UNIT V

MEMORY AND I/O SYSTEMS

Memory hierarchy - Memory technologies – Cache basics – Measuring and improving cache performance - Virtual memory, TLBs - Input/output system, programmed I/O, DMA and interrupts, I/O processors.

MEMEORY SYSTEM

- ✓ Programs and data are held in the main memory of the computer during execution.
- ✓ The execution speed of programs is dependent on the speed with instruction and data that can be transferred between the CPU and the main memory.
- ✓ It is also important to have a large memory to facilitate execution of programs that are large and deal with huge amounts of data.
- ✓ The memory would be fast, large, and inexpensive.
- ✓ Unfortunately, it is impossible to meet all three of these requirements simultaneously.
- ✓ Increases speed and size are archived at increases cost.
- ✓ To solve this problem, to develop a clever structure the improve the speed and size of the memory, yet keep the cost reasonable.

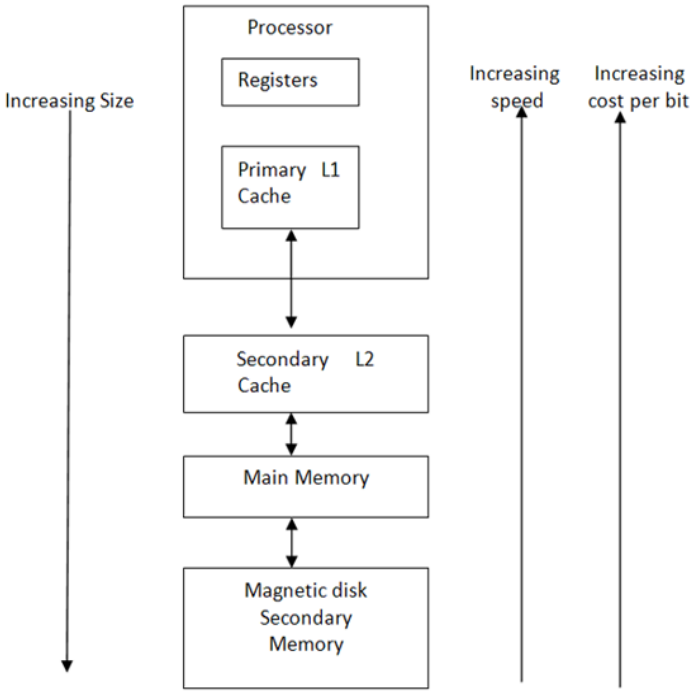
.

SPEED, SIZE, AND COST

- ✓ An ideal memory would be fast, large, and inexpensive.
- ✓ A very fast memory can be implemented, if SRAM chips are used.
- ✓ But these chips are expensive because their basic cells have six transistors, which precludes packing a very large number of cells onto a single chip.
- ✓ Thus, for cost reasons, it is impractical to build a large memory using SRAM chips.
- ✓ The alternative is to use Dynamic RAM chips, which have much simpler basic cells and thus are much less expensive. But such memories are significantly slower.

- ✓ Although dynamic memory units in the range of hundreds of megabytes can be implemented at a reasonable cost, the affordable size is still small compared to the demands of large programs.
- ✓ A solution is provided by using secondary storage, mainly magnetic disks, to implement large memory spaces.
- ✓ Very large disks are available at a reasonable price, and they are used extensively in computer systems.
- ✓ But they are much slower than the semiconductor memory units. So we conclude the following,
    - A huge amount of cost-effective storage can be provided by magnetic disks.
    - A large, yet affordable, main memory can be built with dynamic RAM technology.
    - A smaller unit is the SRAM, where speed is more such as in cache memories.

- ✓ All of these different types of memory units are employed effectively in a computer.
- ✓ The entire computer memory can be viewed as the hierarchy shown in following figure.
- ✓ The access is faster in **processor registers** and they are at the top in memory hierarchy.

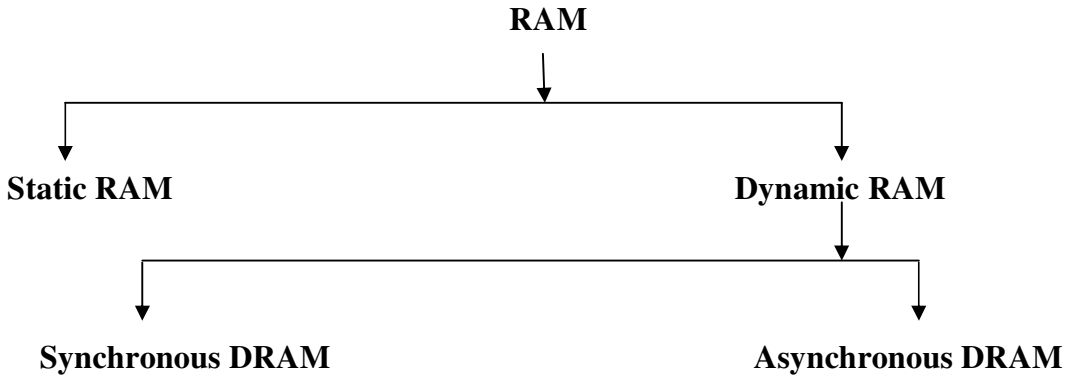


- ✓ At the next level of the hierarchy is a relatively small amount of memory that can be implemented directly on the processor chip.
- ✓ This memory, called a **processor cache,** holds copies of instructions and data stored in a much larger memory that is provided externally.
- ✓ There are two levels of caches,
  - A primary cache is always located on the processor chip. This cache is small because it competes for space on the processor chip, which must implement many other functions. The primary cache is referred to as *level 1* (LI) cache.
  - A larger, secondary cache is placed between the primary cache and the rest of the memory. It is referred to as *level* 2 (L2) cache. It is usually implemented using SRAM chips.

- ✓ The most common way of designing computers is to include a primary cache on the processor chip and a large secondary cache.
- ✓ The next level in the hierarchy is called the main memory.
- ✓ This rather large memory is implemented using dynamic memory components, typically in the form of SIMMs, DIMMs, or RIMMs.

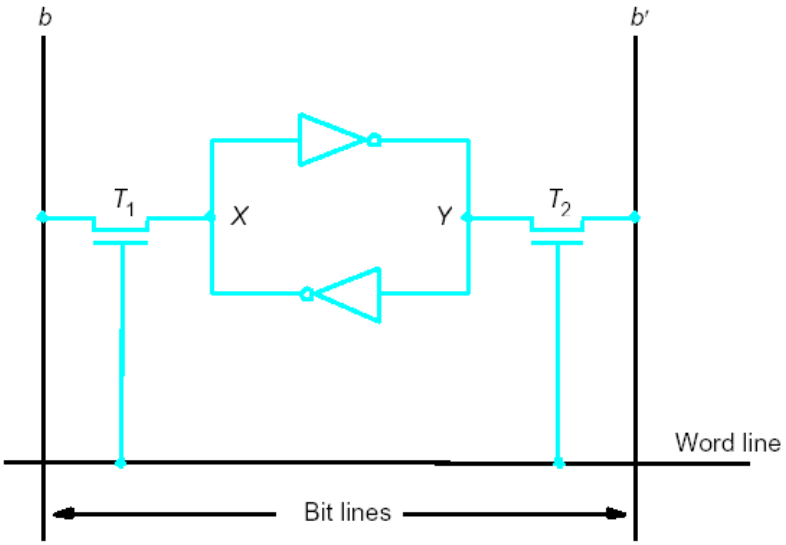
- ✓ The main memory is much larger but significantly slower than the cache memory.
- ✓ Magnetic Disk devices provide a huge amount of inexpensive storage. They are very slow compared to the semiconductor devices used to implement the main memory.

MEMORY TECHNOLOGIES



STATIC MEMORIES

- ✓ Memories that consist of circuits capable of retaining their state as long as power is applied are known as **static memories**.
- ✓ Below diagram illustrates how a static RAM (SRAM) cell may be implemented.



- ✓ Two inverters are cross connected to form a latch.
- ✓ The latch is connected to two bit lines by transistors T1 and T2.
- ✓ These transistors act as switches that can be opened or closed under control of the word line.
- ✓ When the word line is at ground level, the transistors are turned off and the latch retains its state.
- ✓ if the cell is in state 1 then the value at point X is 1.
- ✓ If the cell is in state 0 then the value at point Y is 0.
- ✓ This state is maintained as long as the signal on the word line is at ground level.

Read Operation:

- ✓ In order to read the state of the SRAM cell, the word line is activated to close switches T1 and T2.
  - If the cell is in state 1, then the signal on bit line b is high and the signal on bit line b' is low.
  - If the cell is in state 0, then the signal on bit line b is low and the signal on bit line b' is high.

