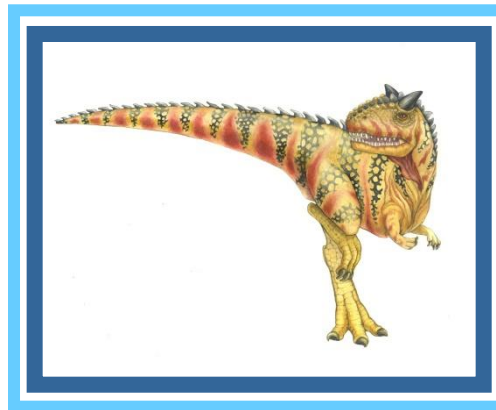


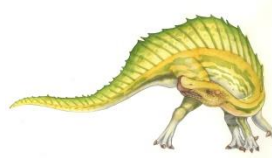
Chương 3: Tiến trình





UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Ôn tập chương 2





Mục tiêu

- Hiểu được khái niệm và các trạng thái của tiến trình
- Biết được các thông số của tiến trình
- Biết được các khái niệm về định thời tiến trình
- Biết được các tác vụ cơ bản của một tiến trình
- Hiểu được cách giao tiếp giữa các tiến trình





Nội dung

- Khái niệm cơ bản
- Trạng thái tiến trình
- Khởi điều khiển tiến trình
- Định thời tiến trình
- Các tác vụ đối với tiến trình
- Sự cộng tác giữa các tiến trình
- Giao tiếp giữa các tiến trình
- Tiểu trình





Khái niệm cơ bản

- Các hoạt động của CPU được gọi là gì?
 - Hệ thống bó (Batch system): jobs
 - Time-shared systems: use program, task
 - Các hoạt động là tương tự → gọi là process
- tiến trình (prcess) là gì?
 - Một chương trình đang thực thi





Khái niệm cơ bản (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Một tiến trình bao gồm:
 - Text section (program code)
 - Data section (chứa global variables)
 - Program counter (PC)
 - Process status word (PSW)
 - Stack pointer (SP)
 - Memory management registers
 - ...

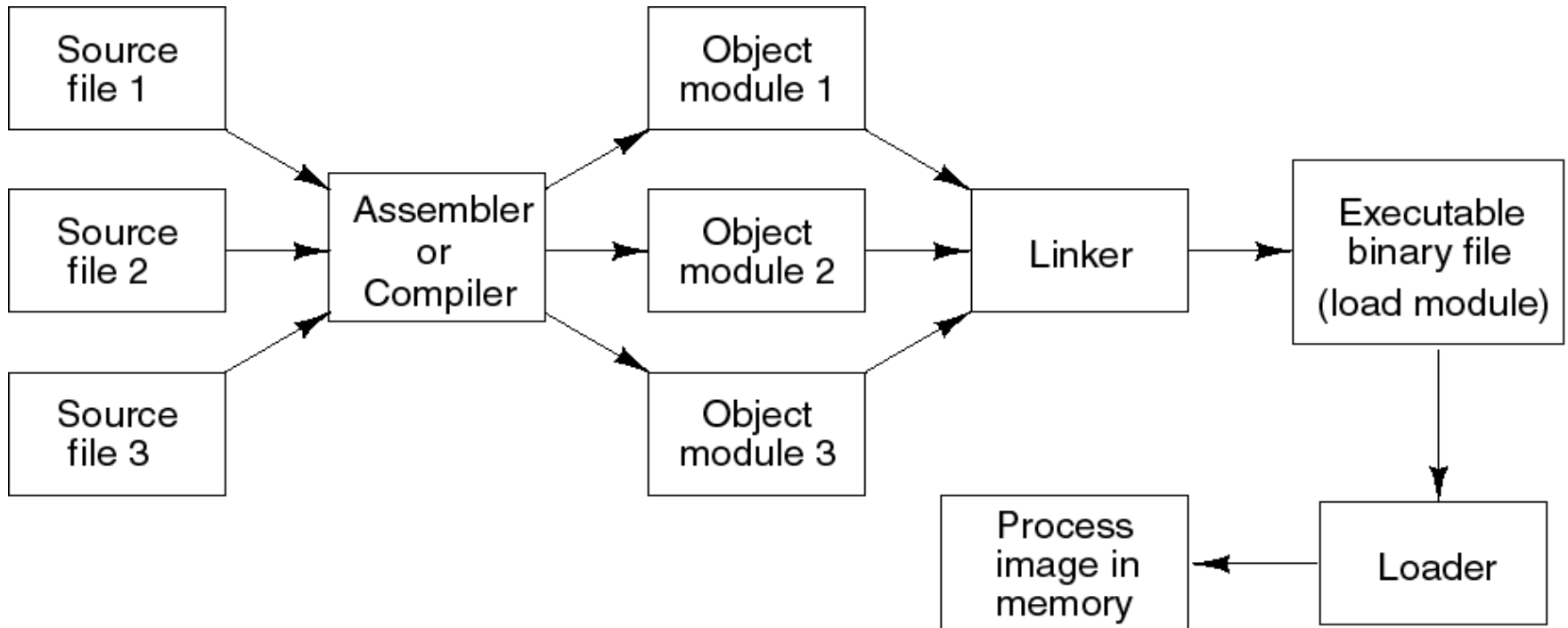




Khái niệm cơ bản (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

■ Các bước nạp chương trình vào bộ nhớ:

