CS310 Operating Systems

Lecture 5: Process – PCB, System Calls

Ravi Mittal

IIT Goa

Acknowledgements !

- Contents of this class presentation has been taken from various sources. Thanks are due to the original content creators:
  - CS162, Operating System and Systems Programming, Profs. Natacha Crooks and Anthony D. Joseph, University of California, Berkeley
  - Book: Operating Systems: Three Easy Pieces, by Remzi and Andrea Arpaci-Dusseau
    - Chapter 5: Process APIs
  - Book: The Operating System Concepts, third edition: Silberschatz, Peter Galvin, Greg Gagne,
  - CS 423 Operating System Design, Uinv of Illinois, Prof Fagen-Ullmschneider

Read the following:

- Book: Operating Systems: Principles and Practice (2nd Edition) Anderson and Dahlin
  - Volume 1, Kernel and Processes
    - Chapter 2: Kernel Abstraction
- Book: Operating Systems: Three Easy Pieces, by Remzi and Andrea Arpaci-Dusseau
  - Chapter 5: Process APIs

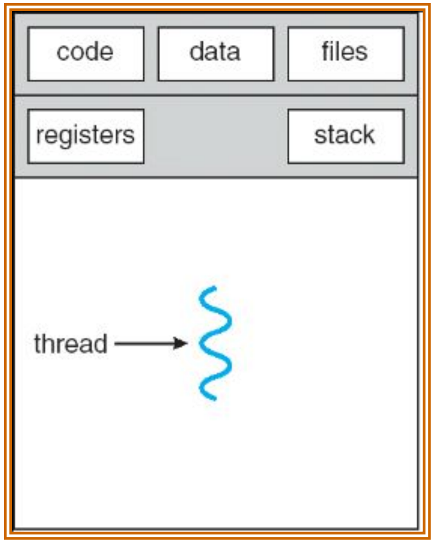
We will study..

- Last Class - Revision
- Process State
- Process Control Block
- Process Scheduling
- Process APIs and System Calls - Introduction

Last Class

Recall: Process

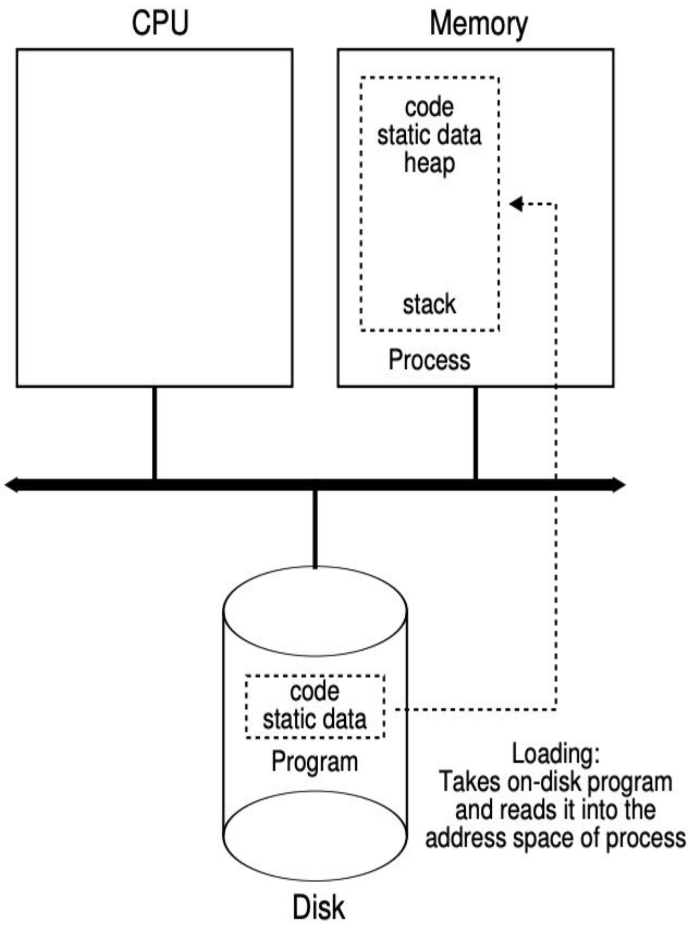
- A unique Identifier (PID)
- Memory Image
  - Code and Data (Static)
  - Stack and Heap (Dynamic)
- CPU Context
  - Registers
    - PC
    - Stack Pointer
    - Frame Pointer
    - General Purpose Registers
- File Descriptors
  - Pointers to open files and devices



