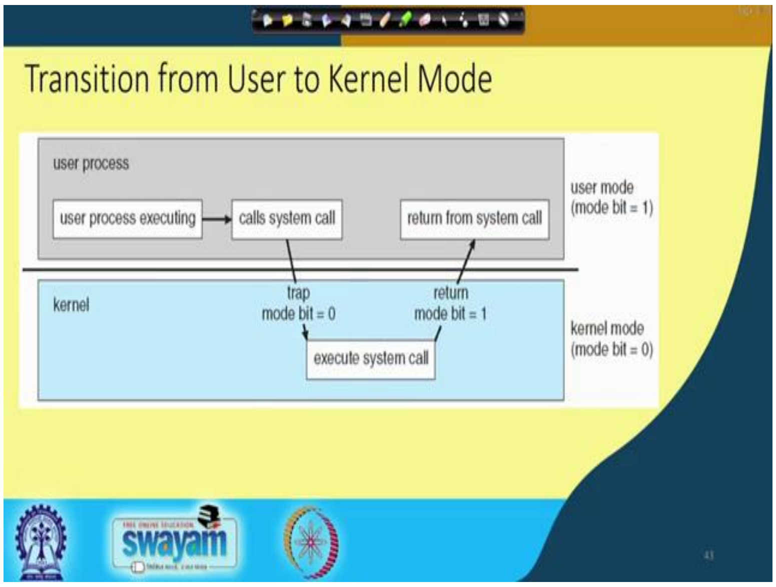
Operating Systems Fundamentals

Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 06
Introduction (Contd.)

So, transition from user mode Kernel mode so, this is done by means of system calls.

(Refer Slide Time: 00:31)



Transition from User to Kernel Mode

So, in this diagram, so we can see that normally when the user process are executing, they are executing with in user mode with the mode bit equal to 1. So, this 1 or 0, that is depicted by the processor designer. So, in this particular example it has been taken that a for user mode, the mode bit is 1 and for kernel mode, mode bit is 0. So, this may be the reverse also and there can be multiple such mode bits also. So, this is just an example.

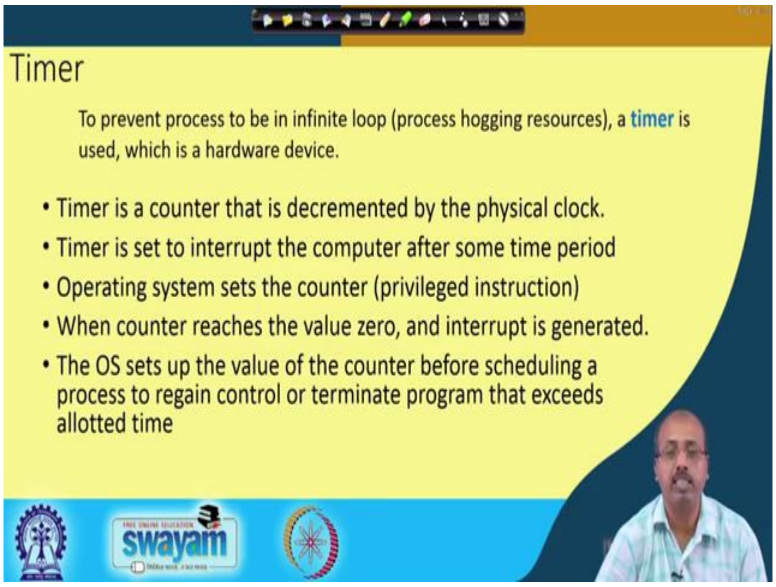
So, user process executing, as I was telling, the normal statements that are there. So, they are getting executed by the user process and after some time there will be a system call when the program needs to access system resources. So, it will be making system calls and when the system call is made, this is normally implemented by trap type of instructions that we have in the computer architecture.

So, corresponding processor designer, there is an instruction set, normally there are some trap type of instruction. So, they are used for this purpose. And trap instruction so they transfer control of the system to some specific addresses and what the OS designer does is that, they put some particular web services in those trap numbers. So, depending upon the type of system call that is made by the user process so, the corresponding trap is invoked and then this it comes to the kernel mode and this strap instruction when it is executing, the system designer, the architecture designer they have ensure that the mode bit becomes equal to 0.

At this point it executes the system call. So, this is in a more protected environment and you see at this is a part, in the kernel part the code that the processor executing is written by the OS designer. Whereas, when a process is in a user space, user process, user mode then it is executing the code written by the user. Of course, there are some library routines that are there so, that can be executed in the user mode also and that is not developed by the user. But in general, I can say that when a program is executing in the user mode so, it is executing the code written by the user and when it is going to the kernel mode of execution so, it is executing a piece of code which is written by the OS designer.

So, these codes are written much more carefully so that there cannot be any problem, unwanted problem in the system. So, this executing system call so, it is making mode bit equal to 0 with that it is executing the system call. After the system call execution is over it will return setting mode bit to 1 and then it returns from the system call and that goes back to the user process execution.

(Refer Slide Time: 03:27)



Timer

To prevent process to be in infinite loop (process hogging resources), a timer is used, which is a hardware device.

