HUTECH

Đại học Công nghệ Tp.HCM

Bài giảng Hệ điều hành

HIENLTH

Ths. Lương Trần Hy Hiến
https://hutechos.weebly.com

Bài 9: Quản lý Nhập xuất

- Giới thiệu về thiết bị nhập xuất
- Các kỹ thuật quản lý thao tác nhập xuất
- Các vấn đề về thiết kế hệ thống quản lý nhập xuất trong HĐH
- Kỹ thuật vùng đệm nhập xuất
- Quản lý hệ thống nhập xuất đĩa

Mục tiêu của Quản lý Nhập xuất

- Tạo thành một lớp giao tiếp độc lập thiết bị
  - Che giấu các chi tiết kỹ thuật của các thiết bị phần cứng.
  - Quản lý và sửa lỗi
- Làm cho các thiết bị phần cứng đơn giản và dễ dùng
- Cho phép chia sẻ các thiết bị phần cứng
  - Xây dựng các cơ chế bảo vệ các thiết bị được chia sẻ
  - Điều phối thiết bị để phục vụ cho nhiều nhu cầu sử dụng cùng lúc .

Ví dụ về các thiết bị nhập xuất

- Các thiết bị giao tiếp:
  - Các thiết bị chỉ nhập : bàn phím, chuột, joystick…
  - Các thiết bị chỉ xuất : màn hình, máy in
  - Các thiết bị vừa nhập vừa xuất: card mạng.
- Các thiết bị lưu trữ:
  - Thiết bị vừa xuất, vừa nhập: đĩa (cứng/mềm), băng từ
  - Thiết bị chỉ xuất: CD-ROM.

Ví dụ về các thiết bị nhập xuất

Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Telephone channel	8 KB/sec
Dual ISDN lines	16 KB/sec
Laser printer	100 KB/sec
Scanner	400 KB/sec
Classic Ethernet	1.25 MB/sec
USB (Universal Serial Bus)	1.5 MB/sec
Digital camcorder	4 MB/sec
IDE disk	5 MB/sec
40x CD-ROM	6 MB/sec
Fast Ethernet	12.5 MB/sec
ISA bus	16.7 MB/sec
EIDE (ATA-2) disk	16.7 MB/sec
FireWire (IEEE 1394)	50 MB/sec
XGA Monitor	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec
Sun Gigaplane XB backplane	20 GB/sec

Phân loại các thiết bị nhập xuất

- Phân loại theo mục đích sử dụng:
  - Các thiết bị giao tiếp:
    - Các thiết bị chỉ nhập : bàn phím, chuột, joystick…
    - Các thiết bị chỉ xuất : màn hình, máy in
    - Các thiết bị vừa nhập vừa xuất: card mạng.
  - Các thiết bị lưu trữ
    - Thiết bị vừa xuất, vừa nhập: đĩa (cứng/mềm), băng từ
    - Thiết bị chỉ xuất: CD-ROM
- Phân loại theo phương pháp truy xuất:
  - Thiết bị khối:
    - Tổ chức theo từng khối riêng biệt và truy xuất ngẫu nhiên (VD: CD-ROM, HDD)
  - Thiết bị tuần tự
    - Gởi nhận theo chuỗi bit và phải truy xuất tuần tự (VD: Card mạng, Bàn phím)

Phân loại các thiết bị nhập xuất (tt)

- HĐH phải gom nhóm các thiết bị khác nhau thành những nhóm cơ bản để dễ dàng quản lý:
  - Storage
    - Hard drives, Tapes, CDROM
  - Networking
    - Ethernet, radio, serial line
  - Multimedia
    - DVD, Camera, microphones
- HĐH phải cung cấp các phương thức nhất quán để truy cập các nhóm đối tượng trên. Nếu không, lập trình sẽ rất khó khăn

Sự khác nhau giữa các thiết bị I/O

- Tốc độ
- Ứng dụng
- Mức độ phức tạp để kiểm soát
- Đơn vị truyền
- Biểu diễn dữ liệu
- Phát sinh lỗi

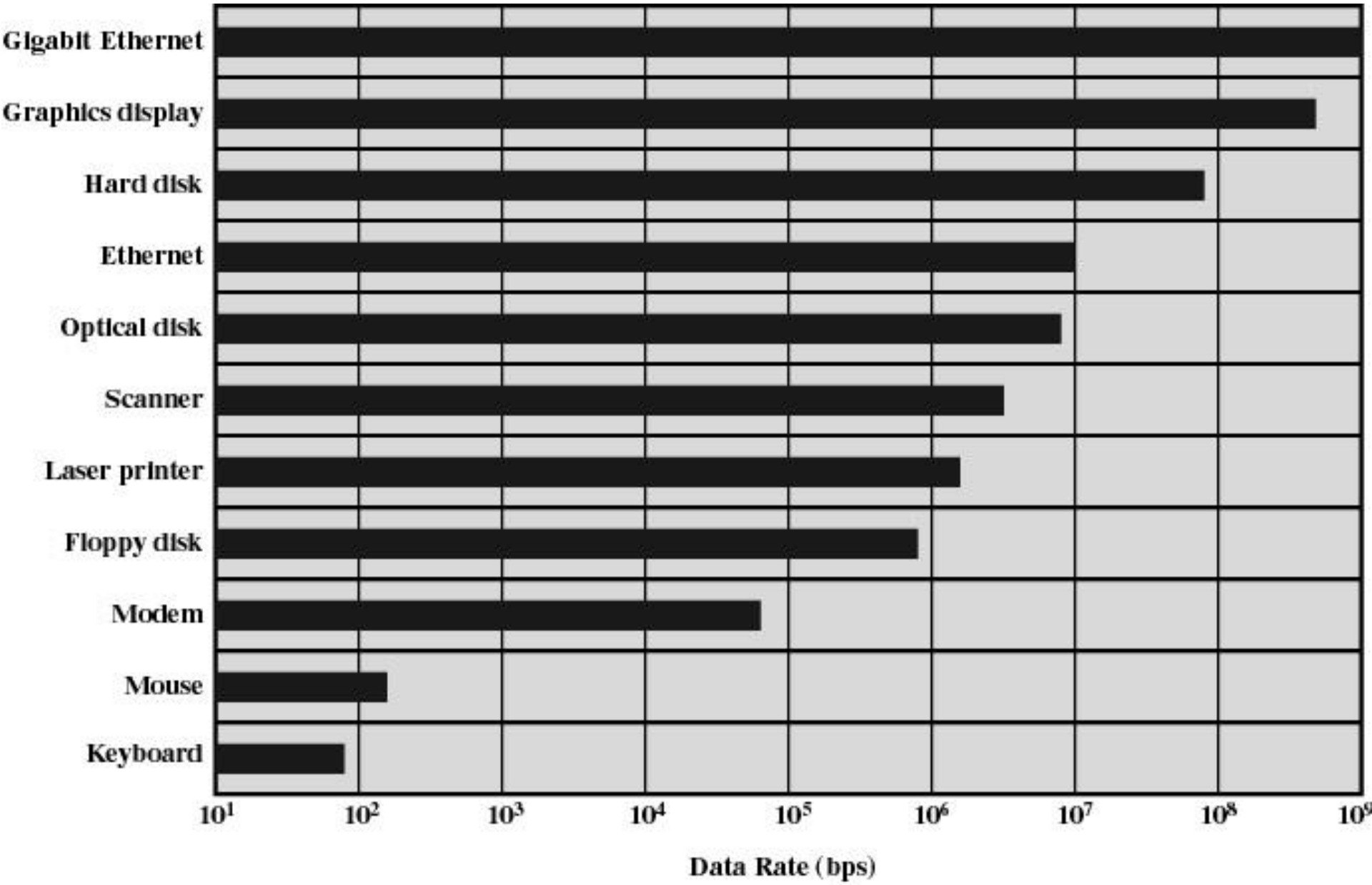


Figure 11.1 Typical I/O Device Data Rates

Sự khác nhau giữa các thiết bị I/O (2)

- Ứng dụng
  - Đĩa dùng để lưu trữ đòi hỏi phải có hệ thống quản lý tập tin đi kèm
  - Đĩa dùng để làm bộ nhớ ảo đòi hỏi phải có sự hỗ trợ phần cứng và phần mềm.
  - Các thiết bị đầu cuối được sử dụng bởi quản trị sẽ có độ ưu tiên cao hơn.

