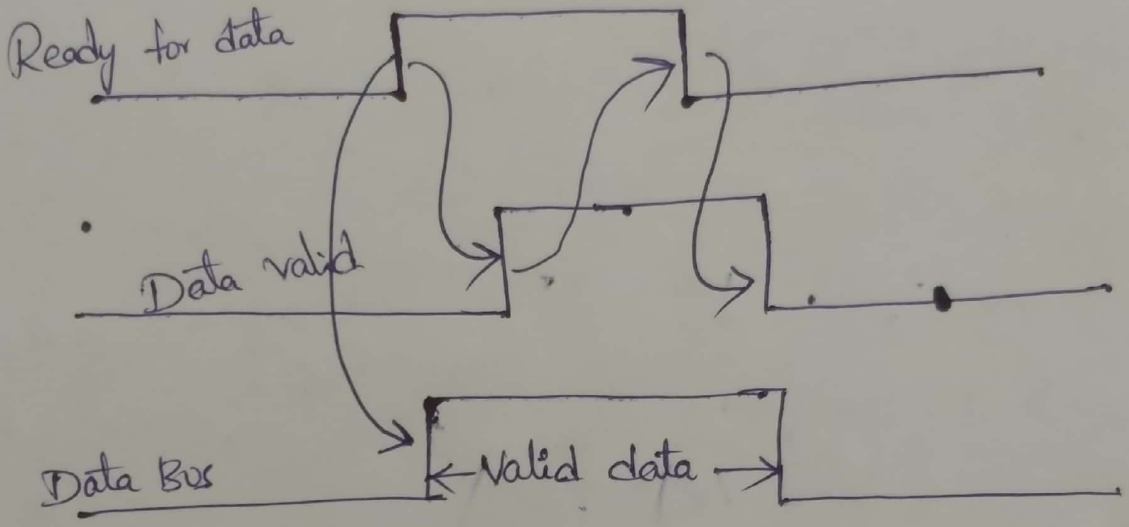
→ The destination unit then disables its data accepted signal and the system goes into initial state.

→ The source does not send the next data item until after the destination unit shows its readiness to accept new data by disabling its data accepted signal.

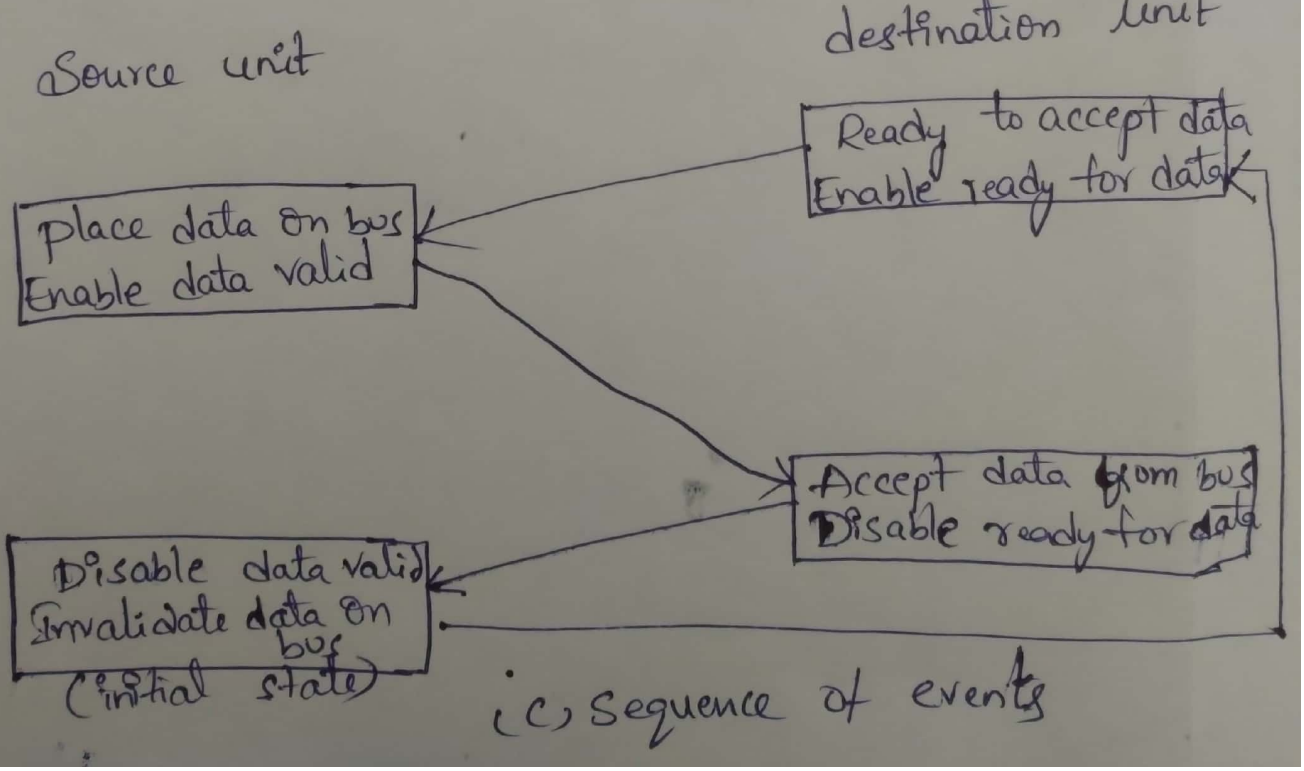
Destination-initiated transfer using handshaking



(a) block diagram



(b) Timing diagram.



(c) Sequence of events

→ The handshaking scheme provides a high degree of flexibility and reliability because the successful completion of a data transfer relies on active participation by both units.

→ the signal generated by the destination unit has been changed to ready for data to reflect its new meaning

→ The source unit in this case does not place data on the bus until after it receives the ready for data signal from the destination unit.

→ From there on, the handshaking procedure will follows the same pattern as in the source initiated case.

→ The only difference between the source-initiated and the destination-initiated transfer is in their choice of initial state.

Asynchronous Serial Transfer :-

→ The transfer of data between two units may be done in parallel & serial.

→ In parallel data transmission, each bit of the message has its own path and the total message must be transmitted through n seperate conductor paths.

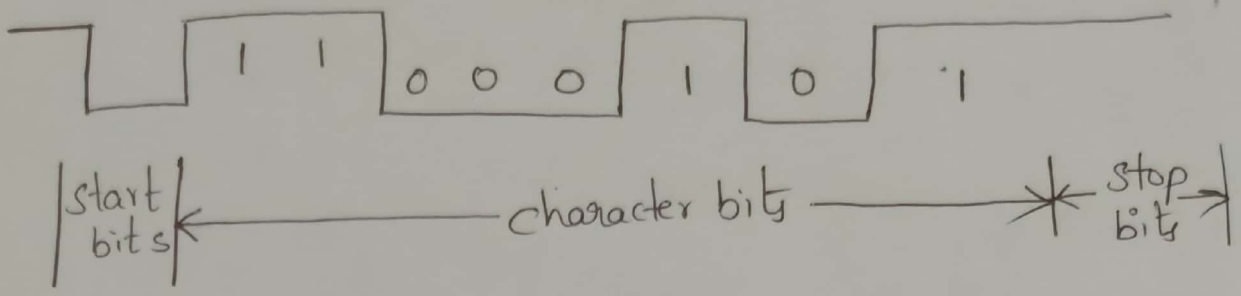
→ In serial data transmission, each bit in the message is sent in sequence one at a time.

→ This method requires the use of one pair of conductors & one conductor and a common ground.

→ Parallel transmission is faster but requires many wires. It is used for short distances and where speed is important.

→ Serial transmission is slower but is less expensive since it requires only one pair of conductors.

→ Serial transmission can be Synchronous & asynchronous.



Asynchronous Serial transmission

→ A transmitted character can be detected by the receiver from knowledge of the transmission rule

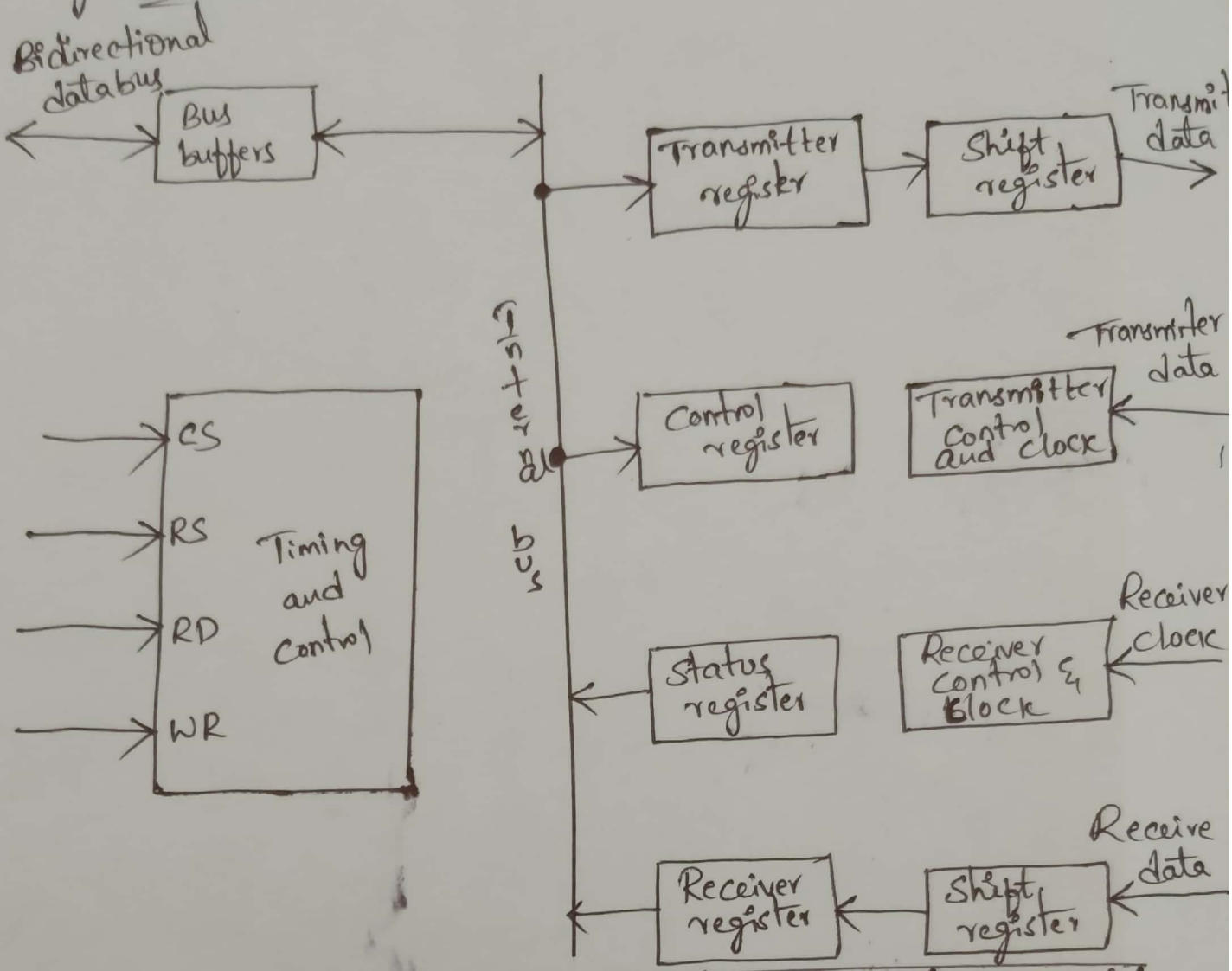
1) When a character is not being sent, the line is kept in the 1-state.