- Timer is a counter that is decremented by the physical clock.
- Timer is set to interrupt the computer after some time period
- Operating system sets the counter (privileged instruction)
- When counter reaches the value zero, and interrupt is generated.
- The OS sets up the value of the counter before scheduling a process to regain control or terminate program that exceeds allotted time

So, we have got this user mode and kernel mode. Many a time, we need to have some timers. So, a timer is used to prevent the processes to be in infinite loop. Timer is a hardware device. So, actually when you start a process so, how do you know that this process is taking too much of system resources? Too much of system resources maybe in terms of CPU time, maybe in term some printer or some other system resource.

So, how do you know that it is taking too much of time? So, for that purpose we can put a limit on the system, on the processes usage of that particular resource or a process going on to infinite loop. So, this is by means of starting a timer, when the timer expires so, it means that the process should have been over by this time, but it must be misbehaving, that is why it is giving this type of performance. So, that process may be needed to be stopped. So, normally some system so, they put a limit on the amount of CPU time that a process can use and when that time expires the process gets killed automatically.

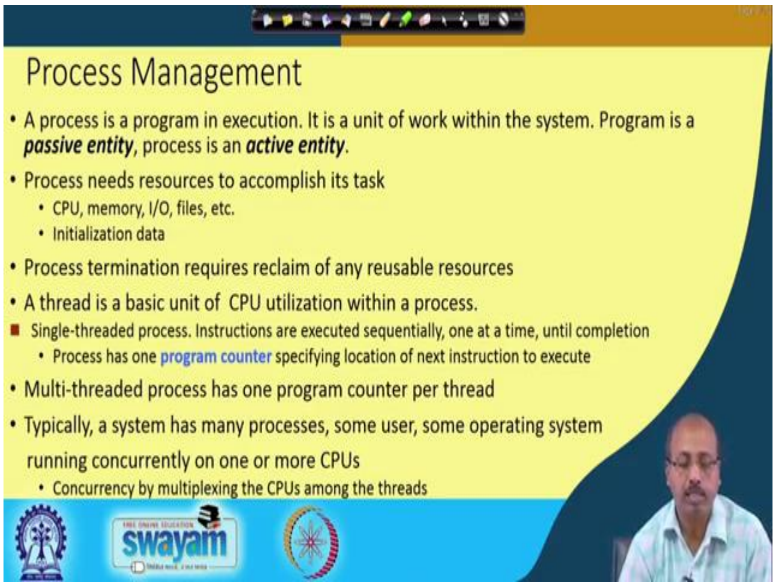
So, this timer is provided by the hardware designer that is the processor designer. It should have a timer in the system. So, timer is a counter that is decremented by the physical clock. So, say, maybe I start a timer 10 seconds. So, after every physical clock tick the value will be reduced by 1 and then when the value becomes 0, then it will interrupt the computer after the time period. Now, when this interrupt comes then we know that the process has not released the resource by this time. So, we may identify the process to be one which is misbehaving.

So, operating system sets the counter by means so, that is the privilege instruction that will set for how much time the process should execute. And when the counter reaches value 0, interrupt will be generated and then OS will set up the value of the counter before scheduling a process to regain control or terminate program that exceeds allotted time. So, what we mean is that suppose I have got a system that has got multiple users in the system. So, in a multi user or multiprocessing system the basic requirement is that every process or every user should get the access to the CPU after the time. Maybe I give, to be fair to everybody, we may decide that we will give 2 second to each of them.

So, when we are doing that so, this 2 second thing, somehow the system has to know that 2 second has expired. So, how does it know it? So, it knows by starting a timer for 2 second and when the 2 second is over the timer will come and there can be 2 possibilities; there are 2 possibilities, one possibility is that the process that was given 2 second does not execute for 2 second, it finishes before 2 second and comes back. Or, it goes into an IO operation without executing for 2 seconds. So, that maybe one type of situation when we say that the process is good, their/its behavior is good because of a because it is not using lot of CPU time. It is giving others chances for execution.

And the other possibility is that the process continually uses that 2 second and it does not stop in between. Then when this timer interrupt will come then the OS will know that the process has exceeded, the process has completely used its 2 second time. So, the operating system now regains control of the system and then it will remove that process from the CPU and give turn to the next process that is there in the queue. And again, this process will be executed sometime later when all other processes are got their turns and again this is a turn for this process. That way it can go on. So, this timer is very important.

(Refer Slide Time: 07:33)



Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity***, process is an ***active entity***.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files, etc.
  - Initialization data
- Process termination requires reclaim of any reusable resources
- A thread is a basic unit of CPU utilization within a process.
- Single-threaded process. Instructions are executed sequentially, one at a time, until completion
  - Process has one program counter specifying location of next instruction to execute
- Multi-threaded process has one program counter per thread
- Typically, a system has many processes, some user, some operating system running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the threads

A very important part of operating system is known as the process management. So, we’ll see this process in more detail, but just to tell you a process is a program in execution, it is a unit of work within the system. So, any program that I write so, is ultimately going for execution by the system. So, that becomes a process. So, when I write a program, I have translated it using some compiler and I have got the machine code of it. So, machine code that is generated is kept as an executable file in the disc. So, that is a passive entity. So, that is not taking the CPU for execution.

But when this program is given for execution so, the user wants to run the program then the program is loaded into main memory and it is starts executing. So, it starts consuming CPU time and all. So now, the process becomes active. So, if I say that the program is a passive entity, process is going to be an active entity.

Process needs resources to accomplish the task, like it needs CPU, memory, IO files, etc. Whereas, a program, it just resides in the disc. So, it is just taking some disc resource, nothing more than that. There is some the program. The process will need CPU memory IO file plus some initialization data, some data variable will be required, they are to be initialized to some values and all so that will be required.