Sự khác nhau giữa các thiết bị I/O (3)

- Độ phức tạp để kiểm soát:
  - Máy in so với đĩa
- Đơn vị truyền
  - Có thể được truyền theo byte hoặc khối (có thể là bit)
- Biểu diễn dữ liệu
  - Cơ chế mã hóa
- Phát sinh lỗi
  - Thiết bị khác nhau xử lý lỗi phát sinh khác nhau

Kỹ thuật xử lý I/O

- Programmed I/O
  - Tiến trình phải busy-waiting cho thao tác nhập xuất hoàn thành
- Interrupt-driven I/O
  - Phát sinh lệnh I/O
  - Bộ xử lý tiếp tục thực thi các chỉ thị khác
  - Module I/O gởi một ngắt đến bộ xử lý khi hoàn thành thao tác I/O

Kỹ thuật quản lý I/O (2)

- Direct Memory Access (DMA)
  - Cơ chế truy cập bộ nhớ trực tiếp
  - Cho phép thiết bị làm việc trực tiếp với bộ nhớ (phần buffer đã được đăng ký) ➔ giải phóng CPU
  - Module DMA điều khiển việc chuyển đổi dữ liệu giữa bộ nhớ chính và thiết bị I/O
  - Bộ xử lý chỉ bị ngắt khi toàn bộ khối đã được chuyển

Các vấn đề thiết kế liên quan đến HĐH

- Tính hiệu quả
  - Hầu hết các thiết bị I/O đều rất chậm so với bộ nhớ chính
    - Sử dụng cơ chế đa chương cho phép một vài tiến trình chờ I/O trong khi tiến trình khác thi hành
  - Cơ chế swapping
    - Bản chất cũng là thao tác nhập/xuất

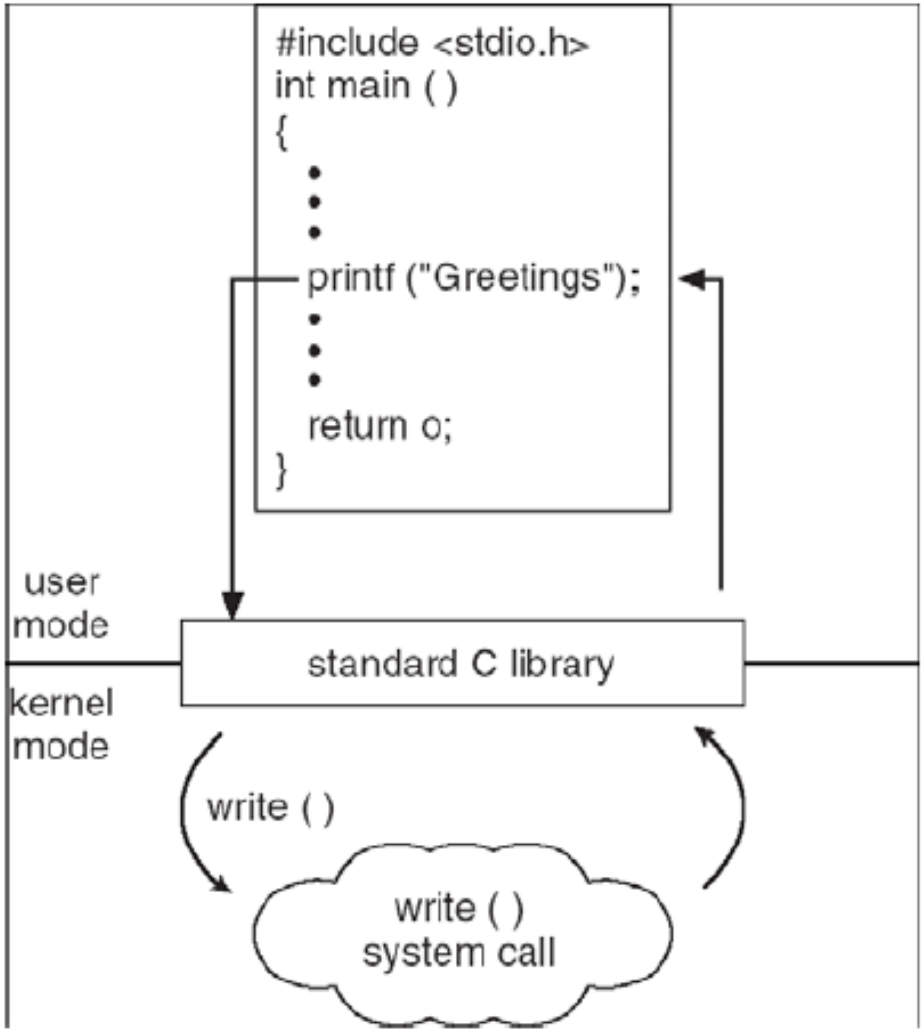
Các vấn đề thiết kế liên quan đến HĐH

- Tính tổng quát
  - Mong muốn xử lý tất cả các thiết bị I/O giống nhau
  - Che giấu hầu hết chi tiết của thiết bị nhập xuất để tiến trình (mức cao) xem các thiết bị ở một vài chức năng thông dụng như read, write, open, close, lock, unlock

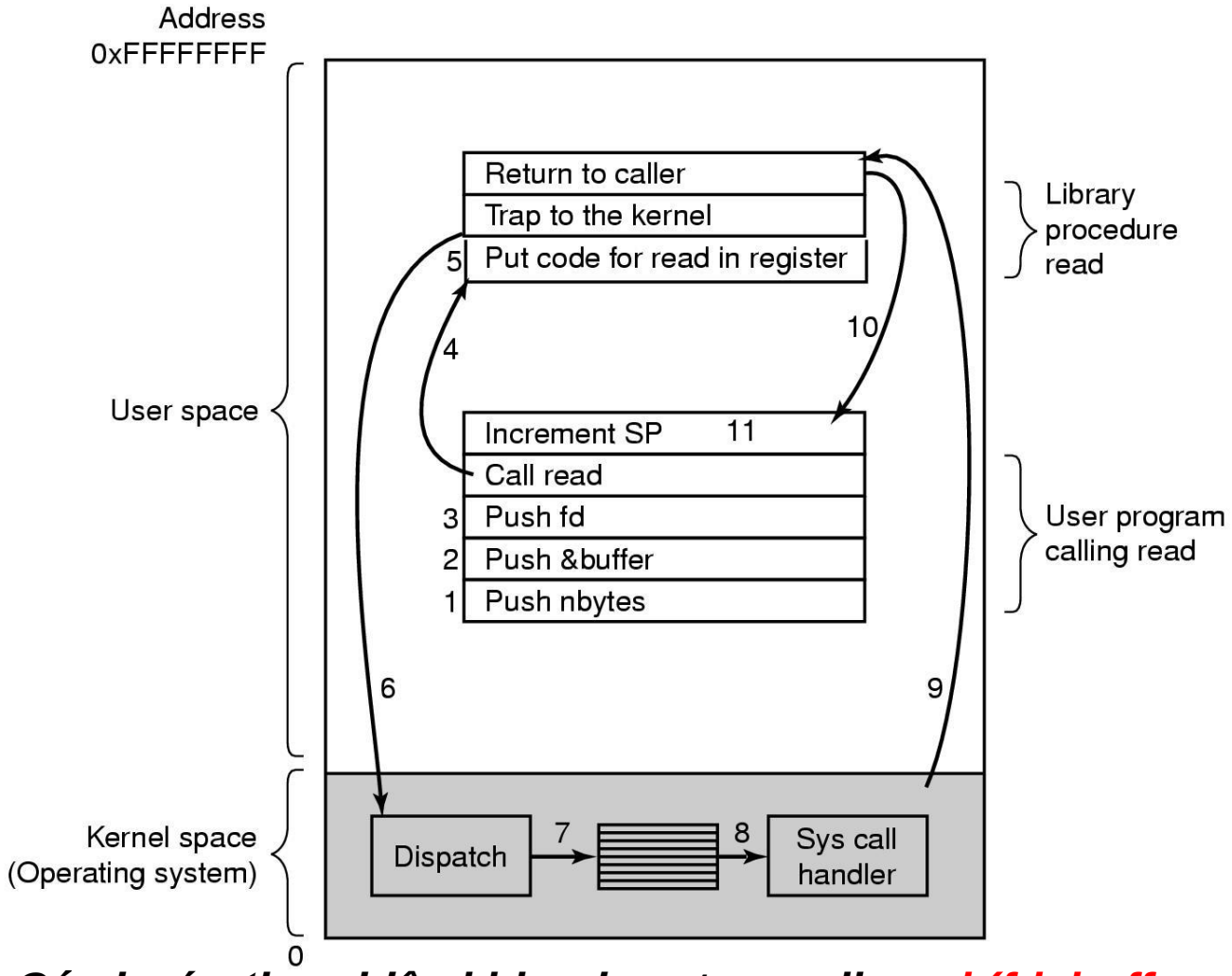
Các phương thức truy cập IO - 1

- HĐH sẽ cung cấp thư viện giao tiếp chung cho các ứng dụng khác nhau:
  - Thư viện đó là 1 tập các hàm có tên chung là system calls
- Ví dụ , với HĐH Unix, sử dụng 4 phương thức chính:
  - open()
  - close()
  - read()
  - write()
- Các phương thức (hàm) này là các system calls được cung cấp bởi HĐH để cho phép các ứng dụng tương tác với các thiết bị xuất nhập.

