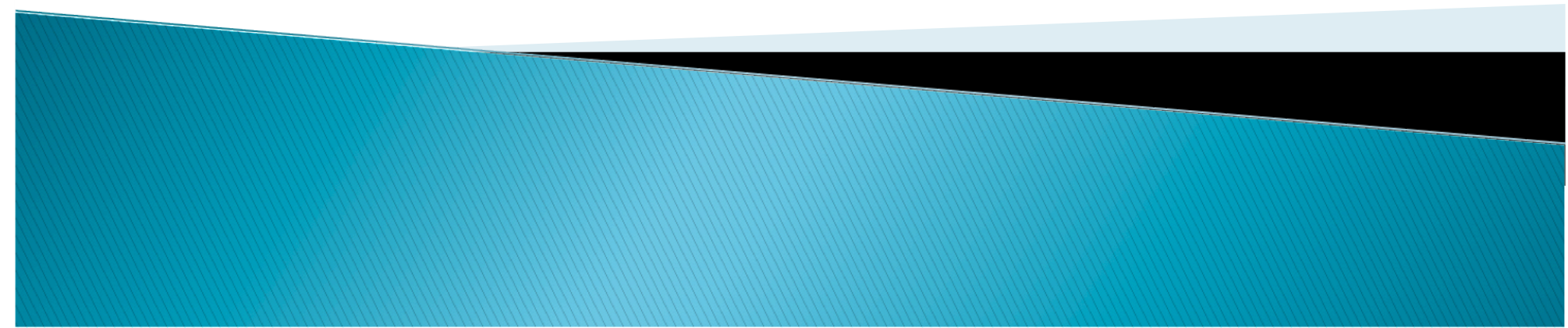
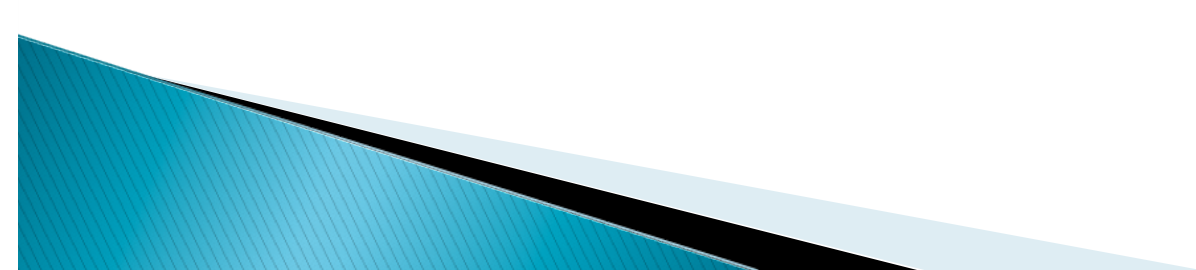


