IO Management and Disk Scheduling

Chapter 11

Categories of I/O Devices

- **Human readable**
  - used to communicate with the user
  - video display terminals
  - keyboard
  - mouse
  - printer

Categories of I/O Devices

- **Machine readable**
  - used to communicate with electronic equipment
  - disk drives
  - tape drives
  - controllers
  - actuators

Categories of I/O Devices

- **Communication**
  - used to communicate with remote devices
  - digital line drivers
  - modems

Differences in I/O Devices

- **Data Transfer Rate**
- **Application**
  - disk used to store files must have file-management software
  - disk used to store virtual memory pages needs special hardware to support it
  - terminal used by system administrator may have a higher priority

Differences in I/O Devices

- **Complexity of control**
- **Unit of transfer**
  - data may be transferred as a stream of bytes for a terminal or in larger blocks for a disk
- **Data representation**
  - encoding schemes
- **Error conditions**
  - devices respond to errors differently

Techniques for Performing I/O

- **Programmed I/O**
  - process is busy-waiting for the operation to complete
- **Interrupt-driven I/O**
  - I/O command is issued
  - processor continues executing instructions
  - I/O module sends an interrupt when done

Techniques for Performing I/O

- **Direct Memory Access (DMA)**
  - DMA module controls exchange of data between main memory and the I/O device
  - processor interrupted only after entire block has been transferred

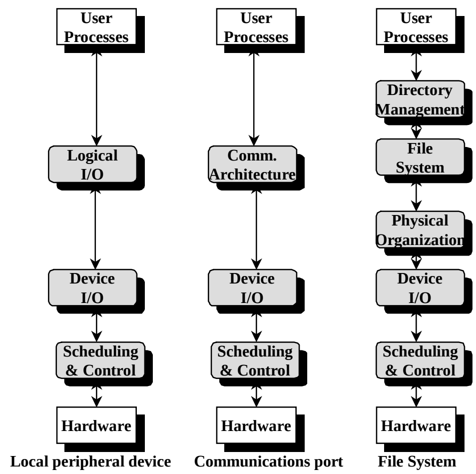
Operating System Design Objectives

- I/O extremely slow compared to main memory
- Use of multiprogramming allows for some processes to be waiting on I/O while another process executes
- I/O cannot keep up with processor speed
- Swapping is used to bring in additional Ready processes which is an I/O operation
- Efficiency is an important issue

Operating System Design Objectives

- Efficiency is an important issue
- Desirable to handle all I/O devices in a uniform manner
- Hide most of the details of device I/O in lower-level routines so that processes and upper levels see devices in general terms such as Read, Write, Open, and Close
- Generality is an important issue

A Model of I/O Organization



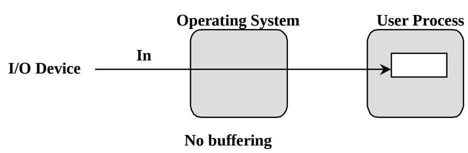
I/O Buffering

- **Reasons for buffering**
  - Processes must wait for I/O to complete before proceeding
  - Certain pages must remain in main memory during I/O

I/O Buffering

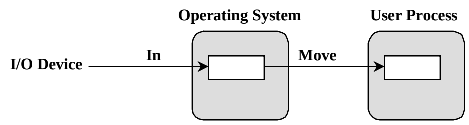
- **Block-oriented**
  - information is stored in fixed sized blocks
  - transfers are made a block at a time
  - used for disks and tapes
- **Stream-oriented**
  - transfer information as a stream of bytes
  - used for terminals, printers, communication ports, mouse, and most other devices that are not secondary storage

No Buffering



Single Buffer

- **Operating system assigns a buffer in main memory for an I/O request**
- **Block-oriented**
  - input transfers made to buffer
  - block moved to user space when needed
  - another block is moved into the buffer
    - read ahead



Single buffering

Single Buffer

- **Block-oriented**
  - user process can process one block of data while next block is read in
  - swapping can occur since input is taking place in system memory, not user memory
  - operating system keeps track of assignment of system buffers to user processes
  - output is accomplished by the user process writing a block to the buffer and later actually written out

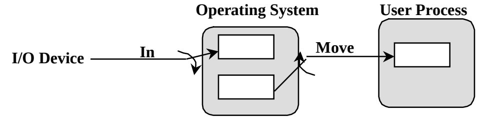
Single Buffer

- **Stream-oriented**
  - used a line at time
  - user input from a terminal is one line at a time with carriage return signaling the end of the line
  - output to the terminal is one line at a time

369

Double Buffer

- **Use two system buffers instead of one**
- **A process can transfer data to or from one buffer while the operating system empties or fills the other buffer**

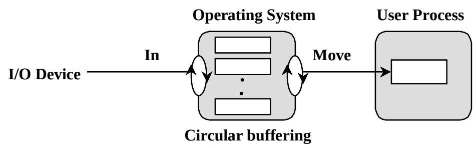


Double buffering

370

Circular Buffer

- More than two buffers are used
- Each individual buffer is one unit in a circular buffer
- Used when I/O operation must keep up with process



Disk Performance Parameters

- To read or write, the disk head must be positioned at the desired track and at the beginning of the desired sector
- Seek time
  - time it takes to position the head at the desired track
- Rotational delay or rotational latency
  - time its takes until desired sector is rotated to line up with the head

Disk Performance Parameters

- **Access time**
  - sum of seek time and rotational delay
  - the time it takes to get in position to read or write
- **Data transfer occurs as the sector moves under the head**
- **Data transfer for an entire file is faster when the file is stored in the same cylinder and in adjacent sectors**