Write Operation:

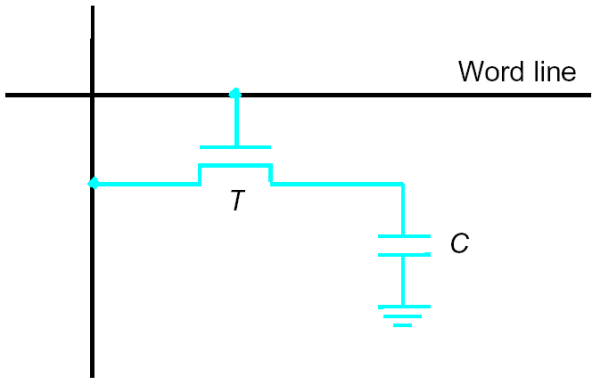
- ✓ The state of the cell is set by placing the appropriate value on bit line b and its complement on b', and then activating the word line.
- ✓ This forces the cell into the corresponding state.
- ✓ The required signal on the bit lines are generated by the Sense/Write circuit.

ASYNCHRONOUS DRAMS

- ✓ Static RAM are fast, but they come at a high cost because their cells require several transistors.
- ✓ Less expensive RAMs can be implemented if simpler cells are used.
- ✓ Such cells do not retain their state indefinitely; hence, they are called dynamic RAMs DRAMs).

- ✓ Information is stored in a dynamic memory cell in the form of a charge on a capacitor, and this charge can be maintained for only tens of milliseconds.
- ✓ Since the cell is required to store information for a much longer time, its contents must be periodically refreshed by restoring the capacitor charge to its full value.

For example, dynamic memory cell that consists of a capacitor, C and a transistor, T is shown below,



- ✓ In order to store information in this cell, transistor T is turned on and an appropriate voltage is applied to the bit line.
- ✓ This causes a known amount of charge to be stored in the capacitor.
- ✓ After the transistor is turned off, the capacitor begins to discharge.
- ✓ All cells in a selected row are read at the same time, which refreshes the contents of the entire row.

Refresh Circuit

- ✓ To ensure that the contents of a DRAM are maintained, each row of cells must be accessed periodically.
- ✓ A **refresh circuit** usually performs this function automatically.
- ✓ Many dynamic memory chips incorporate a refresh facility within the chips themselves.
- ✓ In this case, the dynamic nature of these memory chips is almost invisible to the user.

Timing Control

- ✓ In DRAM, the timing of the memory device is controlled asynchronously.
- ✓ A specialized memory controller circuit provides the necessary control signals, RAS and CAS that govern the timing.
- ✓ The processor must take into account the delay in the response of the memory.
- ✓ Such memories are referred to as **asynchronous DRAMs.**

SYNCHRONOUS DRAMS

- ✓ More recent developments in memory technology have resulted in DRAMs whose operation is directly synchronized with a clock signal.
- ✓ Such memories are known as **synchronous DRAMs (SDRAMs).**

SDRAM basics

- ✓ Traditional forms of memory including DRAM operate in an asynchronous manner.
- ✓ They react to changes as the control inputs change, and also they are only able to operate as the requests are presented to them, dealing with one at a time.
- ✓ SDRAM is able to operate more efficiently.
- ✓ It is synchronized to the clock of the processor and hence to the bus With SDRAM having a synchronous interface, it has an internal finite state machine that pipelines incoming instructions.
- ✓ This enables the SDRAM to operate in a more complex fashion than an asynchronous DRAM. This enables it to operate at much higher speeds.
- ✓ As a result of this SDRAM is capable of keeping two sets of memory addresses open simultaneously.
- ✓ By transferring data alternately from one set of addresses, and then the other, SDRAM cuts down on the delays associated with asynchronous RAM, which must close one address bank before opening the next.
- ✓ The term pipelining is used to describe the process whereby the SDRAM can accept a new instruction before it has finished processing he previous one.
- ✓ In other words, it can effectively process two instructions at once.

- ✓ For writing, one write command can be immediately followed by another without waiting for the original data to be stored within the SDRAM memory itself.
- ✓ For reading the requested data appears a fixed number of clock pulses after the read instruction was presented. It is possible to send additional instructions during the delay period which is termed the latency of the SDRAM.

SDRAM types and development

- ✓ Since SDRAM was introduced, it has been developed to make it faster and more effective.
- ✓ As a result there are a number of different types of SDRAM that are available.
  - ***SDR SDRAM:*** This is the basic type of SDRAM that was first introduced. It has now been superseded by the other types below. It is referred to as single data rate SDRAM, or just SDRAM.
  - ***DDR SDRAM:*** DDR SDRAM gains its name from the fact that it is Double Data Rate SDRAM. This type of SDRAM provides data transfer at twice the speed of the traditional type of SDRAM memory. This is achieved by transferring data twice per cycle.
  - ***DDR2 SDRAM:*** DDR2 SDRAM can operate the external bus twice as fast as its predecessor and it was first introduced in 2003.
  - ***DDR3 SDRAM:*** DDR3 SDRAM is a further development of the double data rate type of SDRAM. It provides further improvements in overall performance and speed. As a result its use is becoming more widespread.
  - ***DDR4 SDRAM:*** This is a further type of SDRAM being developed and anticipated to be available in 2012.

READ ONLY MEMORY

- ✓ Both SRAM and DRAM chips are volatile, which means that they lose the stored information if power is turned off.
- ✓ There are many applications that need memory devices which retain the stored information if power is turned off.

For example, in a typical computer a hardware drive is used to store a large amount of information, including the operating system software. When a computer is turned on, the operating system software has to be loaded from the disk into the memory. This requires execution of a program that boots the operating system. Since the boot program is quite large, most of it is stored on the disk. The processor must execute some instructions that load the boot program into the memory.

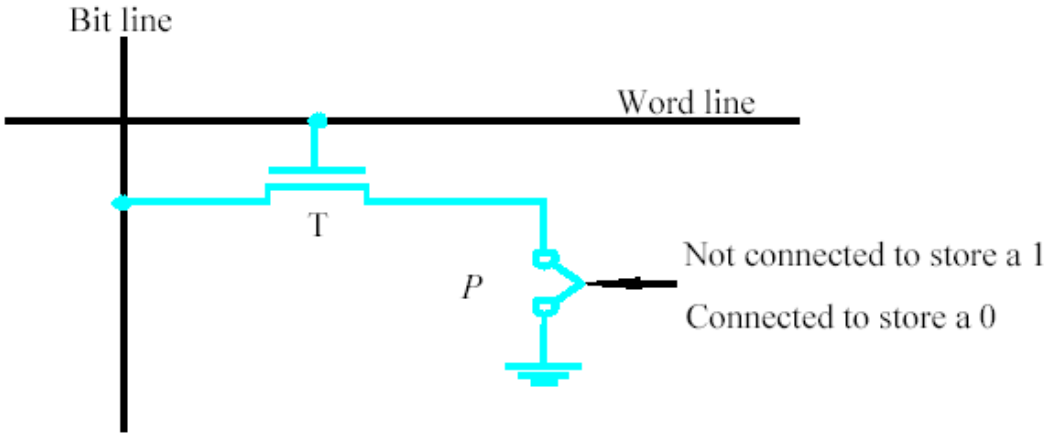
Nonvolatile memory is used extensively in embedded systems. Such systems typically do not use disk storage devices. Their programs are stored in nonvolatile semiconductor memory devices.

- ✓ Different types of nonvolatile memory have been developed.
- ✓ Generally, the contents of such memory can be read as if they were SRAM or DRAM memories.
- ✓ But, a special writing process is needed to place the information into this memory.
- ✓ Since its normal operation involves only reading of stored data, a memory of this type is called **read only memory (ROM).**

ROM CELL

- ✓ A logic value 0 is stored in the cell if the transistor is connected to ground at point P otherwise, 1 is stored.
- ✓ The bit line is connected through a resistor to the power supply.
- ✓ To read the state of the cell, the word line is activated.

- ✓ Thus, the transistor switch is closed and the voltage on the bit line drops to near zero if there is a connection between the transistor and ground.
- ✓ If there is no connection to ground, the bit line remains at the high voltage, indicating a 1.
- ✓ Data are written into a ROM when it is manufactured.
- ✓ Following figure shows a configuration for a ROM cell.



PROM

- ✓ Programmable ROM in which data can be loaded by user.
- ✓ Programmability is achieved by inserting a fuse at point P in above figure.
- ✓ Before it is programmed, the memory contains all 0s.
- ✓ The user can insert 1s at the required locations by burning out the fuses at these locations using high current pulses.
- ✓ This process is irreversible.