System call


#include <stdio.h>
int main ( )
{
printf ("Greetings");
return 0;
}
user mode
kernel mode
standard C library
write ( )
write ( )
system call

System call



Các bước thực hiện khi gọi system call read (fd, buffer, nbytes)

Các phương thức truy cập IO - 2

- Các phương pháp truyền tham số cho system call
  - Thông qua thanh ghi
  - Thông qua bộ nhớ
  - Thông qua stack khi gọi chương trình

Một số API system call (UNIX + Win32)

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Các đặc tính nhập xuất

- Ba đặc tính khác nhau cần xem xét khi xử lý 1 thao tác nhập xuất:
  - Blocking vs. Non-blocking
  - Buffered vs. Unbuffered
  - Synchronous vs. Asynchronous

Blocking vs. Non-Blocking

- Blocking – ứng dụng dừng lại cho đến khi hoàn tất thao tác đọc ghi
  - Ví dụ: Trong thiết bị mạng, nếu muốn ghi 1000 bytes, thì HĐH ghi tất cả các byte cho đến khi ghi hoàn tất.
  - Nếu thiết bị không thể thực hiện lệnh ghi được (ví dụ hỏng dây nối)?

    ➔ kết thúc và trả về số bytes đã ghi được.
- Nonblocking – HĐH đọc và ghi các bytes khi có thể, không cần ứng dụng phải dừng lại.

Buffered vs. Unbuffered

- Buffered:
  - Trong trường hợp buffer dữ liệu của thiết bị quá nhỏ, để không phải chờ quá lâu khi thực hiện IO
    - buffered I/O cho phép kernel copy lại dữ liệu
    - Bên write(): cho phép ứng dụng tiếp tục ghi dữ liệu
    - Bên read(): khi thiết bị báo có dự liệu đến, kernel chép dữ liệu vào buffer. Khi tiến trình gọi read(), kernel chỉ việc copy từ buffer.
  - Khuyết điểm buffered I/O?
    - Thêm chi phí để thực hiện copy
    - Chậm trễ việc gửi dữ liệu
- Unbuffred: Không chấp nhận ghi dữ liệu vào kernel

Synchronous vs. Asynchronous

- Synchronous: các xử lý khác thuộc ứng dụng của người dùng cuối sẽ phải tạm dừng lại (paused) để chờ các thao tác nhập xuất của nó hoàn tất
- Asynchronous: các xử lý khác của ứng dụng có thể thực thi song song với các thao tác nhập xuất

Mô hình lôgic các chức năng nhập xuất

- Giao tiếp đơn giản (thiết bị ngoại vi cục bộ)
  - Logical I/O
  - Device I/O
  - Scheduling and Control
- Giao tiếp thông quan cổng kết nối
  - Communication architecture
  - Device I/O
  - Scheduling and Control

Mô hình lôgic các chức năng nhập xuất

- Hệ thống tập tin
  - Directory management
  - File system
  - Physical organization
  - Device I/O
  - Scheduling and Control

Kỹ thuật vùng đệm nhật/xuất

- Lý do
  - Các tiến trình phải chờ hoàn thành I/O trước khi xử lý tiếp
  - Thao tác I/O vẫn phải tốn bộ nhớ

Kỹ thuật vùng đệm nhật/xuất (2)

- Các thiết bị I/O có hai dạng hỗ trợ nhập/xuất cơ bản:
  - Hướng khối (block – oriented)
  - Hướng dòng (stream – oriented)
- Block-oriented
  - Thông tin được lưu trong những khối có kích thước cho trước
  - Đơn vị chuyển là một khối / một lần
  - Dùng cho đĩa
- Stream-oriented
  - Truyền thông tin như là một dòng các byte
  - Dùng cho các thiết bị đầu cuối, máy in, chuột, cổng giao tiếp,…

Single Buffer

- HĐH dành riêng một vùng đệm trong bộ nhớ chính cho thao tác I/O
- Block-oriented
  - Dữ liệu vào được chuyển vào vùng đệm theo đơn vị khối
  - Khối đó sẽ chuyển đến cho tiến trình người dùng khi cần
  - Khối khác được chuyển vào vùng đệm, trong khi tiến trình đang xử lý khối trước đó.
- Stream-oriented
  - Như khối nhưng với đơn vị là dòng. Một dòng được kết thúc bởi dấu CF.

I/O Buffering

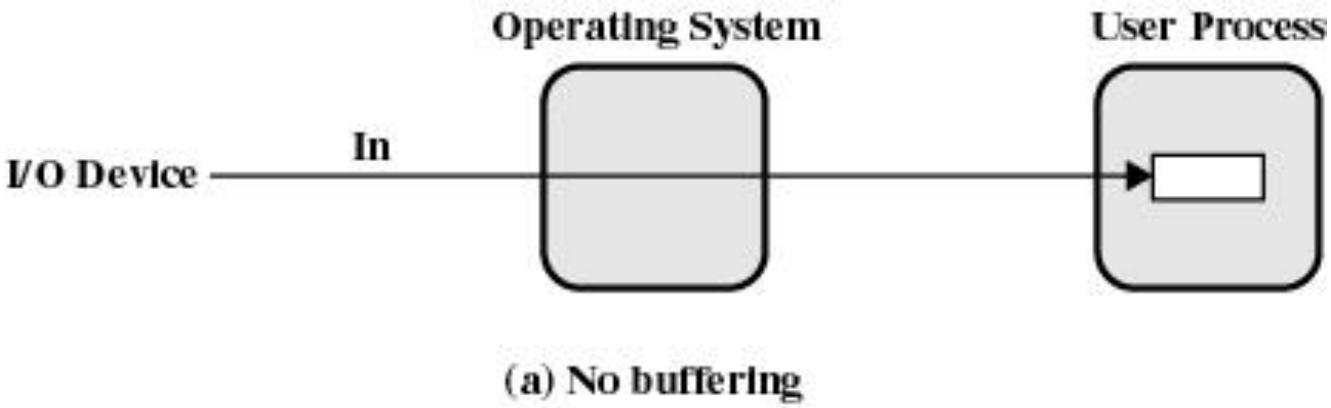


Figure 11.6 I/O Buffering Schemes (input)

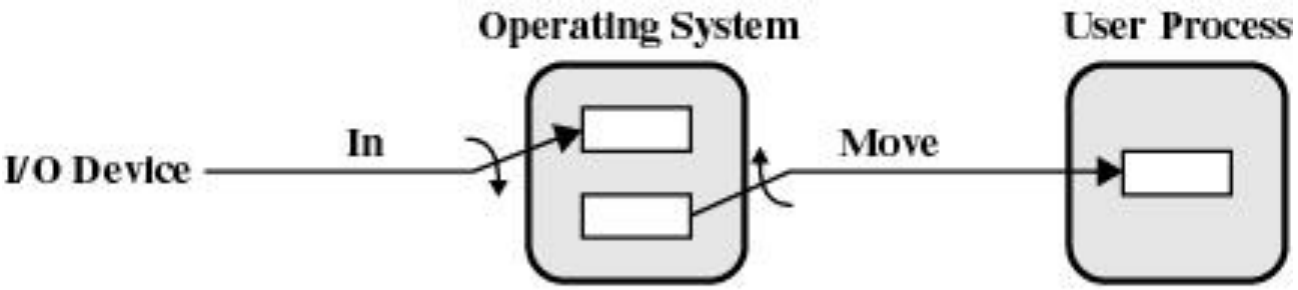
Double Buffer

- Dùng hai vùng đệm thay vì một
- Một tiến trình có thể chuyển dữ liệu vào và ra một vùng đệm trong khi hệ điều hành truyền thông tin hoặc thao tác trên vùng đệm còn lại

Circular Buffer

- Nhiều hơn hai vùng đệm
- Trong trường hợp thao tác nhập xuất gắn liền với tiến trình.

