Thread

- Khái niệm tổng quan
- Các mô hình multithread
- Pthread (POSIX thread)

Xem xét lại khái niệm quá trình

- Nhìn lại và phân tích khái niệm quá trình truyền thống: quá trình gồm
  - 1. Không gian nhớ / địa chỉ
    - chứa text (code), data, heap
  - 2. Một luồng thực thi duy nhất (single thread of execution)
    - program counter
    - các register
    - stack
  - 3. Các tài nguyên khác (các open file, các quá trình con,…)

Mở rộng khái niệm quá trình

- Nhìn lại ‘cooperating processes’
  - Web server tạo một process con cho mỗi client để phục vụ yêu cầu trang web, hình ảnh, âm thanh…
    - Tạo process tốn thời gian và tài nguyên
- Mở rộng khái niệm quá trình truyền thống bằng cách hiện thực nhiều luồng thực thi trong cùng một môi trường của quá trình

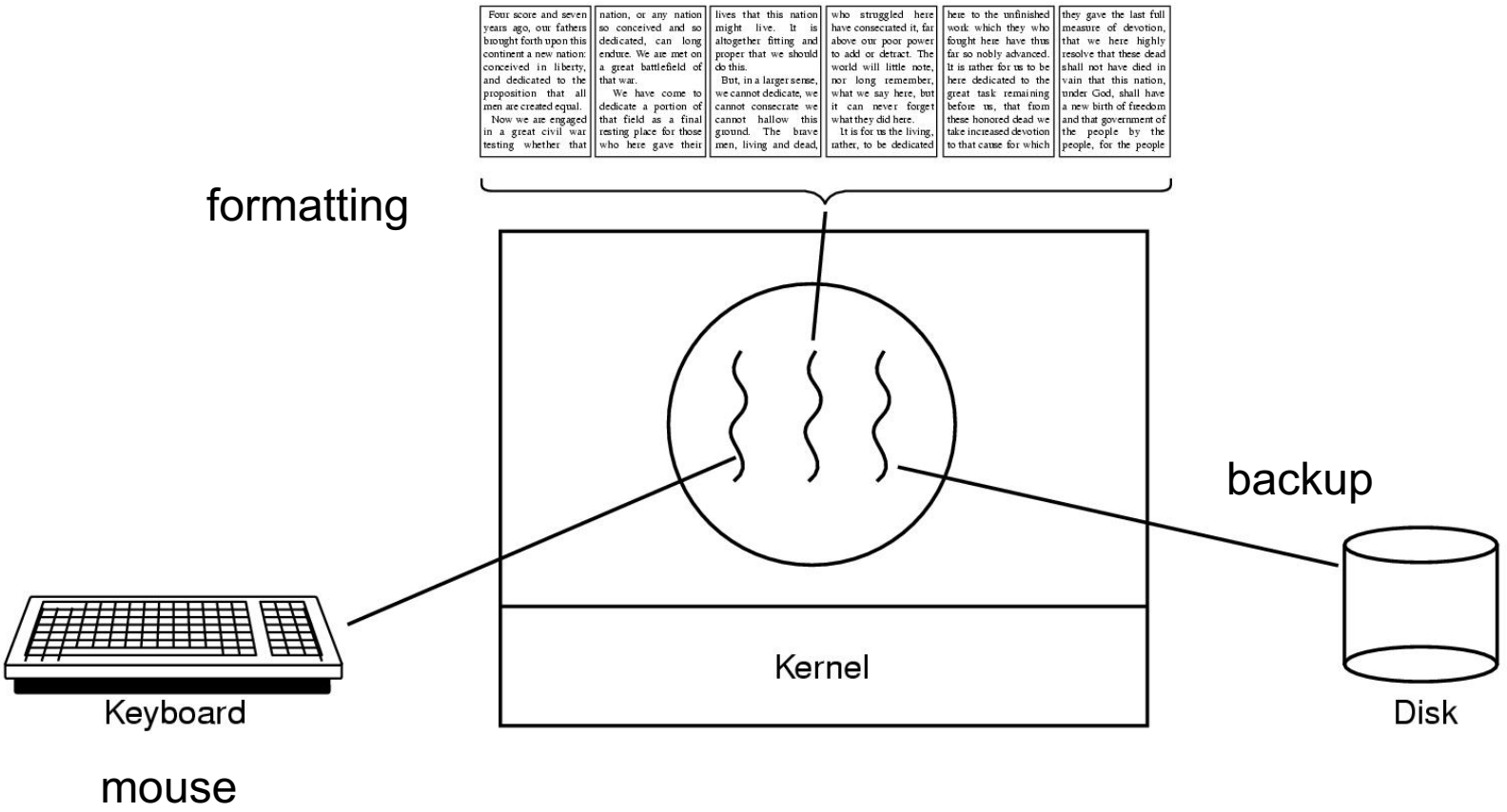
Mở rộng khái niệm quá trình

- Quá trình gồm
  - 1. Không gian địa chỉ
  - 2'. Một hay nhiều luồng thực thi, mỗi luồng thực thi (thread) có riêng
    - program counter
    - các register
    - stack
  - 3. Các tài nguyên khác (các open file, các quá trình con,…)

