MODERN OPERATING SYSTEMS

ANDREW S. TANENBAUM

Chapter 5
Input & Output

1

Content

- I/O hardware (classification, device drivers)
- I/O techniques (programmed, interrupt driven, DMA)
- Structuring I/O software
- Disks (performance, arm scheduling, common disk errors)
- RAID configurations

2

2

Categories of I/O Devices

- Human readable
  - Used to communicate with the user
  - Printers
  - Video display terminals
    - Display
    - Keyboard
    - Mouse

3

Categories of I/O Devices

- Machine readable
  - Used to communicate with electronic equipment
  - Disk and tap drives
  - Sensors
  - Controllers
  - Actuators

4

Categories of I/O Devices

- Communication
  - Used to communicate with remote devices
  - Digital line drivers
  - Modems

5

Differences in I/O Devices

- Data rate
  - May be differences of several orders of magnitude between the data transfer rates

6

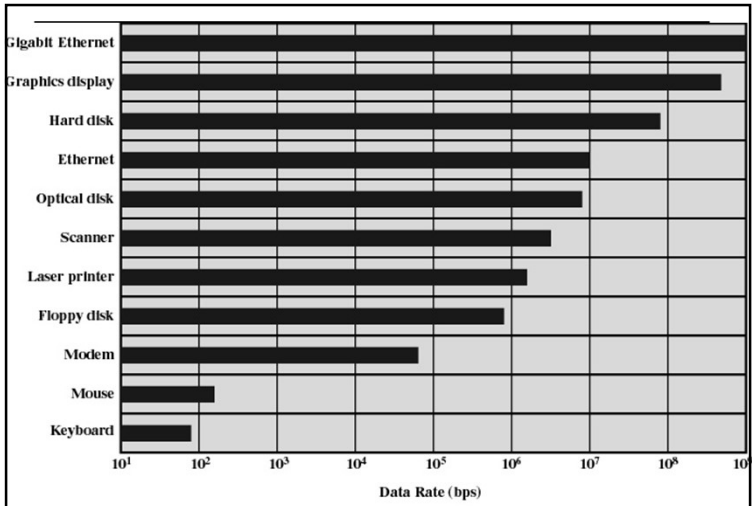


Figure 11.1 Typical I/O Device Data Rates

7

Differences in I/O Devices

- Application
  - Disk used to store files requires file-management software
  - Disk used to store virtual memory pages needs special hardware and software to support it
  - Terminal used by system administrator may have a higher priority

8

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

9

Differences in I/O Devices

- 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

10

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

11

I/O Hardware

- Classification of I/O devices
- Device controllers

12

12

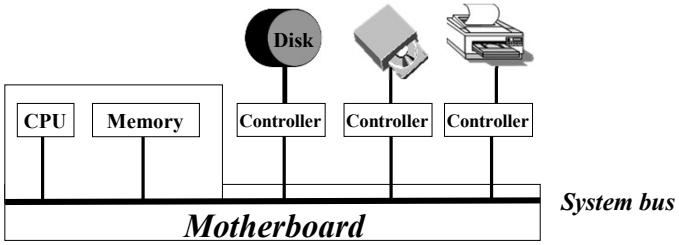
Classification of I/O Devices

- Block devices
  - Information is stored in fixed size blocks
  - Block sizes range from 512-32768 bytes
  - I/O is done by reading/writing blocks
  - Hard disks, floppies, CD ROMS, tapes are in this category
- Character devices
  - I/O is done as characters (ie., no blocking)
  - Terminals, printers, mouse, joysticks are in this category

13

13

Device Controllers



- A controller is an electronic card (PC's) or a unit (mainframes) which performs blocking (from serial bit stream), analog signal generation (to move the disk arm, to drive CRT tubes in screens), execution of I/O commands

14

14

Evolution of the I/O Function

- Processor directly controls a peripheral device
- Controller or I/O module is added
  - Processor uses programmed I/O without interrupts
  - Processor does not need to handle details of external devices

15

Evolution of the I/O Function

- Controller or I/O module with interrupts
  - Processor does not spend time waiting for an I/O operation to be performed
- Direct Memory Access
  - Blocks of data are moved into memory without involving the processor
  - Processor involved at beginning and end only

16

Evolution of the I/O Function

- I/O module is a separate processor
- I/O processor
  - I/O module has its own local memory
  - Its a computer in its own right

17

I/O Techniques

- Programmed I/O
- Interrupt-driven I/O
- Direct memory access (DMA)

18

18

Programmed I/O

- The processor issues an I/O command on behalf of a process to an I/O module
- The process ***busy-waits*** for the operation to be completed before proceeding

19

19

Interrupt-driven I/O

- Processor issues an I/O command on behalf of a process
- Process is suspended and the I/O starts
- Processor may execute another process
- Controller reads a block from the drive serially, bit by bit into controller's internal buffer. A checksum is computed to verify no reading errors
- When I/O is finished, the processor is interrupted to notify that the I/O is over
- OS reads controller's buffer a byte (or word) at a time into a memory buffer.