2) The initiation of a character transmission is detected from the start bit, which is always 0.

3) The character bit always follow the start bit.

4) After the last bit of the character is transmitted, a stop bit is detected when the line returns to the 1-state for at least one bit time.

Asynchronous Communication Interface :-



CS	RS	operation	Register selected
0	X	X	None: data bus in high impedence
1	0	WR	Transmitter register
1	1	WR	Control register
1	0	RD	Receiver register
1	1	RD	Status register

→ The block diagram of an asynchronous communication interface is shown in figure.
→ It functions as both transmitter and a receiver.
→ The transmitter register accepts a data byte from the CPU through the data bus.
→ This byte is transferred to a shift register for serial transmission.
→ The operation of the asynchronous communication interface is initialized by the CPU by sending a byte to the control register.
→ The CPU can select the receiver register to read the byte through the data bus.
→ The bits in the status register are used for input and output flags and for recording certain errors that may occur during the transmission. The CPU can read the status register to check the status of the flag bits.
→ The chip select (CS) input is used to select the interface through the address bus.
→ The (RS) Register select is associated with the (RD) read and write (WR) controls.

Modes of Transfer :-

Data transfer between the central computer and I/O devices may be handled in a variety of modes.

→ 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 (DMA)

Programmed I/O :-

In the programmed I/O method, the I/O device does not have direct access to memory.

→ It requires the execution of several instructions by the CPU.

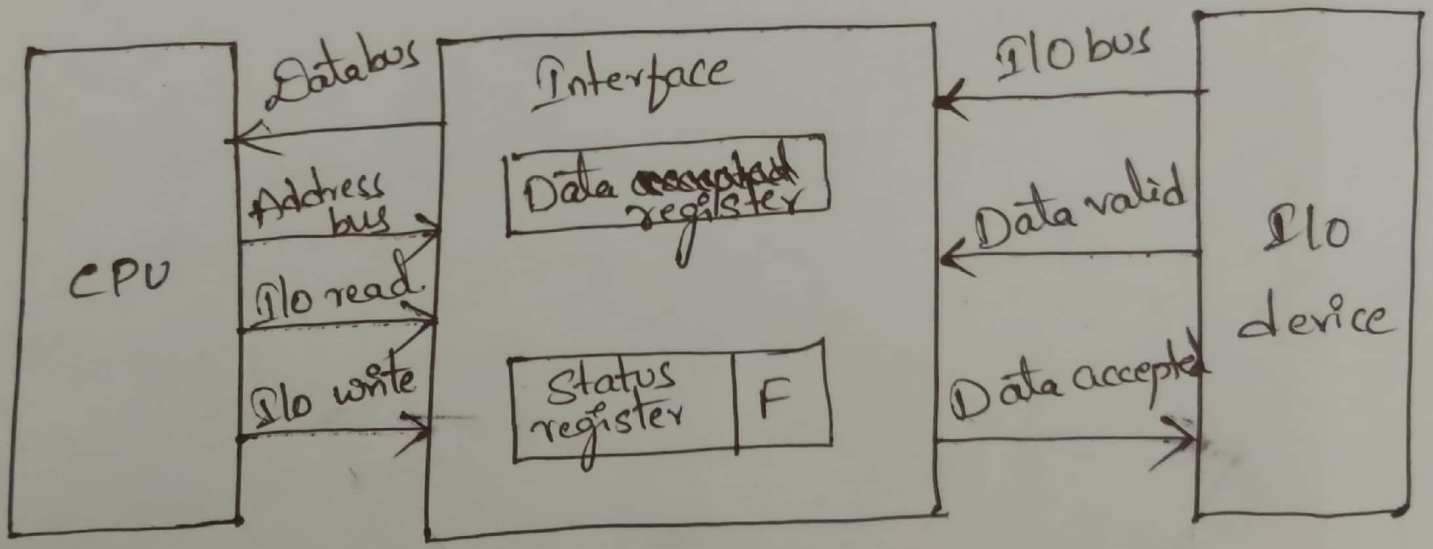
→ The device transfers byte of data one at a time as they are available

→ When a byte of data is available, the device places it in the I/O bus and enables its data valid line.

→ The Interface accepts the byte into its data register and enables the data accepted line.

→ The Interface sets a bit in the status register that we will refer to as an I or "flag" bit.

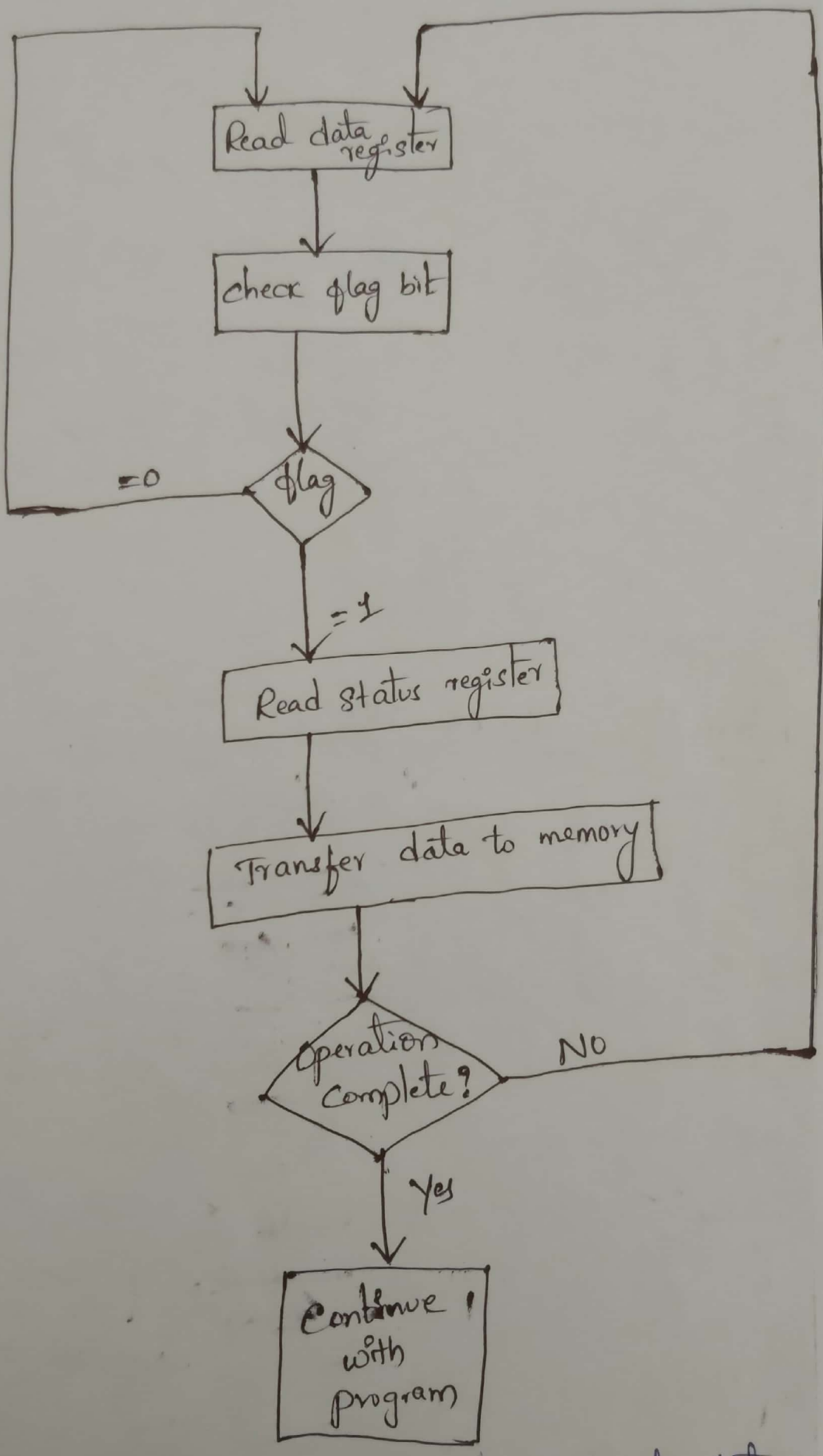
→ The device can now disable the data valid line but it will not transfer another byte untill the data accepted line disabled by the interface.



F = flag bit

fig:- Data transfer from I/O device to CPU.

Flowchart :-



-Flowchart for CPU program to input data

→ A flowchart of the program that must be written for the CPU is shown in fig.
→ It is assumed that the device is sending a sequence of bytes that must be stored in memory
→ The transfer of each byte requires three instructions

1. Read the status register
2. Check the status of the flag bit and branch to step 1 if not set & to step 3 if set.
3. Read the data register.

Interrupt - Initiated I/O :-

→ Instead of continous monitoring of CPU, interface will be informed to issue an interupt request signal.
→ Meanwhile, CPU proceeds to execute another program and the interface keeps monitoring the device.
→ When device is ready for data transfer, it generates interupt request.
→ Here the CPU stops the task it is performing processes the data transfer then resumes the Riginal task.

→ There are various sources of interrupts; those could be both internal and external.

Program interrupts :- They are generated by some condition that occurs as a result of an instruction execution.

Timer Interrupts :- They are generated within the processor, and allow the OS to perform certain operations on regular basis.