EPROM

- ✓ Another type of ROM chip allows the stored data to be erased and new data to be loaded.
- ✓ Such an erasable, reprogrammable ROM is usually called an EPROM.
- ✓ It provides considerable flexibility during the development phase of digital systems.
- ✓ Since EPROMs are capable of retaining stored information for a long time, they can be used in place of ROMs while software is being developed.
- ✓ In this way, memory changes and updates can be easily made.

- ✓ The important advantage of EPROM chips is that their contents can be erased and reprogrammed.
- ✓ Erasing done by UV (ultraviolet) light.

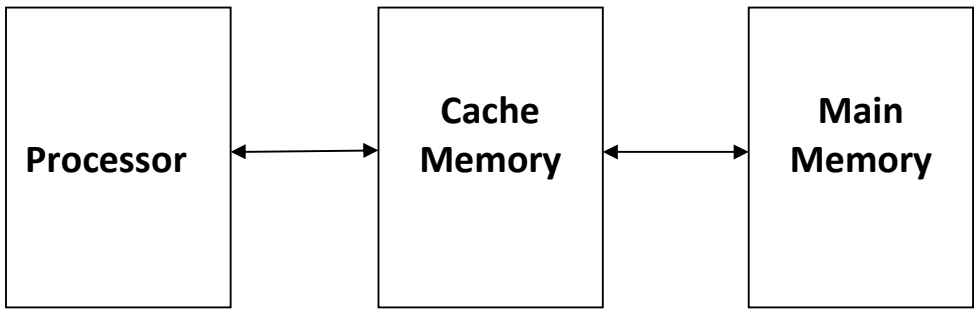
EEPROM

- ✓ A significant disadvantage of EPROMs is that a chip must be physically removed from the circuit for reprogramming and that its entire contents are erased by the ultraviolet light.
- ✓ It is possible to implement another version of erasable PROMs that can be both programmed and erased electrically.
- ✓ Such chips called EEPROMs, do not have to be removed for erasure. Moreover, it is possible to erase the cell contents selectively.
- ✓ The only disadvantage of **EEPROM**s is that different voltages are needed for erasing writing, and reading the stored data.

CACHE MEMORY

Cache memory is a small-sized type of volatile computer memory that provides high-speed data access to a processor and stores frequently used computer programs, applications and data.

- ✓ Cache memory faster than main memory.
- ✓ Cache memory, also called CPU memory, It can access more quickly than it can regular RAM.
- ✓ This memory is typically integrated directly with the CPU chip or placed on a separate chip that has a separate bus interconnect with the CPU.
- ✓ Consider the simple arrangement in following figure,



- ✓ When a read request is received from the processor, the contents of the memory location are transferred into the cache one word at a time.
- ✓ When the program references any of the locations in this block, the desired contents are read directly from the cache.
- ✓ Usually the cache memory can store a reasonable number of blocks at any given time, but

  this number is small compared to the total number of block in the main memory.
- ✓ The correspondence between the main memory blocks and cache is specified by a mapping function.

MAPPING FUNCTIONS:

- ✓ There are three types of mapping techniques used in cache memory system
  - Direct mapping
  - Associative mapping
  - Set associative mapping

Direct mapping:

- ✓ The simplest way to determine cache locations in which to store memory blocks is the direct mapping technique.
- ✓ Consider a cache of 128 blocks.
- ✓ The j^{th} block in main memory is mapped onto block j modulo 128 of the cache.
- ✓ Thus, whenever one of the main memory blocks 0, 128, 256… is loaded in the cache, it is stored in cache block 0. Block1, 129, 257… are stored in cache block 1, and so on.

✓ Placement of a block in the cache is determined from the memory address.

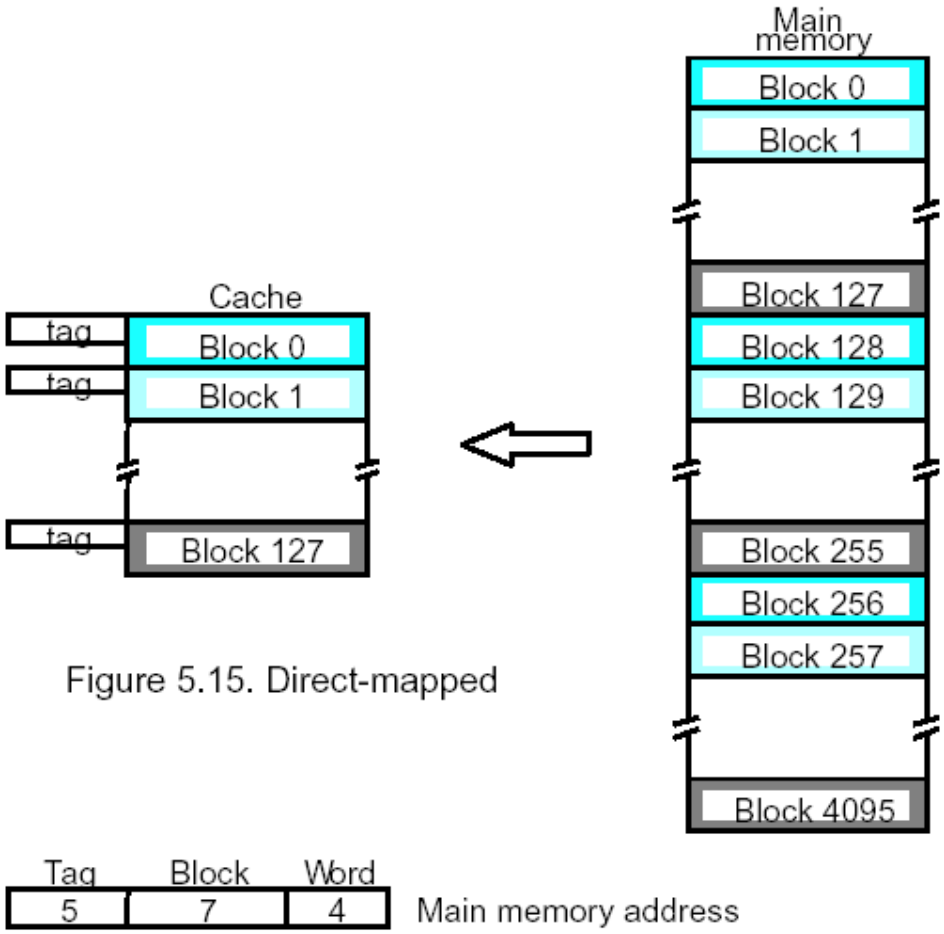


Figure 5.15. Direct-mapped

✓ The memory address can be divided into **three** fields.

- Tag (higher order 5 bits)
- Block (7 bits)
- Word (lower order 4 bits)

Associative mapping

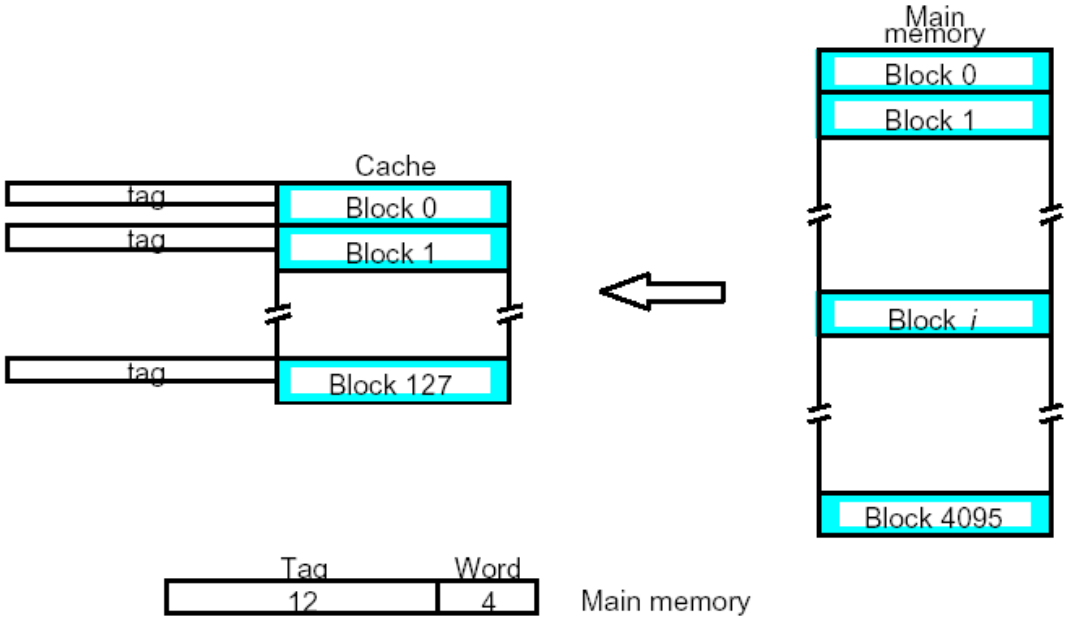
✓ The main memory block can be placed into any cache block position.