Process termination, when a process terminates then, when it was executing it held a number of resources. So, when the process terminates all these resources are to be taken back by the system and maybe it is now used by some other process. So, process termination requires a reclaiming of any reusable resources.

We’ll be talking about thread in more detail later, but thread is a basic unit of CPU utilization within a process. So, within a process also there can be several sequences of executions that are possible and it may so happen that a number of sequences, they go parallel. A typical example is like this, suppose I have got a server, some web server and there are a number of requests coming from different users of the system. Now, all of them are executing the same server program, but within that every user or every client that connects to the server has got a different entity. So, all of them are called threads.

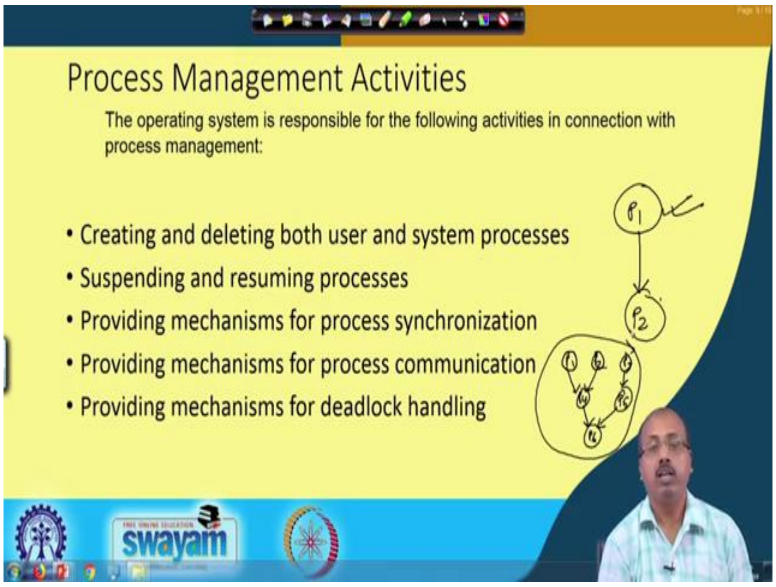
So, in this sense a thread is the basic unit of CPU utilization within a process. For a single threaded process, instructions are executed sequentially one at a time until completion. So, this type of things is a single threaded process. So, if you look into a process that has got a program counter that specifies the location of the next instruction to execute. So, that is very general. For any system any program that we have to execute, this processor will have a program counter register that will tell like which instruction to execute next.

Compare to single threaded processes we can have multithreaded process in which we have got one program counter per thread. So, as I was telling that there can be several threads for that web server program and then for each thread, I need to remember which statement I was executing. So, in some sense you can say we have got multiple program counters, one for each thread.

Typically, a system will have many processes, some user and some operating system processes, they are running concurrently on one or more CPU. So, this is the most general most general scenario. So, we can have some user programs and some user processes and some operating system processes. So, they will run concurrently and there may be a single CPU, there can be multiple CPU. So, depending upon the scheduling policy so, this processes are given to this CPUs and then they are executing.

So, their concurrency has to be maintained. So, concurrency is done by multiplexing the CPUs among the thread. So, if I have got say 4 CPUs and there are 10 processes or 10 threads, then I cannot execute all 10 of them simultaneously, but I can take a 4 of them, give to 4 CPUs, 4 cores of 4 CPUs then after sometime I take back these 4 and give another set of 4 threads for execution so, that can be done. So, that way there is a concept of time multiplexing among the threads.

(Refer Slide Time: 12:18)



Process Management Activities

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

So, process management activities so, operating system is responsible for these activities, one is creating and deleting user and system processes. So, when are the operating system comes up so, it create some processes and whenever the user programs come up, they create some processes, after sometime, some of this processes will die. So, we have to delete them. So that creation and deletion of processes, that is an important responsibility of operating system process management module.

Then we may want to suspend a process or resume some process. So, as I was telling that a process may be taking lot of CPU resources or IO resources like that. So, how to stop one process executing and if I stop it in between then maybe I need to restart it later. So, how do you suspend it and how do you resume the operation.

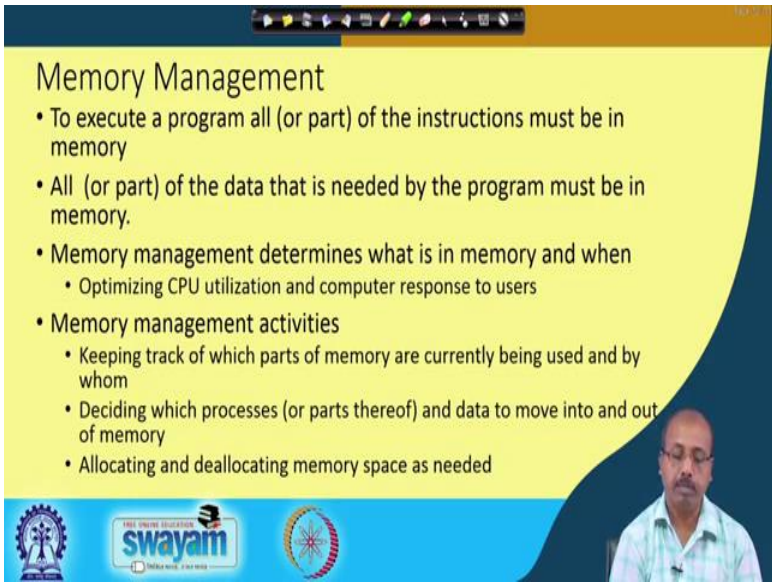
Then how to provide mechanism for process synchronization? So, I must do something so, that I can synchronize between the execution of processes. So, it may so happen that I have got an application where I have got 2 processes P1 and P2 and it is the case that this P2, it uses some value which is computed by P1. So, it is required that in my system, though P1 and P2 they are 2 parallel processes or parallel threads, so, P1 should be executed first and then only P2 should be executed. So, there is an explicit synchronization requirement between P1 and P2.