I/O Buffering



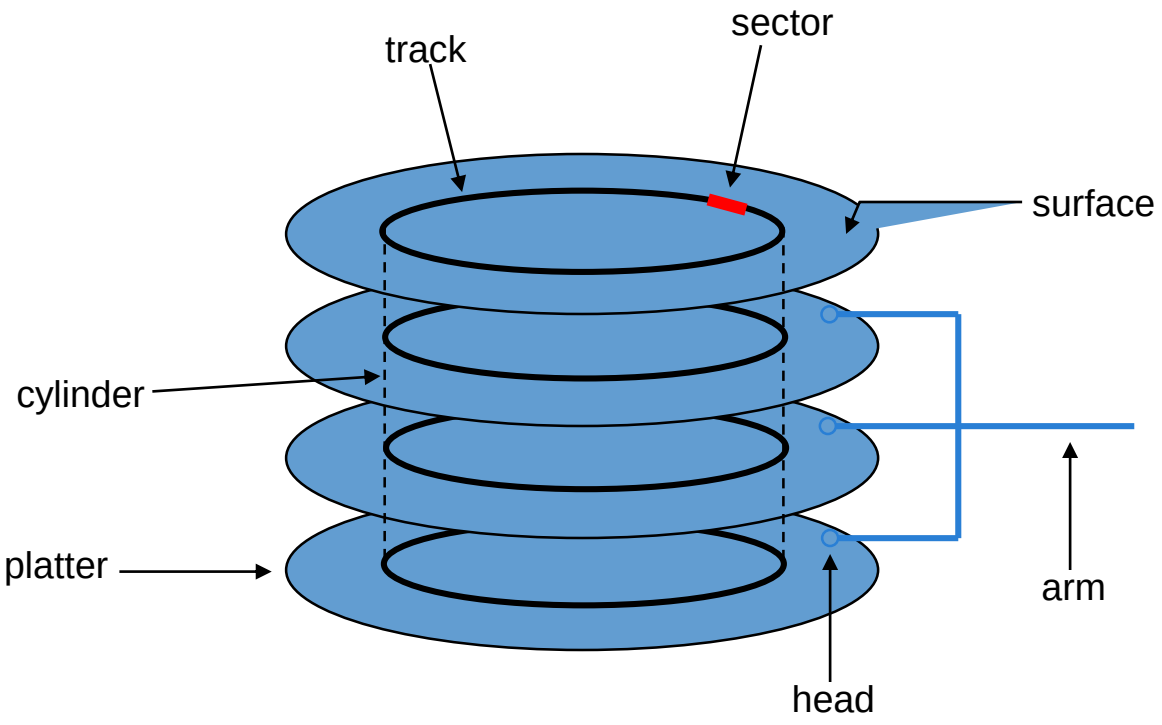
(c) Double buffering

(d) Circular buffering

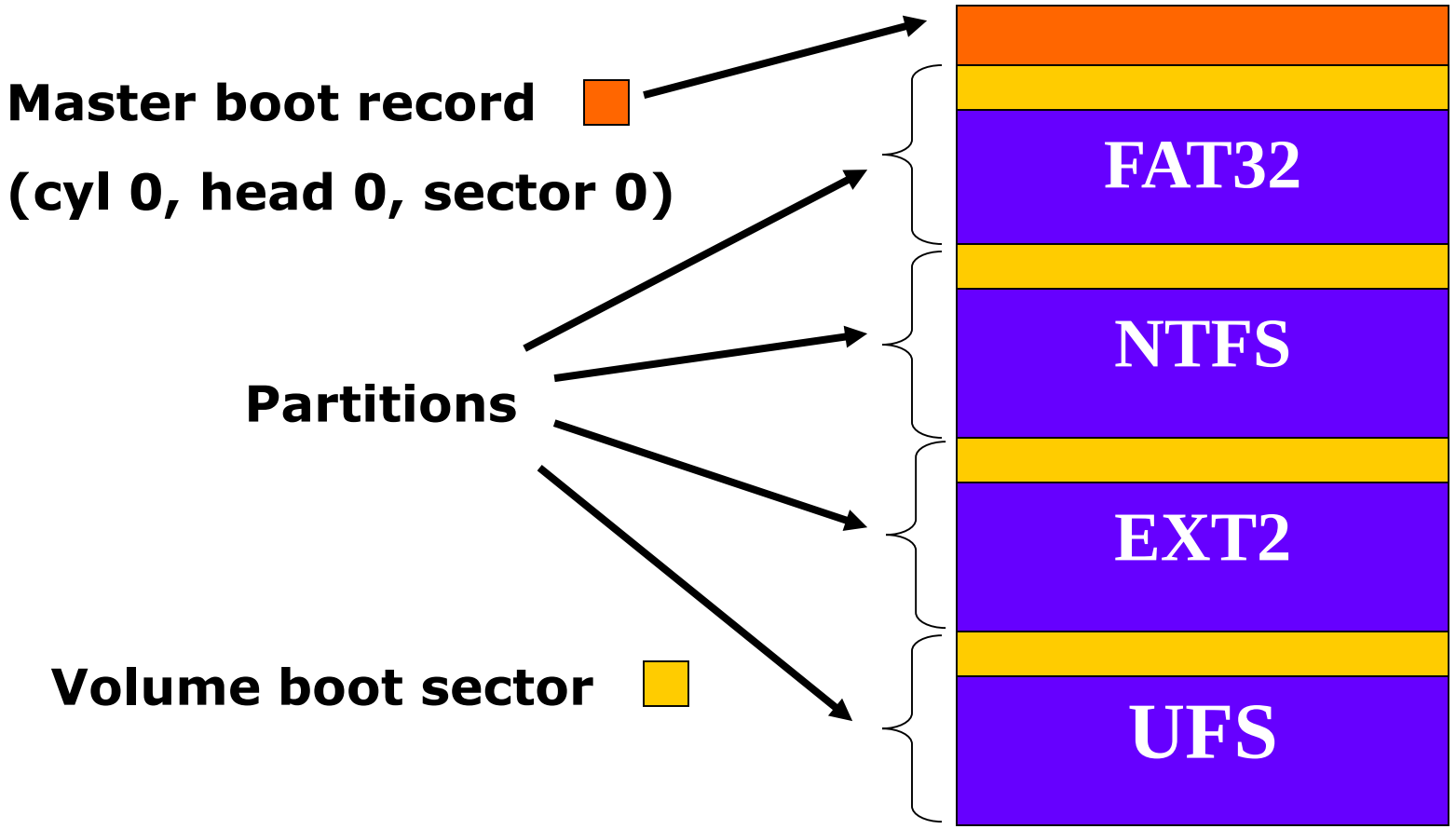
Figure 11.6 I/O Buffering Schemes (input)

Nhắc lại cấu trúc đĩa

- Các thành phần của đĩa
  - platters
  - surfaces
  - tracks
  - sectors
  - cylinders
  - arm
  - heads



Nội dung



Các tham số hiệu năng đĩa

- Để đọc hay ghi, đầu đĩa phải di chuyển đến track và sector tương ứng
- **Seek time**
  - Thời gian dùng để định vị đầu đĩa đến track (hay cylinder) mong muốn.
    - Thời gian trung bình trên các đĩa cứng hiện nay là dưới **10 ms**
- **Rotational delay** hoặc rotational latency
  - Thời gian dùng để di chuyển đầu đĩa đến sector tương ứng (vị trí đầu tiên của sector).
  - Thời gian rotational delay trung bình là $t_r/2$
    - t_r là thời gian quay 1 vòng
  - Thời gian quay của các đĩa hiện nay từ 3600 rpm đến 15000 rpm (có thời gian rotational delay trung bình là 2 ms)