✓ In this case, 12 tag bits are required to identify a memory block when it is resident in the cache.

✓ The tag bits of an address received from the processor are compared to the tag bits of each block of the cache to see if the desired block is present.

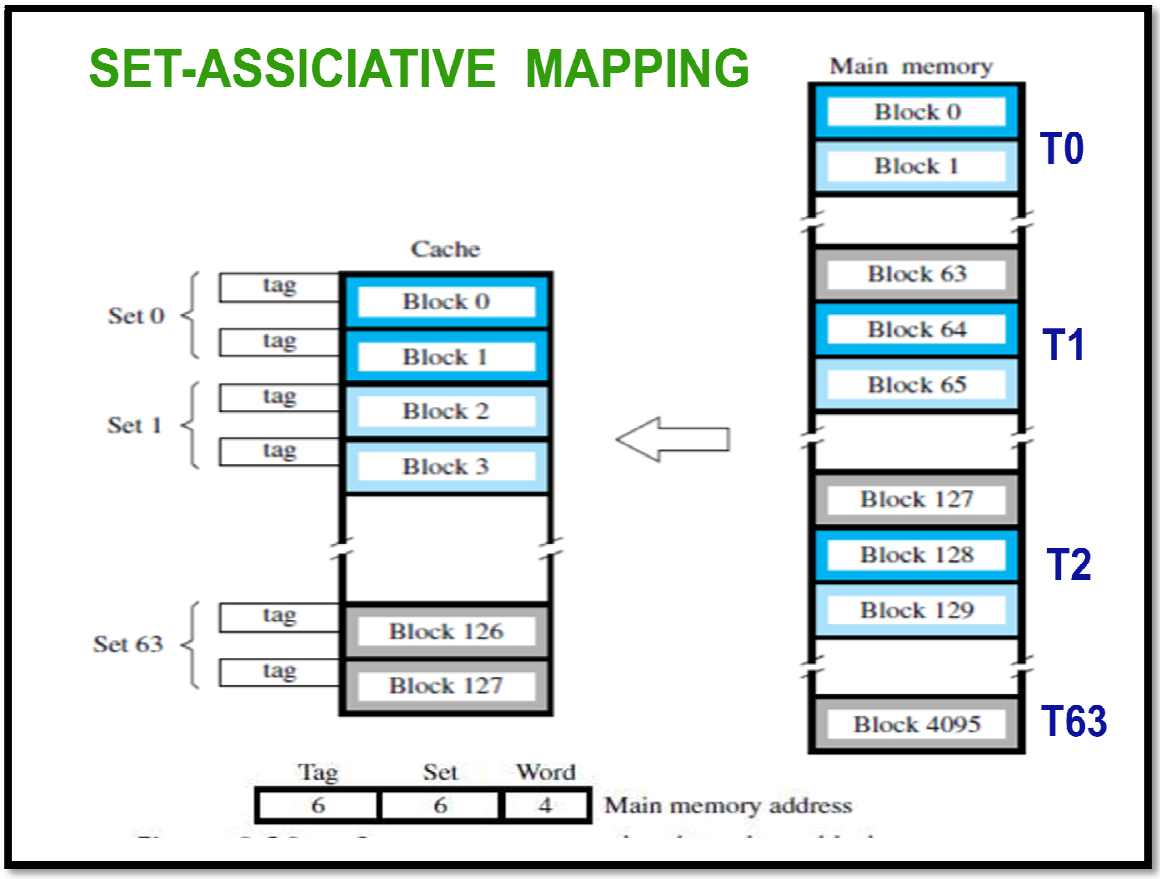
✓ This is called the associative mapping technique.

- ✓ It gives complete freedom in choosing the cache location in which to place the memory block.
- ✓ Thus, the space in the cache can be used more efficiently.
- ✓ Following figure shows a mapping method.



Set Associative mapping

- ✓ A combination of the direct and associative mapping techniques can be used.
- ✓ Blocks of the cache are grouped into sets, and the mapping allows a block of the main memory to reside in any block of a specific set.
- ✓ Hence the contention problem of the direct method is eased by having a few choices for block placement.
- ✓ At the same time, the hardware cost is reduced by decreasing the size of the associative search.
- ✓ This is called the set associative mapping technique is shown in figure.



-

- ✓ It has the cache with two blocks per set.
- ✓ In this case, memory blocks 0, 64, 128,….4032 map into cache set 0, and they can occupy either of the two block positions within this set.
- ✓ Having 64 sets means that the 6 bit set field of the address determines which set of the cache might contain the desired block.
- ✓ The tag field of the address must then be associatively compared to the tags of the two blocks of the set to check if the desired block is present.
- ✓ This two way associative search is simple to implement.

Measuring and Improving Catch Performance

- ✓ Two different techniques are used to improve cache performance,
  1. Reducing the miss rate.
  2. Reducing the miss penalty by adding additional catch (multilevel caching)

- ✓ The performance of the cache memory system incorporated with CPU execution time.
- ✓ Then the formula is,

CPU execution time = (CPU clock cycles + Memory stall clock cycles) * Clock cycle time

- ✓ The effectiveness of cache memory is based on locality of reference.
- ✓ It is of two types,
  - Temporal locality
  - Spatial locality

Temporal locality: Temporal locality means recently executed instructions are executed very soon.

Spatial locality : Spatial locality means that instructions in close proximity to a recently executed instruction (with respect to the instructions addresses) are also likely to be executed soon.

Replacement algorithm

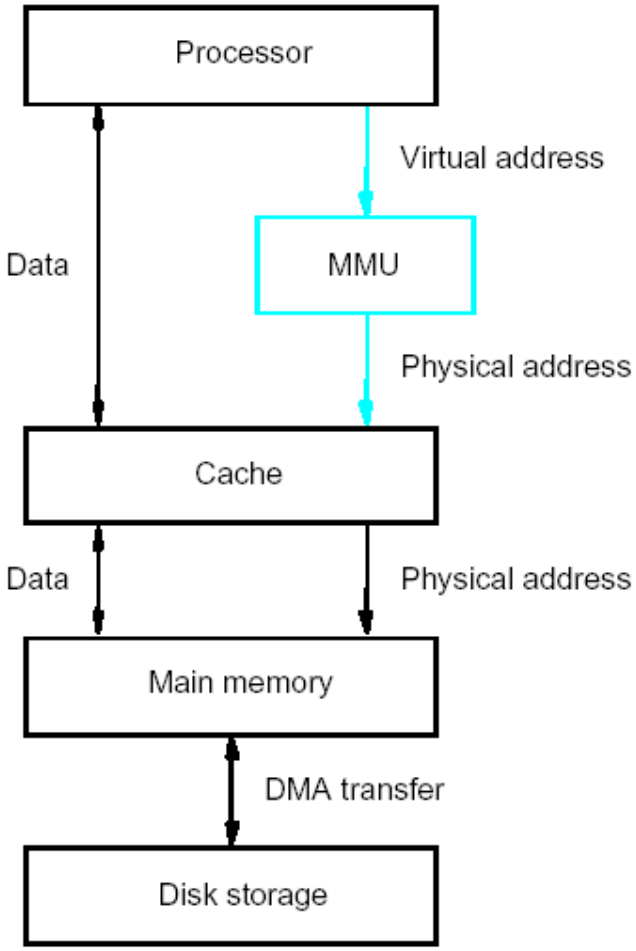
- ✓ When the cache is full and a memory word (instruction or data) that is not in the cache is referenced, the cache control hardware must decide which block should be removed to create space for the new block that contains the referenced word.
- ✓ This is done by using replacement algorithm.
- ✓ The processor needn't known explicitly that a cache is present.
- ✓ It simply issues Read and Write requests using addresses that refer to locations in the memory.
- ✓ The cache control circuitry determines whether the requested word currently exists in the cache.
- ✓ it does, the Read or Write operation is performed on the appropriate cache location.
- ✓ This case, a read or write hit is said to have occurred.
- ✓ In a **read operation**, the main memory is not involved.

- ✓ In a **write operation**, the main memory is involved.