Disk Scheduling Policies

- **Seek time is the reason for differences in performance**
- **For a single disk there will be a number of I/O requests**
- **If requests are selected randomly, we will get the worst possible performance**

Disk Scheduling Policies

- **First-in, first-out (FIFO)**
  - process request sequentially
  - fair to all processes
  - approaches random scheduling in performance if there are many processes

Disk Scheduling Policies

- **Priority**
  - goal is not to optimize disk use but to meet other objectives
  - short batch jobs may have higher priority
  - provide good interactive response time

Disk Scheduling Policies

- **Last-in, first-out**
  - good for transaction processing systems
    - the device is given to the most recent user so there should be little arm movement
  - possibility of starvation since a job may never regain the head of the line

Disk Scheduling Policies

- **Shortest Service Time First**
  - select the disk I/O request that requires the least movement of the disk arm from its current position
  - always choose the minimum Seek time

Disk Scheduling Policies

- **SCAN**
  - arm moves in one direction only, satisfying all outstanding requests until it reaches the last track in that direction
  - direction is reversed

379

Disk Scheduling Policies

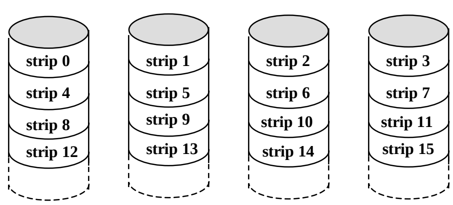
- **C-SCAN**
  - restricts scanning to one direction only
  - when the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again

380

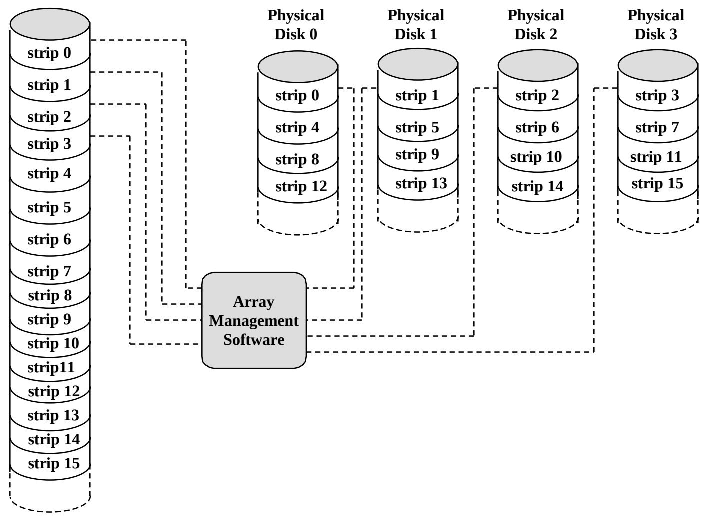
Disk Scheduling Policies

- **N-step-SCAN**
  - segments the disk request queue into subqueues of length N
  - subqueues are process one at a time, using SCAN
  - new requests added to other queue when queue is processed
- **FSCAN**
  - two queues
  - one queue is empty for new request

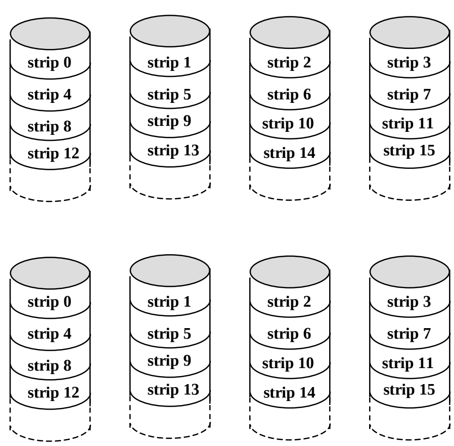
RAID 0 (non-redundant)



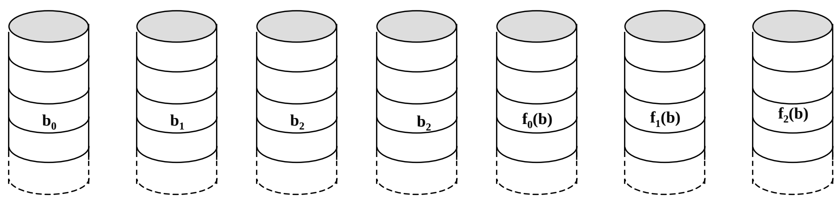
Data Mapping for RAID Level 0 Array



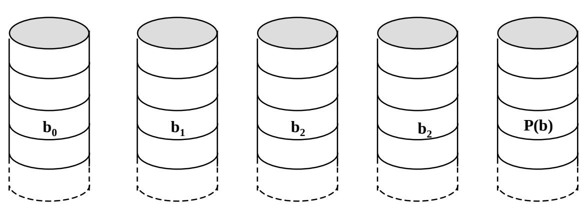
RAID 1 (mirrored)



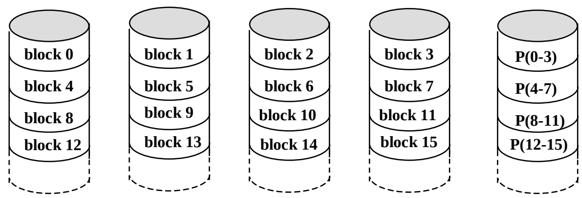
RAID 2 (redundancy through Hamming code)



RAID 3 (bit-interleaved parity)



RAID 4 (block-level parity)



RAID 5 (block-level distributed parity)