Single-Threaded Process

Recall: Process Creation

- Allocates memory and creates memory image
    - Loads code, static data from disk executable (eg a.out)
    - Creates and initialized runtime stack and heap
- Opens basic files
    - Standard Input, Output, Error
        - Standard Input Output let programs read input from terminal and print output to screen
- Initializes CPU registers
    - PC points to first instruction
- In modern OS, process loading is



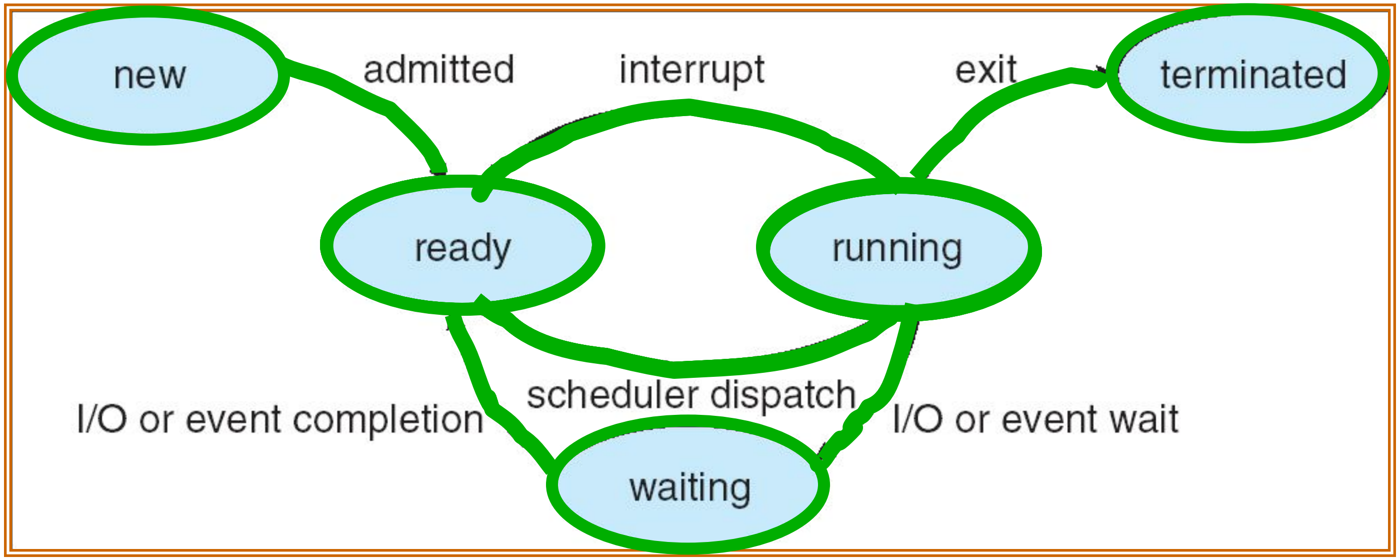
Process States

Process States

- Running
  - It is executing instructions
- Ready
  - A process is ready to run; Yet to be scheduled
- Waiting (blocked)
  - Suspended
  - Waiting for some event to be completed
    - Processes has initiated an I/O request to disk and gets blocked

- As a process executes, it moves from state to state

Lifecycle of a Process



- As a process executes, it changes state:
  - new: The process is being created
  - ready: The process is waiting to run
  - running: Instructions are being executed
  - waiting: Process waiting for some event to occur
  - terminated: The process has finished execution

Process States

Time	Process 0	Process 1	Notes
1	Running	Ready	
2	Running	Ready	Proc 0 initiates I/O
3	Blocked	Running	Proc 0 is blocked; Proc 1 running
4	Blocked	Running	
5	Ready	Running	Proc 0 I/O done
6	Ready	Running	Proc 1 now done
7	Running	-	Proc 0 is running
8	Running	-	Proc 0 is now done

Process Control Block

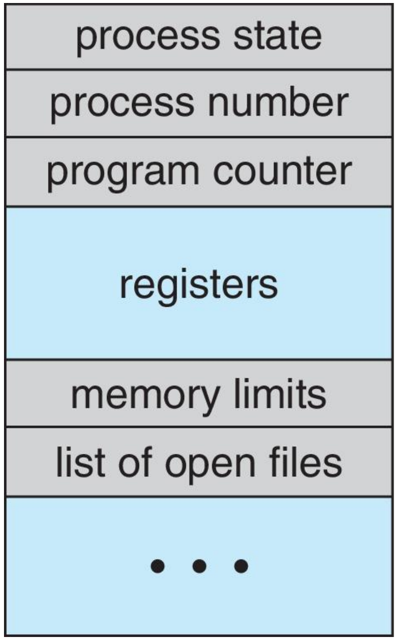
Process Control Block: OS Data Structure