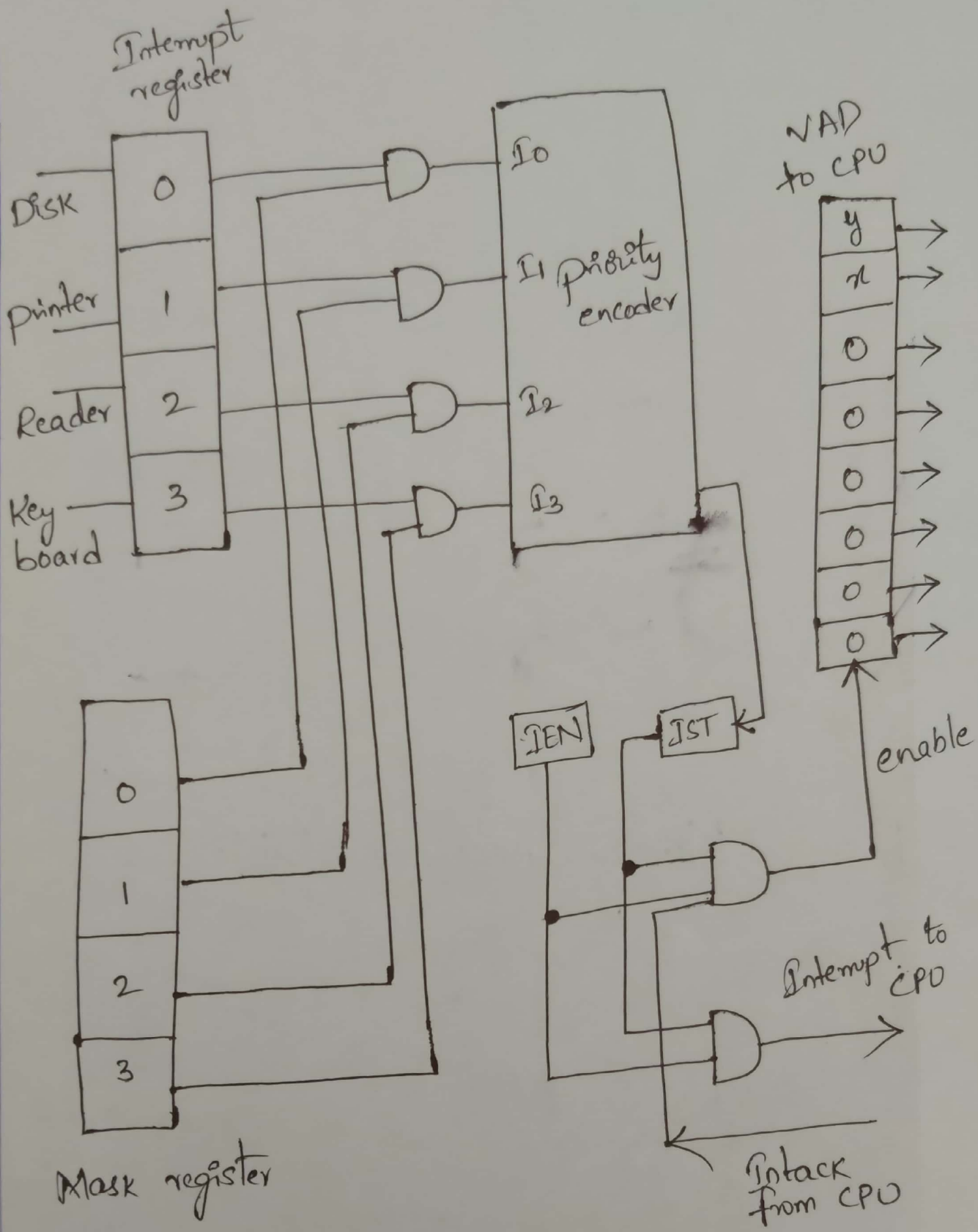
I/O Interrupts :- They are generated for initiation & completion of I/O operation.

Hardware failure interrupts :- They are generated by failure, such as power failure or memory priority error.

Priority Interrupt :-

Data transfer between the CPU and an I/O device is initiated by the CPU.

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

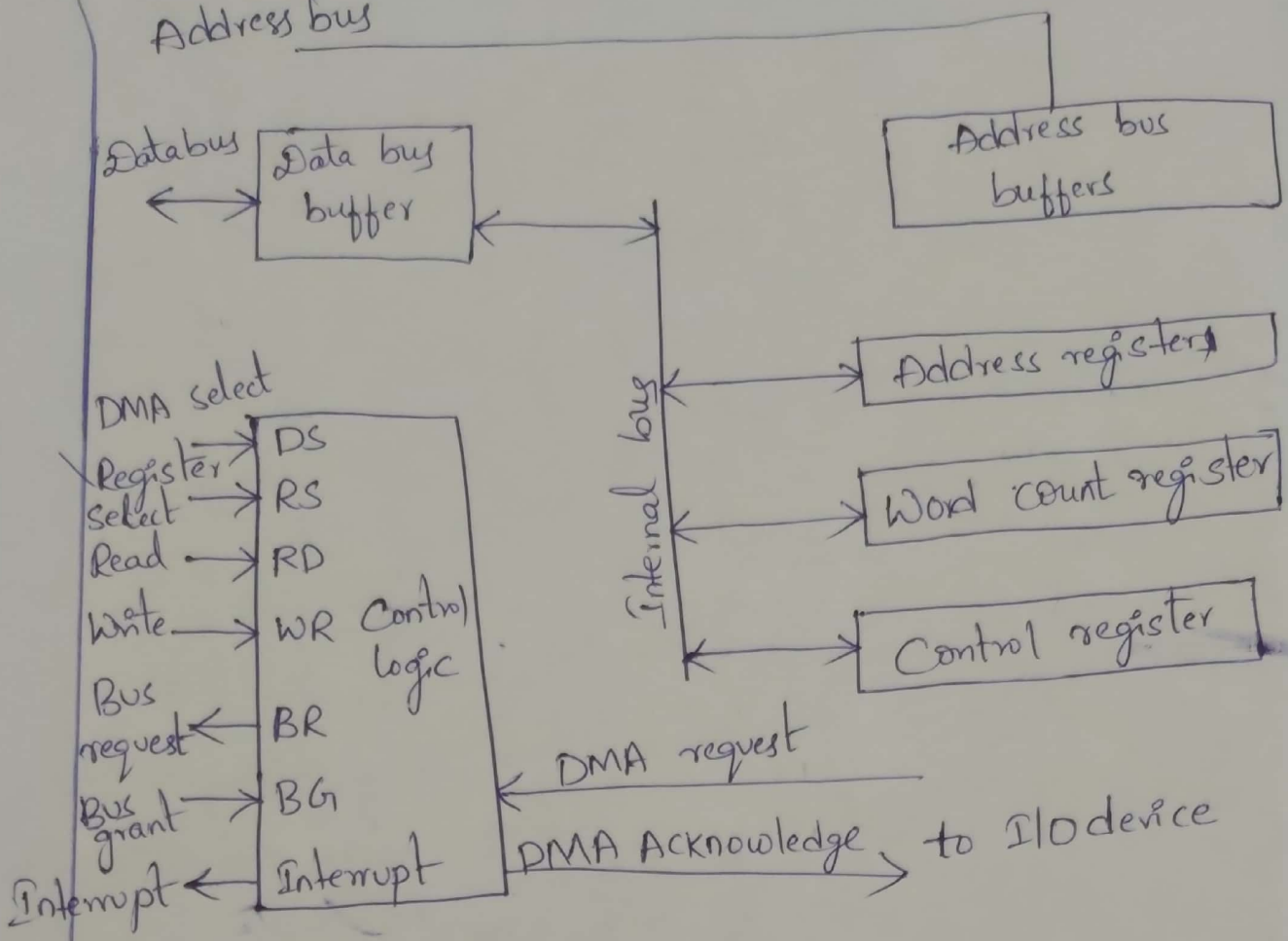


Priority interrupt hardware

→ The priority logic for a system of four interrupt sources is shown in figure.
→ It consist of an interrupt register whose individual bits are set by external conditions and cleared by program instructions.
→ The magnetic disk, being a high-speed device, is given the highest priority.
→ The printer has the next priority, followed by a character reader and a keyboard.
→ The mask register has the same number of bits as the interrupt register.

Direct Memory Access :- (DMA) :-

→ The transfer of data between a fast storage device such as magnetic disk and memory is often limited by the speed of the CPU.
→ Removing the CPU from the path and letting the peripheral device manage the memory buses directly would improve the speed of transfer.
→ This transfer technique is called direct memory access.



Block diagram of DMA controller

→ The unit communicates with the CPU via the data bus and control lines.

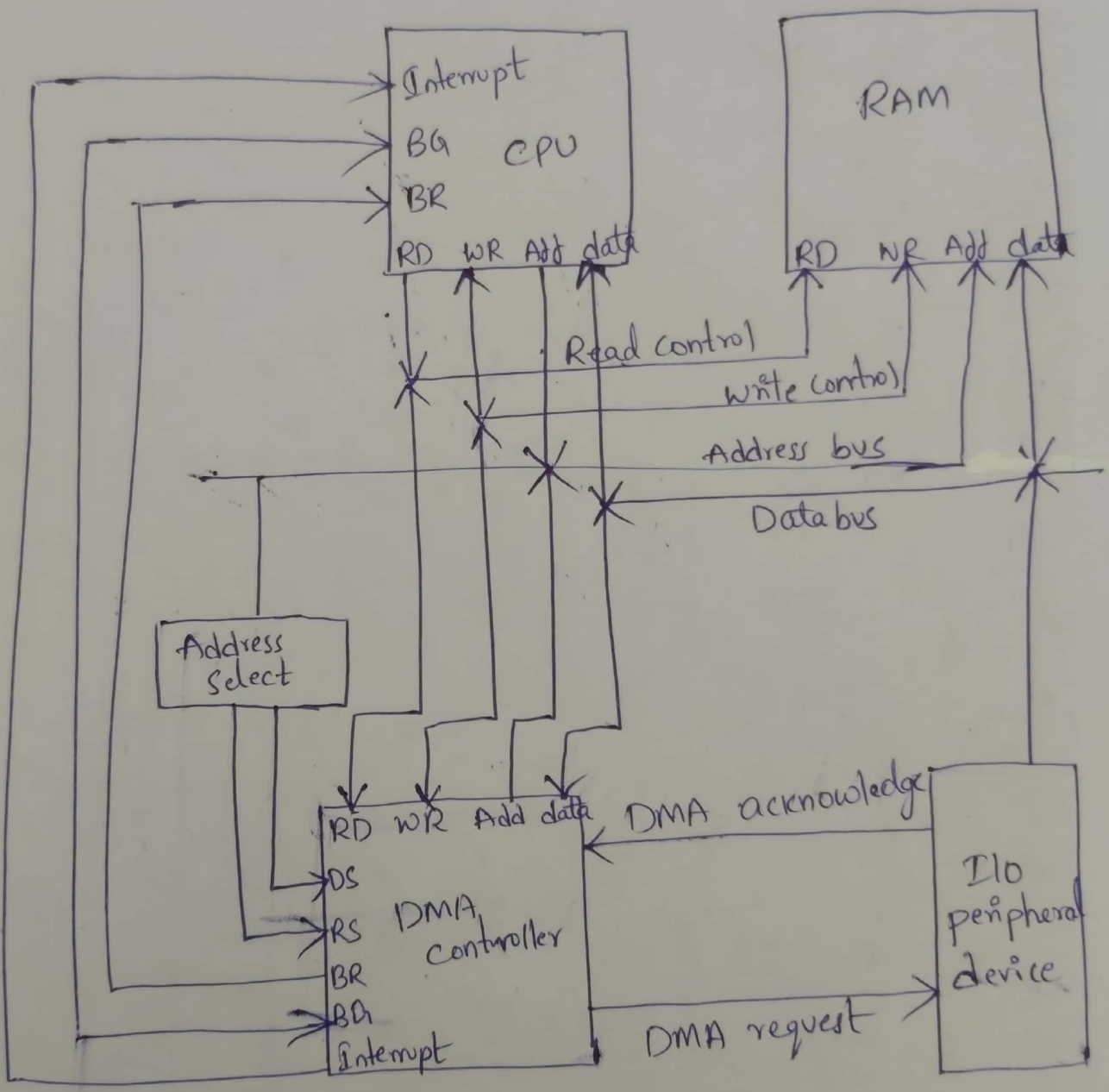
→ The registers in the DMA are selected by the CPU through the address bus by enabling the DS (DMA select) and RS (Register select) inputs.