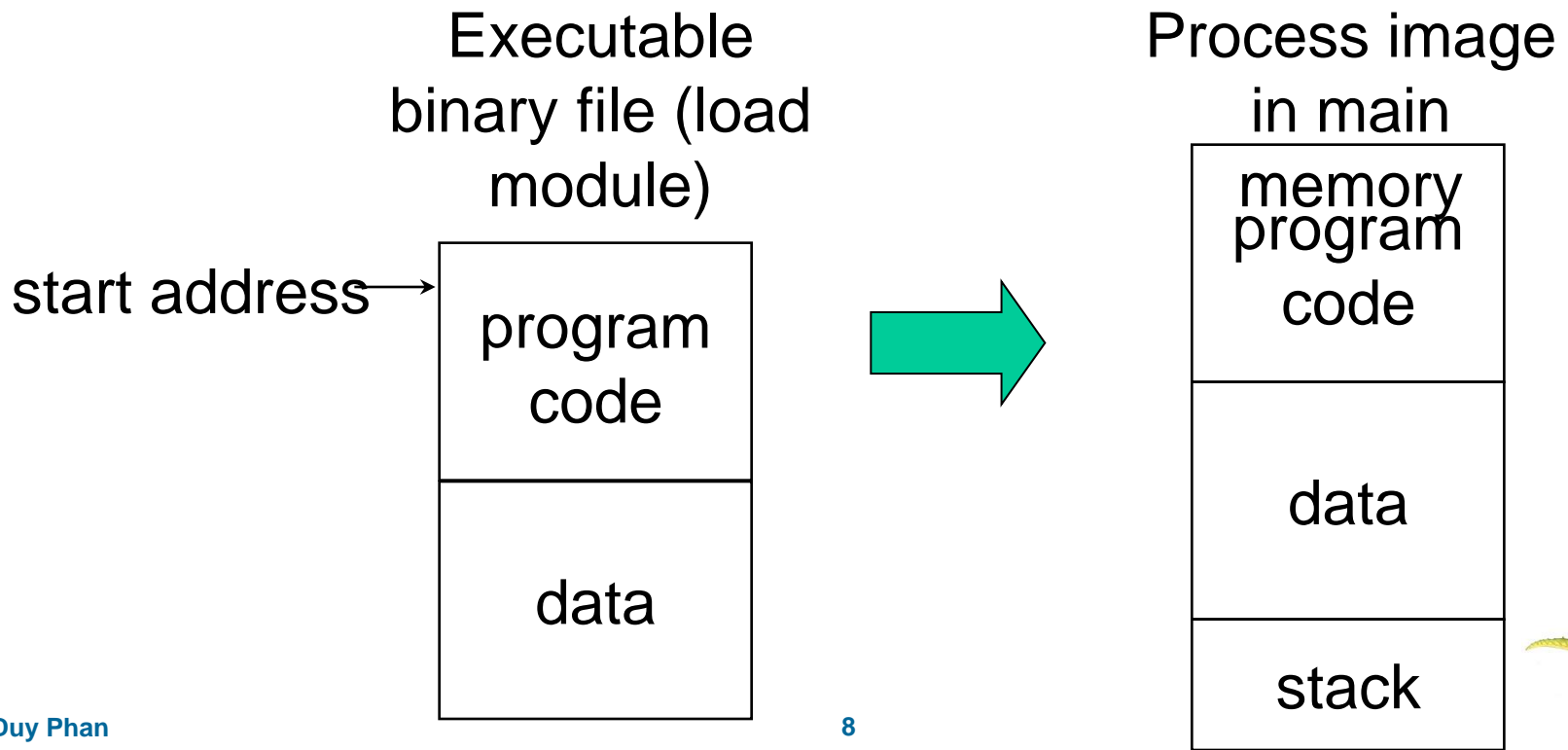




Khái niệm cơ bản (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Chương trình -> tiến trình:
 - Dùng load module để biểu diễn chương trình thực thi được
 - Layout luận lý của process image

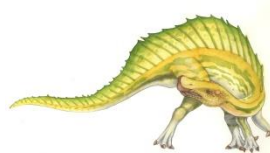




Khái niệm cơ bản (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Các bước khởi tạo tiến trình:
 - Cấp phát một định danh duy nhất cho tiến trình
 - Cấp phát không gian nhớ để nạp tiến trình
 - Khởi tạo khối dữ liệu Process Control Block (PCB) cho tiến trình
 - Thiết lập các mối liên hệ cần thiết (ví dụ: sắp PCB vào hàng đợi định thời, ...)





Trạng thái tiến trình

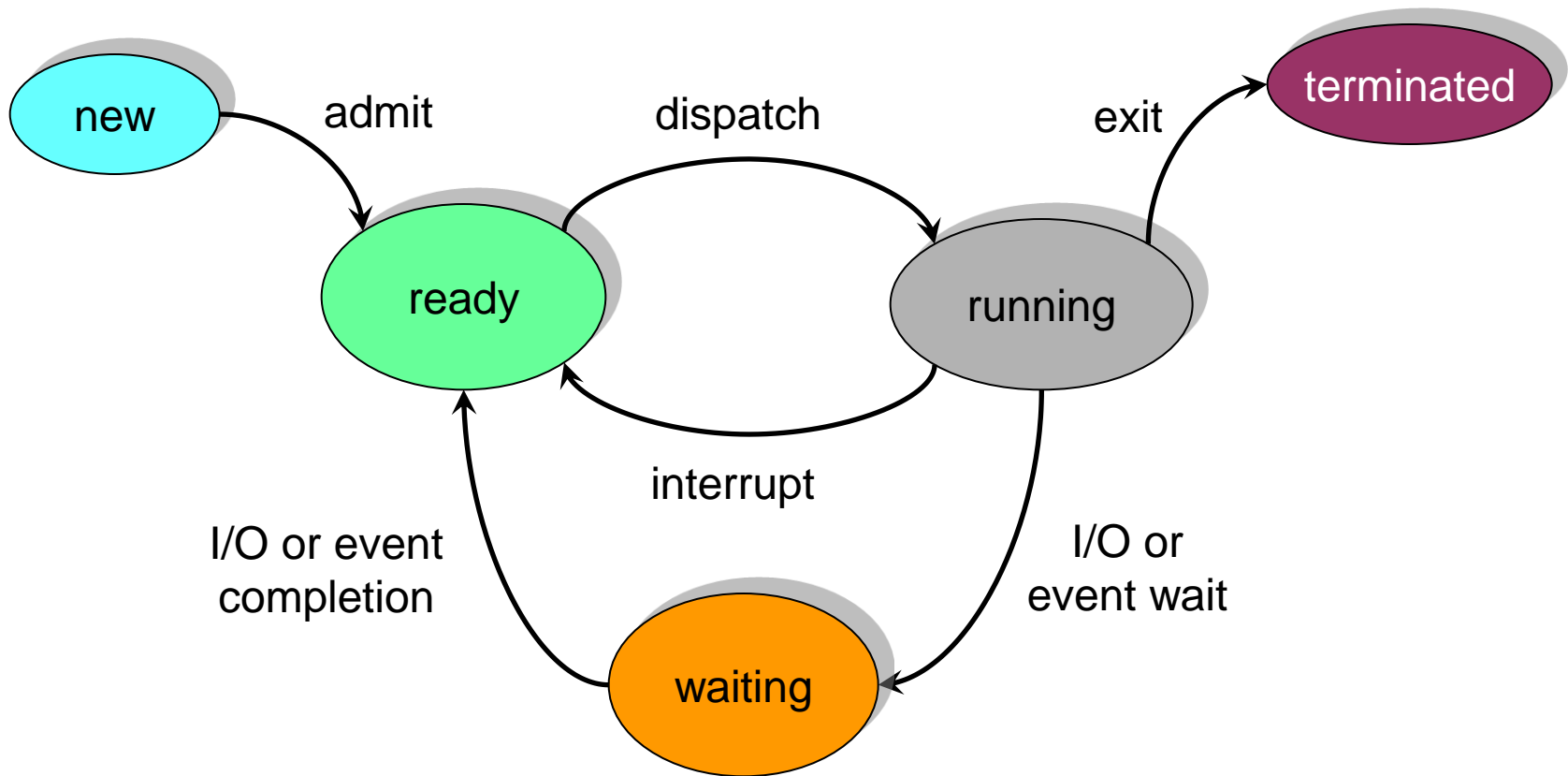
- **new**: tiến trình vừa được tạo
- **ready**: tiến trình đã có đủ tài nguyên, chỉ còn cần CPU
- **running**: các lệnh của tiến trình đang được thực thi
- **waiting**: hay là blocked, tiến trình đợi I/O hoàn tất, tín hiệu
- **terminated**: tiến trình đã kết thúc





Trạng thái tiến trình (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN



Chuyển đổi giữa các trạng thái của tiến trình





Trạng thái tiến trình (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

```
/* test.c */  
int main(int argc, char** argv)  
{  
    printf("Hello world\n");  
    exit(0);  
}
```

Biên dịch chương trình trong Linux: **gcc test.c -o test**

Thực thi chương trình test:
./test

Trong hệ thống sẽ có một tiến trình test được tạo ra, thực thi và kết thúc.