Các tham số hiệu năng đĩa

- **Transfer time**
  - Thời gian chuyển dữ liệu (từ hoặc vào bộ nhớ chính)
  - Transfer time = $b*t_r/N$
  - b: số byte cần đọc, N: tổng số byte trên 1 track
- **Access time** = Seek time + Rotational time + Transfer time

$$= T_{seek} + t_r/2 + b*t_r/N$$

  - Thời gian để đặt đầu đĩa và đúng vị trí sector để đọc và ghi
  - Thực tế : Seek time >> latency time > transfer time
  - Tối ưu seek time ➔ định thì truy xuất đĩa
  - Tối ưu latency time:
    - Đĩa KT nhỏ, quay nhanh, lưu trữ dữ liệu gần kề
    - Chọn kích thước sector, nơi lưu trữ các tệp tường dùng hợp lý

Ví dụ truy xuất đĩa từ

- Xét 1 đĩa cứng có 5 mặt, mỗi mặt có 200 track, mỗi track có 500 sector, seek time trung bình là 4 ms, tốc độ quay của đĩa (rotation speed) là 15000 rpm
  - Xác định dung lượng của đĩa?
  - Giả sử có 1 tập tin có dung lượng 1.28MB, chiếm trọn 5 track (2500 sector) trên cùng một cylinder. Hãy xác định thời gian truy xuất trung bình tập tin này?
- Giải:
  - Thời gian đọc track đầu tiên
    - Thời gian seek time trung bình: 4 ms
    - Thời gian rotational delay trung bình: 2 ms
    - Thời gian đọc 1 track (500 sectors): 4 ms
  - Do tập tin nằm trên 1 cylinder nên không tốn thời gian seek time khi đọc các track còn lại. Do đó, tổng thời gian truy xuất tập tin này là:

$$10 + (4 \times 6) = 34 \text{ ms} = 0.034 \text{ seconds}$$

Chính sách lập lịch trên đĩa

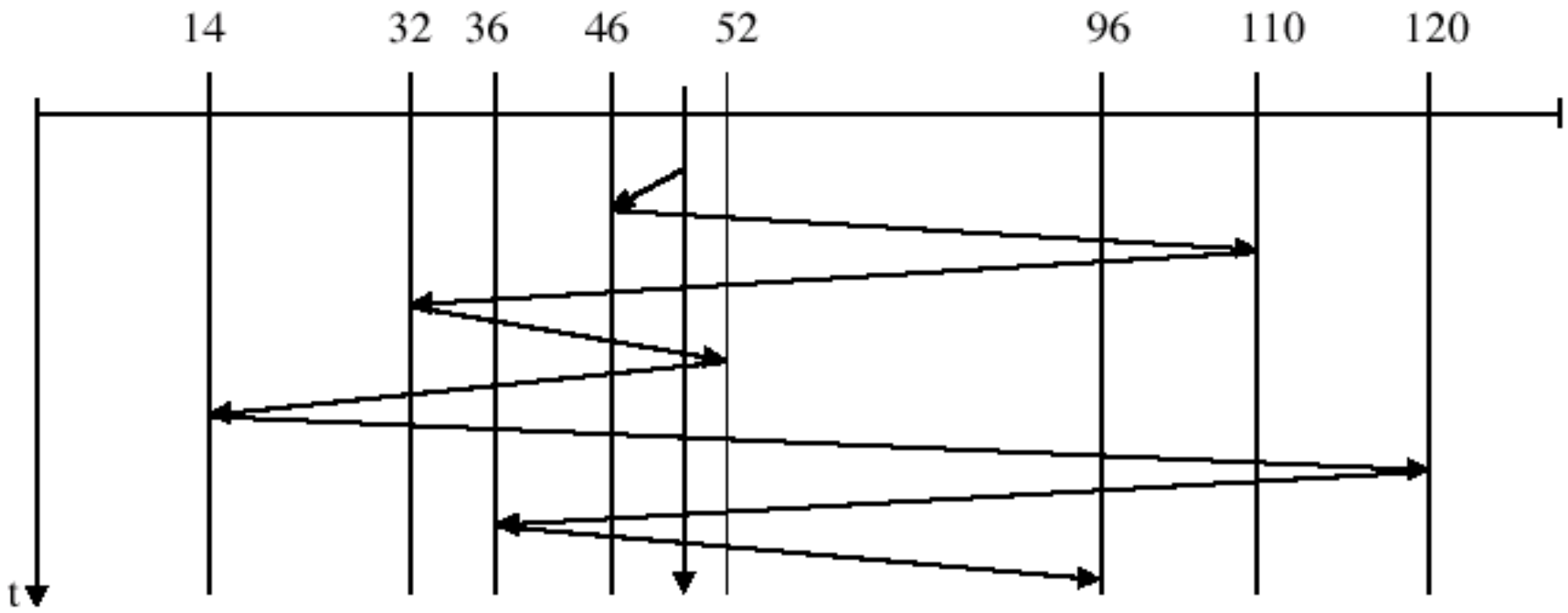
- **FCFS (First Come First Serve)**
  - Di chuyển đầu đọc theo thứ tự yêu cầu
  - Không sắp xếp lại hàng đợi công việc
  - Dễ lập trình nhưng kết quả thường không tốt
- **SSTF (Shortest Seek Time First)**
  - Sau khi phục vụ 1 yêu cầu, di chuyển đầu đọc tới vị trí cần đọc gần với vị trí hiện hành của đầu đọc nhất
  - Giảm seek time so với FCFS
  - Vấn đề starvation
- **SCAN (elevator algorithms)**
  - Đầu đọc sẽ di chuyển về một phía của đĩa và từ đó di chuyển qua phía kia
  - Vấn đề: mật độ dày các yêu cầu chưa được phục vụ ở đầu còn lại
- **C-SCAN**
  - Tương tự SCAN, chỉ khác là khi di chuyển tới 1 đầu nào đó của đĩa, nó sẽ lập tức trở về đầu bắt đầu của đĩa
- **LOOK:**
  - Tương tự SCAN nhưng đầu đọc chỉ di chuyển tới khối xa nhất ở mỗi hướng chứ không đến cuối
- **C-LOOK**

First Come First Served - FCFS

- Còn gọi là: First-in, first-out (FIFO)
- Xử lý yêu cầu một cách tuần tự (*Yêu cầu nào đến trước được thực hiện trước*)
- Công bằng cho tất cả tiến trình
- Nếu nhiều tiến trình thì xác suất xem như là ngẫu nhiên
- Phương pháp này dễ lập trình nhưng không cung cấp dịch vụ tốt.
  - Ví dụ: cần phải đọc các khối theo thứ tự sau **98, 183, 37, 122, 14, 75** và đầu đọc đang ở vị trí 59.
  - Đầu đọc sẽ lần lượt đi qua các khối **59, 98, 183, 37, 122, 14, 75**.

First Come First Served - FCFS

Arrival order: 46, 110, 32, 52, 14, 120, 36, 96 (track addresses)
Head current position: 50