Chapter 3: Processes

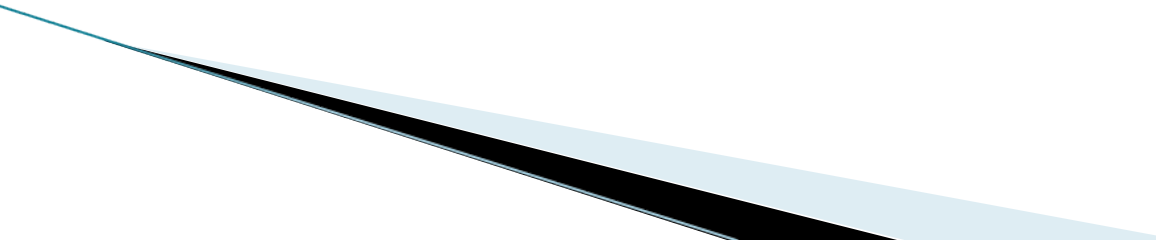


Process Scheduling

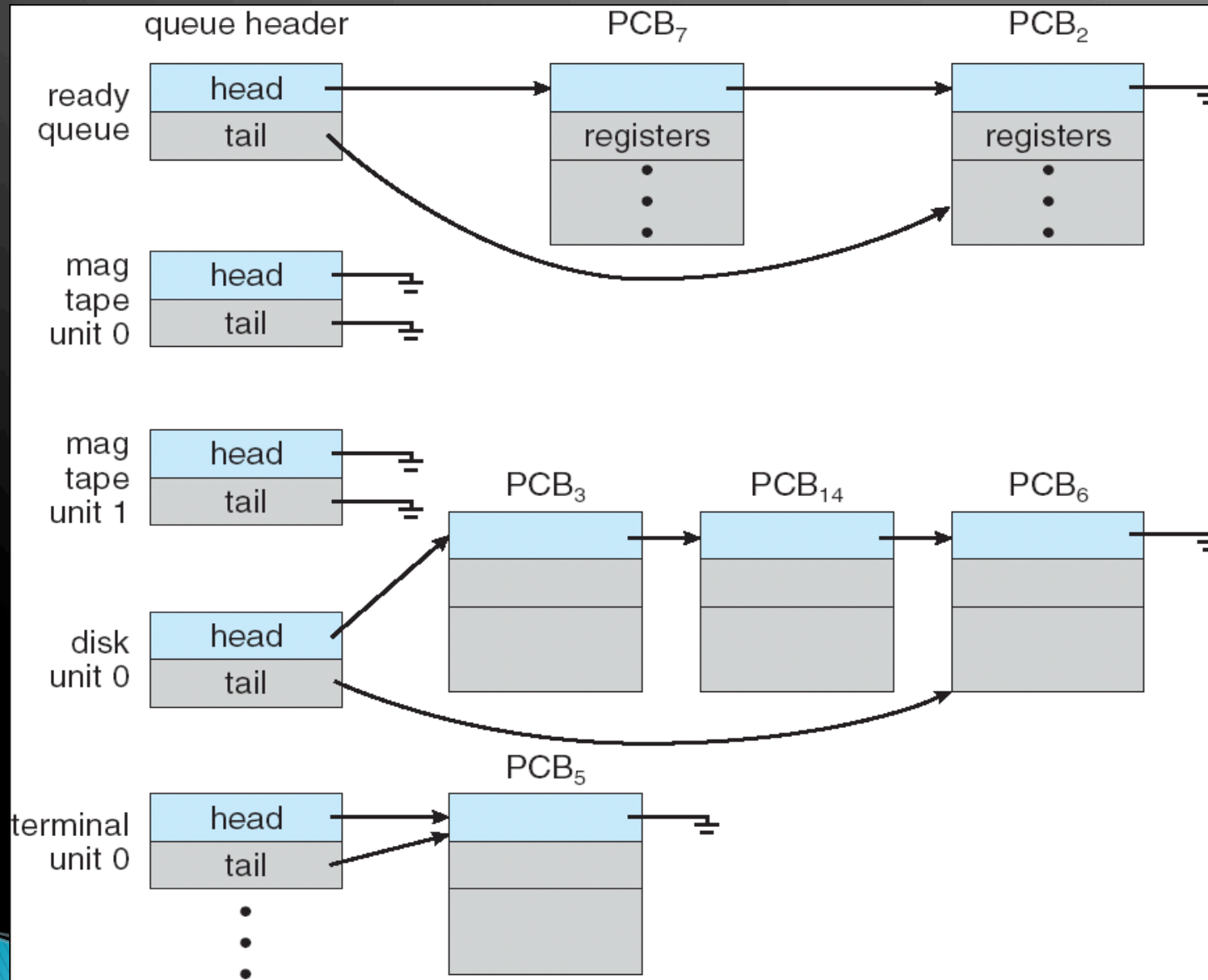
- The objective of multiprogramming is to have some process running at all times, to maximize CPU utilization.
- For a single-processor system, there will never be more than one running process.
- If there are more processes, the rest will have to wait until the CPU is free and can be rescheduled.