- ✓ The main memory location is updated when the location is removed from cache.
- ✓ This technique is known as the write back or copy back protocol**.**
- ✓ When the addressed word in a read operation is not in the cache, a **read miss** occurs.
- ✓ During a write operation, if the addressed word is not in the cache, a **write miss** occurs.

VIRTUAL MEMORIES

- ✓ The physical main memory is not as large as the address space spanned by the processor.
- ✓ For example, when A processor that issues 32 bit addresses has an addressable space of 4G bytes.
- ✓ Then a program does not completely fit into the main memory, the parts of it not currently being executed are stored on secondary storage devices, such as magnetic disks.
- ✓ All parts of a program that are eventually executed are first brought into the main memory.
- ✓ When a new segment of a program is to be moved into a full memory, it must place another segment already in the memory.
- ✓ In modern computer, the operating system moves programs and data automatically between the main memory and secondary storage.



- ✓ Techniques that automatically move program and data blocks into the physical main memory when they are required for execution are called virtual memory technique.
- ✓ Above figure shows a typical organization that implements virtual memory.
- ✓ A special hardware unit, called the memory management unit (MMU), translates virtual addresses into physical addresses.
- ✓ When the desired data are in the main memory, these data are fetched as described in our presentation of the cache mechanism.
- ✓ If the data are not in the main memory, the MMU causes the operating system to bring the data into the main memory from the disk.
- ➢ Transfer of data between the disk and the main memory is performed using the **DMA (Direct Memory Access).**

ADDRESS TRANSLATION

- ✓ Assume that all programs and data are composed of fixed-length units called **pages**, each of which consists of a block of words that occupy contiguous locations in the main memory.
- ✓ Pages commonly range from 2K to 16K bytes in length.
- ✓ They constitute the basic unit of information that is moved between the main memory and the disk whenever the translation mechanism determines that a move is required.
- ✓ Pages should not be too small, because the access time of a magnetic disk is much longer (several milliseconds) than the access time of the main memory.
- ✓ By adding the virtual page number to the contents of this register, the address of the corresponding entry in the page table is obtained.
- ✓ This location gives the starting address of the page, if that page currently resides in the main memory.
- ✓ If it is not present, it is brought to the suitable location in main memory.
- ✓ Memory Management Unit translates virtual address to physical address.
- ✓ Each entry in the page table also includes some control bits that describe the status of the page while it is in the main memory.
- ✓ The page table information is used by the MMU for every read and write access, so the page table should be situated within the MMU.
- ✓ Therefore, the page table is kept in the main memory.
- ✓ However a copy of a small portion of the page table can be accommodated within the MMU.

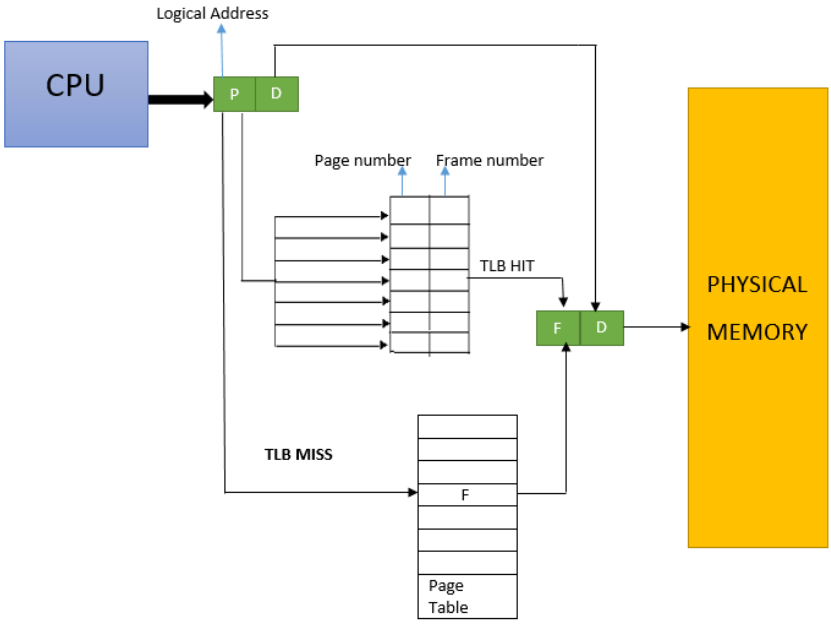
TLBs

- ✓ Translation-Lookaside Buffer (TLB) A cache that keeps track of recently used address mappings to try to avoid an access to the page table.
- ✓ (TLB) is a memory cache that is used to reduce the time taken to access a user memory location.
- ✓ It is a part of the chip's memory-management unit (MMU).

- ✓ The TLB stores the recent translations of virtual memory to physical memory and can be called an address-translation cache.
- ✓ A TLB may reside between the CPU and the CPU cache, between CPU cache and the main memory or between the different levels of the multi-level cache.
- ✓ The majority of desktop, laptop, and server processors include one or more TLBs in the memory management hardware, and it is nearly always present in any processor that utilizes paged or segmented virtual memory.
- ✓ A TLB has a fixed number of slots containing page table entries and segment table entries; page table entries map virtual addresses to physical addresses and intermediate table addresses, while segment table entries map virtual addresses to segment addresses, intermediate table addresses and page table addresses.
- ✓ The virtual memory is the memory space as seen from a process; this space is often split into pages of a fixed size (in paged memory), or less commonly into segments of variable sizes (in segmented memory).
- ✓ The page table, generally stored in main memory, keeps track of where the virtual pages are stored in the physical memory.
- ✓ This method uses two memory accesses (one for the page table entry, one for the byte) to access a byte.
- ✓ First, the page table is looked up for the frame number.
- ✓ Second, the frame number with the page offset gives the actual address.
- ✓ Thus any straightforward virtual memory scheme would have the effect of doubling the memory access time.
- ✓ Hence, the TLB is used to reduce the time taken to access the memory locations in the page table method.
- ✓ The TLB is a cache of the page table, representing only a subset of the page table contents.

- ✓ Referencing the physical memory addresses, a TLB may reside between the CPU and the CPU cache, between the CPU cache and primary storage memory, or between levels of a multi-level cache.
- ✓ The placement determines whether the cache uses physical or virtual addressing.

- ✓ If the cache is virtually addressed, requests are sent directly from the CPU to the cache, and the TLB is accessed only on a cache miss.
- ✓ If the cache is physically addressed, the CPU does a TLB lookup on every memory operation and the resulting physical address is sent to the cache.



- ✓ The TLB can be used as a fast lookup hardware cache.
- ✓ The above figure shows the working of a TLB.
- ✓ Each entry in the TLB consists of two parts: a tag and a value.
- ✓ If the tag of the incoming virtual address matches the tag in the TLB, the corresponding value is returned.
- ✓ Since the TLB lookup is usually a part of the instruction pipeline, searches are fast and cause essentially no performance penalty.
- ✓ However, to be able to search within the instruction pipeline, the TLB has to be small.

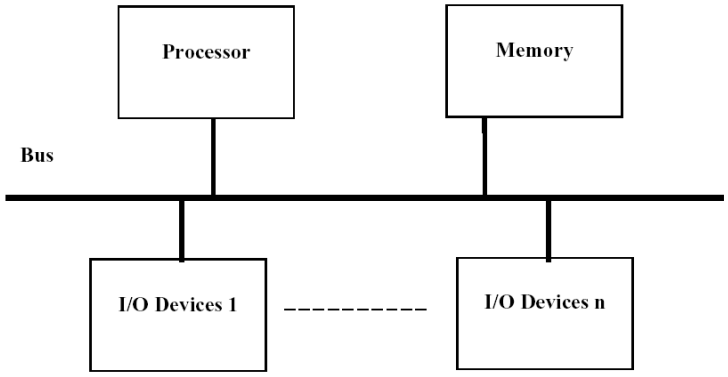
I/O SYSTEM

Introduction

- ✓ One of the basic features of a computer is its ability to exchange data with other devices.
- ✓ This communication capability enables a human operator, for example, to use a keyboard and a display screen to process text and graphics.
- ✓ We make extensive use of computers to communicate with other computers over the internet and access information around the globe.
- ✓ In other applications, computers are less visible but equally important.
- ✓ They are an integral part of home appliances, manufacturing equipment, transportation systems, banking and point-of-sale terminals.
- ✓ In such applications, input to a computer may come from a sensor switch, a digital camera, a microphone, or a fire alarm.
- ✓ Output may be a sound signal to be sent to a speaker or a digitally coded command to change the speed of a motor, open a valve, or cause a robot to move in a specified manner.
- ✓ In short, a general-purpose computer should have the ability to exchange information with a wide range of devices in varying environments

ACCESSING INPUT OUTPUT DEVICES

- ✓ A simple arrangement to connect I/O devices to a computer is to use a single bus arrangement, as shown in Figure,



- ✓ The bus enables all the devices connected to it to exchange information.
- ✓ Typically, it consists of three sets of lines used to carry address, data, and control signals. Each I/O device is assigned a unique set of addresses.
- ✓ When the processor places a particular address on the address lines, the device that recognizes this address responds to the commands issued on the control lines.
- ✓ The processor requests either a read or a write operation, and the requested data are transferred' over the data lines.
- ✓ When I/O devices and the memory share the same address space, the arrangement is called memory –mapped I/O.
- ✓ With memory mapped I/O any machine instruction that can access memory can be used to transfer data to or from an I/O device.
- ✓ For example, if DATAIN is the address of the input buffer associated with the keyboard, the instruction

Move DATAIN , RO

- ✓ Which reads the data from DATAIN and stores them into processor register RO.
- ✓ Similarly, the instruction

Move RO , DATAOUT

- ✓ Which sends the contents of register RO to location DATAOUT, which may be the
- ✓ output data buffer of a display unit or a printer.
- ✓ Most computer systems use memory-mapped I/O. Some processors have special In and Out instructions to perform I/O transfers.