- Chuỗi trạng thái của tiến trình test như sau (trường hợp tốt nhất):
 - new
 - ready
 - running
 - waiting (do chờ I/O khi gọi printf)
 - ready
 - running
 - terminated

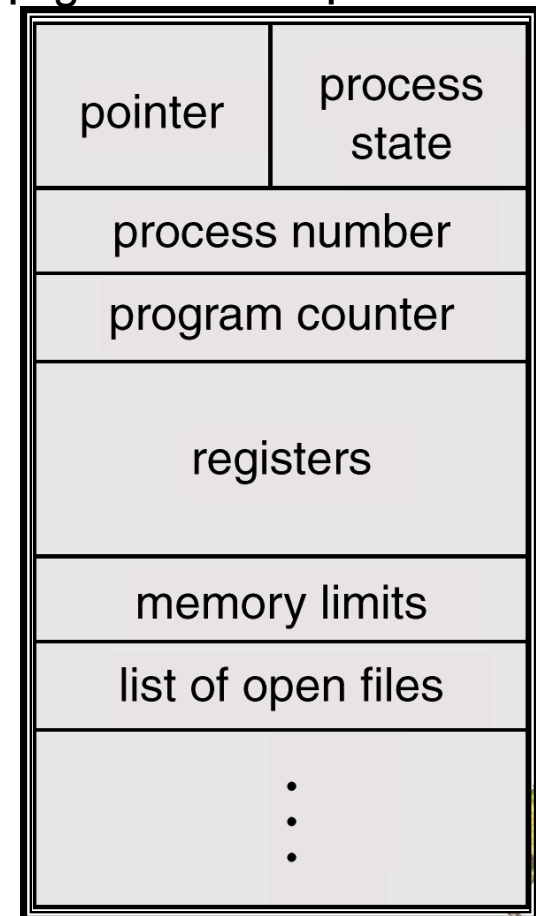




Process control block

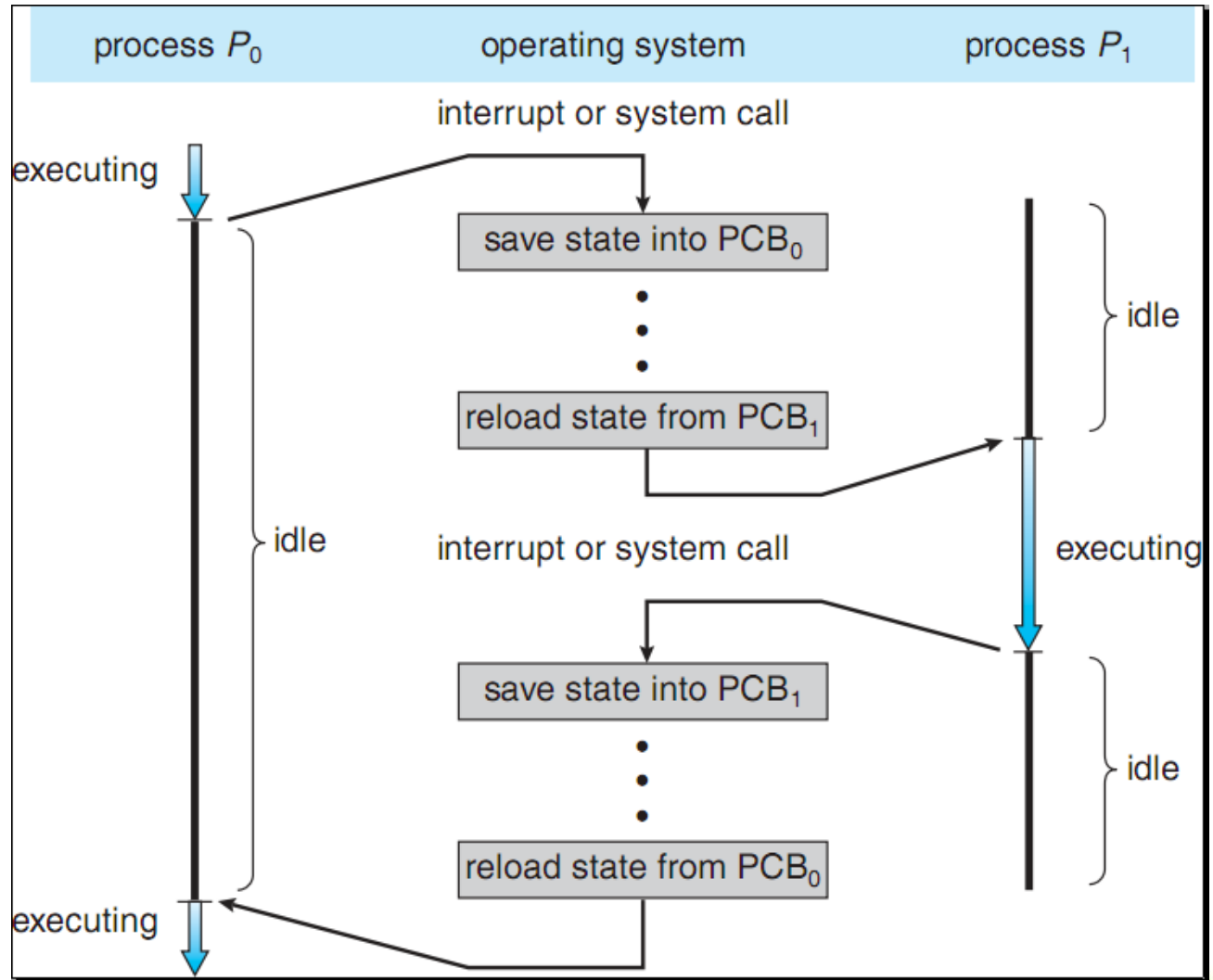
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Mỗi tiến trình trong hệ thống đều được cấp phát một Process Control Block (PCB)
 - PCB là một trong các cấu trúc dữ liệu quan trọng nhất của hệ điều hành
- PCB gồm:
 - Trạng thái tiến trình: new, ready, running,...
 - Bộ đếm chương trình
 - Các thanh ghi
 - Thông tin lập thời biểu CPU: độ ưu tiên, ...
 - Thông tin quản lý bộ nhớ
 - Thông tin: lượng CPU, thời gian sử dụng,
 - Thông tin trạng thái I/O



Process control block (tt)