→ The RD (read) and WR (write) inputs are bidirectional.

→ When BG (bus grant) input is 0, the CPU can communicate with the DMA registers through the data bus to read from & write to DMA registers.

When BG=1, the CPU has the RD and WR are output lines from the DMA controller to the random access memory to specify the read or write operation for the data.

DMA transfer :-



DMA transfer in a Computer System

→ The CPU communicates with the DMA through the address and data buses as with any interface unit.

→ When the peripheral device sends a DMA request, the DMA controller activates the BR line, informing the CPU to relinquish the buses.

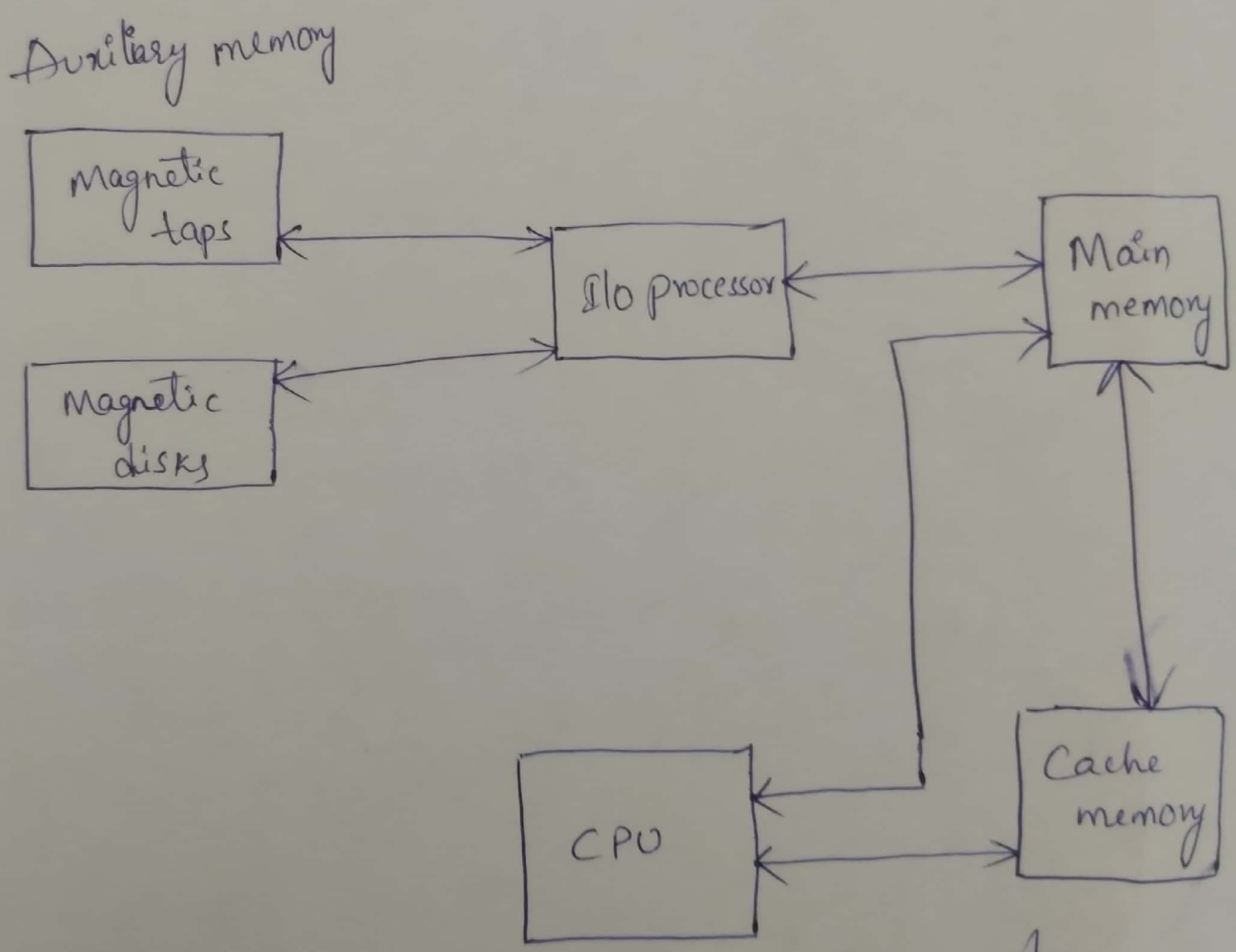
→ The CPU responds with its BG line, informing the DMA that its buses are disabled.

→ The DMA then puts the current value of its address register into the address bus, initiates the RD or WR signal, and sends a DMA acknowledge to the peripheral device.

Memory Organization :-

- Memory hierarchy
- Main memory
- Auxiliary Memory
- Associative Memory
- Cache memory.

* Memory hierarchy *:-



Memory hierarchy in a Computer System

→ The memory hierarchy diagram shows the components in a typical memory hierarchy.

Auxiliary memory :-

→ At the bottom of the hierarchy are the relatively slow magnetic tapes used to store removable files.

→ Next are the magnetic disks used as backup storage.

Main memory :-

→ The main memory occupies a central position by being able to communicate directly with the CPU and with auxiliary memory devices through an I/O Processor.

→ When ~~main~~ main memory need to execute the program, they are brought in from auxiliary memory

Cache memory :- A special very-high-speed memory called a cache memory. The cache is used for storing segments of programs currently being executed in the CPU.

* Main Memory *

The main memory is the central storage unit in a computer system. It is a relatively large and fast memory used to store programs and data during the computer operation.

→ The principal technology used for the main memory is based on semiconductor integrated circuits.

→ Integrated circuit RAM chips are available in two possible operating modes, static and dynamic.

→ static RAM stores the binary information in flipflops.

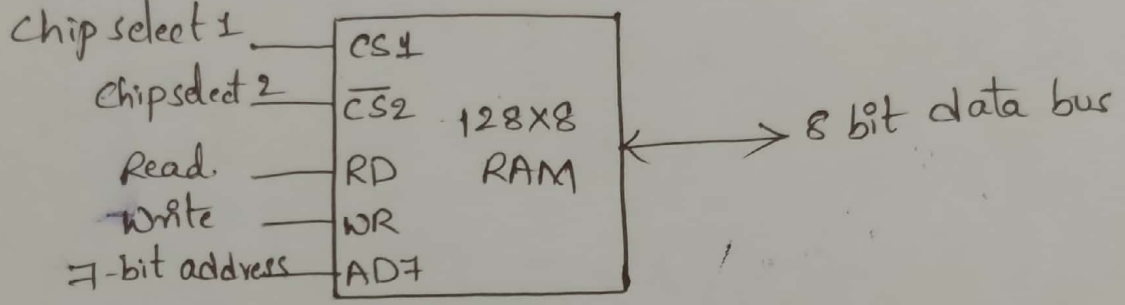
→ dynamic RAM stores the binary information in the form of electric charges that applied to capacitors.

→ most of the main memory in a general-purpose computer is made up of RAM integrated circuit chips, but a portion of memory may be constructed with ROM chips.

→ Among other things, the ROM portion of main memory is needed for storing an initial program called a bootstrap loader.

→ The startup of a computer consists of turning the power on and starting the execution of an initial program.

RAM and ROM chips:-



(a) block diagram of RAM

CS1	CS2	RD	WR	memory function	State of the Bus
0	0	X	X	Inhibit	High-Impedence
0	1	X	X	Inhibit	High-Impedence
1	0	0	0	Inhibit	High-Impedence
1	0	0	1	Write	Input data to RAM
1	0	1	X	Read	Output data from RAM
1	1	X	X	Inhibit	High-Impedence

(b) function table

Fig:- Typical RAM chip.

→ The block diagram of a RAM chip is shown in fig. The capacity of the memory is 128 words of eight bits (one byte) per word.

→ This requires a 7-bit address an 8-bit bidirectional data bus.

→ read and write inputs specify the memory operation and the two chips select (CS) control inputs are for enabling the chip only when it is selected by the microprocessor.