- ✓ When building a computer system based on these processors, the designer has the option of connecting I/O devices to use the special I/O address space or simply incorporating them as part of the memory address space.
- ✓ The latter approach is the most common as it leads to simpler software.
- ✓ One advantage of a separate I/O address space is that I/O devices deal with fewer address lines.
- ✓ Note that a separate I/O address space does not necessarily mean that the I/O address lines are physically separate from the memory address lines.
- ✓ A special signal on the bus indicates that the requested read or write transfer is an I/O operation. When this signal is asserted, the memory unit ignores the requested transfer.
- ✓ The I/O devices examine the low order bits of the address bus to determine whether they should respond.

- ✓ The following figure illustrates the hardware required to connect an I/O device to the bus.

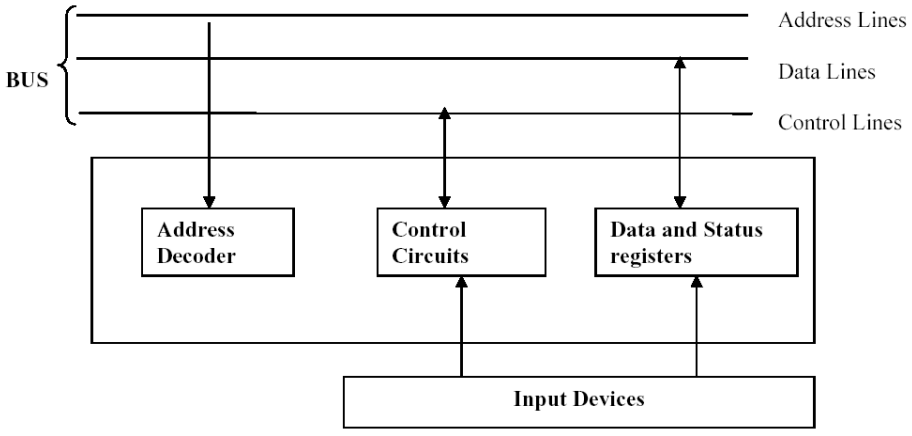


Fig:5.2 I/O interface for an input device

- ✓ The address decoder enables the device to recognize its address when this address appears on the address lines.
- ✓ The data register hold the data being transferred to or from the processor.
- ✓ The status register contains the information relevant to the operation of the I/O devices.

- ✓ Both the data and status register are connected to the data bus and assigned unique address.
- ✓ The address decodes ,the data and status registers ,and the control circuitry required to coordinate I/O transfer constitute the device interface circuit.
- ✓ When a human operator is entering characters at a keyboard, the processor is capable of executing millions of instructions between successive character entries.
- ✓ An instruction that reads a Character from the keyboard should be executed only when a character is available in the input buffer of the keyboard interface.
- ✓ Also, we must make 'sure that an input character is read only once.

- ✓ Three modes of transfer of device data,
  (i) Programmed IO
  (ii) Interrupt driven IO
  (iii) Direct memory access (DMA)

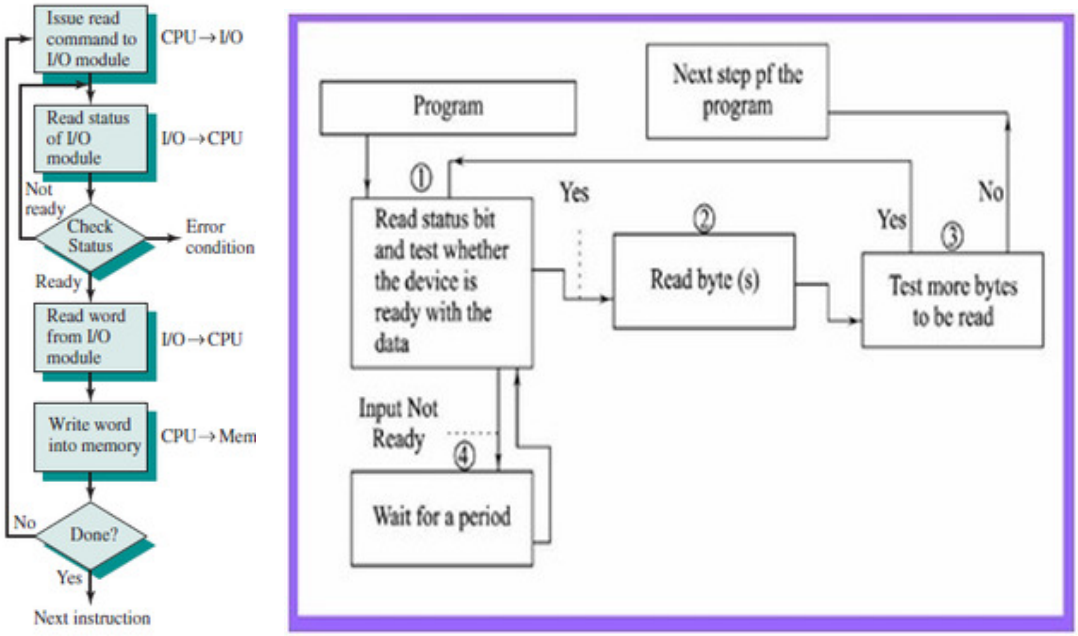
(i) Programmed IO

- ✓ Programmed input/output (PIO) is a method of transferring data between the CPU and a peripheral, such as a network adapter or an ATA storage device.
- ✓ programmable I/O is one of the I/O technique other than the interrupt-driven I/O and direct memory access (DMA).
- ✓ The programmed I/O was the most simple type of I/O technique for the exchanges of data or any types of communication between the processor and the external devices.
- ✓ With programmed I/O, data are exchanged between the processor and the I/O module.
- ✓ The processor executes a program that gives it direct control of the I/O operation, including sensing device status, sending a read or write command, and transferring the data.
- ✓ When the processor issues a command to the I/O module, it must wait until the I/O operation is complete.

✓ If the processor is faster than the I/O module, this is wasteful of processor time. The overall operation of the programmed I/O can be summaries as follow:

1. The processor is executing a program and encounters an instruction relating to I/O operation.
2. The processor then executes that instruction by issuing a command to the appropriate I/O module.
3. The I/O module will perform the requested action based on the I/O command issued by the processor (READ/WRITE) and set the appropriate bits in the I/O status register.
4. The processor will periodically check the status of the I/O module until it find that the operation is complete.

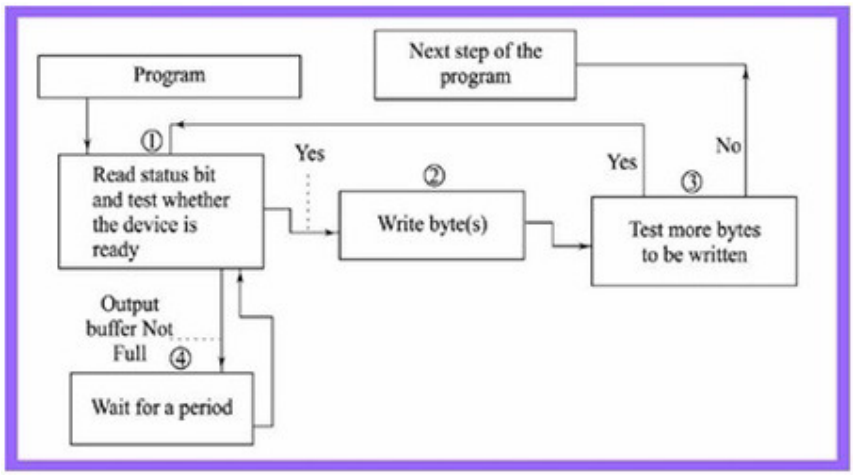
Programmed I/O Mode Input Data Transfer



1. Each input is read after first testing whether the device is ready with the input (a state reflected by a bit in a status register).
2. The program waits for the ready status by repeatedly testing the status bit and till all targeted bytes are read from the input device.