20

20

Direct Memory Access (DMA)

- A DMA module controls the exchange of data between main memory and an I/O device
- The processor sends a request for the transfer of a block of data to the DMA module (block address, memory address and number of bytes to transfer) and continues with other work
- DMA module interrupts the processor when the entire block has been transferred
- When OS takes over, it does not have to copy the disk block to memory; it is already there.

21

21

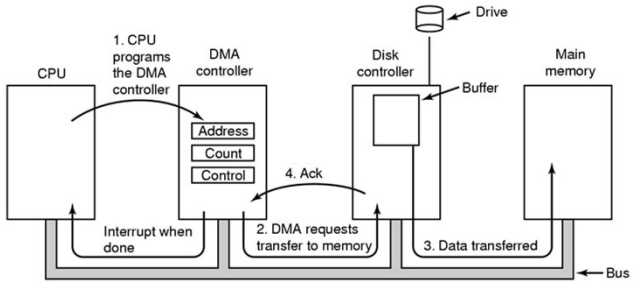
DMA (Cont.)

- DMA unit is capable of transferring data straight from memory to the I/O device
- ***Cycle Stealing***: DMA unit makes the CPU unable to use the bus until the DMA unit has finished
- Instruction execution cycle is suspended, ***NOT*** interrupted
- DMA has less number of interrupts (usually one when I/O is finished to acknowledge). Interrupt driven I/O may need one interrupt for each character if device has no internal buffers.

22

22

Direct Memory Access (DMA)

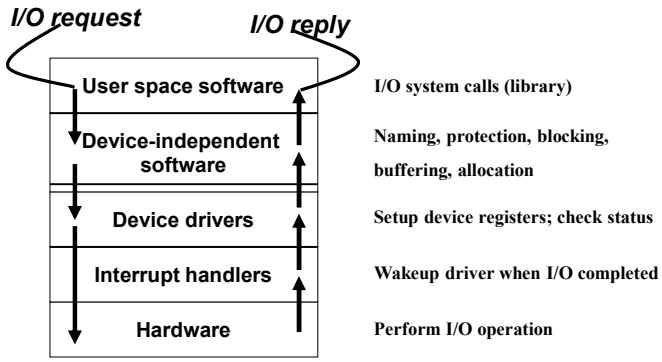


Operation of a DMA transfer

23

23

Structuring I/O Software



24

24

User-Space I/O Software

- Library of I/O procedures (ie., system calls) such as

  bytes-read = read (file_descriptor, buffer, bytes to be read)
- Spooling provides virtual I/O devices

25

25

Device-Independent I/O Software

- Uniform interface for device drivers (ie., different devices)
- Device naming
  - Mapping of symbolic device names to proper device drivers
- Device protection
  - In a multi-user system you can not let all users access all I/O devices

26

26

Device-Independent I/O Software (Cont.)

- Provide device independent block size
  - Physical block sizes for different devices may differ, so we have to provide the same logical block sizes
- Buffering
- Storage allocation on block devices such as disks
- Allocating and releasing dedicated devices such as tapes

27

27

Device-Independent I/O Software (Cont.)

- Error reporting
  - When a bad block is encountered, the driver repeats the I/O request several times and issues an error message if data can not be recovered

28

28

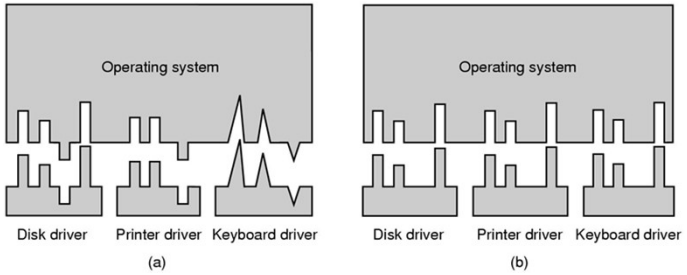
Device Drivers

- One driver per device or device class
- Device driver
  - Issues I/O commands
  - Checks the status of I/O device (eg. Floopy drive motor)
  - Queues I/O requests

29

29

Device Drivers (Cont.)