- Lưu đồ chuyển CPU từ tiến trình này đến tiến trình khác





Yêu cầu đối với hệ điều hành về quản lý tiến trình

- Hỗ trợ sự thực thi luân phiên giữa nhiều tiến trình
 - Hiệu suất sử dụng CPU
 - Thời gian đáp ứng
- Phân phối tài nguyên hệ thống hợp lý
- Tránh deadlock, trì hoãn vô hạn định
- Cung cấp cơ chế giao tiếp và đồng bộ hoạt động các tiến trình
- Cung cấp cơ chế hỗ trợ user tạo/kết thúc tiến trình

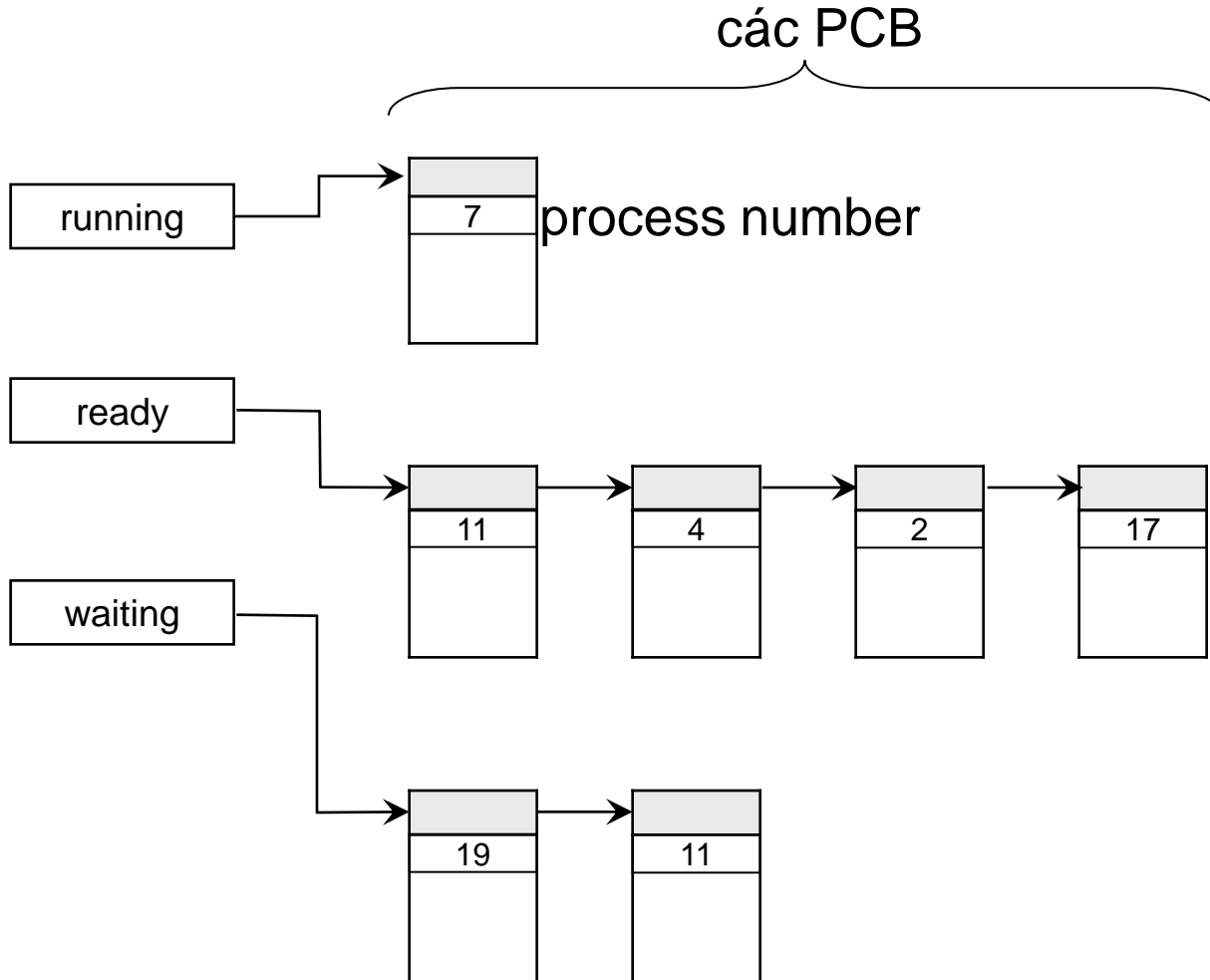




Quản lý các tiến trình: các hàng đợi

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Ví dụ





Định thời tiến trình

■ Tại sao phải định thời?