So, in a more general case I can think that a system has got a number of processes in it and there are some synchronization requirements like say, if this is say P1, P2, P3, P4, like that. Then it may so happen that, P4 it requires P1 and P2 should be over. There is another process P5 that can be executed only after P3 is over. And there is a process P6 that can be executed, after this P4 and P5 both are over. Now, how do you ensure that this type of synchronization? So, if I create 6 independent processes and give it to the system then say how to ensure that system will not execute P4 before P1, P2? So, that is the synchronization problem.

Sometimes we need some mechanism for communication like say some data item which is computed by P1 is used by P2. So, there must be some communication mechanism between the processes and there is a typical problem that occurs whenever you have got this type of multiple processes which is known as that problem of deadlock.

So, deadlock is a situation when none of the processes can proceed and system throughput will be very low, because all the processes, they start waiting for data computed by each other or all of them start waiting for some resource to be freed by some other process and that process again waits for some other process to finish its resource. So, ultimately all the processes they go into a cyclic situation and then we have got a system situation where the entire system in the entire system, nothing is progressing. So, that is a deadlock situation. So, we will see it in more detail when we go to the discussion on deadlock. So, this is the responsibilities of this process management part.

(Refer Slide Time: 15:49)



Memory Management

- To execute a program all (or part) of the instructions must be in memory
- All (or part) of the data that is needed by the program must be in memory.
- Memory management determines what is in memory and when
  - Optimizing CPU utilization and computer response to users
- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

Then, there is another module in the operating system which is known as the memory management module. So, here the job is to execute a program, all of the instructions must be in memory, so, this is the first thing. So, for executing a program, the program must be in the main memory because CPU can talk only to the main memory, it cannot talk to the secondary storage like that. So, the program must be there in the main memory.

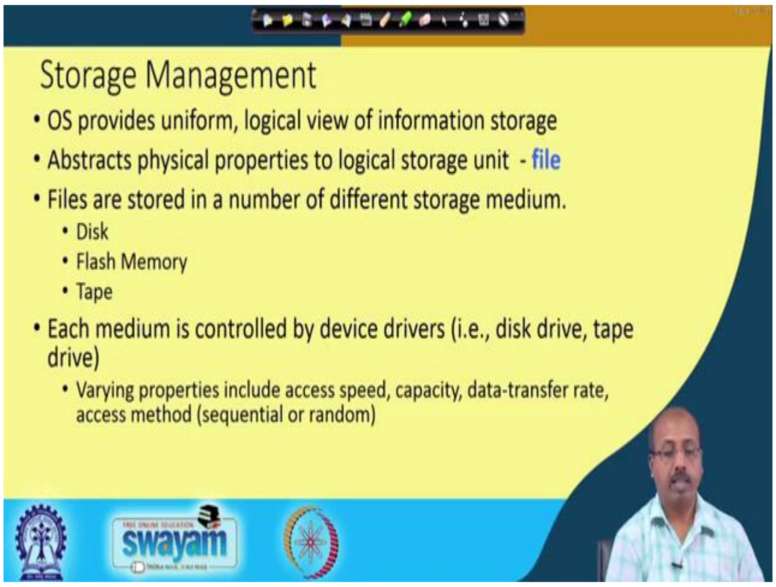
So, ideally the entire program be in the main memory or it may so, happen that entire program cannot be loaded. So, we'll see some schemes in which we'll find that even if we load a part of the program, it can be executed by the system by the CPU. So, this is the first requirement, second requirement is that, all of the data that is needed by the program must be in memory. So, apart from the program code so, all that data part should also be there in the memory.

Memory management determines what is in memory and when. So, it has to decide like we have got the requirement that for executing an instruction, the instruction code must be there in the memory and the variable on which this instruction operates, that must be present in the memory.

So, the memory management policy so, it has to decide that for smooth running of all the processes which are the program segments that should be there in the memory, which are the data segments that be in the memory and at what time those values be available. So, if a program is not executing now , so, it is not mandatory that I should have the corresponding program and data variables loaded with the memory. So, the memory manager may decide, no I will not put this part in the memory now, I will use this space for loading the data and programs for some other program which is currently being executed. So, that way the memory management will decide what should be there in the memory and when should it be in the memory.

So, memory management activity, so, they talk about, keeping track of which parts of memory are currently being used and by whom, deciding which processes and data to move into and out of memory so, and allocating and deallocating memory space as needed. So, these are the major responsibilities of memory management. So, operating system design, if you look into, so they can be divided into a number of modules. So, one of them is the process management module that we have seen its responsibilities in the last slide. So, this slide is telling us like for a memory management part. So, this is another responsibility of the operating system. So, what are the essential jobs that it has to do.