Process Scheduling Queues

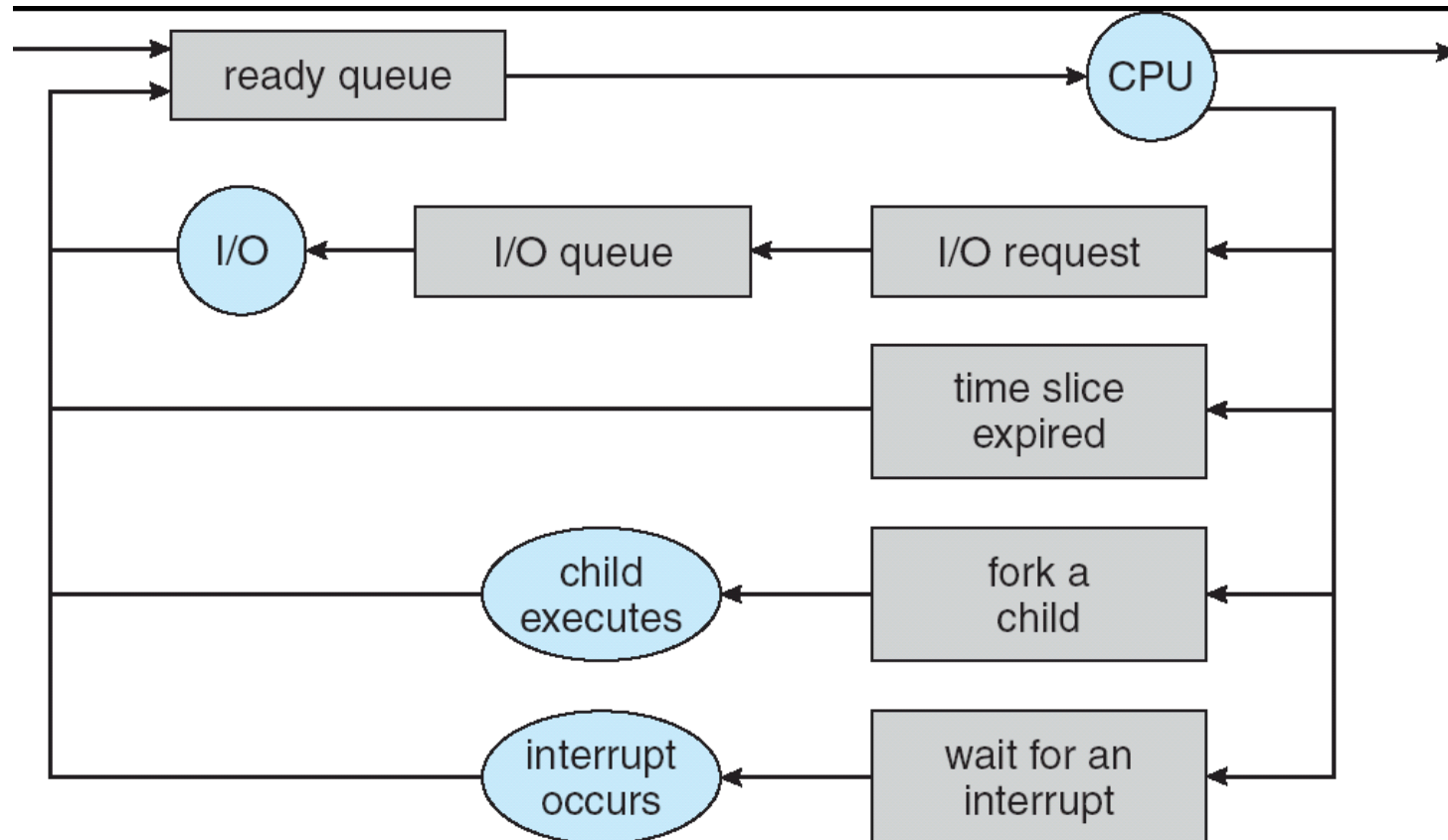
- As processes enter the system, they are put into a **job queue**, which consists of all processes in the system.
 - The processes that are residing in main memory and are ready and waiting to execute are kept on a list called the **ready queue**.
 - This queue is generally stored as a linked list. A ready-queue header contains pointers to the first and final PCBs in the list. Each PCB includes a pointer field that points to the next PCB in the ready queue.
 - Suppose the process makes an I/O request to a shared device, such as a disk. Since there are many processes in the system, the disk may be busy with the I/O request of some other process.
 - The process therefore may have to wait for the disk. The list of processes waiting for a particular I/O device is called a device queue. Each device has its own **device queue**.
- 