● Đa chương

- Có vài tiến trình chạy tại các thời điểm
- Mục tiêu: tận dụng tối đa CPU

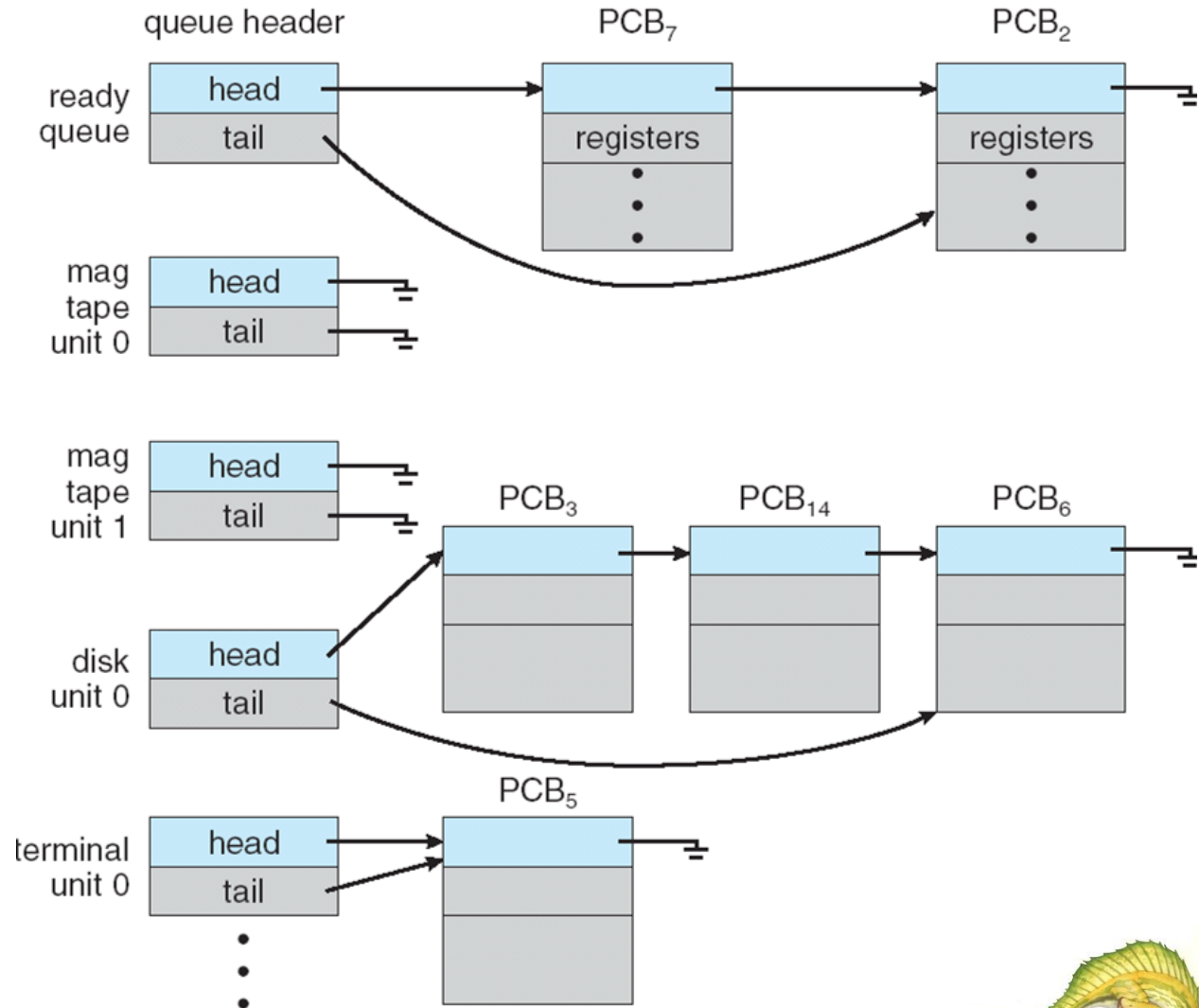
● Chia thời

- User tương tác với mỗi chương trình đang thực thi
- Mục tiêu: tối thiểu thời gian đáp ứng

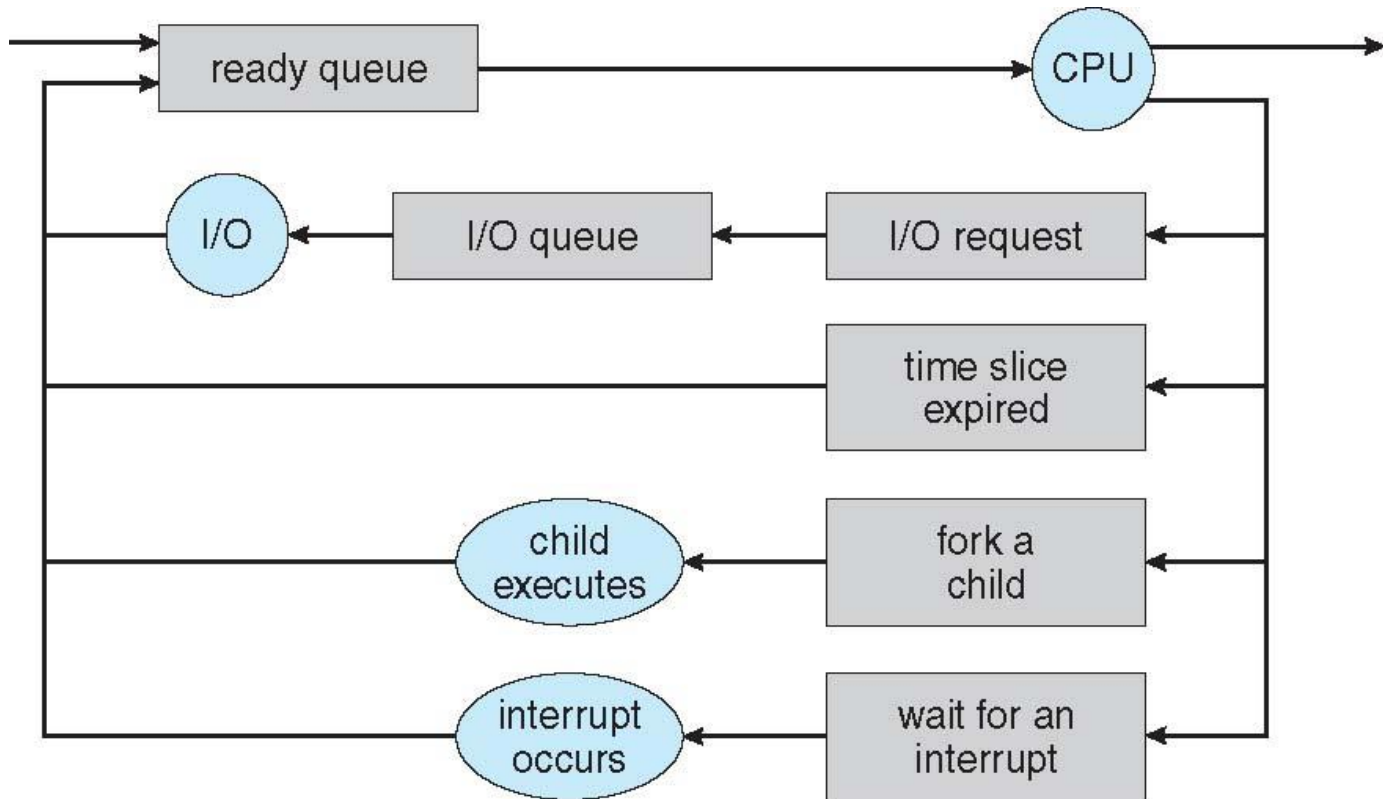


Các hàng đợi định thời

- Hàng đợi công việc-Job queue
- Hàng đợi sẵn sàng-Ready queue
- Hàng đợi thiết bị-Device queues
- ...



Các hàng đợi định thời (tt)