(Refer Slide Time: 18:31)



Storage Management

- OS provides uniform, logical view of information storage
- Abstracts physical properties to logical storage unit - file
- Files are stored in a number of different storage medium.
  - Disk
  - Flash Memory
  - Tape
- Each medium is controlled by device drivers (i.e., disk drive, tape drive)
  - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)

Then we have got the storage management part. So, OS provide a uniform logical view of information storage. So, when a program is executing, so, it is not understanding whether the content is coming from main memory or the content was not there in memory it is coming from the secondary storage. So, it will not understand that. But, physically the program may be, some part of the program may be in the main memory, some part in the secondary storage and all that.

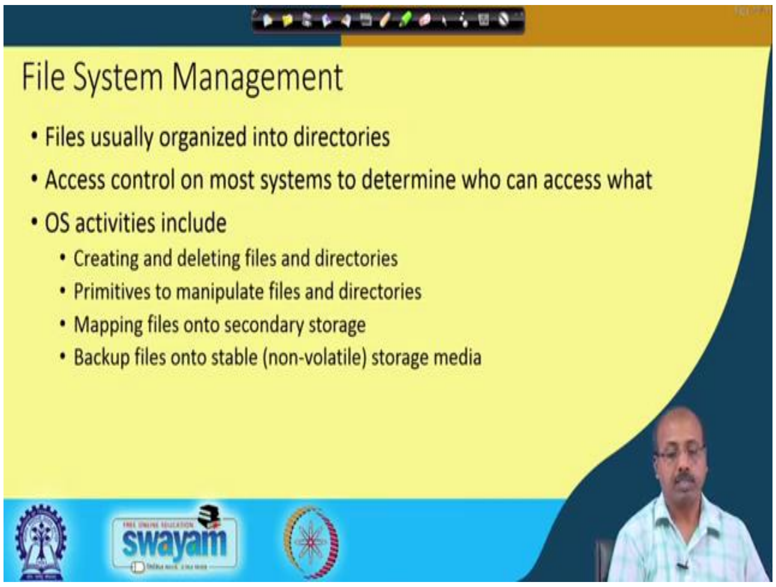
So, logically whole thing is a single storage, main memory, CPU register, then your disks, so, they are all part of this logical storage. So, this access to this logical storage is uniform. So, program, when it is executing, it sees everything as memory location and it just tells that I need the content of this particular memory address and that OS has to see like whether that is available. And if it is available, it will give the value. If it is not available then it has to get the value fetched from the secondary storage and then return the value. So, this way we have got the different storage management problems.

So, this storage management modules they will abstract physical properties of logical storage unit. So, that is called a file. So, it will consist of a set of files and the whenever it has to access some memory location, so, maybe ultimately it has to look into a file. So, files will be stored in a number of different storage medium, it may be in the disk, may be in the flash memory, may be in that tape so, innumerable different storage media that I can have for storing the files.

And each medium is controlled by device drivers so, each device is different like the operation of a disk storage is not the same as the operation of flash memory. So, but as an OS designer so, if we want to have different views for each of them, so, designing operating system becomes complex. So, what is done, we have got the device drivers that can talk to the corresponding devices and that makes the access to all these devices uniform. So, as OS designer it has to send messages to the disk driver, this device driver and this device driver will in turn translate it into command specific for the device. So, that way from the OS view point so, this is uniform view.

So, as you go that the uniform view is provided by the device driver and at the next level the device driver, so, they will have the interface the devices and that will provide the exact commands for doing it. So, there can be proper varying properties that can include access speed, capacity, data transfer rate, data access method, etc. For example, the disk so, their access is random. So, disk head can be moved to any place and the disk can rotate and the head can go to one place. Whereas, for tape type of device so, there the access is sequential. So, if you have accessed location x, at the next time the head will be at position x plus 1 only, then, at x plus 2 only so, that way it goes by location by location so, that way the tape access is sequential whereas, disk access is random in nature. Capacity wise, if you say, then, tape capacity may be huge but the disk capacity may not be that much. So, data transfer rate again, so, disk will be better than tape. So, like that, the storage is there, but the storage management becomes an issue. So, we have to think about the policies by which we can take care of this difference in speed and the difference in behavior of these modules and all.

(Refer Slide Time: 22:21)



File System Management

- Files usually organized into directories
- Access control on most systems to determine who can access what
- OS activities include
  - Creating and deleting files and directories
  - Primitives to manipulate files and directories
  - Mapping files onto secondary storage
  - Backup files onto stable (non-volatile) storage media

So, this will be talking about the file system management. So, files, this is another module of the operating system that we have. So, files are usually organized into directories or folders.