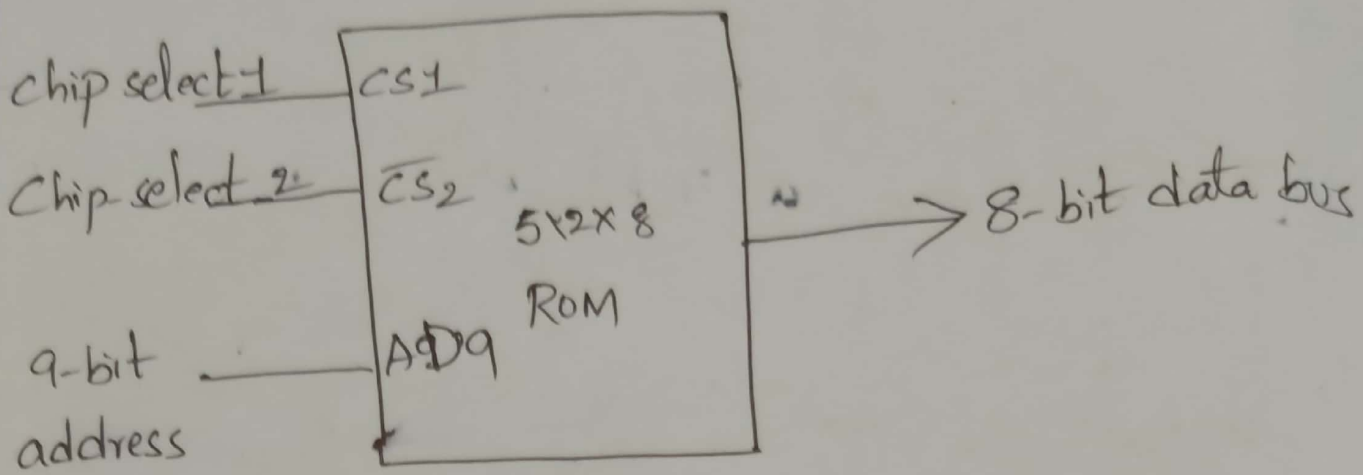
→ The fution table listed in table. Specifies the operation of the RAM chip.

→ The unit is in operation only when CS1=1 and $\overline{CS2}=0$. The bar on top of the second select variable indicates that this input is enabled when it is equal to 0.

→ If the chip select inputs are not enabled, & if they are enabled but read & write inputs are not enabled, the memory is inhibited and its data bus is in high-impedence state.

→ When $\overline{CS2}=0$, CS1 = 1, the memory can be placed in a read and write mode.

ROM chip:-



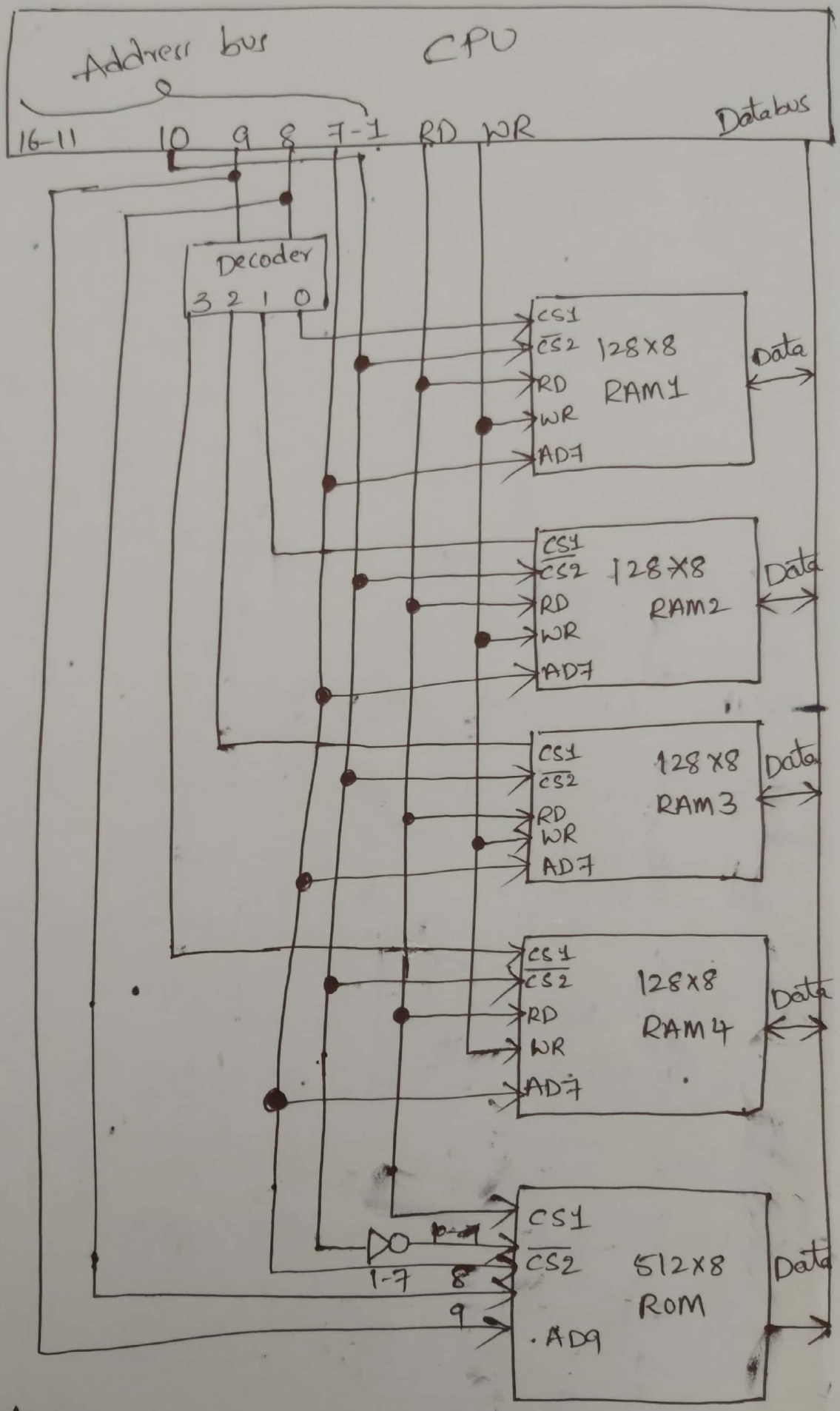
Typical ROM chip

→ The ROM chip has 512 bytes. and need 9 address lines.

→ the two chip select inputs must be CS1 = 1 and $\overline{CS2} = 0$ for the unit to cooperate. Otherwise the data bus is in a high impedence state.

→ there is no need for a read & write control because the unit can only read.

Memory connection to CPU :-



Memory Connection to the CPU

→ The connection of memory chips to the CPU is shown in diagram.
→ This configuration gives a memory capacity of 512 bytes of RAM and 512 bytes of ROM.
→ The particular RAM chip selected is determined from lines 8 and 9 in the address bus.
→ This is done through a 2×4 decoder whose outputs go to the CS1 inputs in each RAM chip.
→ When address lines 8 and 9 are equal to 00, the first RAM chip is selected, when 01, the second RAM chip is selected and so on.
→ The RD and WR outputs from the microprocessor are applied to the inputs of each RAM chip.
→ The selection between RAM and ROM is achieved through bus line 10.
→ The RAMs are selected when the bit in this line is 0, and the ROM when bit is 1.
→ Address buslines 1 to 9 are applied to the input address of ROM.

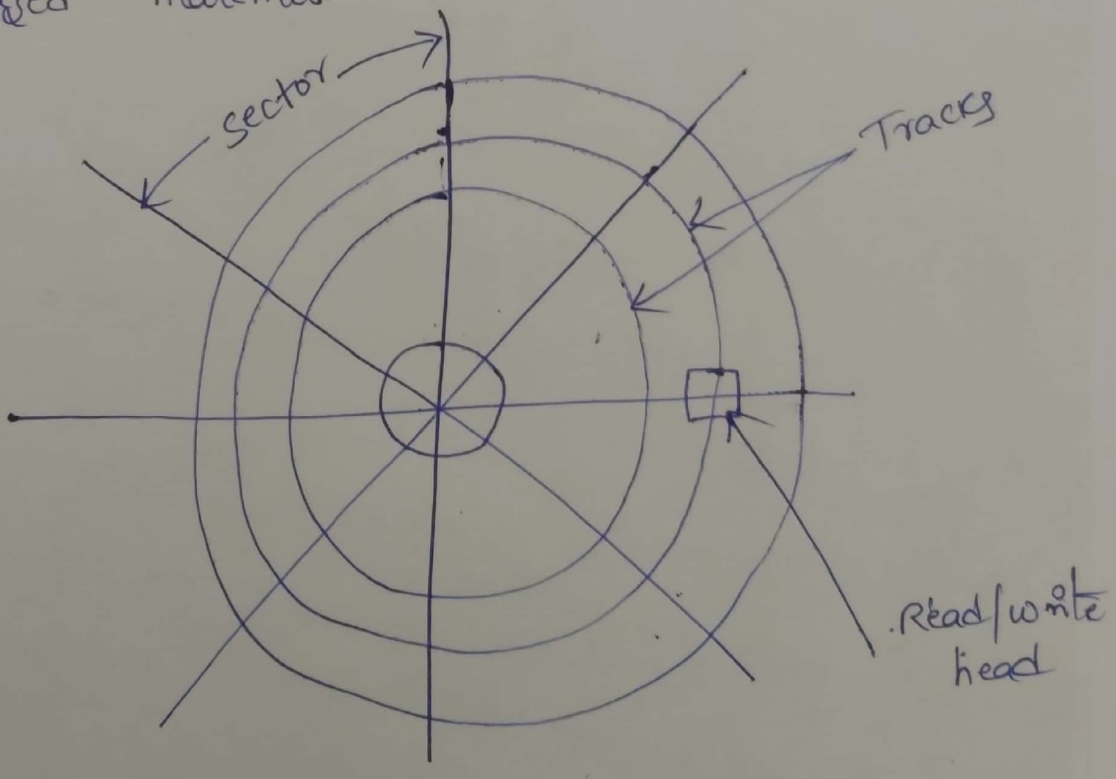
Auxiliary Memory:-

→ The most common auxiliary memory devices used in computer systems are magnetic disks and tapes.

→ The physical properties of these storage devices can be quite complex, the important characteristics of any device are its access mode, access time transfer rate, capacity and soft.

Magnetic Disks

A magnetic disk is a circular plate constructed of metal & plastic coated with magnetized material.



Magnetic disk

→ Several disks may be stacked on one spindle with read/write heads available on each surface.
→ All disks rotate together at high speed and are not stopped & started for access purposes.
→ Bits are stored in the magnetized surface in spots along concentric circles called tracks.
→ Tracks are commonly divided into sections called sectors.
→ the minimum quantity of information which can be transferred is a sector.
→ Disks that are permanently attached to the unit assembly and cannot be removed by the occassional user are called hard disks.
→ A disk drive with removable disks is called a floppy disk.