Ready Queue And Various I/O Device Queues

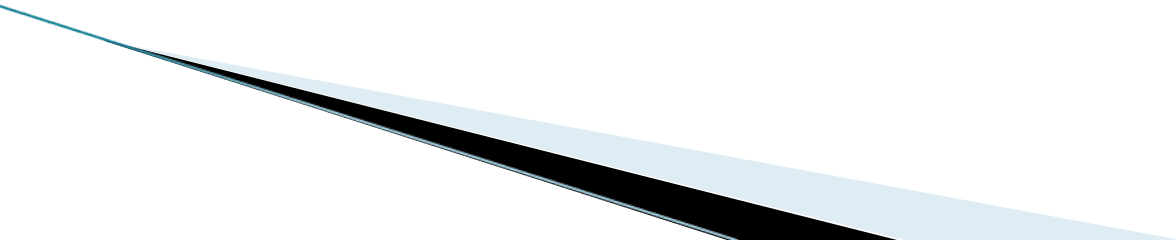


Process Scheduling Queues (cntd)

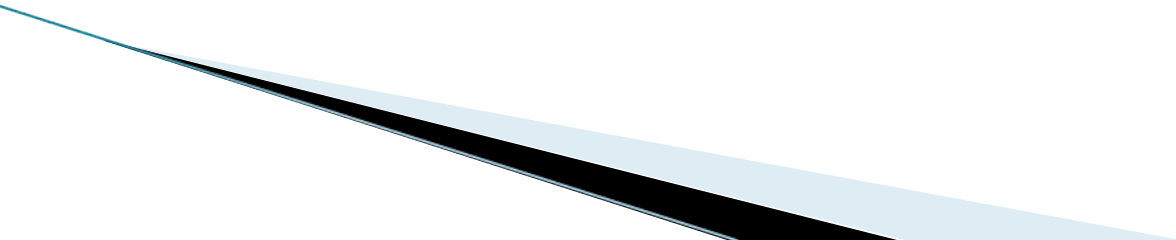
- A common representation for a discussion of process scheduling is a queuing diagram



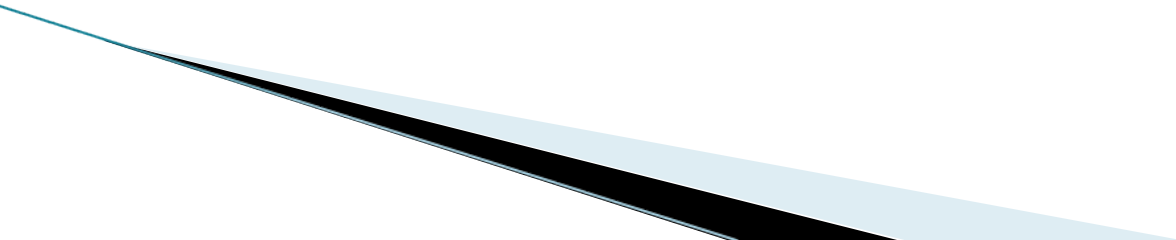
Process Scheduling Queues

- The circles represent the resources that serve the queues, and the arrows indicate the flow of processes in the system.
 - A new process is initially put in the ready queue waiting to be selected for execution.
 - Once the process is allocated the CPU and is executing, one of several events could occur:
 - The process could issue an I/O request and then be placed in an I/O queue.
 - The process could create a new subprocess and wait for the subprocess's termination. The parent process goes into waiting state.
 - The process could be removed forcibly from the CPU, as a result of an interrupt, and be put back in the ready queue.
 - A process continues this cycle until it terminates, at which time it is removed from all queues and has its PCB and resources deallocated.
- 