- Operating system keeps track of various processes on a computer using Process Control Block (PCB)
- PCB stores all the information the OS needs about the process
- The OS allocates a new PCB on the creation of each process and places it on a state queue
- The OS deallocates the PCB when the process terminates
- PCBs are dynamically allocated in OS memory
  - User process can not access it

Process Control Block (PCB) - 1

Information associated with each process

- Process state – running, waiting, etc.,
- Program counter – location of instruction to next execute
- Stack pointer (user stack ; Kernel stack)
- CPU registers – contents of all process-centric registers
- CPU scheduling information- priorities, scheduling queue pointers
- Memory-management information – memory allocated to the process
  - Pointers to text, stack and heap segments
  - Page table related info
- Process Identification – process id (pid), Parent process id (ppid), user id (uid),
- Accounting information – CPU used, clock time elapsed since start, time limits
- I/O status information – I/O devices allocated to process, list of open files



process state

process number

program counter

registers

memory limits

list of open files

• • •

Book: The Operating System Concepts: Silberschatz, Galvin, Gagne

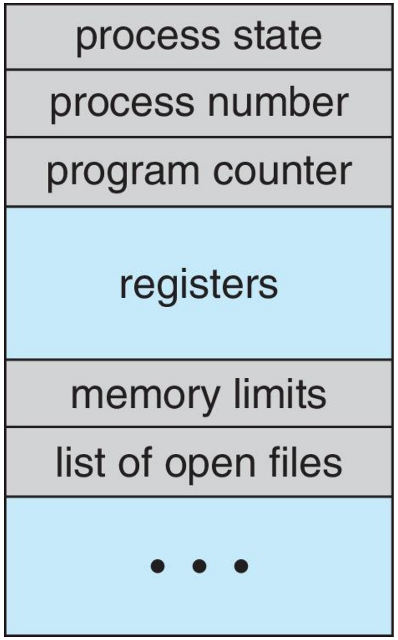
Process Control Block (PCB) - 2

- Data Structuring: pointers to other PCBs

- Process Privileges: Access privileges to certain memory area, critical structures etc

- Resource ownership
  - Pointer to opened files etc

- PCBs need to be protected from inadvertent destruction by any routine
  - Protection of PCBs is a critical issue in the design of an OS