3. The program is in busy (non-waiting) state only after the device gets ready else in wait state.

✓ Programmed I/O Mode Output Data Transfer



1. Each output written after first testing whether the device is ready to accept the byte at its output register or output buffer is empty.
2. The program waits for the ready status by repeatedly testing the status bit(s) and till all the targeted bytes are written to the device.
3. The program in busy (non-waiting) state only after the device gets ready else wait state.

I/O Commands

✓ To execute an I/O-related instruction, the processor issues an address, specifying the particular I/O module and external device, and an I/O command.

✓ There are four types of I/O commands that an I/O module may receive when it is addressed by a processor:

✓ Control: Used to activate a peripheral and tell it what to do.

✓ For example, a magnetic-tape unit may be instructed to rewind or to move forward one record.

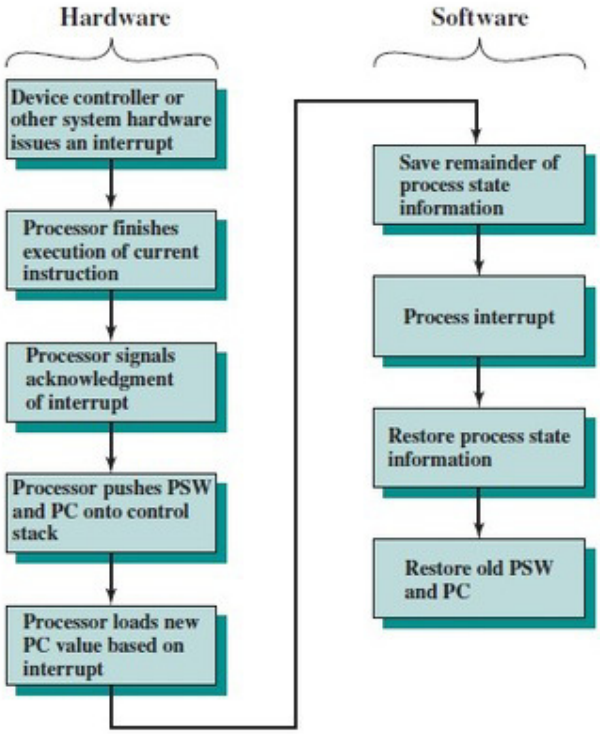
✓ These commands are tailored to the particular type of peripheral device.

- ✓ Test: Used to test various status conditions associated with an I/O module and its peripherals.
- ✓ The processor will want to know that the peripheral of interest is powered on and available for use.
- ✓ It will also want to know if the most recent I/O operation is completed and if any errors occurred.
- ✓ Read: Causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer.
- ✓ The processor can then obtain the data item by requesting that the I/O module place it on the data bus.
- ✓ Write: Causes the I/O module to take an item of data (byte or word) from the data bus and subsequently transmit that data item to the peripheral.
- ✓ Advantages & Disadvantages of Programmed I/O

Advantages	simple to implement
	very little hardware support
Disadvantages	busy waiting
	ties up CPU for long period with no useful work

(ii) Interrupt driven IO

- ✓ Interrupt driven I/O is an alternative scheme dealing with I/O. Interrupt I/O is a way of controlling input/output activity whereby a peripheral or terminal that needs to make or receive a data transfer sends a signal.
- ✓ This will cause a program interrupt to be set. At a time appropriate to the priority level of the I/O interrupt.
- ✓ Relative to the total interrupt system, the processors enter an interrupt service routine.
- ✓ The function of the routine will depend upon the system of interrupt levels and priorities that is implemented in the processor.
- ✓ The interrupt technique requires more complex hardware and software, but makes far more efficient use of the computer's time and capacities. Figure 2 shows the simple interrupt processing.



- ✓ For input, the device interrupts the CPU when new data has arrived and is ready to be retrieved by the system processor.
- ✓ The actual actions to perform depend on whether the device uses I/O ports or memory mapping.
- ✓ For output, the device delivers an interrupt either when it is ready to accept new data or to acknowledge a successful data transfer.
- ✓ Memory-mapped and DMA-capable devices usually generate interrupts to tell the system they are done with the buffer.
- ✓ Here the CPU works on its given tasks continuously.
- ✓ When an input is available, such as when someone types a key on the keyboard, then the CPU is interrupted from its work to take care of the input data.
- ✓ The CPU can work continuously on a task without checking the input devices, allowing the devices themselves to interrupt it as necessary.

Basic Operations of Interrupt

CPU issues read command.

1. I/O module gets data from peripheral whilst CPU does other work.

2. I/O module interrupts CPU.
3. CPU requests data.
4. I/O module transfers data.

Interrupt Processing

1. A device driver initiates an I/O request on behalf of a process.
2. The device driver signals the I/O controller for the proper device, which initiates the requested I/O.
3. The device signals the I/O controller that is ready to retrieve input, the output is complete or that an error has been generated.
4. The CPU receives the interrupt signal on the interrupt-request line and transfer control over the interrupt handler routine.
5. The interrupt handler determines the cause of the interrupt, performs the necessary processing and executes a "return from" interrupt instruction.
6. The CPU returns to the execution state prior to the interrupt being signaled.
7. The CPU continues processing until the cycle begins again.

Advantages & Disadvantages of Interrupt Drive I/O

Advantages	fast
	efficient
Disadvantages	can be tricky to write if using a low level language
	can be tough to get various pieces to work well together
	usually done by the hardware manufacturer / OS maker, e.g. Microsoft

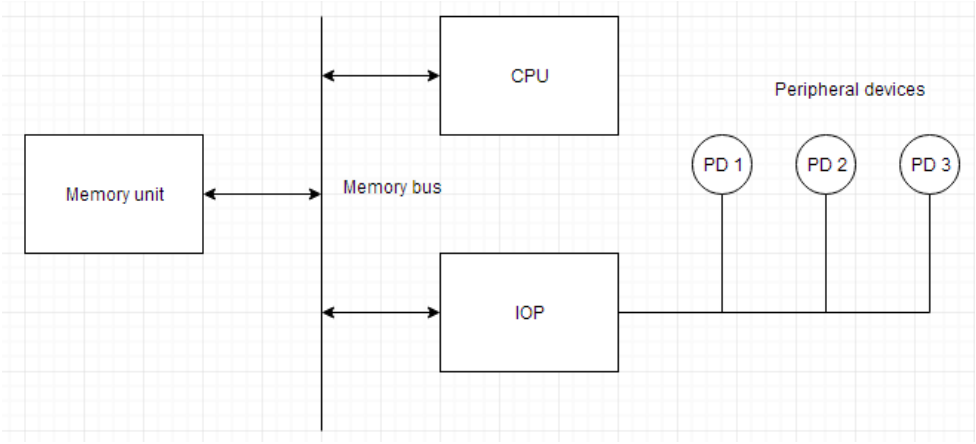
INPUT/OUTPUT PROCESSOR

✓ An input-output processor (IOP) is a processor with direct memory access capability.

- ✓ In this, the computer system is divided into a memory unit and number of processors.
- ✓ Each IOP controls and manage the input-output tasks.
- ✓ The IOP is similar to CPU except that it handles only the details of I/O processing.
- ✓ The IOP can fetch and execute its own instructions. These IOP instructions are designed to manage I/O transfers only.

Block Diagram Of IOP

- ✓ Below is a block diagram of a computer along with various I/O Processors.
- ✓ The memory unit occupies the central position and can communicate with each processor.
- ✓ The CPU processes the data required for solving the computational tasks.
- ✓ The IOP provides a path for transfer of data between peripherals and memory.
- ✓ The CPU assigns the task of initiating the I/O program.
- ✓ The IOP operates independent from CPU and transfer data between peripherals and memory.



- ✓ The communication between the IOP and the devices is similar to the program control method of transfer.
- ✓ And the communication with the memory is similar to the direct memory access method.