Lưu đồ hàng đợi của định thời tiến trình



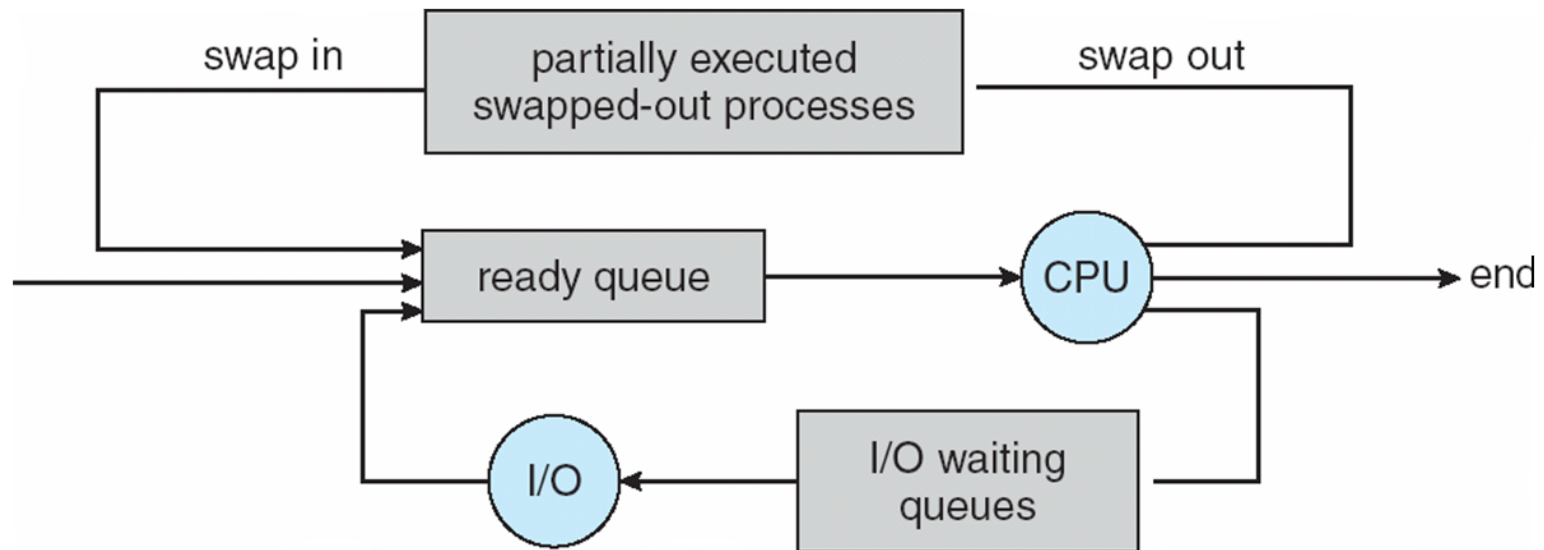


Bộ định thời

- Bộ định thời công việc (Job scheduler) hay bộ định thời dài (long-term scheduler)
- Bộ định thời CPU hay bộ định thời ngắn
- Các tiến trình có thể mô tả như:
 - tiến trình hướng I/O
 - tiến trình hướng CPU
- Thời gian thực hiện khác nhau -> kết hợp hài hòa giữa chúng



- Đôi khi hệ điều hành (như time-sharing system) có thêm medium-term scheduling để điều chỉnh mức độ đa chương của hệ thống
- Medium-term scheduler
 - chuyển tiến trình từ bộ nhớ sang đĩa (swap out)
 - chuyển tiến trình từ đĩa vào bộ nhớ (swap in)





Các tác vụ đối với tiến trình

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

■ Tạo tiến trình mới:

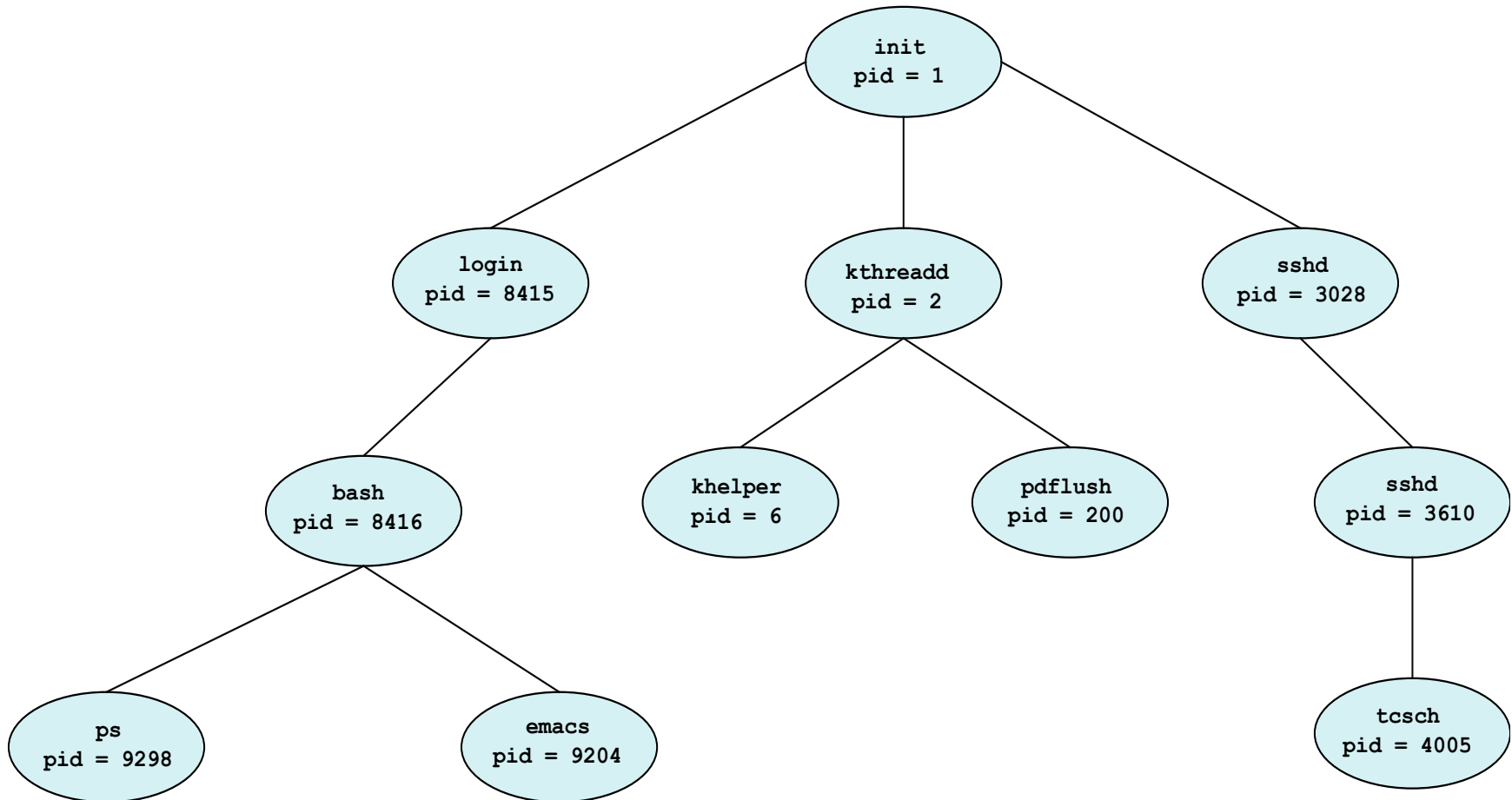
- Một tiến trình có thể tạo nhiều tiến trình mới thông qua một lời gọi hệ thống create-process (vd: hàm fork trong Unix)
 - Ví dụ: (Unix) Khi user đăng nhập hệ thống, một command interpreter (shell) sẽ được tạo ra cho user
- tiến trình được tạo là tiến trình con của tiến trình tạo (tiền trình cha)
 - Quan hệ cha-con định nghĩa một cây tiến trình





Cây tiến trình trong Linux/Unix

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN





Các tác vụ đối với tiến trình (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

■ Tạo tiến trình mới:

- tiến trình con nhận tài nguyên: từ HĐH hoặc từ tiến trình cha
- Chia sẻ tài nguyên của tiến trình cha
 - tiến trình cha và con chia sẻ mọi tài nguyên
 - tiến trình con chia sẻ một phần tài nguyên của cha
- Trình tự thực thi
 - tiến trình cha và con thực thi đồng thời (concurrently)
 - tiến trình cha đợi đến khi các tiến trình con kết thúc