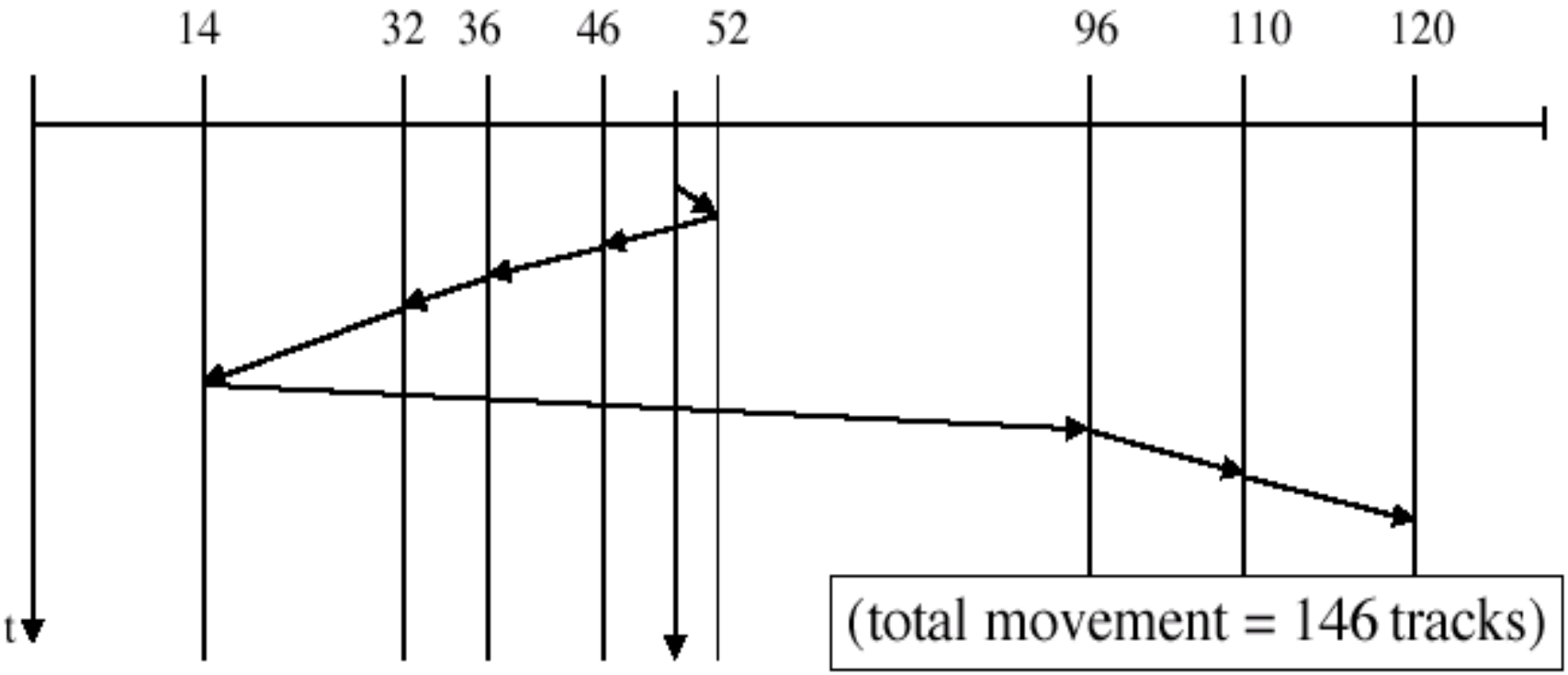
Total head movement = 454 tracks

Shortest Service Time First (SSTF)

- Di chuyển đầu đọc đến các khối cần thiết theo vị trí lần lượt ***gần với vị trí hiện tại của đầu đọc nhất***.
- Nói cách khác: *Chọn yêu cầu I/O mà đòi hỏi ít di chuyển nhất đầu đĩa từ vị trí hiện tại.*
- Luôn luôn cho ra seek time nhỏ nhất.
- Ví dụ: cần đọc các khối **98, 183, 37, 122, 14, 124, 65 và 67**. Giả sử đầu đọc đang ở vị trí 53.
- Đầu đọc sẽ lần lượt qua các khối 53, 65, 67, 37, 14, 98, 122, 124, 183.

Shortest Service Time First (SSTF)

Arrival order: 46, 110, 32, 52, 14, 120, 36, 96
Head current position: 50

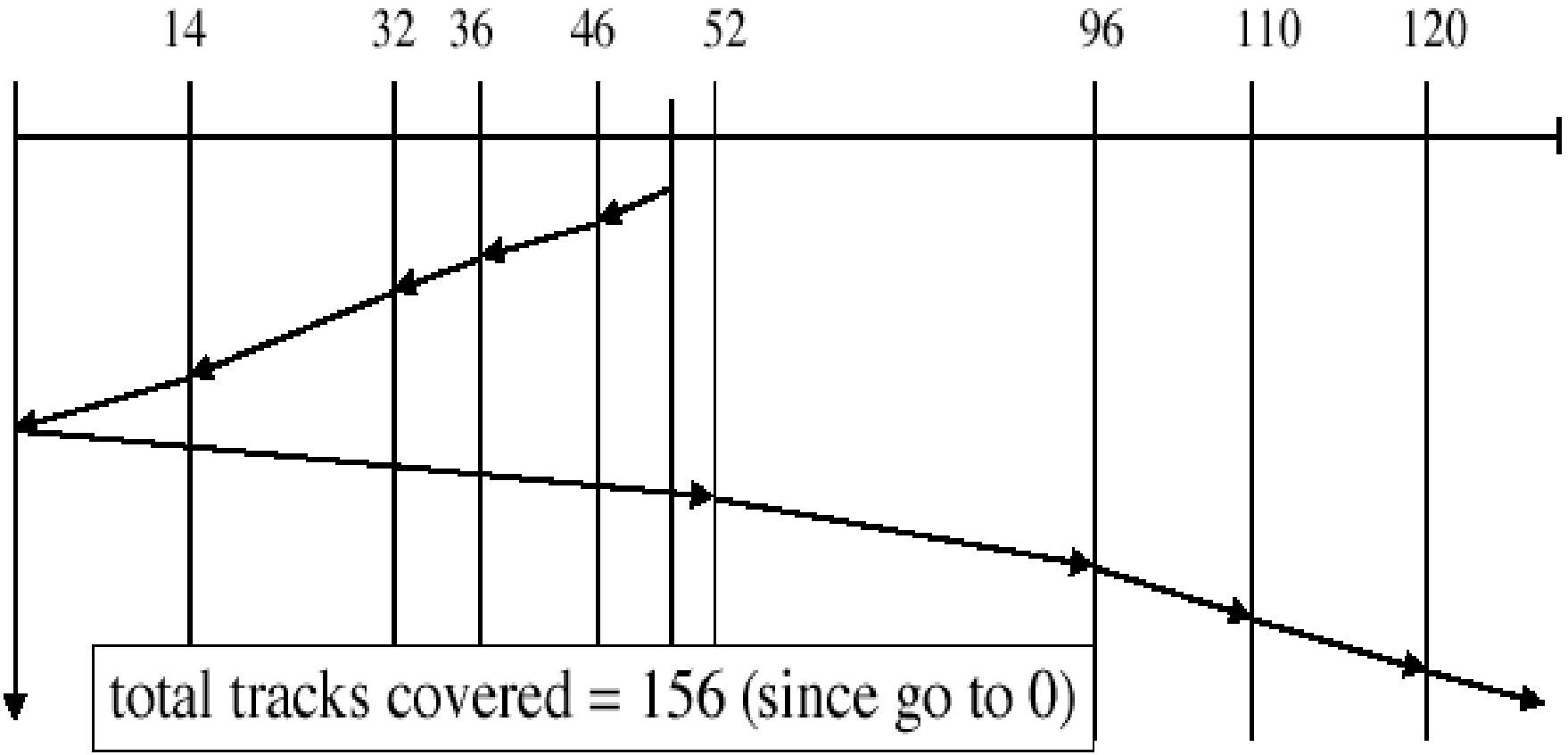


SCAN

- Đầu đọc di chuyển về một phía của đĩa và từ đó di chuyển về phía ngược lại.
  - Ban đầu chọn REQ theo hướng IN → OUT (chạm biên 0)
  - Khi đầu đọc ở ngoài cùng, chọn REQ theo hướng OUT→ IN
- Ví dụ cần đọc các khối 98, 183, 37, 122, 14, 124, 65, 67. Giả sử đầu đọc ở vị trí 53.
- Đầu đọc sẽ lần lượt qua các khối 53, 37, 14, 0, 65, 67, 98, 122, 124, 183 (theo SCAN định hướng về 0).

SCAN

Arrival order: 46, 110, 32, 52, 14, 120, 36, 96
Head current position: 50, moving toward to 0.