Quá trình multithreaded

- Khi quá trình khởi đầu chỉ có main (hay initial) thread thực thi
  - Main thread sẽ tạo các thread khác
- Các thread trong cùng một process chia sẻ code, data và tài nguyên khác (các file đang mở,...) của process
- Quá trình *đa luồng* (multithreaded process) là quá trình có nhiều luồng

Sử dụng thread



Trình soạn thảo văn bản với ba thread

Các trường tiêu biểu của PCB

Process management	**Memory management**	**File management**
Registers	Pointer to text segment	Root directory
Program counter	Pointer to data segment	Working directory
Program status word	Pointer to stack segment	File descriptors
Stack pointer		User ID
Process state		Group ID
Priority		
Scheduling parameters		
Process ID		
Parent process		
Process group		
Signals		
Time when process started		
CPU time used		
Children's CPU time		
Time of next alarm		

Tanenbaum

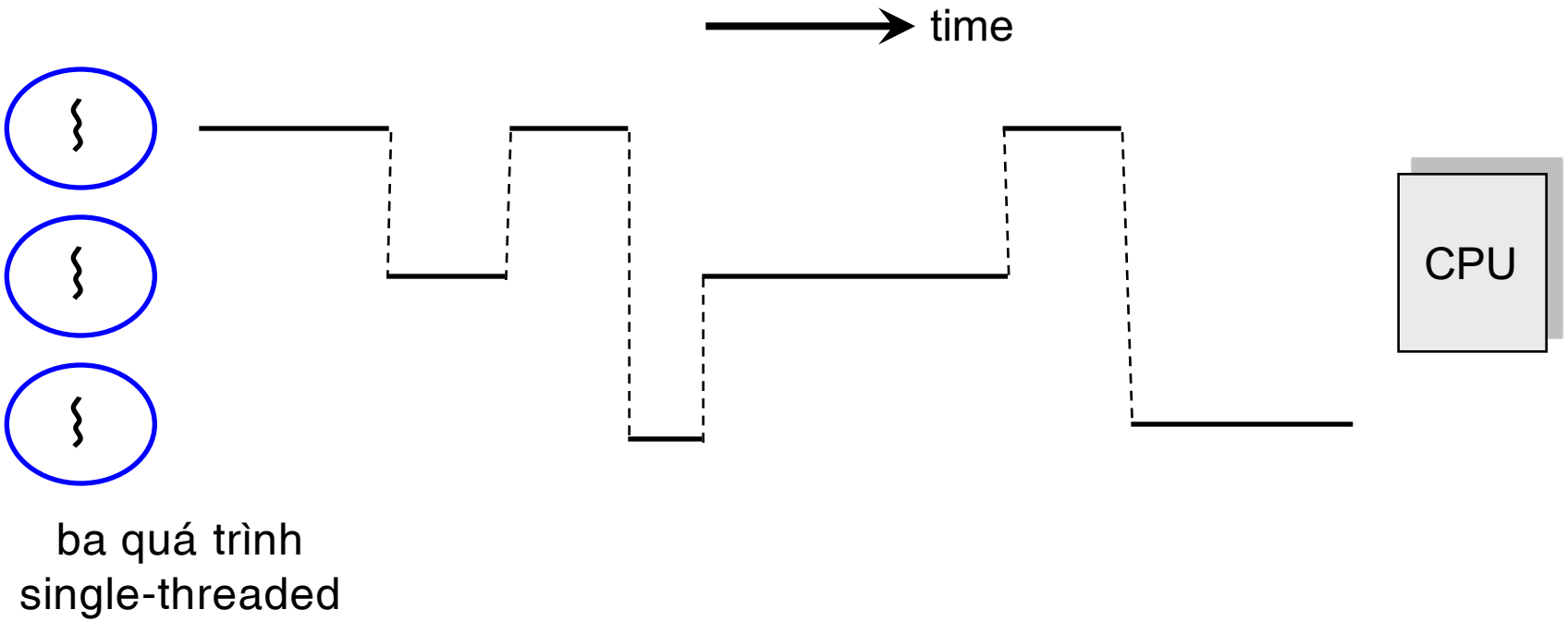
Process & thread information

Per process items
Address space
Open files
Child processes
Signals & handlers
Accounting info
Global variables