Magnetic tape :- A magnetic tape transport consist of the electrical, mechanical and electronic components to provide the parts and control mechanism for a magnetic tape unit. magnetic tapes can be stopped, started to move forward.

Associative Memory :-

→ TO Search particular data in memory, data is read from certain address and compared if the match is not found content of the next address is accessed and compared.

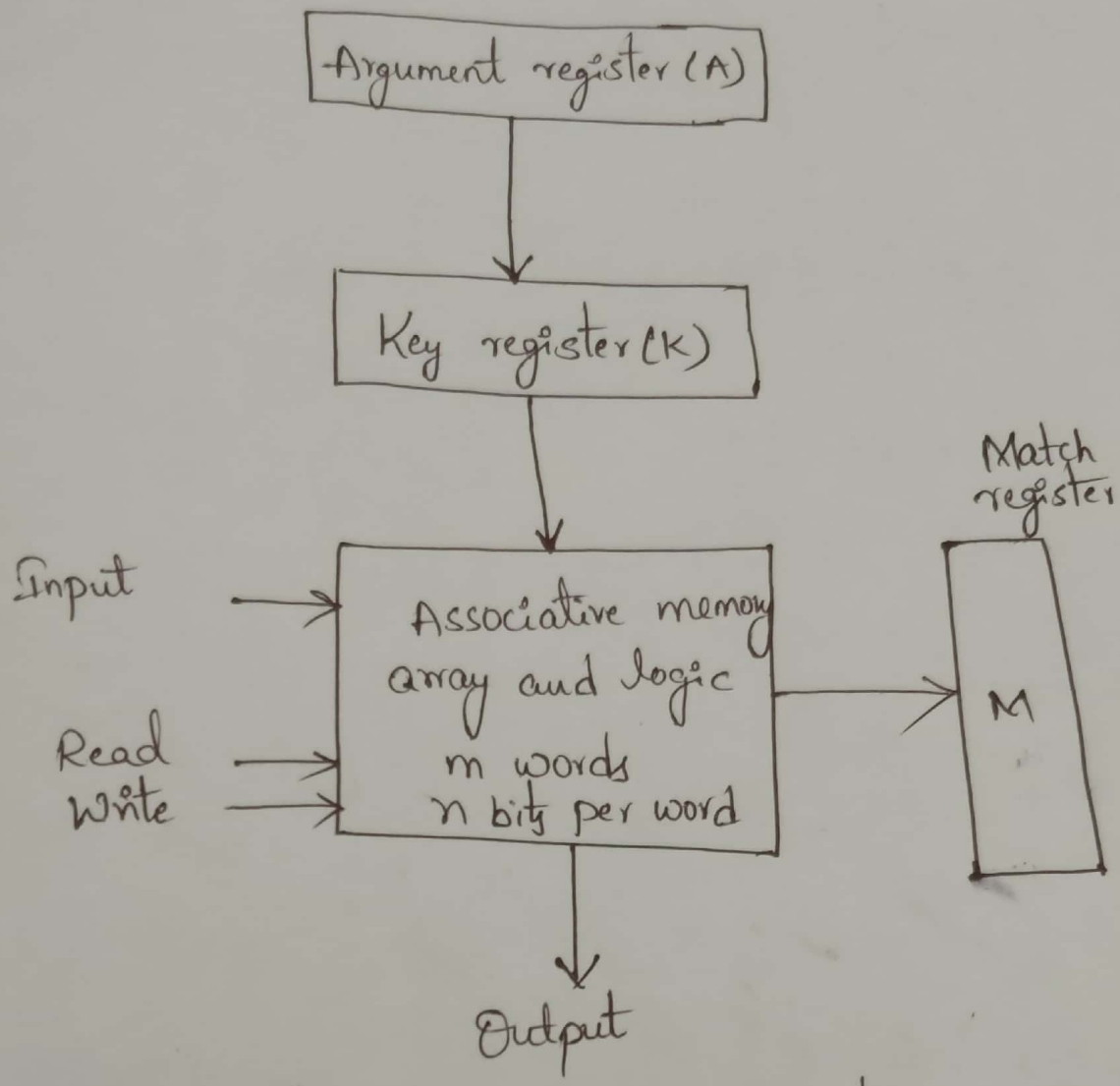
→ This goes on untill required data is found. The number of access depend on the location of data and efficiency of searching algorithm.

→ This Searching time can be reduced if data is ~~cleared~~. searched on the basis of content.

→ A memory unit accessed by content is called associative memory & content addressable memory (CAM) & associative storage & associative array.

→ This type of memory accessed simultaneously and in parallel on the basis of data content.

→ Associative memory contain Argument register, Key Register, Associative memory array and Match register.



Block diagram of associative memory

Argument register :- It contains the word to be searched. It has n bits.

Key register :- This specifies which part of the argument word needs to be compared with words in memory. If all bits in register are 1, the entire word should be compared. Otherwise, only the bits having K-bit set to 1 will be compared.

Associative memory array:- It contains the words which are to be compared with the argument word.

Match register:-

It has m bits, one bit corresponding to each word in the memory array. After the matching process, the bits corresponding to matching words in match register are set to 1.

Example

A	101	111100	
K	111	000000	
Word 1	100	111100	No match
Word 2	101	000001	match.

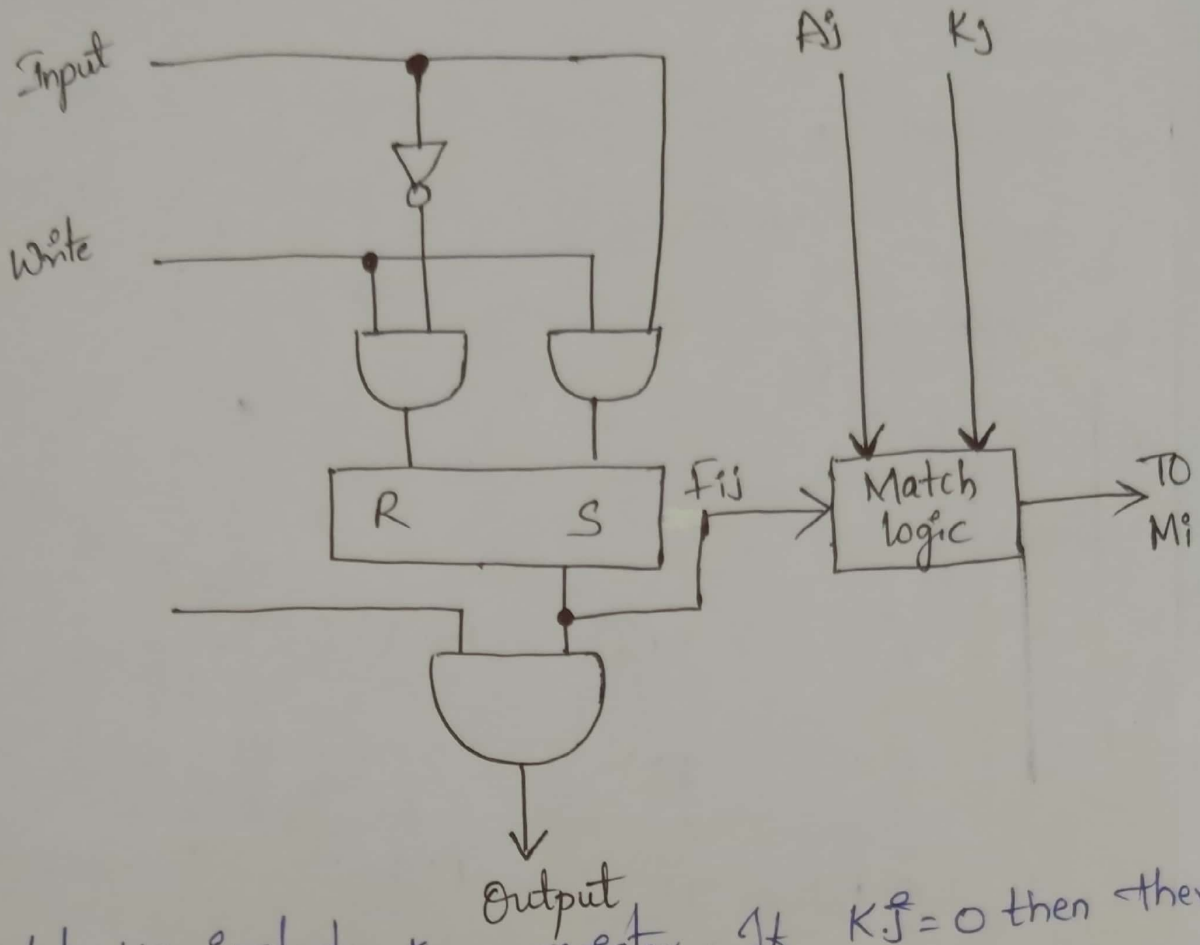
→ Key register provide the mask for choosing the particular field in A register.

→ The entire content of A register is compared if Key register content all 1.

→ Otherwise only bit that have 1 in Key register are compared.

→ If the compared data is matched corresponding bits in the match register are set.

Hardware implementation of one cell associative memory



→ Let us include key register. If $K_j = 0$ then there is no need to compare A_j and F_{ij}.

→ Only when $K_j = 1$, comparision is needed.

⇒ This achieved by ORing each term with K_j.

$$M_i = (x_1 + K_1)(x_2 + K_2')(x_3 + K_3')\ldots(x_n + K_n')$$

Write operation :-

If the entire memory is loaded with new information at once prior to search operation then writing can be done by addressing each location in sequence.