process state

process number

program counter

registers

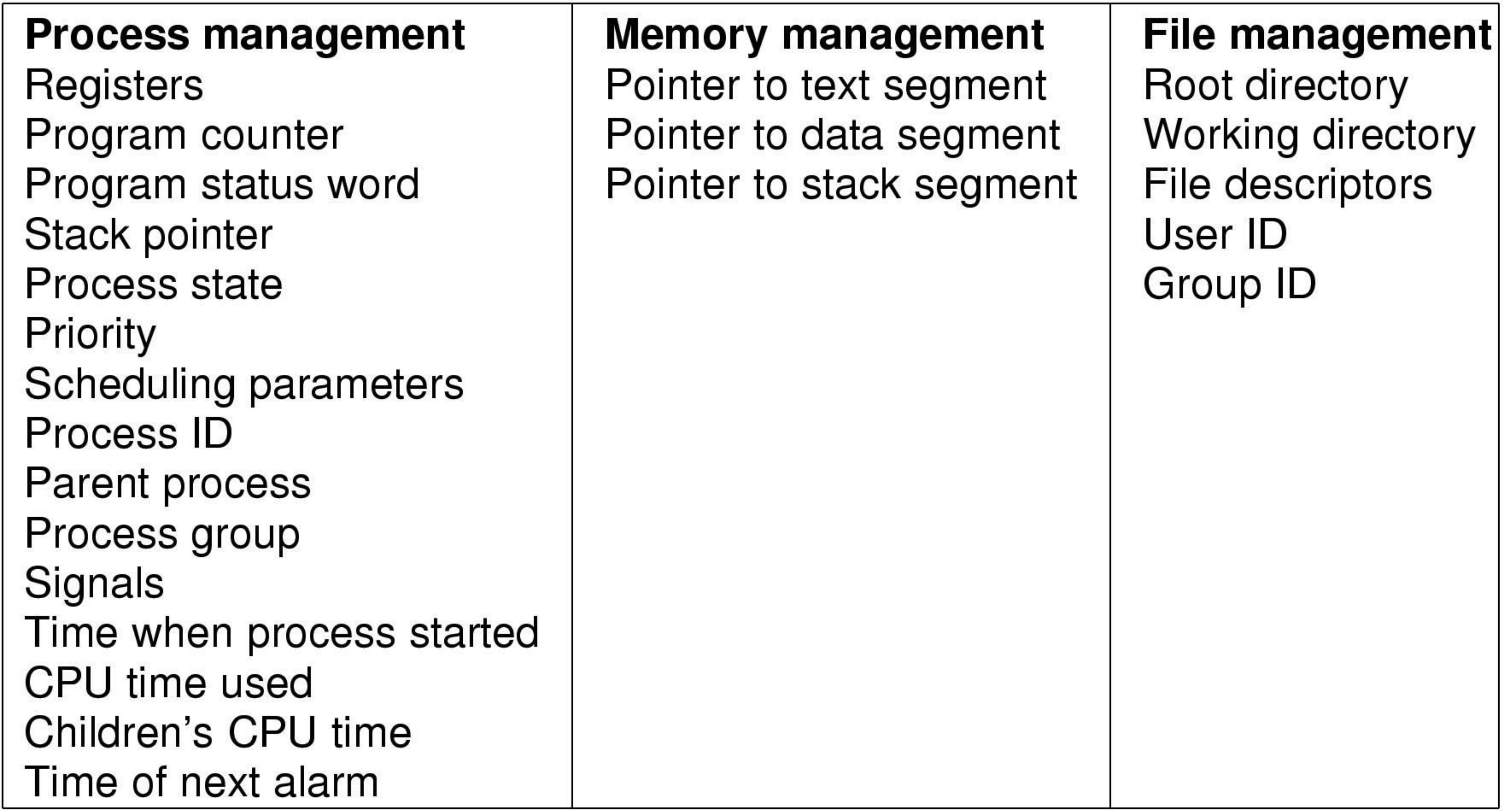
memory limits