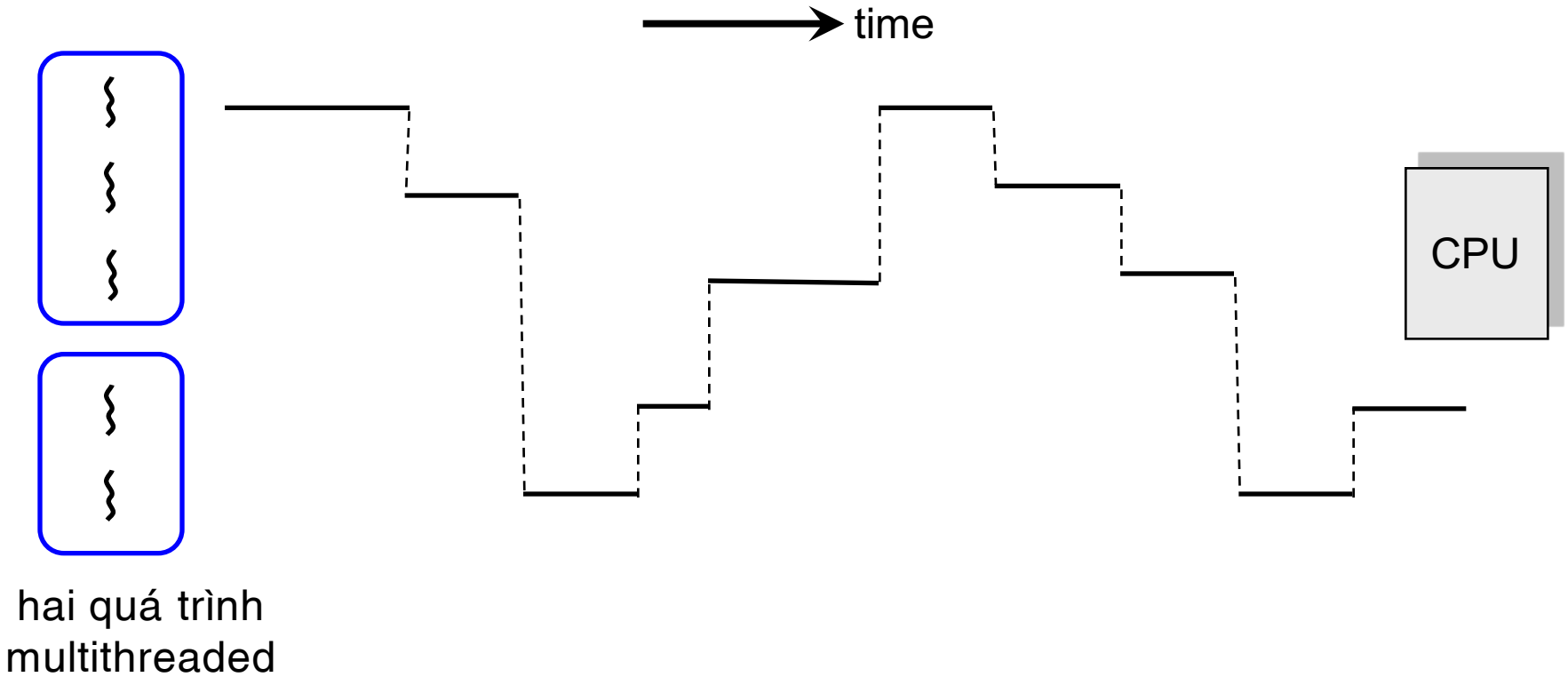
Per thread items	**Per thread items**	**Per thread items**
Program counter	Program counter	Program counter
Registers	Registers	Registers
Stack & stack pointer	Stack & stack pointer	Stack & stack pointer
State	State	State

Quá trình có ba thread

Chia sẻ CPU giữa các thread (1/2)