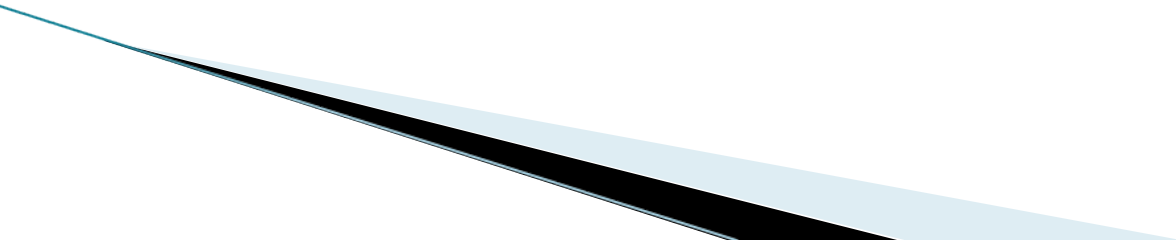
Schedulers

- A process migrates among the various scheduling queues throughout its lifetime.
 - The operating system must select, for scheduling purposes, processes from these queues in some fashion. The selection process is carried out by the appropriate scheduler from these queues in some fashion.
 - A **long-term scheduler** is typical of a batch system. It runs infrequently, (such as when one process ends selecting one more to be loaded in from disk in its place), and can afford to take the time to implement intelligent and advanced scheduling algorithms.
 - The **short-term scheduler**, or CPU Scheduler, runs very frequently, on the order of 100 milliseconds, and must very quickly swap one process out of the CPU and swap in another one.
- 

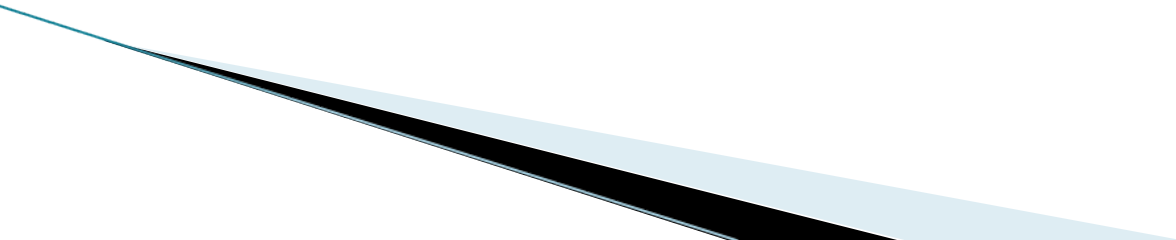
Schedulers (cntd)

- In general, most processes can be described as either I/O bound or CPU bound. An
 - I/O-bound process is one that spends more of its time doing I/O than it spends doing computations.
 - A CPU-bound process, in contrast, generates I/O requests infrequently, using more of its time doing computations.
 - It is important that the long-term scheduler make a careful selection. If all processes are CPU bound, the I/O waiting queue will almost always be empty, devices will go unused, and again the system will be unbalanced. The system with the best performance will thus have a combination of CPU-bound and I/O-bound processes.
- 

Schedulers (cntd)

- On some systems, the long-term scheduler may be absent or minimal.
 - For example, time-sharing systems such as UNIX and Microsoft Windows systems often have no long-term scheduler but simply put every new process in memory for the short-term scheduler.
 - Time-sharing systems, may introduce an additional, intermediate level of scheduling known as **medium-term scheduler**.
 - The key idea behind a medium-term scheduler is that sometimes it can be advantageous to remove processes from memory and thus reduce the degree of multiprogramming.
- 

Schedulers (cntd)

- Later, the process can be reintroduced into memory, and its execution can be continued where it left off. This scheme is called swapping.
 - The process is swapped out, and is later swapped in, by the medium-term scheduler.
 - Swapping may be necessary to improve the process mix (IO bound vs Cpu bound) or available memory is exhausted and new process needs to be run requiring memory to be freed up.
- 

Addition of Medium Term Scheduling