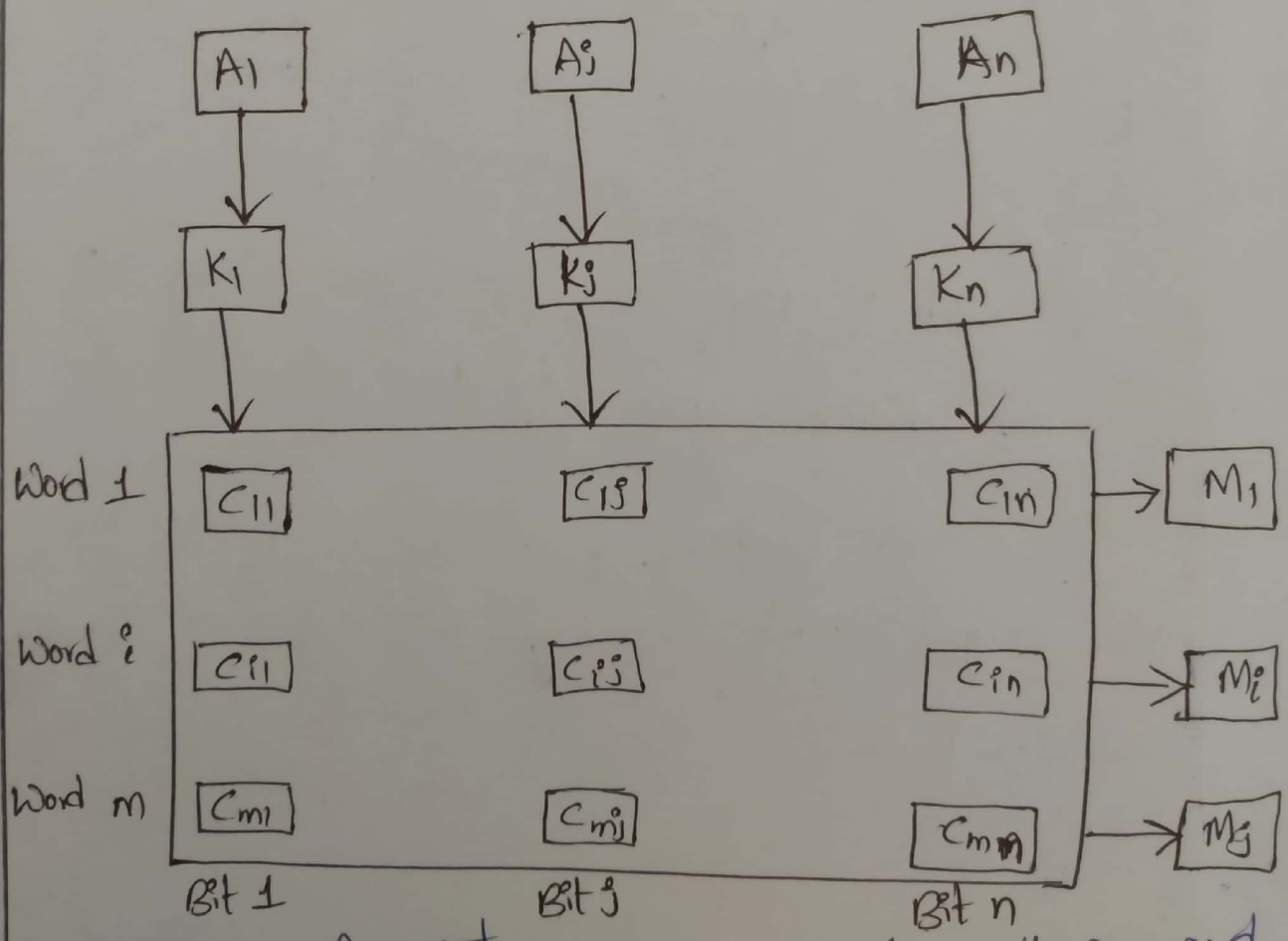
→ Tag register contain as many bits as there core words in memory.

→ It contain 1 for active word and 0 for inactive word.

→ If the word is to be inserted, tag register is scanned until 0 is founded and word is written at that position and bit is change to 1.

Read operation :-

When a word is to be read from an associative memory., the content of the word, or a part of the word is specified.



Associative memory m word, n cells per word

Advantages :-

→ This is suitable for parallel searches.

→ To speed up data bases, in neural networks and page tables used by the virtual memory.

Disadvantages :-

→ More expensive than a RAM

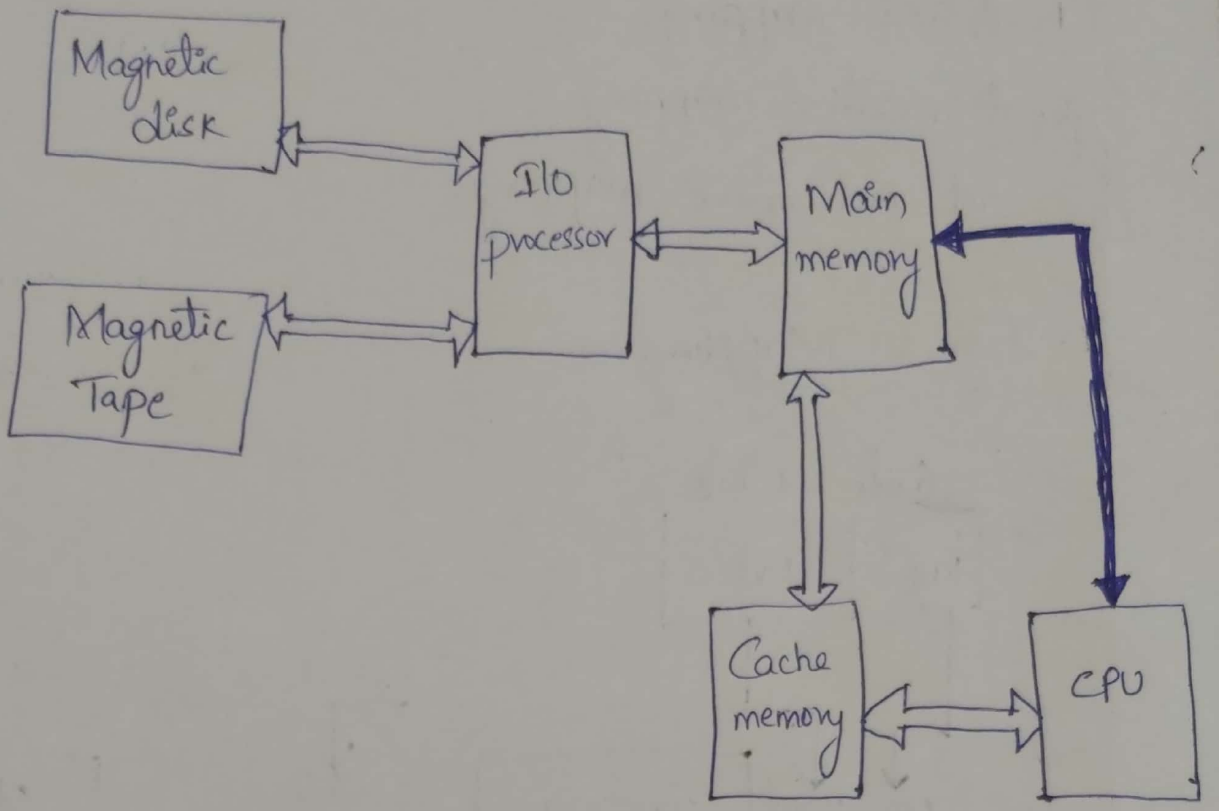
→ These memories are used in applications where the search time is very critical and must be very short.

Cache Memory :-

→ Cache memory is a small, high speed RAM buffer located between the CPU and main memory.

→ It holds a copy of the instructions & data currently being used by the CPU.

→ The main purpose of a cache is to accelerate your computer while keeping the price of the computer low.

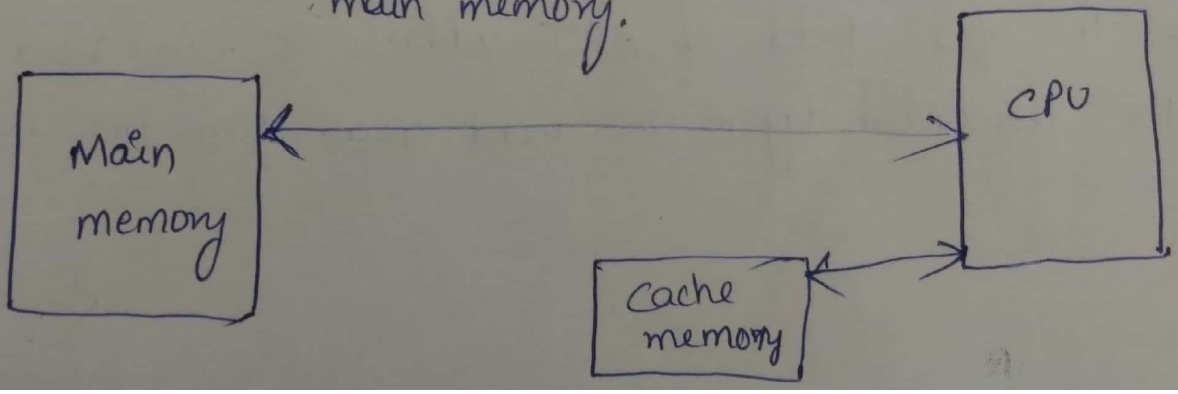


Placement of cache in computer

Hit ratio :- The ratio of the total no-of hits divided by the total CPU accesses to memory is called Hit ratio.

Hit ratio = Total no-of hits / (Total no-of hits + Total no-of miss).

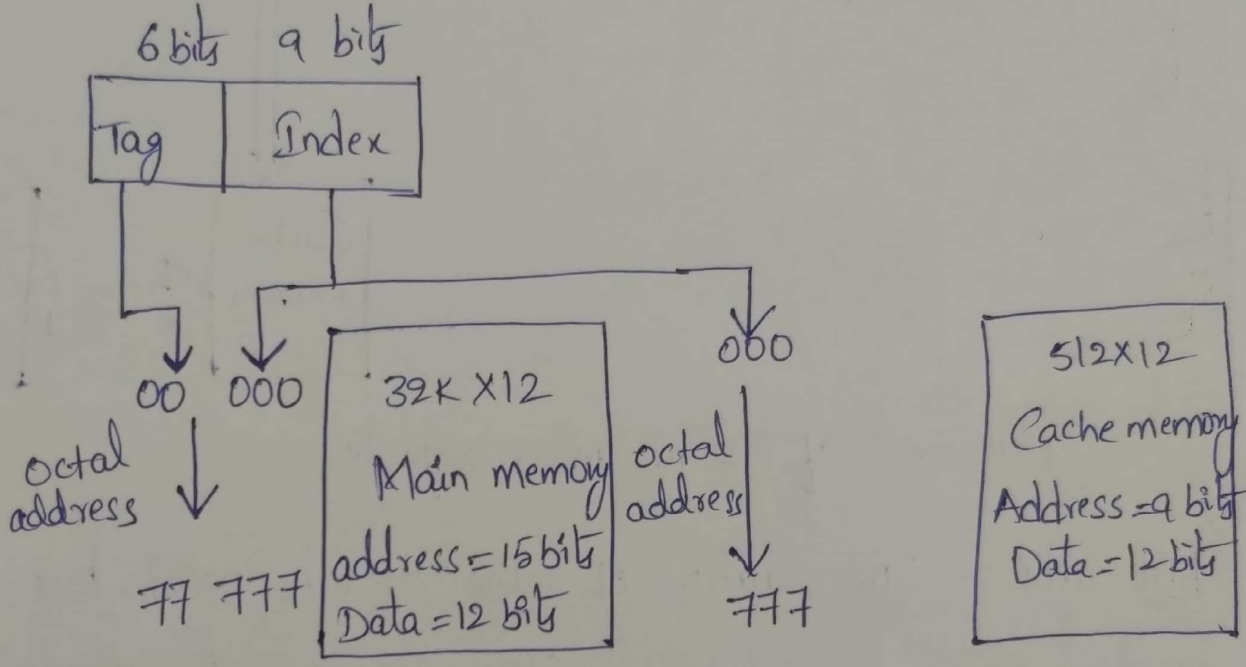
example :- A system with 512X12 cache and 32K X12 of main memory.