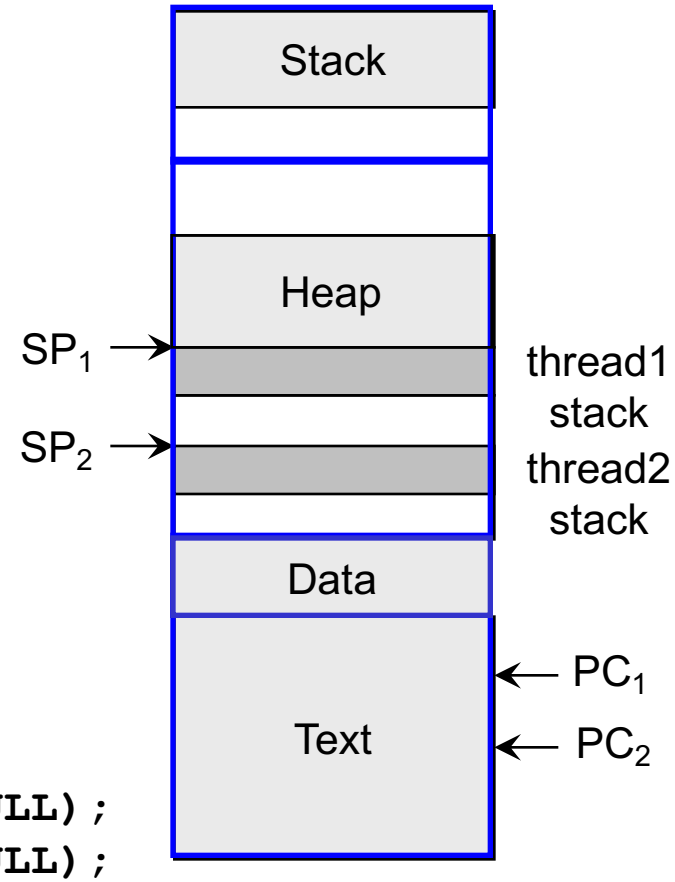


Chia sẻ CPU giữa các thread (2/2)



Ví dụ chương trình sử dụng Pthread

```
#include <stdio.h>
void* thread1(){
   int i;
   for (i = 0; i < 10; i++){
        printf("Thread 1\n"); sleep(1);
   }
}
void* thread2(){
   int i;
   for (i = 0; i < 10; i++){
        printf("Thread 2\n"); sleep(1);
}
int main(){
   pthread_t th1, th2;
   pthread_create(&th1, NULL, thread1, NULL);
   pthread_create(&th2, NULL, thread2, NULL);
   sleep(20);
   return 0;
}
```



Sơ đồ bộ nhớ

Chương trình này khi chạy có bao nhiêu thread?

Ưu điểm của thread

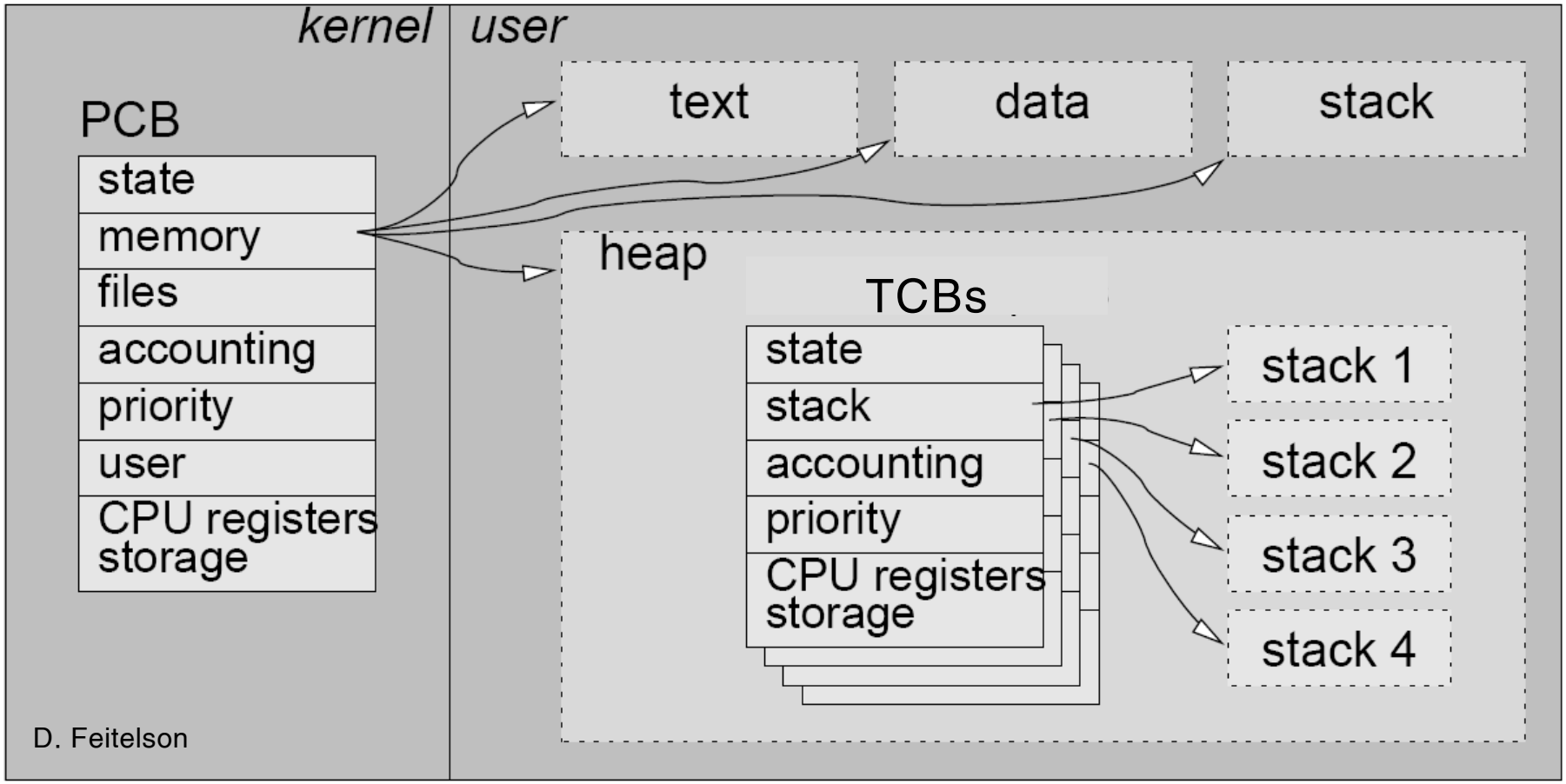
- Tính đáp ứng cao cho các ứng dụng tương tác
- Chia sẻ tài nguyên giữa các thread: vd memory
- Tiết kiệm chi phí hệ thống
  - Chi phí tạo/quản lý thread nhỏ hơn so với quá trình
  - Chi phí chuyển ngữ cảnh giữa các thread nhỏ hơn so với quá trình
- Tận dụng được đa xử lý (multiprocessor)
  - Mỗi thread chạy trên một processor riêng, do đó tăng mức độ song song của chương trình