list of open files

• • •

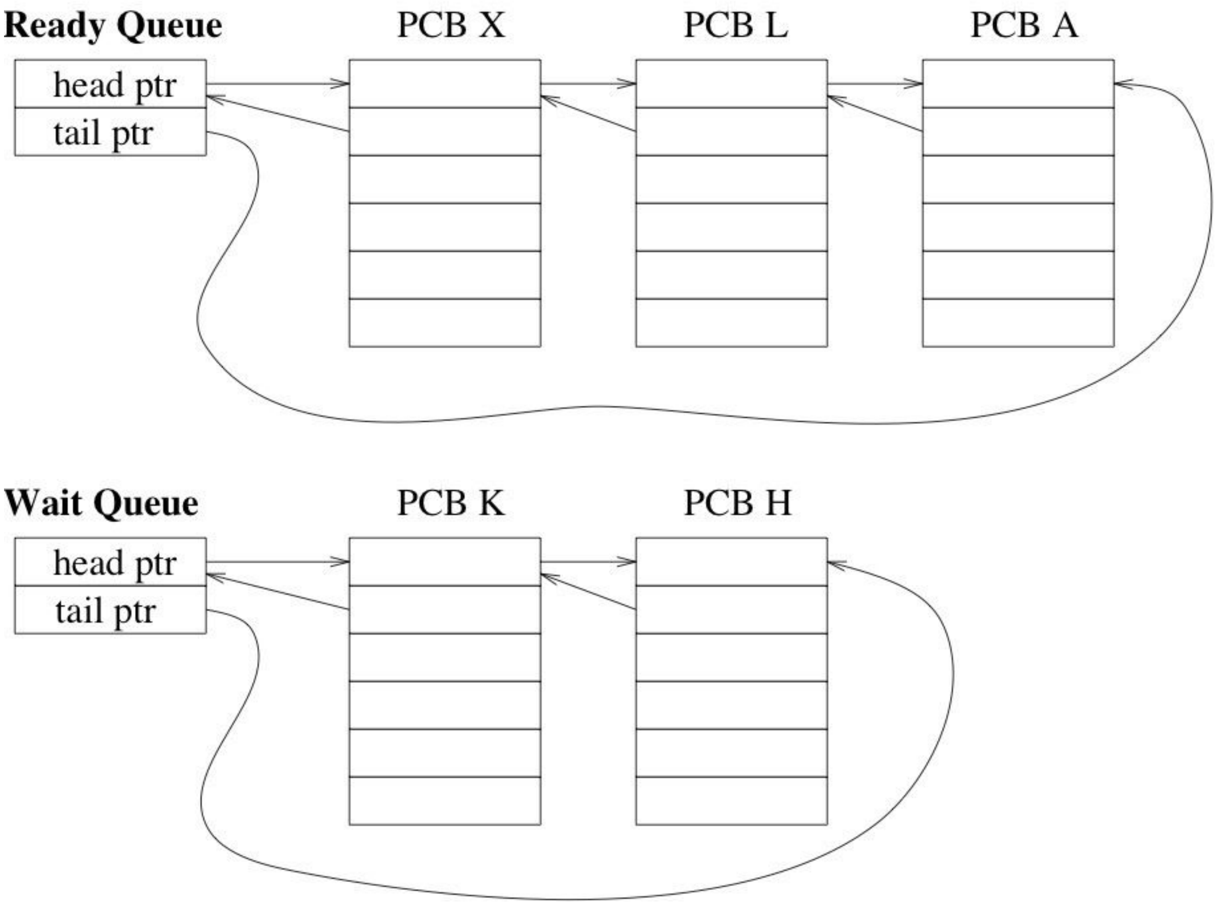
Book: The Operating System Concepts: Silberschatz, Galvin, Gagne