Về quan hệ cha/con

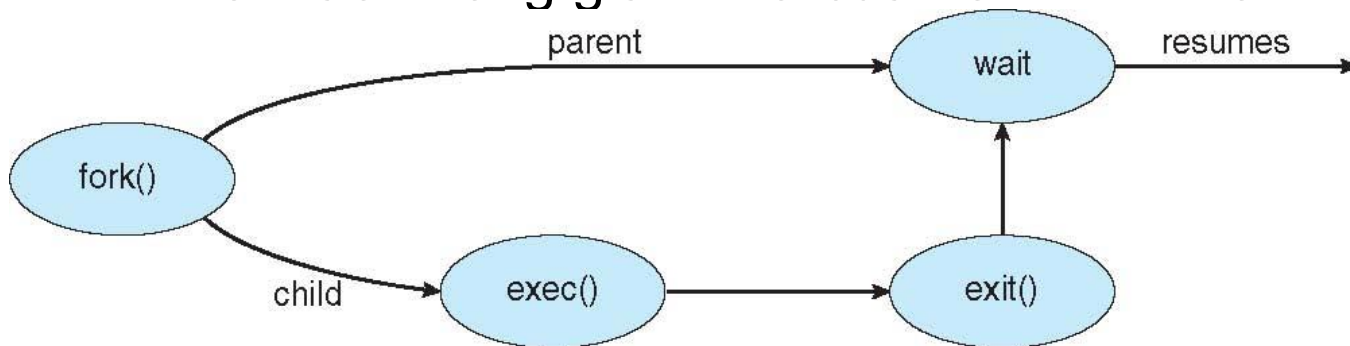
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

■ Không gian địa chỉ:

- Không gian địa chỉ của tiến trình con được nhân bản từ cha
- Không gian địa chỉ của tiến trình con được khởi tạo từ template

■ Ví dụ trong Unix/Linux

- System call `fork()` tạo một tiến trình mới
- System call `exec()` dùng sau `fork()` để nạp một chương trình mới vào không gian nhớ của tiến trình mới





Ví dụ tạo process với fork()

```
#include <stdio.h>
#include <unistd.h>
int main (int argc, char *argv[]){
    int pid;
    /* create a new process */
    pid = fork();
    if (pid > 0){
        printf("This is parent process");
        wait(NULL);
        exit(0);}
    else if (pid == 0) {
        printf("This is child process");
        execlp("/bin/ls", "ls", NULL);
        exit(0);}
    else {    // pid < 0
        printf("Fork error\n");
        exit(-1);
    }
}
```





Các tác vụ đối với tiến trình (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

■ Kết thúc tiến trình:

- tiến trình tự kết thúc
 - ▶ tiến trình kết thúc khi thực thi lệnh cuối và gọi system routine exit
- tiến trình kết thúc do tiến trình khác (có đủ quyền, vd: tiến trình cha của nó)
 - ▶ Gọi system routine abort với tham số là pid (process identifier) của tiến trình cần được kết thúc
- Hệ điều hành thu hồi tất cả các tài nguyên của tiến trình kết thúc (vùng nhớ, I/O buffer,...)





Cộng tác giữa các tiến trình

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Trong tiến trình thực thi, các tiến trình có thể cộng tác (cooperate) để hoàn thành công việc
- Các tiến trình cộng tác để
 - Chia sẻ dữ liệu (information sharing)
 - Tăng tốc tính toán (computational speedup)
 - ▶ Nếu hệ thống có nhiều CPU, chia công việc tính toán thành nhiều công việc tính toán nhỏ chạy song song
 - Thực hiện một công việc chung
 - ▶ Xây dựng một phần mềm phức tạp bằng cách chia thành các module/process hợp tác nhau
- Sự cộng tác giữa các tiến trình yêu cầu hệ điều hành hỗ trợ cơ chế giao tiếp và cơ chế đồng bộ hoạt động của các tiến trình





Bài toán người sản xuất-người tiêu thụ

- Producer tạo ra các dữ liệu và consumer tiêu thụ, sử dụng các dữ liệu đó
- Sự trao đổi thông tin thực hiện qua buffer
 - unbounded buffer: kích thước buffer vô hạn (không thực tế)
 - bounded buffer: kích thước buffer có hạn
- Producer và consumer phải hoạt động đồng bộ vì
 - Consumer không được tiêu thụ khi producer chưa sản xuất
 - Producer không được tạo thêm sản phẩm khi buffer đầy





Giao tiếp liên tiến trình

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- IPC là cơ chế cung cấp bởi hệ điều hành nhằm giúp các tiến trình:
 - Giao tiếp với nhau
 - Đồng bộ hoạt động
 - mà không cần chia sẻ không gian địa chỉ
- IPC có thể được cung cấp bởi message passing system





Hệ thống truyền thông điệp

Làm thế nào để các tiến trình giao tiếp nhau?

- Đặt tên (Naming)
 - Giao tiếp trực tiếp
 - ▶ `send(P, msg)`: gửi thông điệp đến tiến trình P
 - ▶ `receive(Q, msg)`: nhận thông điệp đến từ tiến trình Q
 - Giao tiếp gián tiếp: thông qua mailbox hay port
 - ▶ `send(A, msg)`: gửi thông điệp đến mailbox A
 - ▶ `receive(Q, msg)`: nhận thông điệp từ mailbox B
- Đồng bộ hóa (Synchronization): blocking send, nonblocking send, blocking receive, nonblocking receive





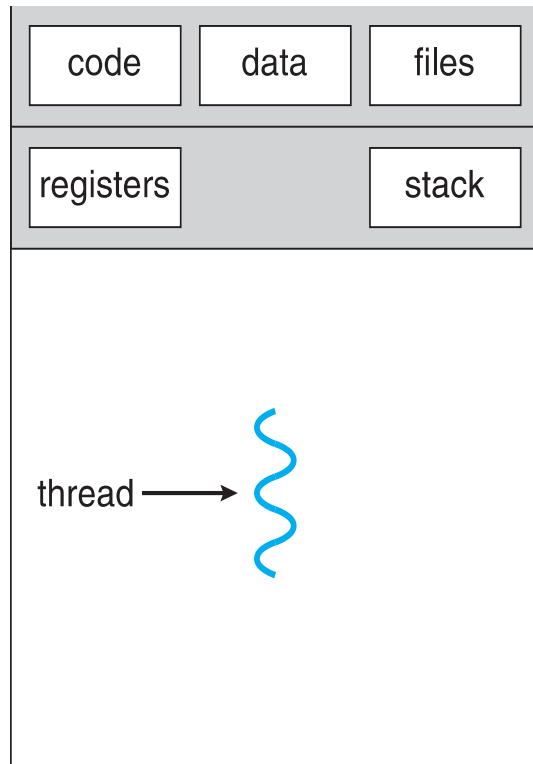
Làm thế nào để các tiến trình giao tiếp nhau?