User thread (1/4)

- Một *thư viện thread* (thread library, run-time system) được hiện thực trong user space để hổ trợ các tác vụ lên thread
  - Thư viện thread cung cấp các hàm khởi tạo, định thời và quản lý thread như
    - thread_create
    - thread_exit
    - thread_join
    - thread_yield
  - Thư viện thread dùng *Thread Control Block* (TCB) để lưu thông tin về user thread (program counter, các register, stack)

User thread (2/4)

- Cấu trúc dữ liệu và memory layout để hiện thực user thread

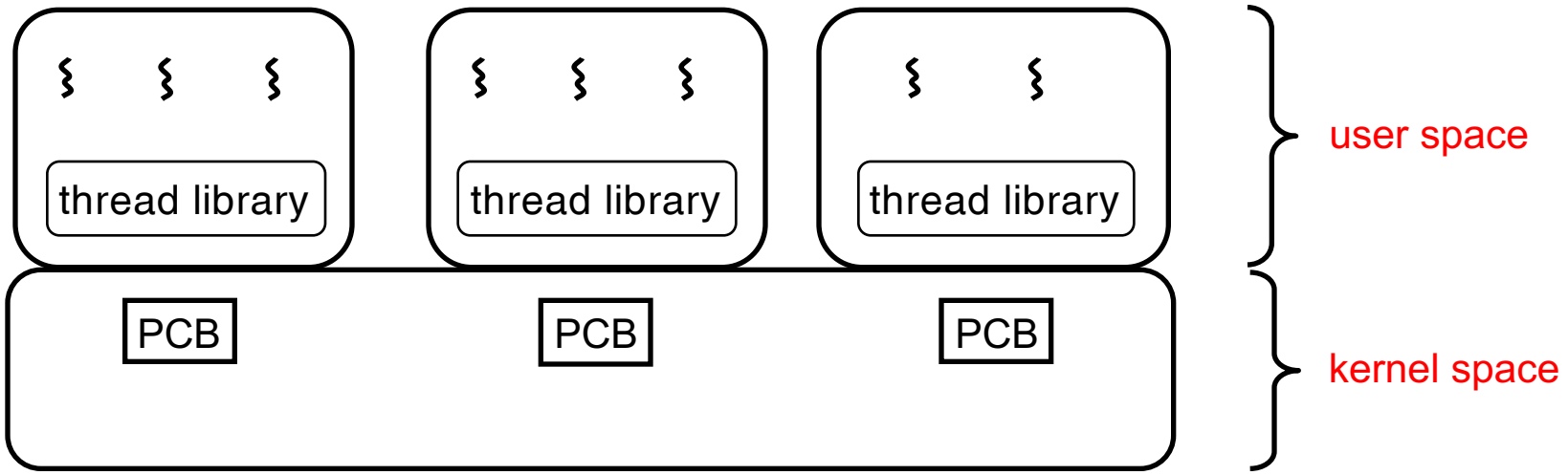


User thread (3/4)