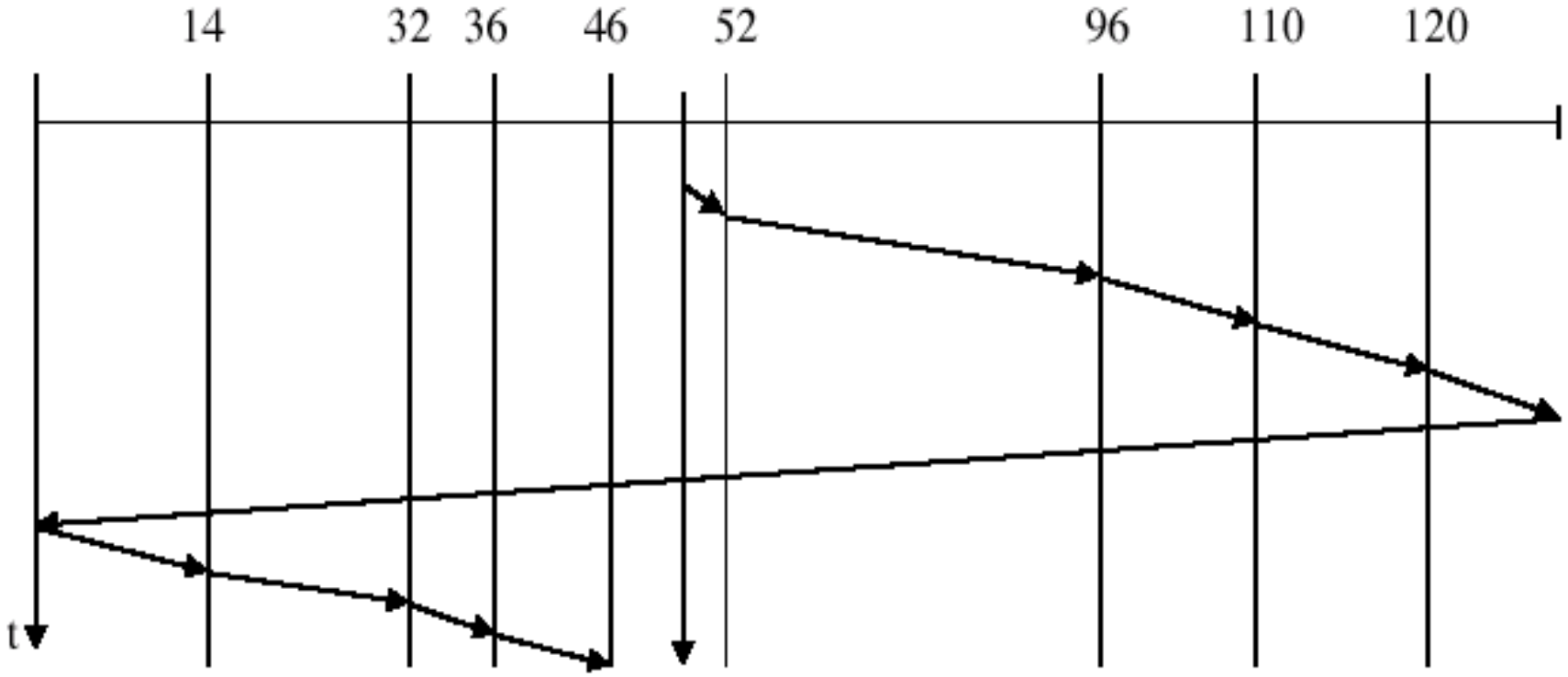


C-SCAN

- Như SCAN, giới hạn chỉ di chuyển theo 1 chiều duy nhất (từ OUT → IN).
- Quay về vị trí bắt đầu (0) ngay khi đụng track ở xa nhất (có thể đụng biên).
- Ví dụ cần đọc các khối 98, 183, 37, 122, 14, 124, 65, 67. Giả sử đầu đọc ở vị trí 53 và được đánh dấu từ 0 → 199
- Đầu đọc sẽ lần lượt qua các khối 53, 65, 67, 98, 122, 124, 183, 199, 0, 14, 37.

C-SCAN

Arrival order: 46, 110, 32, 52, 14, 120, 36, 96
Head current position: 50, moving direction 0 --> 140



LOOK, C-LOOK

- Tương ứng giống SCAN, C-SCAN nhưng không đụng 2 biên.
- Ví dụ cần đọc các khối 98, 183, 37, 122, 14, 124, 65, 67 theo C-LOOK. Giả sử đầu đọc ở vị trí 53.
- Đầu đọc sẽ lần lượt qua các khối 53, 65, 67, 98, 122, 124, 183, 14, 37.

Các thuật toán đọc đĩa : ưu - khuyết điểm

- FCFS : thích hợp trong trường hợp dữ liệu được lưu trữ liên tục.
- SCAN, C-SCAN, LOOK, C-LOOK : thích hợp với lượng dữ liệu cần truy xuất lớn.
- SSTF : thường được sử dụng.

Câu hỏi ôn tập

Ví dụ: cần đọc các khối 98, 183, 124, 65, 12 và 67. Giả sử đầu đọc đang ở vị trí 53, đầu đọc được đánh số từ 0 - 199.

- Đầu đọc sẽ lần lượt qua các khối nào và vẽ sơ đồ theo các thuật toán đọc đĩa FCFS, SSTF, SCAN và C-SCAN.
- Từ đó kết luận xem cách đọc nào tối ưu nhất?

Gợi ý

Yêu cầu đọc đĩa: 98, 183, 124, 65, 12, 67.
Giả sử đầu đọc đang ở vị trí 53.

- FCFS: 53, 98, 183, 124, 65, 12, 67.
  - S= |98-53| + |183 - 98| + |124-183| + |65-24|+|12-65|+|67-12|
- SSTF: 53, 65, 67, 98, 124, 183, 12.
- SCAN: 53, 12, 0, 65, 67, 98, 124, 183.
- C-SCAN: 53, 65, 67, 98, 124, 183, 199, 0, 12.

Thông tin thêm

Địa chỉ tương đối = (head, cylinder, sector)

Head : 0 → Số mặt trên đĩa – 1

Cylinder: 0 → Số cylinder trên đĩa – 1

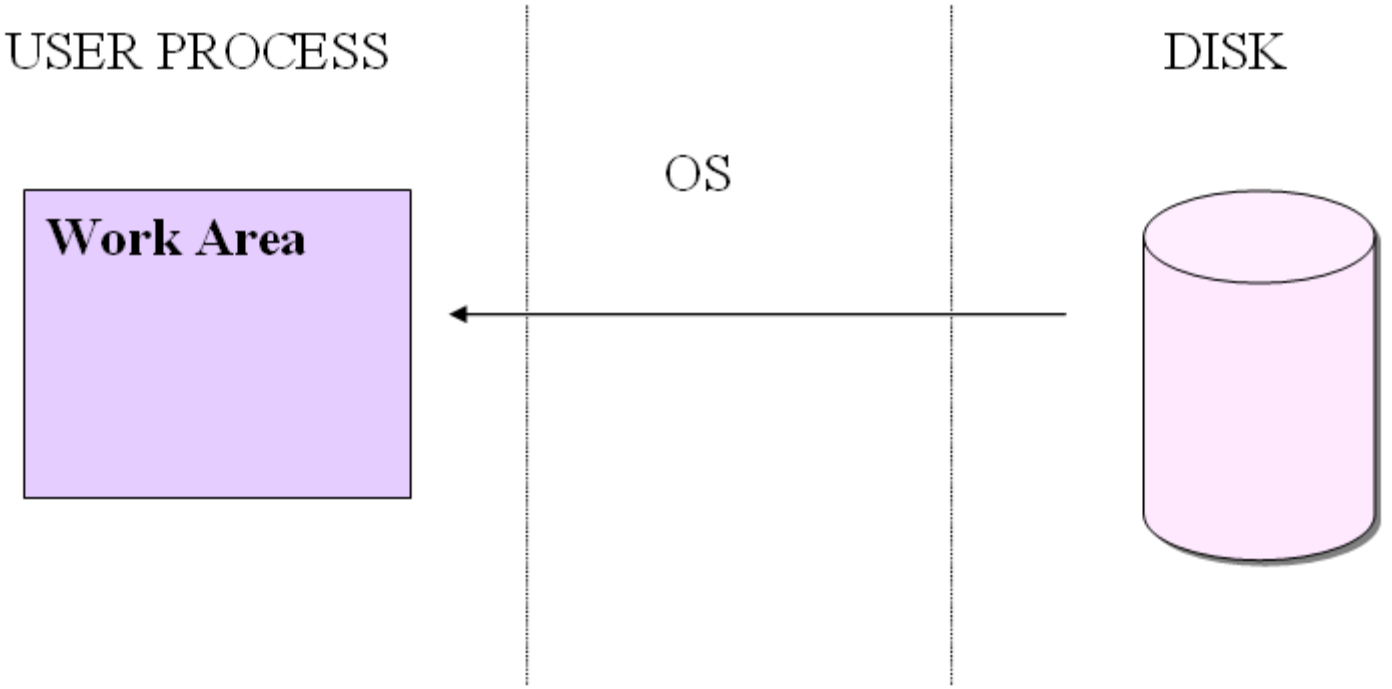
Sector : 1 → Số sector trên track

Địa chỉ tuyệt đối = head * Số sector/track + track * số sector/cylinder + sector - 1

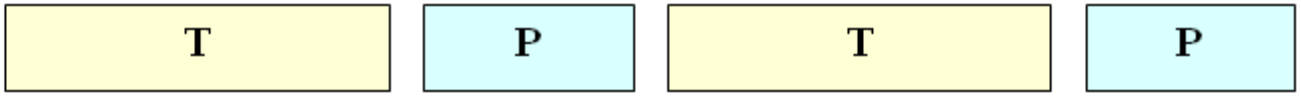
Đệm dữ liệu (Buffering)