- Tạo vùng đệm (Buffering): dùng queue để tạm chứa các message
 - Khả năng chứa là 0 (Zero capacity hay no buffering)
 - Bounded capacity: độ dài của queue là giới hạn
 - Unbounded capacity: độ dài của queue là không giới hạn

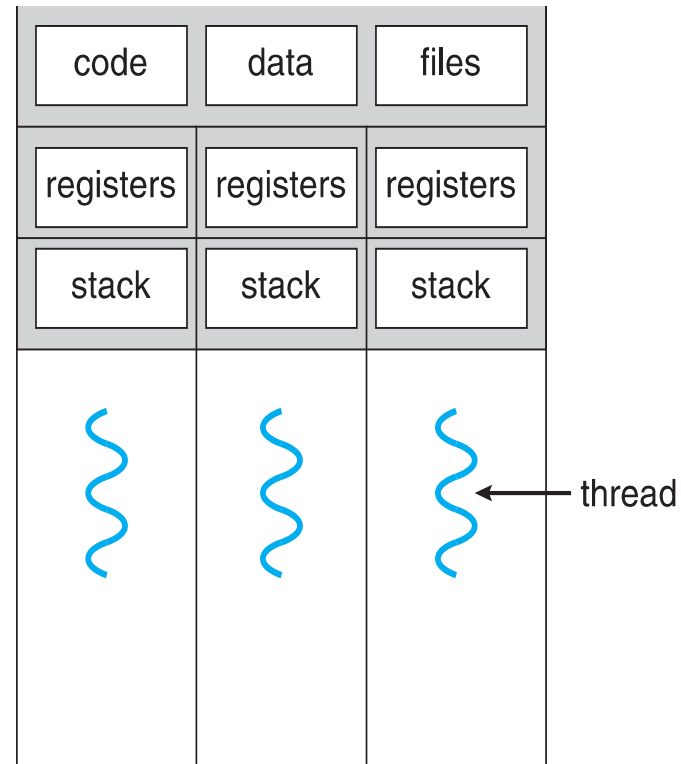


Tiểu trình - Thread

- Tiểu trình: là một đơn vị cơ bản sử dụng CPU gồm:
- Thread ID, PC, Registers, Stack và chia sẻ chung code, data, resources (files)



single-threaded process



multithreaded process



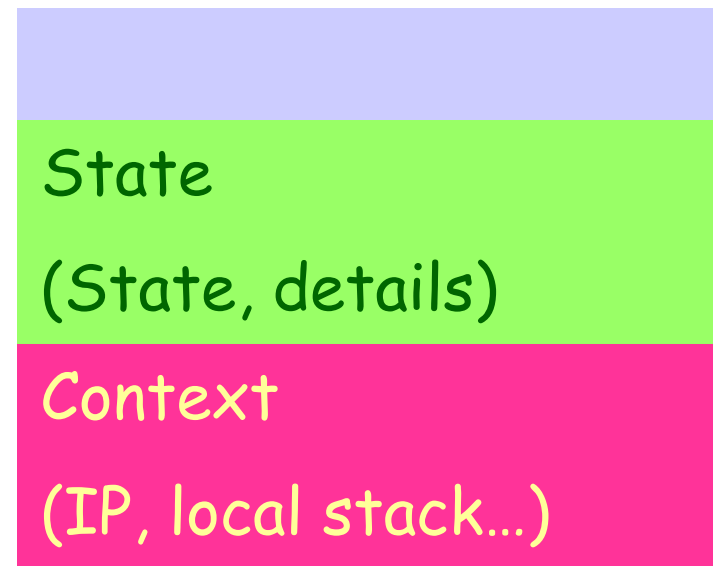


PCB và TCB trong mô hình multithreads

PCB



Thread Control Block TCB





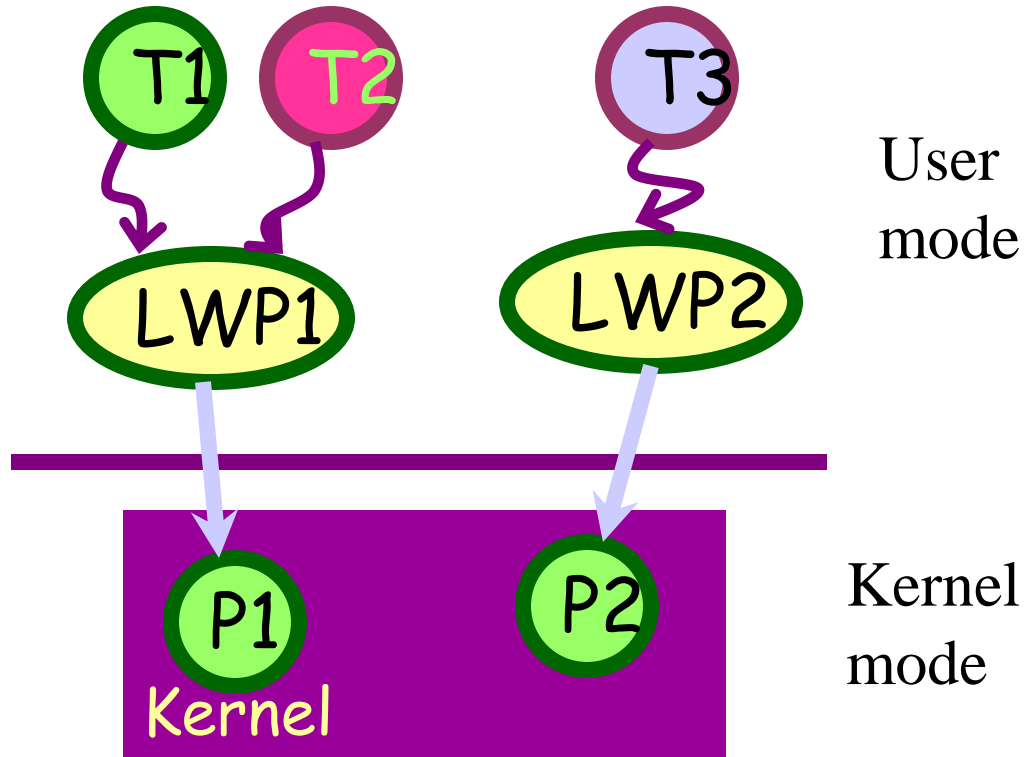
Lợi ích của tiến trình đa luồng

- Đáp ứng nhanh: cho phép chương trình tiếp tục thực thi khi một bộ phận bị khóa hoặc một hoạt động dài
- Chia sẻ tài nguyên: tiết kiệm không gian nhớ
- Kinh tế: tạo và chuyển ngữ cảnh nhanh hơn tiến trình
 - Ví dụ: Trong Solaris 2, tạo process chậm hơn 30 lần, chuyển chậm hơn 5 lần so với thread
- Trong multiprocessor: có thể thực hiện song song





Tiểu trình người dùng (User thread)



Khái niệm tiểu trình được hỗ trợ bởi một thư viện hoạt động trong user mode





Tiểu trình hạt nhân (Kernel thread)



Khái niệm tiểu trình được xây dựng bên trong hạt nhân





Tổng kết

- Trạng thái tiến trình
- Khối điều khiển tiến trình
- Định thời tiến trình
- Các tác vụ đối với tiến trình
- Sự cộng tác giữa các tiến trình
- Giao tiếp giữa các tiến trình
- Tiểu trình





UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập



Kết thúc chương 3