- Kernel không biết sự có mặt của user thread
  - Kernel chỉ biết PCB của quá trình
- Ví dụ thư viện user thread
  - POSIX Pthread

User thread (4/4)

- Vấn đề: hệ điều hành cấp phát duy nhất một PCB cho mỗi process (→ main/initial thread)
  - *Blocking problem:* Khi một thread trở nên blocked thì mọi thread khác của process sẽ không tiến triển được



Kernel thread (1/3)

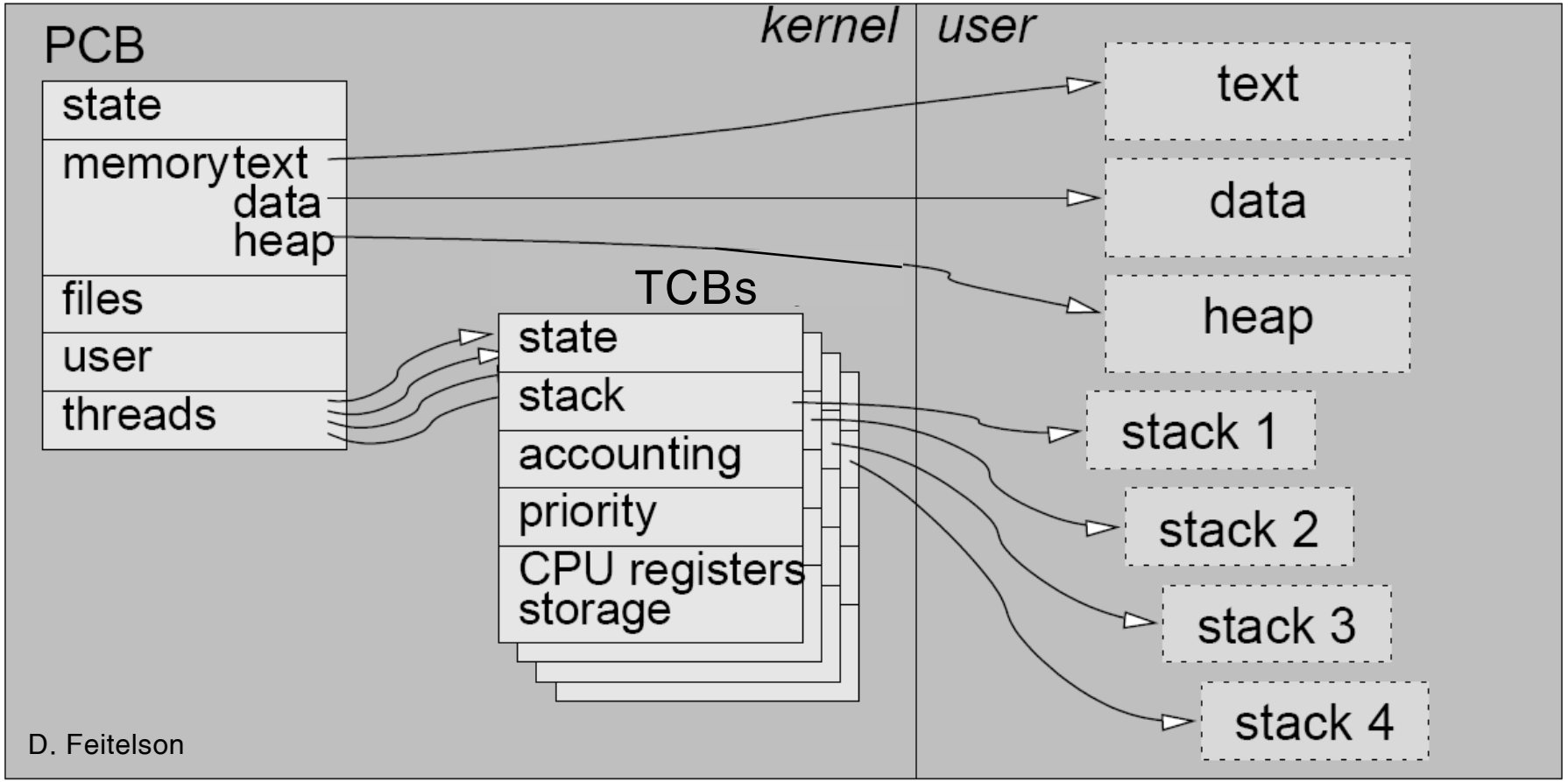
- Khi kỹ thuật multithreading được hệ điều hành trực tiếp hỗ trợ
  - Kernel quản lý cả process và các thread – **kernel thread**
  - Kernel thực hiện việc định thời CPU cho các thread của quá trình