Types of cache mapping :-

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

1. Direct Mapping :-



→ The direct mapping technique is simple and inexpensive to implement.

→ When the CPU wants to access data from memory it places a address. the Index field of CPU address is used to access address.

→ The tag field of CPU address is compared with the associated tag in the word read from the cache

→ If the tag bits of CPU address is matched with the tag-bits of cache, then there is a hit and the required data word is read from cache.

→ If there is no match, then there is a miss and the required data word is stored in main memory

→ It is then transferred from main memory to cache memory with the new tag.

Example:-

Main memory

Address	Data
00 000	5670
00 777	7523
01 000	1256
01 777	5321
67 125	7432
77 777	5432

Cache memory

Index	Tag	Data
000	00	5670
777	00	7523
000	01	1256
125	51	1560
777	77	5432

2. Associative Mapping :-

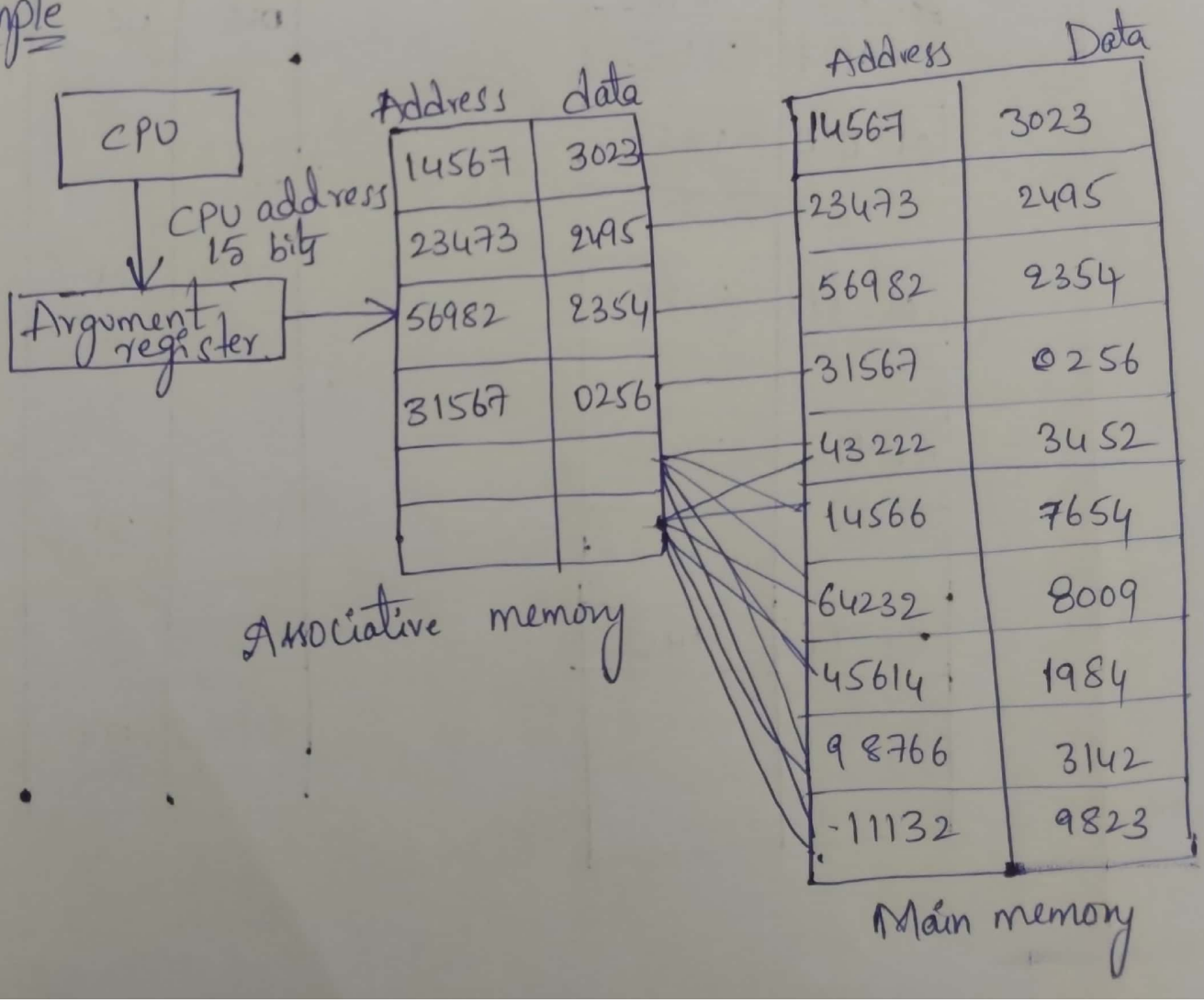
→ An associative mapping uses an associative memory

→ This memory is being accessed using its contents

→ Each line of cachememory will accommodate the address (main memory) and the contents of that address from the main memory.

→ That is why this memory is also called Content Addressable Memory (CAM). It allows each block of memory to be stored in the cache.

Example



3. Set Associative Mapping :-

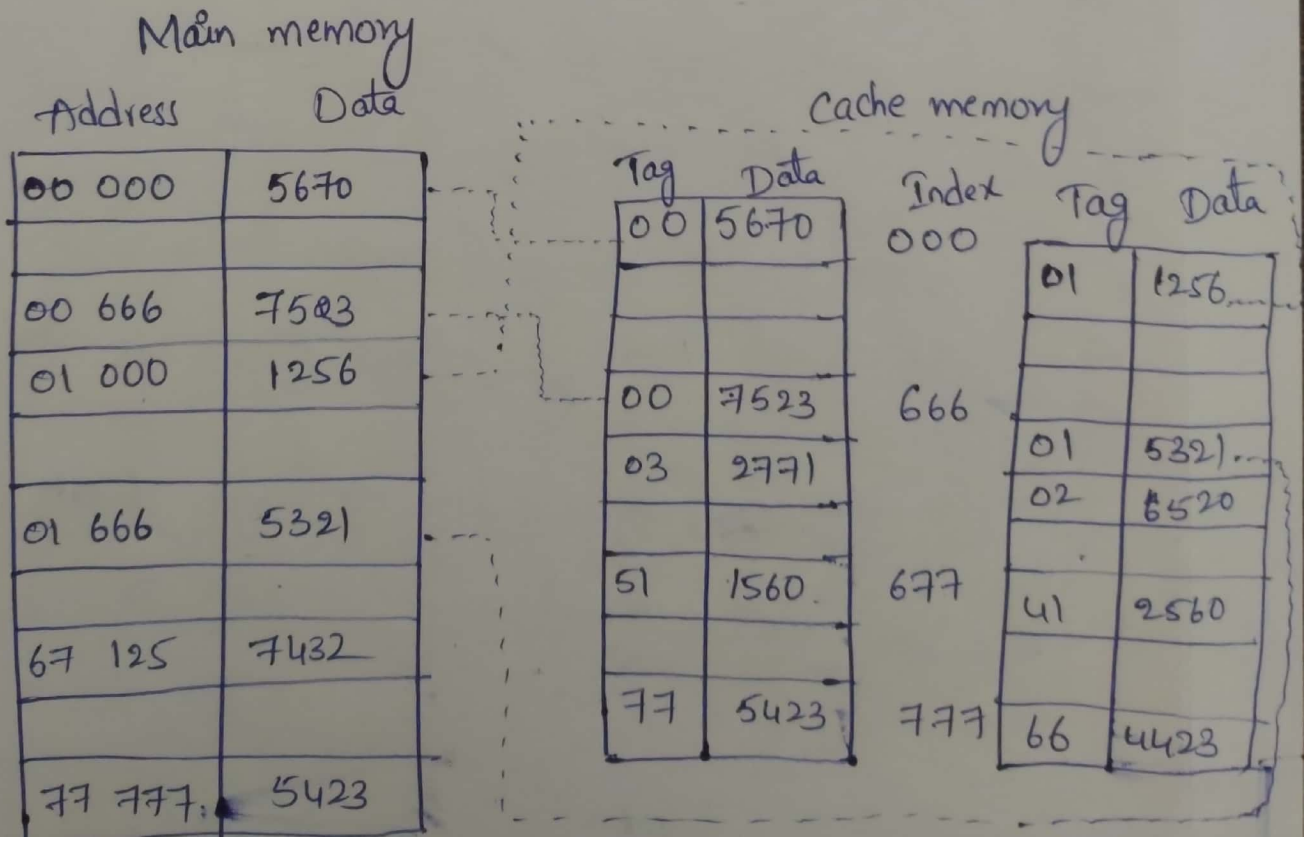
→ The each control of the direct mapping cache and the more flexible mapping of the fully associative cache.

→ In set associative mapping, each cache location can have more than one pair tag + data items.

→ That is more than one pair of tag and data are residing at the same location of cache memory.

→ In one cache location is holding two pair of tag + data items, that is called 2-way set associative mapping.

Example



Writing into cache :-

When memory write operations are performed, CPU first writes into the cache memory. These modifications made by CPU during a write operation, on the data saved in cache, need to be written back to main memory & to auxiliary memory.

→ These two popular cache write policies are:

a) Write - Through

b) Write - Back.

a) Write-through :-

→ In write-through cache, the main memory is updated each time the CPU writes into cache.

→ The advantage of the write-through cache is that the main memory always contains the same data as the cache contains.

→ The characteristic is disirable in a system which uses direct memory access scheme of data transfer.

→ The IIO devices communicating through DMA receive the most recent data.

Write-Back :-

→ In write-Back scheme, only the cache memory is updated during a write operation.

→ The updated locations in the cache memory are marked by a flag so that later on, when the word is removed from the cache, it is copied into the main memory.

→ The words are removed from the cache time to time to make room for a new block of words.

* Unit 4 completed *.