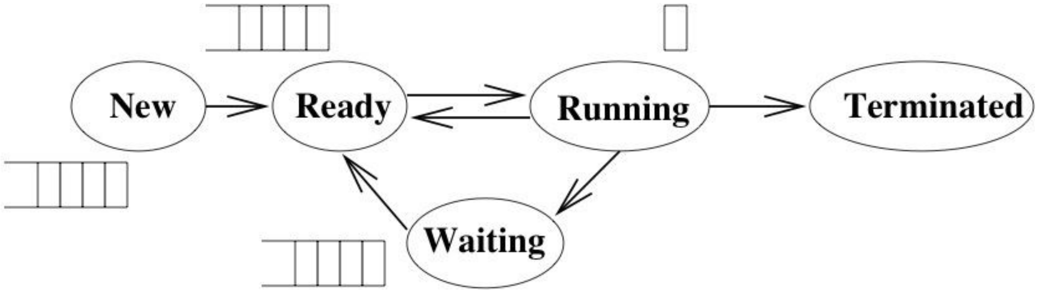
PCB Fields



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		

State Queues Examples



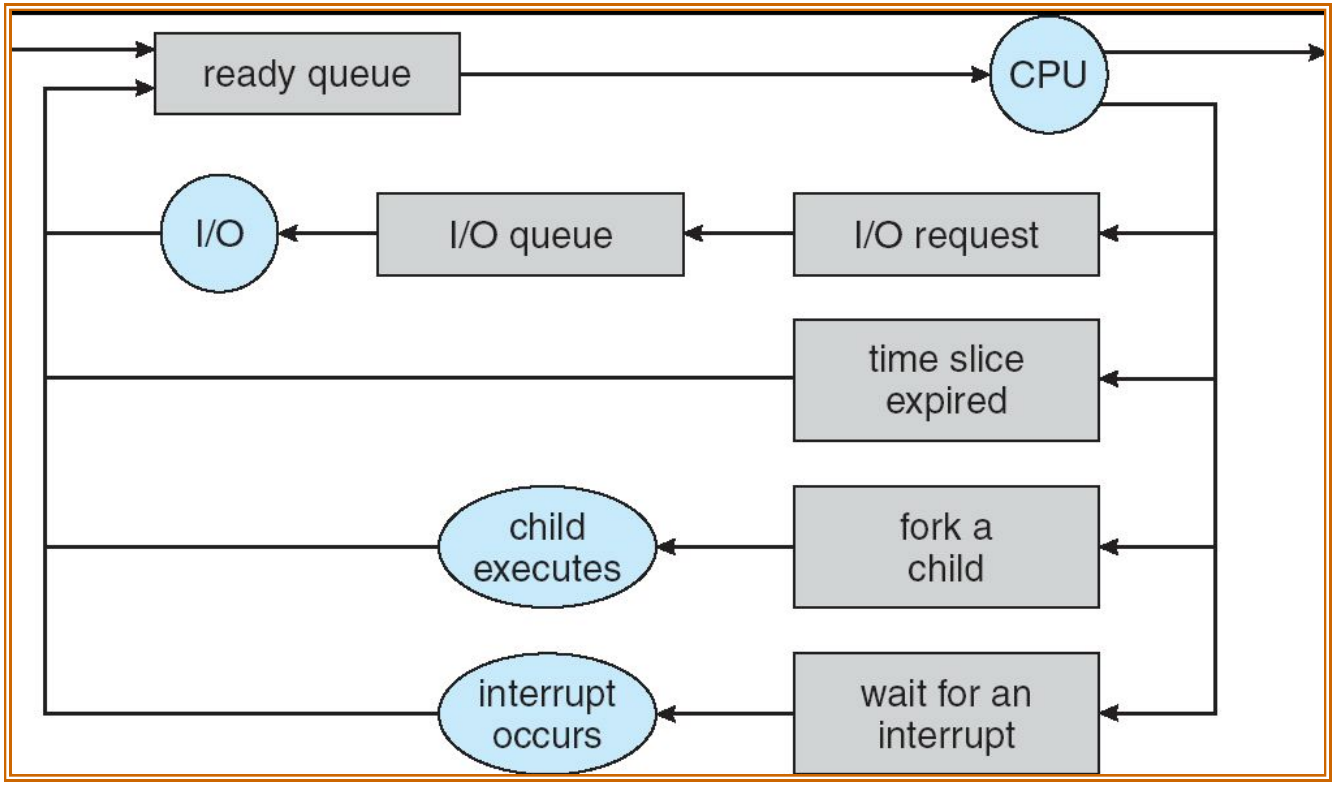


PCB and Context Switching

- The process of switching the CPU from one process to another (stopping one and starting the next) is the context switch
- The current state of process held in a process control block (PCB):
  - This is a “snapshot” of the execution and protection environment
  - Only one PCB active at a time
- One one process is ”running” at a time
- The OS starts executing a ready process by loading hardware registers (PC, SP, etc) from its PCB
- While a process is running, the CPU modifies the Program, Counter (PC), Stack Pointer (SP), registers, etc.
- When the OS stops a process, it saves the current values of the registers, (PC, SP, etc.) into its PCB
- Time sharing systems may do 100 to 1000 context switches a second.