- Lưu tạm dữ liệu trong bộ nhớ khi truyền giữa các thiết bị nhằm:
  - Tốc độ truyền giữa các thiết bị
  - Kích thước dữ liệu truyền giữa các thiết bị

Không sử dụng vùng đệm dữ liệu

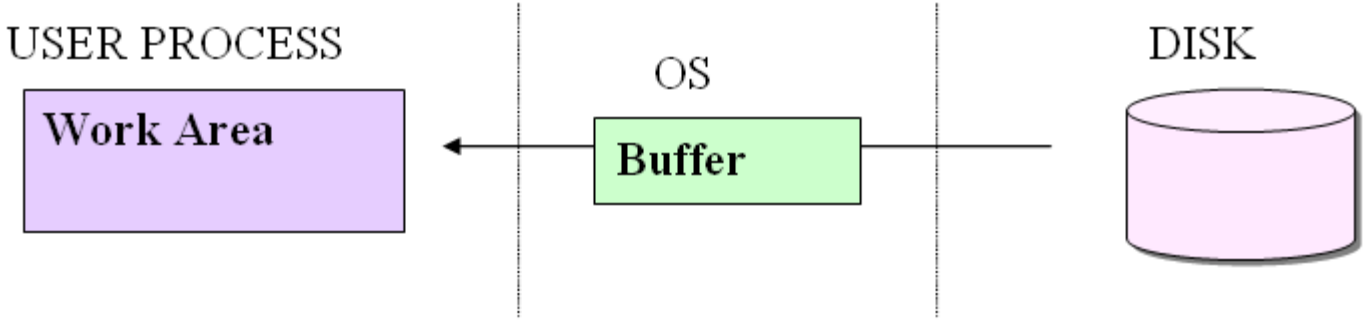


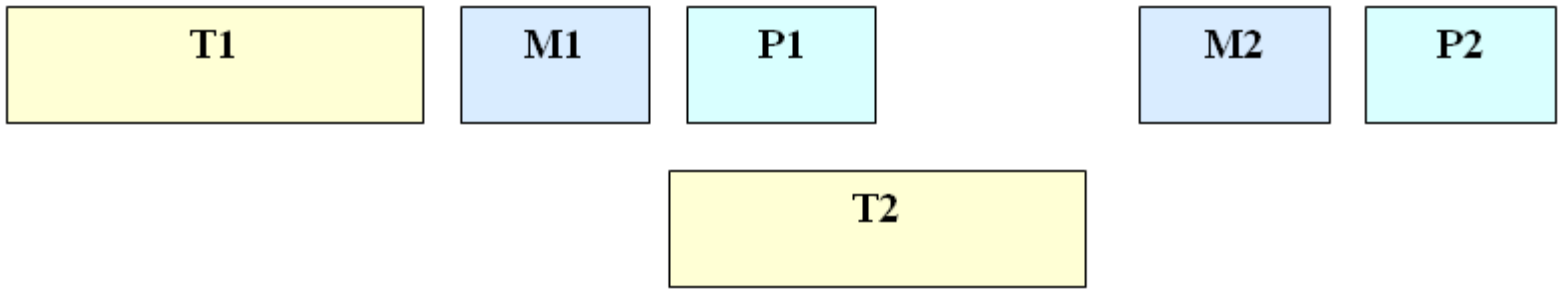
P – Processor Time
T – Transfer or read time



Unbuffered transfer (Timing)

Sử dụng vùng đệm dữ liệu





Assuming M (Move time) to be less than P, this is an improvement over unbuffered transfer.

–Buffered transfer (Timing)

Đệm dữ liệu truy xuất nhanh (Caching)

- Lưu trữ bản sao của dữ liệu tại một vùng nhớ truy xuất nhanh
- Cần phân biệt cache và buffer
  - Buffer lưu tạm bản sao của dữ liệu
  - Cache lưu tạm bản sao của dữ liệu tại nơi có tốc độ truy xuất nhanh hơn nơi lưu dữ liệu

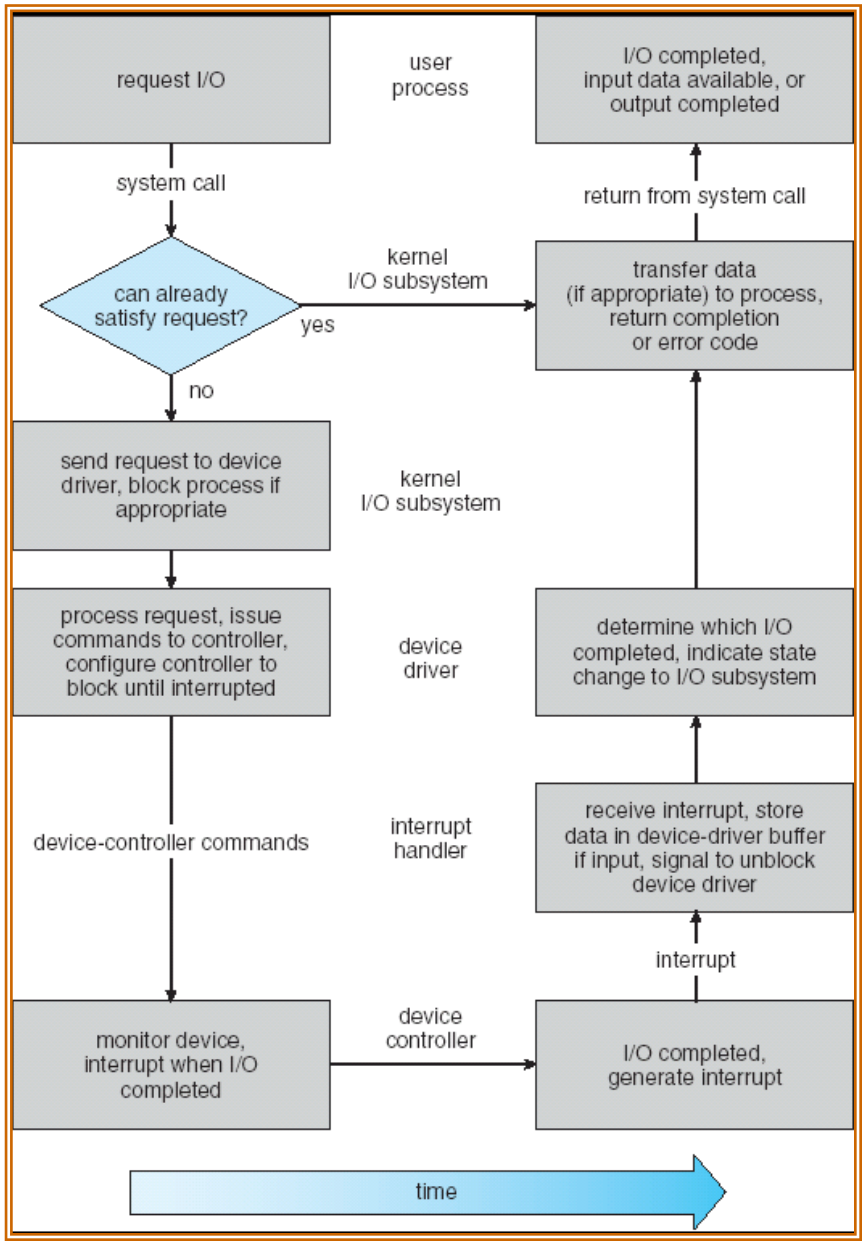
Spooling

- Đặc điểm:
  - Lưu trữ tạm dữ liệu đầu ra cho một thiết bị nếu thiết bị chỉ có thể phục vụ một yêu cầu tại một thời điểm, như máy in
  - Chỉ chấp nhận dòng dữ liệu liên tục
- Thao tác in:
  - Dữ liệu cần in của mỗi ứng dụng được đưa vào một tập tin riêng
  - Hệ thống spooling sẽ chuyển lần lượt các tập tin này cho máy in

Quản lý lỗi (Error Handling)

- Lỗi thường ở 2 dạng:
  - Tạm thời
  - Lâu dài
- I/O Subsystem có thể phục hồi hiệu quả lỗi tạm thời
- Khi yêu cầu nhập xuất xảy ra lỗi → trả về mã lỗi

Vòng đời của một yêu cầu nhập xuất



Q & A