(a) Without a standard driver interface

(b) With a standard driver interface

30

30

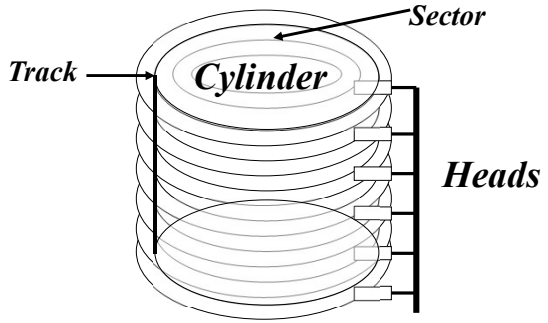
Interrupt Handlers

- When an I/O is issued, the process is suspended until I/O is finished
- When the I/O is finished, the hardware causes an interrupt and the execution is directed to a special routine (interrupt handler)
- Interrupt handler notifies the device driver which in turn passes this to the upper layers

31

31

Disks



32

32

Parameter	IBM 360-KB floppy disk	WD 18300 hard disk
Number of cylinders	40	10601
Tracks per cylinder	2	12
Sectors per track	9	281 (avg)
Sectors per disk	720	35742000
Bytes per sector	512	512
Disk capacity	360 KB	18.3 GB
Seek time (adjacent cylinders)	6 msec	0.8 msec
Seek time (average case)	77 msec	6.9 msec
Rotation time	200 msec	8.33 msec
Motor stop/start time	250 msec	20 sec
Time to transfer 1 sector	22 msec	17 µsec

Disk parameters for the original IBM PC floppy disk and a Western Digital WD 18300 hard disk

33

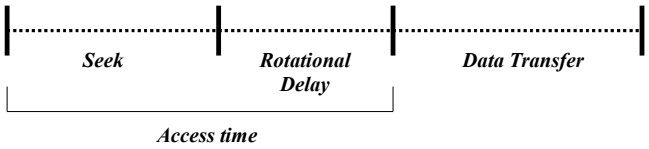
33

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 for the beginning of the sector to reach the head

34

Disk Performance Parameters



- Seek time: Time to move disk arm to the required track
- Rotational delay (rotational latency): Wait for the correct block to be under the head

35

35

Approximate Formulas for Disk Performance Parameters

- Seek time (T_s) = ***m * n + s***
  where ***m*** = a constant depending on the disk drive
  n = number of tracks traversed
  s = startup time
- Rotational delay (T_r) = 1 / (2****r***)
  where ***r*** is the rotation speed in revolutions per second
- Transfer time (T_t) = ***b*** / (***r*N***)
  where ***b*** = number of bytes to be transferred
  N = number of bytes on a track
- Average access time (T_a) = T_s + T_r + T_t

36

36

Example (From "Stallings" Book)

- Read a file of 256 sectors (or 8 tracks) from a disk drive with the following characteristics
  - Average seek time = 20 msec
  - Transfer rate = 1 Mb/s
  - Bytes per sector = 512
  - 32 sectors per track
  - Disk rotates at 3600 rpm
- Consider two cases:
  - Contiguously Stored
  - Randomly Stored

37

37

File is Stored Contiguously

- Time to read one track is
  seek + latency + data transfer (1 track - one revolution) =
  20 msec + 8.3 msec + 16.7 msec = 45 msec
- No seek time for the other 7 tracks
- All tracks = first track + other 7 tracks
  45 msec + 7 * 25 (8.3+16.7) = 220 msec

38

38

Randomly Stored

- Time to read one sector (randomly) is
  seek + latency + data transfer (1 sector) =
  20 msec + 8.3 msec + 0.5 msec = 28.8 msec
- Time to read 256 sectors = 256 * 28.8 = 7.37 seconds

39

39

Disk Scheduling Policies

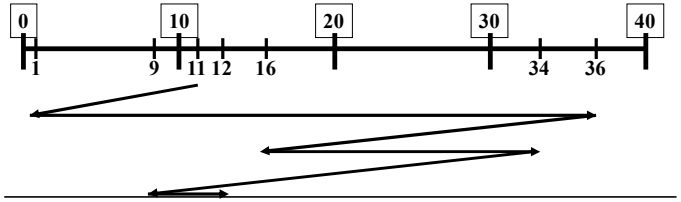
- The order in which sectors are read from the disk has a tremendous effect on I/O performance)
- Scheduling Algorithms
  - FIFO
  - SSF (Shortest seek first)
  - SCAN (Elevator algorithm)
  - C-SCAN (One-way elevator)
  - FSCAN