Process Scheduling

Process Scheduling

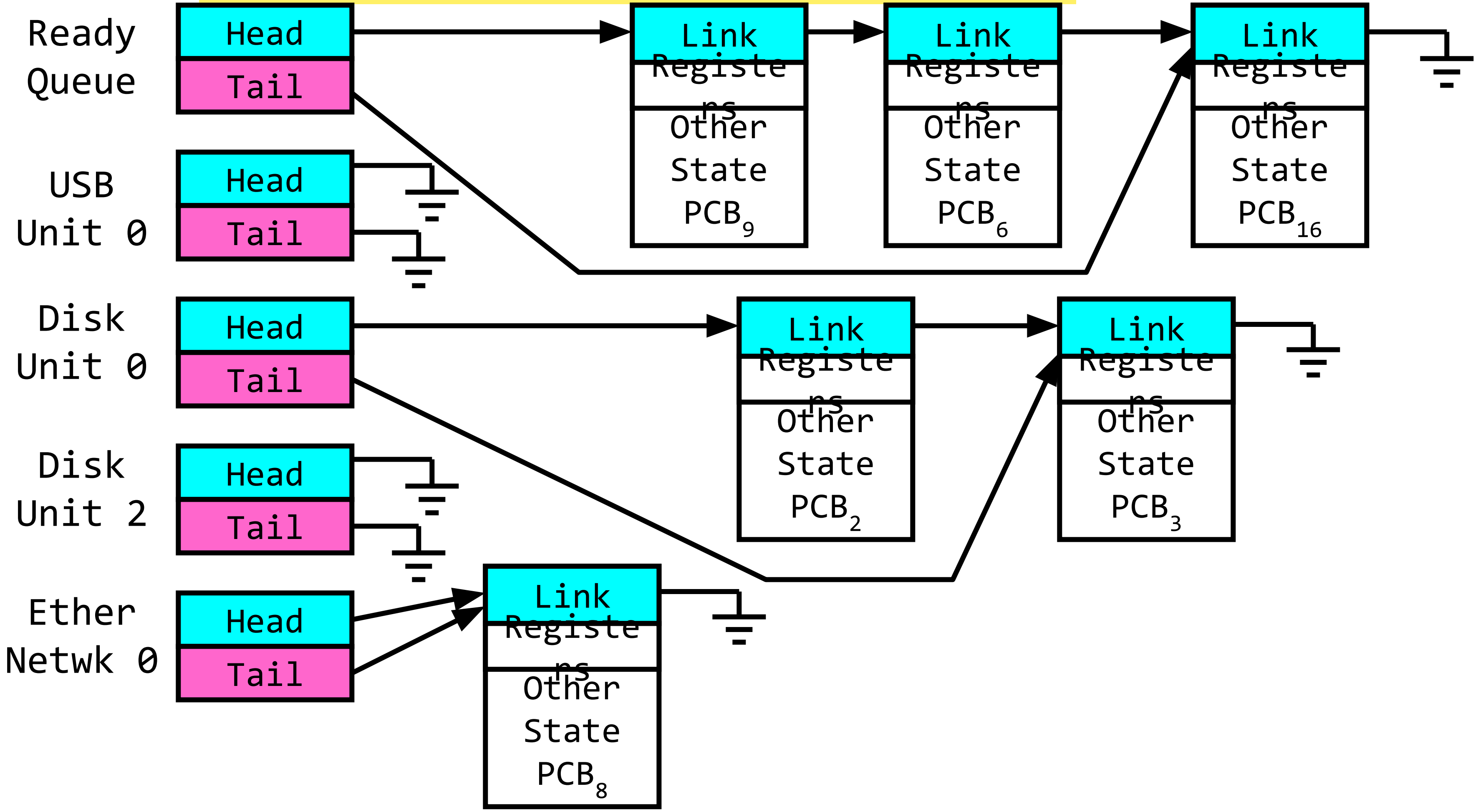


- PCBs move from queue to queue as they change state
  - Decisions about which order to remove from queues are Scheduling decisions
  - Many scheduling algorithms possible (will study a few algos)

Book: The Operating System Concepts: Silberschatz, Galvin, Gagne

Ready Queue And Various I/O Device Queues

- Process not running ⇒ PCB is in some scheduler queue
  - Separate queue for each device, condition (ready, wait etc)
  - Each queue can have a different scheduler policy