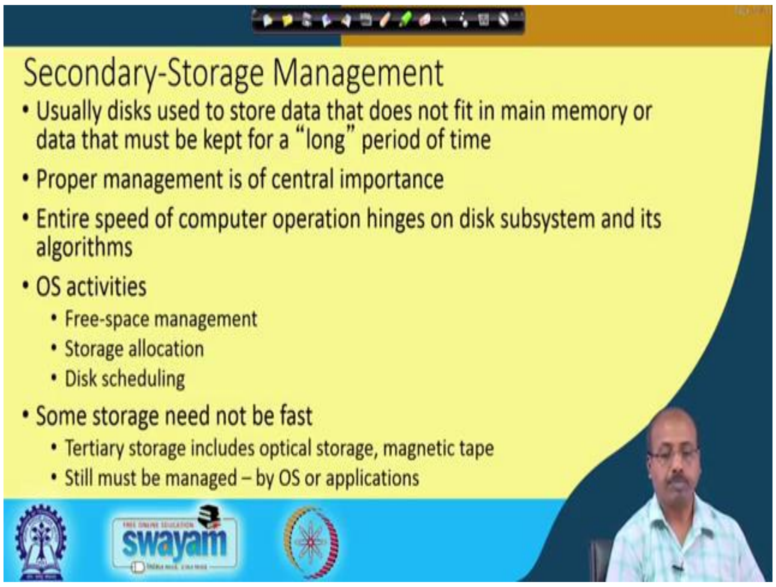
So, access control on most systems to determine who can access what. So, access control is a set of rules that will say like who is allowed to access which file. So, this is very important because when you got multi programmed system or multiprocessing system so, multiple users are there. So, one user should not willingly or unintentionally modify some other users program code or file etc. or data like that. So, whether that is happening or not so, OS should some control over that. So, file management policy so, that uses this access control mechanism to do that.

So, OS activities in file management system, it includes creating and deleting files and directories, primitives to manipulate files and directories, mapping files onto secondary storage, backup files onto stable storage media. So, when a program is executing all on a sudden, it may try to open a file. So, that facility has to be provided. Similarly, a program may want to read or write into the file so, that facility has to be provided. Then for organizing the files into a proper structure, directory structure may be provided.

Then given a filename, how do I locate it? So, maybe I, in some directory, have got a file. Now, in the physical, if it is a disk storage so, ultimately it does not have that directory and everything there. So, what it has is, some sector addresses, that file starts at some sector address. So, how to locate the sector address for a particular file like a program has asked the computer to open file x y z. Now, what is the corresponding disk sector number that has to be known so, that is the problem of mapping.

So, mapping file name on to the storage address, so, that is one important thing to be done and some time. So, we need to take backup so, because if there is a system crash and all then all the contents will be lost. So, it is the requirement that we have got some backup of the system in some stable storage.

(Refer Slide Time: 24:48)



Secondary-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a "long" period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Free-space management
  - Storage allocation
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Still must be managed – by OS or applications

So, the secondary storage management part so, usually disks are used to store data that does not fit in main memory or data that must be kept for a "long" period of time so, they are kept in the disk.

Then proper management is of central importance, because disk is much slower than main memory. So, when I say that, some part of my data did not fit in the main memory so, that is why I have kept it in the secondary storage. Then, I must ensure that when that data is needed, that is taken back into the main memory and that has to be done a bit efficient fashion, because if I start accessing the secondary storage when that particular data is referred to, that may increase the access time.

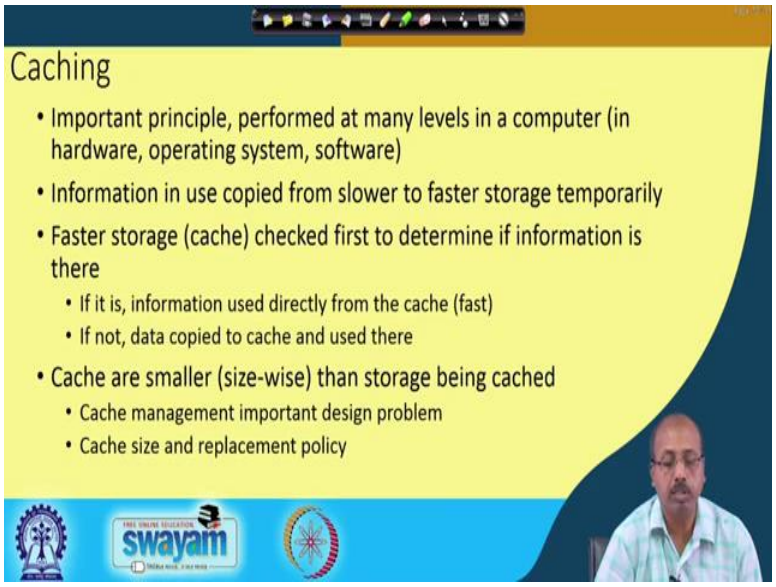
So, if I can be a bit proactive, if I can just get the data a bit earlier, then we can find that data in time. So, that way this storage management becomes an important issue and when do you get the data from the secondary storage to the main memory, so, that is a problem. So, proper management is of central importance. Entire speed of computer operation hinges on the disk subsystem and it is algorithm. So, the disk access is a very important problem that we will see when we come to the disk management portion.

So, as far as o s activities are concerned so, it includes free space management. So, files are getting created and deleted from the system. So, that create some free spaces on the disk, now after sometime that entire disk maybe fragmented into this small-small free spaces that we have. So, that stops us from using all this free spaces together. So, this free space management is a very important issue. Then storage allocation, so, how do you allocate disk for to a process that get wants to create a file? So, that storage allocation is a problem.

Then disk scheduling, so, this talks about if there are multiple request to access the disk from different processes, then in which sequence we provide those accesses, because this is an electromechanical devices so, disk head movement will take time. So, if it has to go from track number 1 to 10, then it will take some good amount of time for doing that. So, if I can schedule all my request in such a fashion that this movement is minimized then I will get a better performance from the system. So, this disk scheduling is an important problem.

Some storage need not be fast, because like the tertiary storage like optical storage, magnetic tape they need not be very fast, but capacity should be huge but still they need to be managed by the OS or applications.