40

40

First in, First out (FIFO)

- Disk driver accepts request one at a time and carries them in that order
- No starvation
- Example: Requests for 1, 36, 16, 34, 9, 12 when positioned on cylinder 11 (mean movement = 18.5 cylinders)

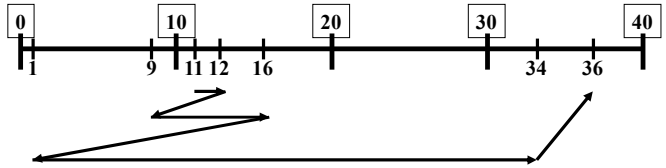


41

41

Shortest Seek First (SSF)

- Request which requires shortest seek is chosen
- Possibility of starvation (if requests are clustered)
- Same example : mean movement = 10.2 cylinders

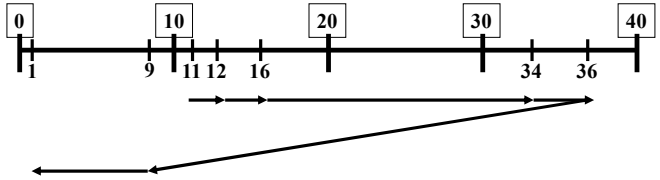


42

42

SCAN (Elevator Algorithm)

- Disk arm moves in one direction, performing all requests until no more are needed in that direction, then turns around and comes back
- Same example : mean movement = 10.0 cylinders



43

43

SCAN (Cont.)

- Favours
  - Tracks nearest to both innermost and outermost cylinders
  - Latest-arriving requests

44

44

C-SCAN (One-way Elevator)

- Modification of SCAN where scanning direction is one way only
- Once arm reaches the end it moves back to the start

45

45

FSCAN

- SSTF, SCAN and C-SCAN may suffer from "arm stickiness" (starvation for some requests)
- If multiple new requests keep arriving for the same track the arm gets "stuck"
- The solution is to maintain multiple queues

46

46

FSCAN (Cont.)

- Two queues, one being used for the scan and the other for new requests during the scan
- When a scan begins, all new requests are in one of the queues, with the other being empty
- During the scan, all new requests are put into the queue that was initially empty
- Thus, service of new requests is deferred until all the old requests have been processed

47

47

Common Disk Errors

- Programming error (e.g., request for nonexistant sector)
  - This type of error should not occur if programming (software development) is done carefully
  - If such an error is encountered, probably the only thing to do is to terminate the request and notify the user or programmer

48

48

Common Disk Errors (Cont.)

- Transient checksum error (e.g., usually caused by dust on the head. Mostly for floppies)
  - The read or write operation is repeated for a couple of times
  - If the operation is not successful the block is marked as bad (Bad CRC)
  - Re-formating may cure the error

49

49

Common Disk Errors (Cont.)

- Permanent checksum error (e.g., disk block physically damaged)
  - Bad blocks are marked so that device drivers do not access them
- Controller error (e.g., controller refuses to accept commands)

50

50

Common Disk Errors (Cont.)

- Seek error (e.g., the arm is directed to cylinder 6 but it goes to 7)
  - The disk arm is positioned on cylinders by pulses (one pulse per cylinder). When the arm reaches its destination the cylinder number is checked (written when the drive was formatted). If the arm is in a wrong position then a seek error occurs

51

51

Common Disk Errors (Cont.)

- Some controllers can correct the seek error by issuing a RECALIBRATE command
- This command moves the arm as far out as it will go to reset the arm on cylinder 0. If this does not solve the problem then the drive has to be repaired (replaced with a new one?)

52

52

RAID

(Redundant Array of Inexpensive Disks)

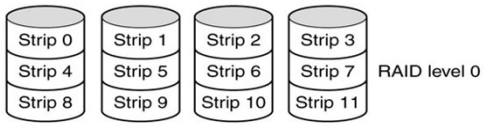
- Security (fault tolerance)
- Performance
- 0-5 levels are defined

53

53

RAID 0 Level

- A file is written (distributed) over several disks
- This permits multiple reads and writes
- Consequently speed is improved
- But no error correction

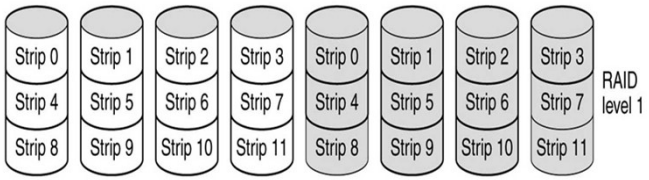


54

54

RAID 1 (Mirroring)