Kernel thread (2/3)

- Khi multithreading được hỗ trợ bởi kernel
  - Khởi tạo và quản lý thread chậm hơn so với user thread do system call overhead chuyển user mode ↔ kernel mode
  - Tận dụng được lợi thế của kiến trúc multiprocessor
  - Dù một thread bị blocked, các thread khác của quá trình vẫn có thể tiến triển
- Một số hệ thống multithreading
  - Windows 9x/NT/200x
  - Solaris
  - Linux

Kernel thread (3/3)

- Cấu trúc dữ liệu và memory layout để hiện thực kernel thread

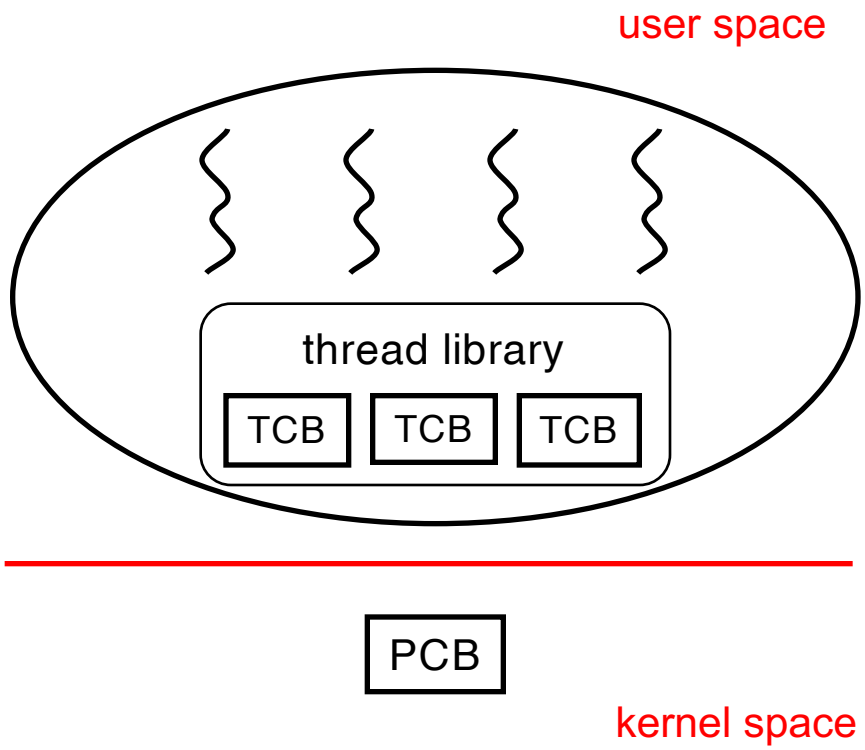


Các mô hình hiện thực thread

- Để thống nhất các ý niệm ‘user thread’ và ‘kernel thread’ đã được trình bày, ta dùng mô hình ‘X-to-X’ sau:
  - Khái niệm ‘user(-level) multithreading’ sẽ được dùng để chỉ multithreading trong user space
  - User(-level) multithreading có thể hiện thực theo một trong các mô hình sau
    - Mô hình *many-to-one* – tương ứng với ‘user thread’ cũ
    - Mô hình *one-to-one* – tương ứng với ‘kernel thread’ cũ
    - Mô hình *many-to-many*