(Refer Slide Time: 27:33)



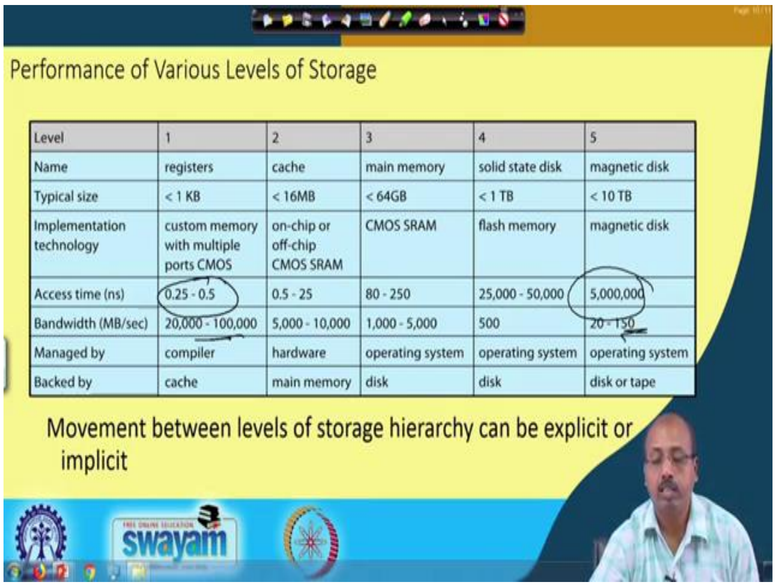
Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache are smaller (size-wise) than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Then we have got the concept of cache memory. So, this an important principle, because data may be kept at several levels and cache is something that is closest to the CPU. So, that way when this cache is accessed then may be if the data is available in the cache then the main memory access or the secondary storage access so, that can be avoided.

So, that way I can have performance improvements, so, it provides faster storage and cache size is small. Naturally that cache size will be small because the cost of the cache is high. So, which data you store in the cache and how do you access it so, that will determine the management policy of the cache and that will be determining the performance of the system.

(Refer Slide Time: 28:25)



Performance of Various Levels of Storage

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

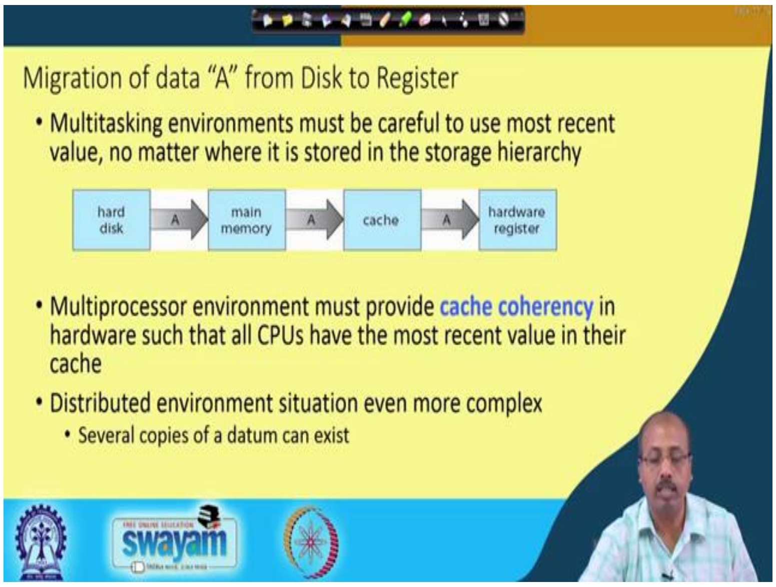
Movement between levels of storage hierarchy can be explicit or implicit

So, if you look into various levels of storage then they have CPU registers typical size is less than 1 kilobyte, cache it is 16 megabyte, main memory 64 gigabyte, then solid state disk 1 terabyte, magnetic disk less than 10 terabyte like that. So, sizes actually increasing in the direction from register to magnetic disk.

On the other hand, implementation technology, there are different implementation technologies, access time is an important factor. So, if you see that access time of the CPU registers so, they are very small in the range of nanoseconds whereas, if you go to magnetic disk so, this is becoming pretty high so that is there. Bandwidth, how many megabyte per second you can transfer so, CPU register it is the highest and as you go to magnetic disks so it is low.

So, depending upon the type of data that you have and how important it is, how frequently it is needed. So, you may decide the level at which you will be keeping that particular data. So, this decision has to be taken by the operating system.

(Refer Slide Time: 29:39)



Migration of data "A" from Disk to Register

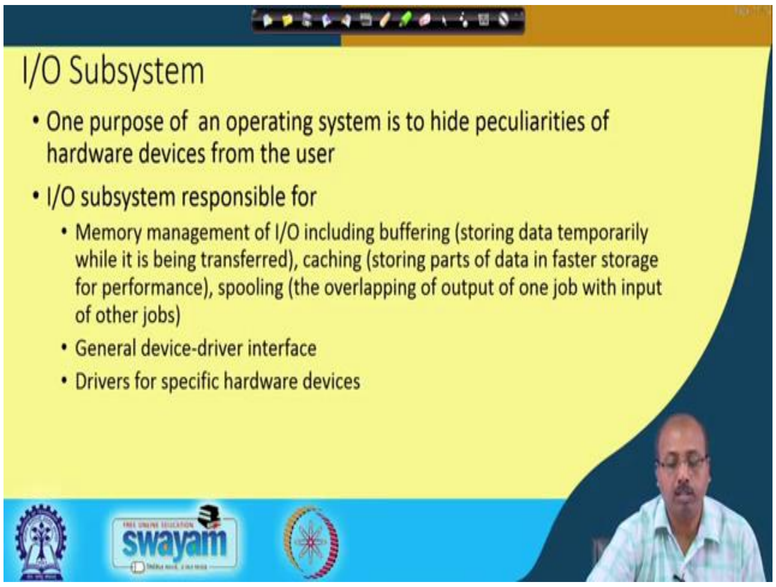
- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy

$$\begin{tikzcd}\boxed{\text{hard disk}} \arrow[r, "A"] & \boxed{\text{main memory}} \arrow[r, "A"] & \boxed{\text{cache}} \arrow[r, "A"] & \boxed{\text{hardware register}}\end{tikzcd}$$

- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist

Next we have. So, these the migration data from hard disk to main memory, so this I have already discussed in this cache coherence problem that can come because the data item should be consistent across all the levels.

(Refer Slide Time: 29:50)

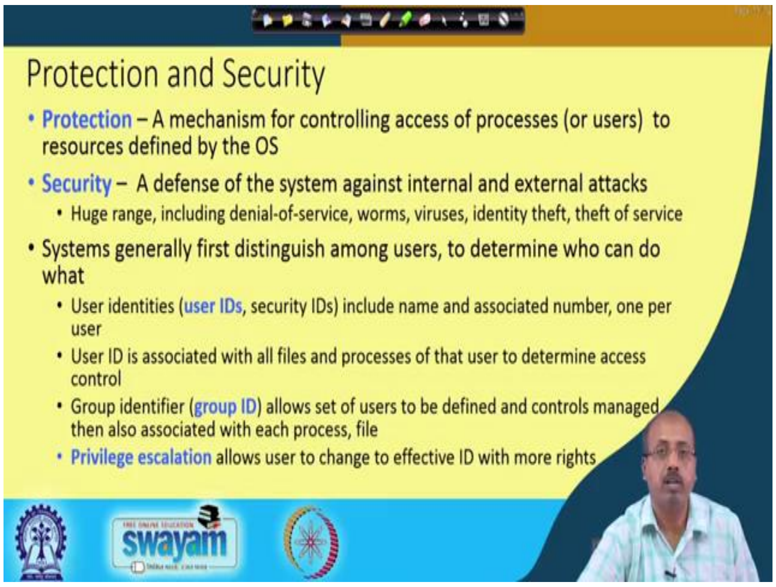


I/O Subsystem

- One purpose of an operating system is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

Then we have called this IO subsystem. So, we have got this, the IO devices are there in the system. So, as an user of the system so, you should not be looking into this each IO device separately. So, I should have an uniform interface. So, the operating system should provide me a facility by which this uniform IO device access can be met.

(Refer Slide Time: 30:16)



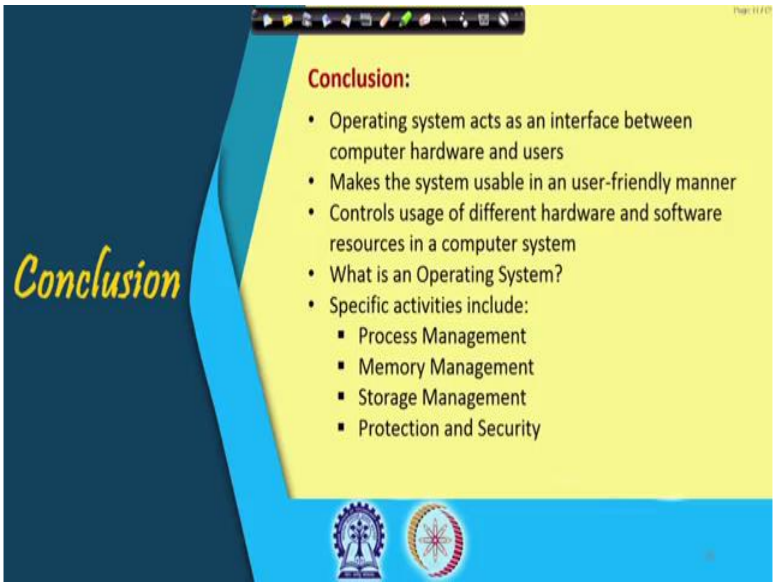
Protection and Security

- **Protection** – A mechanism for controlling access of processes (or users) to resources defined by the OS
- **Security** – A defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID is associated with all files and processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

Then there are protection security mechanism; as I already said that protection has to be provided by the operating system so, that for multiprocessing system. So, I should not be getting a multiple users.

One user should not be getting access to the data by other user or program of other user. Similarly security has to be provided in terms of user ids and all so will be going to this security protection issues later.

(Refer Slide Time: 30:42)



Conclusion

Conclusion:

- Operating system acts as an interface between computer hardware and users
- Makes the system usable in an user-friendly manner
- Controls usage of different hardware and software resources in a computer system
- What is an Operating System?
- Specific activities include:
  - Process Management
  - Memory Management
  - Storage Management
  - Protection and Security

So, to conclude so, operating system it acts as an interface between computer hardware and users. It makes the system usable in a user friendly manner, controls usage of different hardware and software resources in a computer system. So, we have discussed about what is an operating system and we have seen that the specific activities that are there are process management, memory management, storage management, protection and security. So, in successive classes so, will be looking into these thing one by one and continue with that.