- A file is written on at least two drives
- The other drive becomes a mirror image of the first drive



55

55

RAID 1 (Mirroring)

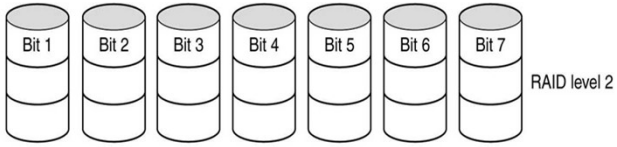
- Reading is improved because of two paths
- Writing is slower as the same data has to be written twice
- Fault tolerance is improved as the failure of two disks at the same time is low

56

56

RAID 2

- A file is distributed on several disks as in Raid 0, but Raid level 2 works on words or bytes basis
- Additional drives contains the parity information which may be used to reconstruct the file if a drive fails (***See "Hamming Codes" for error correction***),



57

57

RAID 2

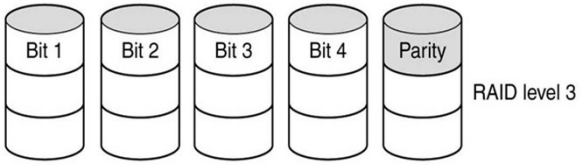
- Raid level 2 requires synchronized drives
- Reading is very fast as all drives can transfer data (portions of the file). In one sector reading time n (number of drives) sectors are read in
- Writing is slower because of parity information (to write parity all requests must access this drive)

58

58

RAID 3

- Raid 3 is a simplifed version of Raid 2
- It uses an additional drive for a parity bit (word/byte)
- Drives must be synchronized
- Provides 1 bit (word/byte) error correction

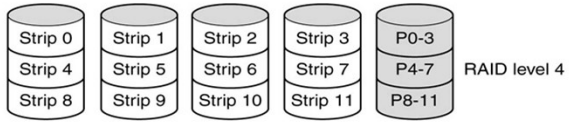


59

59

RAID 4

- A file is distributed as strips on several disks as in RAID 0
- Another drive holds the parity strip
- Reading is fast as all drives can transfer data (portions of the file) independently
- Parity drive is heavily used

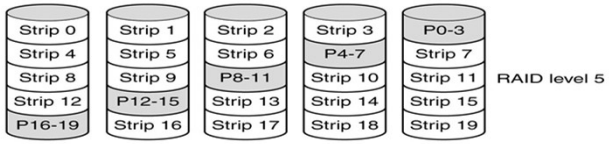


60

60

RAID 5

- Raid level 5 is Raid level 4 with parity information distributed on all drives



61

61

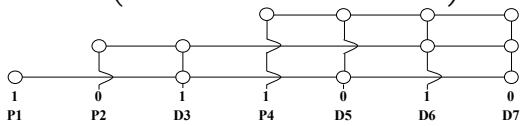
RAID 5

- Similar to RAID 4 but parity is distributed to all disks
- Fast read and writes
- Suitable for transaction oriented processing such as on-line banking, hotel reservations etc.
- Total capacity for a RAID 5 system with ***N*** disks = capacity of one disk * (N-1)

62

62

Hamming Code Example (One Bit Correction)



- 4 data bits (D3, D5, D6, D7) – 1 0 1 0 and 3 parity bits (P1, P2, P4)- 1 0 1
- P1 makes D3, D5, D7 even; P2 makes D3, D6, D7 even; P4 makes D5, D6, D7 even
- Check parities after transmission. 0= parity check is correct; 1= parity check failed

P4	P2	P1	Error in
0	0	0	No error
0	0	1	P1
0	1	0	P2
0	1	1	D3
1	0	0	P4
1	0	1	D5
1	1	0	D6
1	1	1	D7

63

63

Summary

- I/O architecture is the computer system's interface to the outside world
- I/O functions are generally broken up into a number of layers
- A key aspect (diện mạo) of I/O is the use of buffers that are controlled by I/O utilities rather than by application processes
- Buffering smoothes (làm phẳng) out the differences between the speeds
- The use of buffers also decouples (tách riêng) the actual I/O transfer from the address space of the application process
- Disk I/O has the greatest impact (ảnh hưởng) on overall system performance
- Two of the most widely used approaches are disk scheduling and the disk cache
- A disk cache is a buffer, usually kept in main memory, that functions as a cache of disk block between disk memory and ~~the rest (phần còn lại) of~~ main memory

64