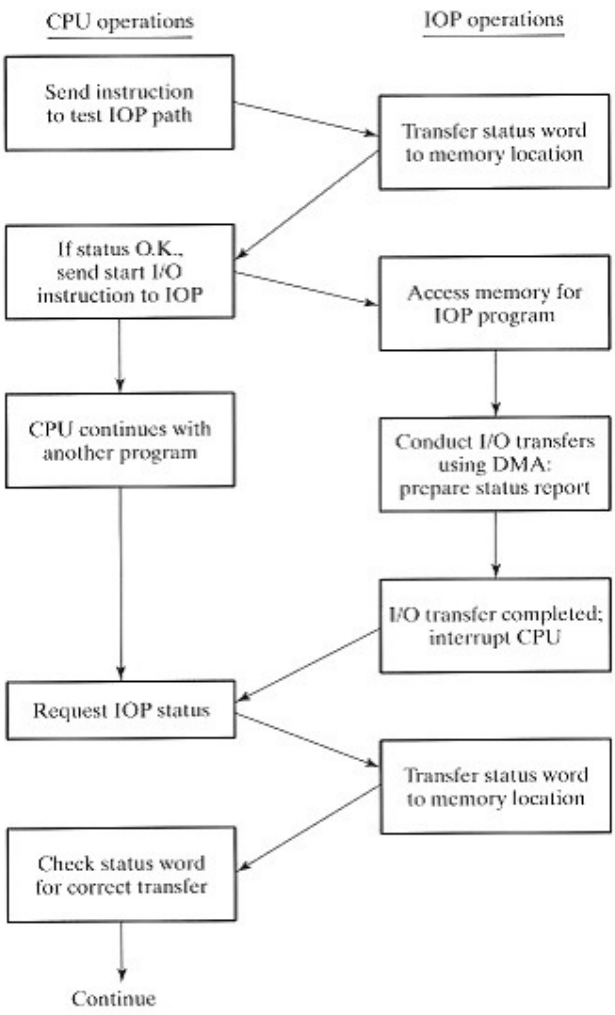
- ✓ In large scale computers, each processor is independent of other processors and any processor can initiate the operation.

- ✓ The CPU can act as master and the IOP act as slave processor.
- ✓ The CPU assigns the task of initiating operations but it is the IOP, who executes the instructions, and not the CPU. CPU instructions provide operations to start an I/O transfer.
- ✓ The IOP asks for CPU through interrupt.
- ✓ Instructions that are read from memory by an IOP are also called commands to distinguish them from instructions that are read by CPU.
- ✓ Commands are prepared by programmers and are stored in memory.
- ✓ Command words make the program for IOP. CPU informs the IOP where to find the commands in memory.

CPU-IO Communication

- ✓ Communication between the CPU and IOP may take different forms depending on the particular computer used.
- ✓ Mostly, memory unit acts as a memory center where each processor leaves information for the other.
- ✓ CPU sends an instruction to test the IOP path. The IOP responds by inserting a status word in memory for the CPU to check.
- ✓ The bits of the status word indicate the condition of IOP and I/O device ("IOP overload condition", "device busy with another transfer" etc).
- ✓ CPU then checks status word to decide what to do next .
- ✓ If all is in order, CPU sends the instruction to start the I/O transfer.
- ✓ The memory address received with this instruction tells the IOP where to find its program.
- ✓ CPU may continue with another program while the IOP is busy with the I/O program.
- ✓ When IOP terminates the transfer (using DMA), it sends an interrupt request to CPU.
- ✓ The CPU responds by issuing an instruction to read the status from the IOP and IOP then answers by placing the status report into specified memory location.

- ✓ By inspecting the bits in the status word, CPU determines whether the I/O operation was completed satisfactorily and the process is repeated again.



PCI

Peripheral Component Interconnect

- The PCI bus is a good example of a system bus that grew out of the need for standardization.
- It supports the functions found on a processor bus but in a standardized format that is independent of any particular processor.
- Devices connected to the PCI bus appear to the processor as if they were connected directly to the processor bus.
- They are assigned addresses in the memory address space of the processor.
- The PCI follows a sequence of bus standards that were used primarily in IBM PCs.
- The PCI was developed as a low cost bus that is truly processor independent.

- Its design anticipated a rapidly growing demand for bus bandwidth to support high speed disks and graphic and video devices.
- As a result, PCI is still popular as an industry standard almost a decade after it was first introduced in 1992.

Feature

- An important feature that the PCI pioneered is a plug –and-play capability for connecting I/O devices.
- To connect a new device, the user simply connects the device interface board to the bus.
- The software takes care of the rest.

Date Transfer

- In today's computers. Most memory transfers involve a burst of data rather than just one word.
- This is because of cache memory.
- Similarly during write operation, the processor sends a memory address followed by a sequence of data words, to be written.
- The PCI is designed primarily to support this mode of operation.
- A read or a write operation involving a single word is simply treated as a burst of length one.
- The bus supports three independent address space.
    - Memory
    - I/O
    - Configuration
- The first two are self-explanatory.
- The I/O address space is intended for use with processors, such as Pentium, that have separate I/O address space.

USB
Universal Serial Bus

- The USB supports two speeds of operation called
    - Low speed(1.5 mega bits/s)
    - Full speed (12 megabits /s)

USB 2.0

- The most recent revision of the bus specification (USB 2.0) introduced a third speed of operation.
- Which is called high speed (480 megabits/s)

Objectives

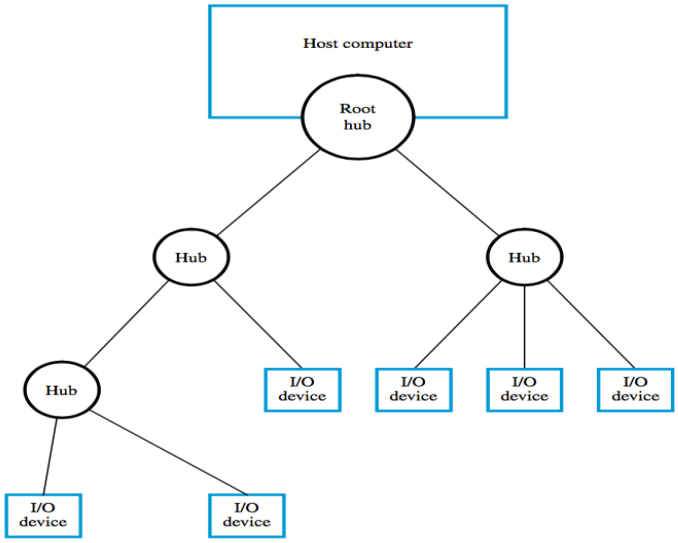
- The USB has been designed to meet several key objectives.
- Provide a simple, low cost, and easy to use interconnection system that overcomes the difficulties due to the limited number of I/O ports available on a computer.
- Accommodate a wide range of data transfer characteristics for I/O devices, including telephone and Internet connection.
- Enhance user convenience through a :plug-and-play" mode or operation.

Port Limitation

- To add a new ports, a user must open the computer box to gain access to the internal expansion bus and install a new interface card.
- The user may also need to know how to configure the device and the software.
- An objective of the USB is to make it possible to add many devices to a computer system at any time, without opening the computer box.

USB Architecture

- The serial transmission format has been chosen for the USB.
- Because the serial bus satisfies the low cost and flexibility requirements.
- USB offers three bit rates, ranging from 1.5 to 480 megabits/s, to suit the needs of different I/O devices.
- To accommodate a large number of devices that can be added or removed at any time, the USB has the following tree structure.



- Each node of the tree has a device called a hub.
- Which acts as an intermediate control pointbetween the host and I/O devices.
- The root hub connect the entire tree to the host computer.
- The leaves of the tree are the I/O devices(key board, speaker, digital TV etc..)
- Which are called functions in USB terminology.
- The tree structure enables many devices to be connected while using only simple point to point serial links.
- Each hub has a number of ports where devices may be connected, including other hubs.
- In normal operation, a hub copies a message that it receives from its upstream connection to all its downstream ports.
- As a result a message sent by the host computer is broadcast to all I/O devices.
- But only the addressed device will respond to that message.

Polling

- USB operates strictly on the basis of polling.
- A device may send a message only in response to a poll message from the host.
- Hence upstream message do not encounter conflicts or interfere with each other as no two devices send messages at the same time.
- This restrictions allows hubs to be simple, low cost devices.

USB Protocols

- All information transferred over the USB is organized in packets.

- Where packets consists of one or more bytes of information.
- There are many types of packets that perform variety of control functions.

- The information transferred on the USB can be divided into two broad categories.
  - Control
  - Data
- Control packets perform such tasks as
  - addressing a device to initiate data transfer
  - acknowledging the received data
  - indicating an error.
- Data packets carry information that is delivered to a device.