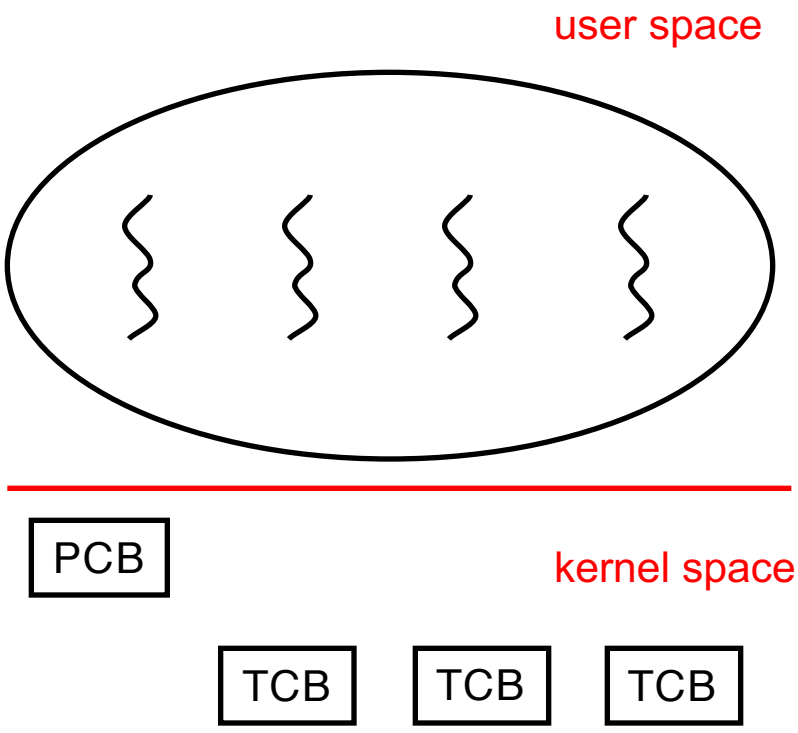
Mô hình many-to-one

- Nhiều user(-level) thread "chia sẻ" một PCB để thực thi
  - Việc quản lý thread được thực hiện thông qua các hàm của một thread library được gọi ở user level
  - Blocking problem: Khi một thread trở nên blocked thì mọi thread khác của process không tiến triển được
- Có thể được hiện thực đối với hầu hết các hệ điều hành



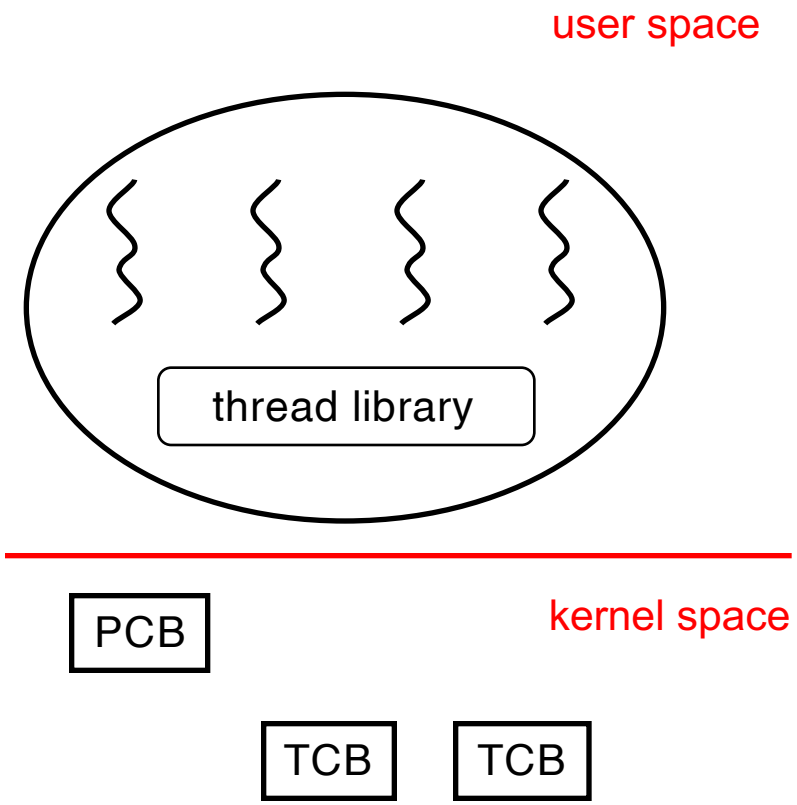
Mô hình one-to-one

- Mỗi user thread được hệ điều hành quản lý thông qua một kernel TCB riêng
  - Hệ điều hành phải cấp phát một kernel TCB để tạo user thread
- Hệ điều hành phải cung cấp được nhiều kernel TCB cho một quá trình
- Ví dụ: Windows NT/2000



Mô hình many-to-many

- Gọi ‘many-to-few’ thì đúng hơn
- Nhiều user-level thread được phân chia thực thi (multiplexed) thông qua một số kernel TCB của quá trình
  - Kết hợp ưu điểm của các mô hình many-to-one và one-to-one
- Ví dụ
  - Solaris 2
  - Windows NT/2000 với package ThreadFiber



Pthread

- Chuẩn POSIX (IEEE 1003.1c) đặc tả API cho các thủ tục tạo thread và đồng bộ thread
- Phổ biến trong các hệ thống UNIX/Linux
- Là một thư viện hỗ trợ user-level thread

- Biên dịch và thực thi chương trình multithreaded C trong Linux

```
$ gcc source_file.c -lpthread -o output_file
$ ./output_file
```