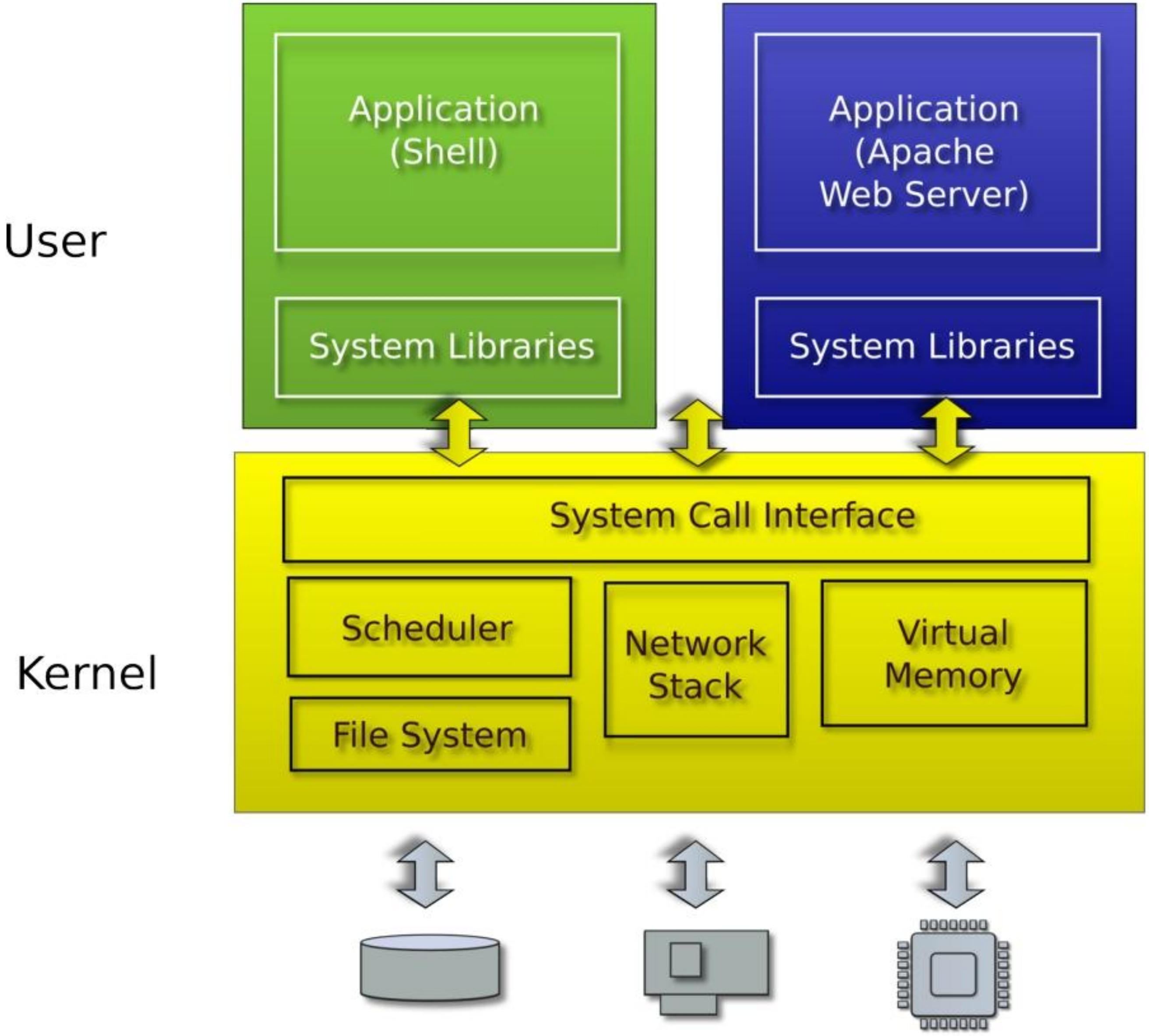


Process APIs and System Calls - Introduction

Process APIs

- What interface should the OS present for process creation and control?
- How should these interfaces be designed to enable?
  - Powerful functionality
  - Ease of use
  - High Performance

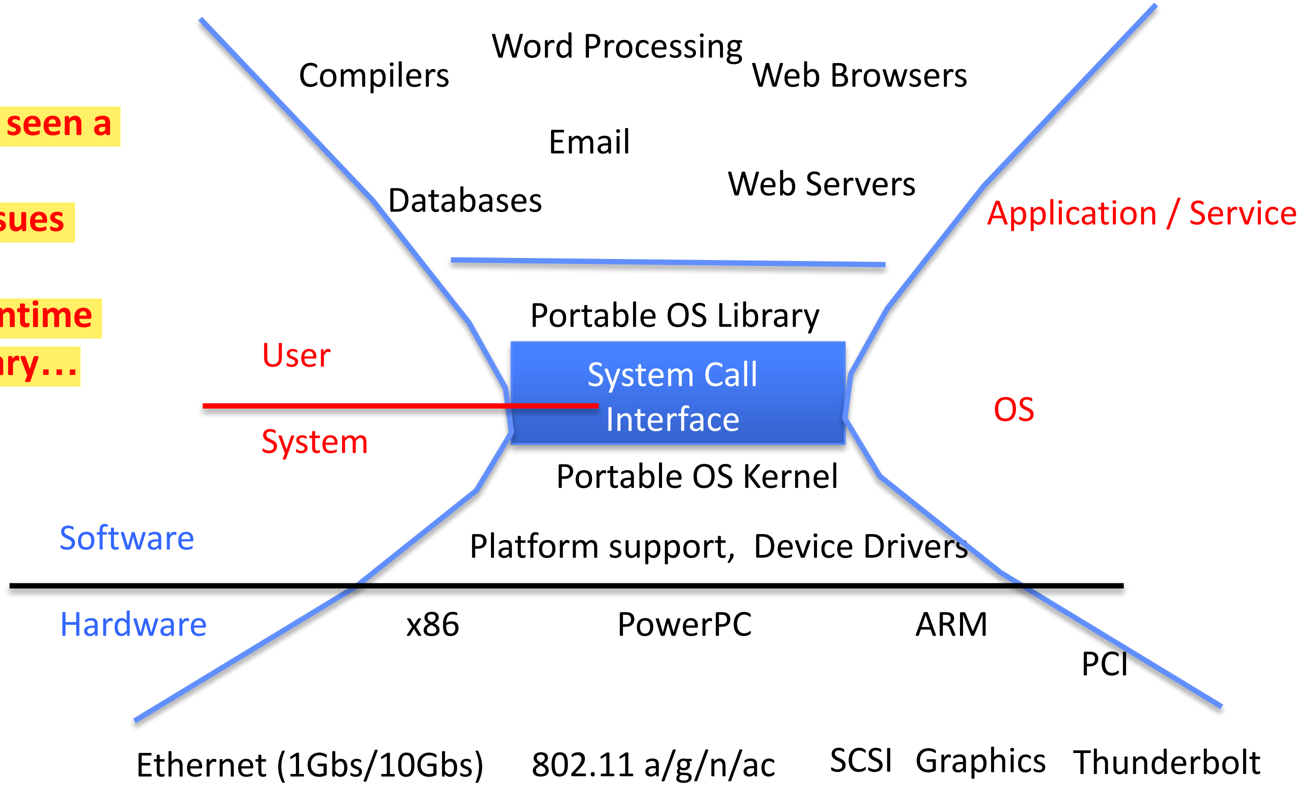
Typical Unix OS



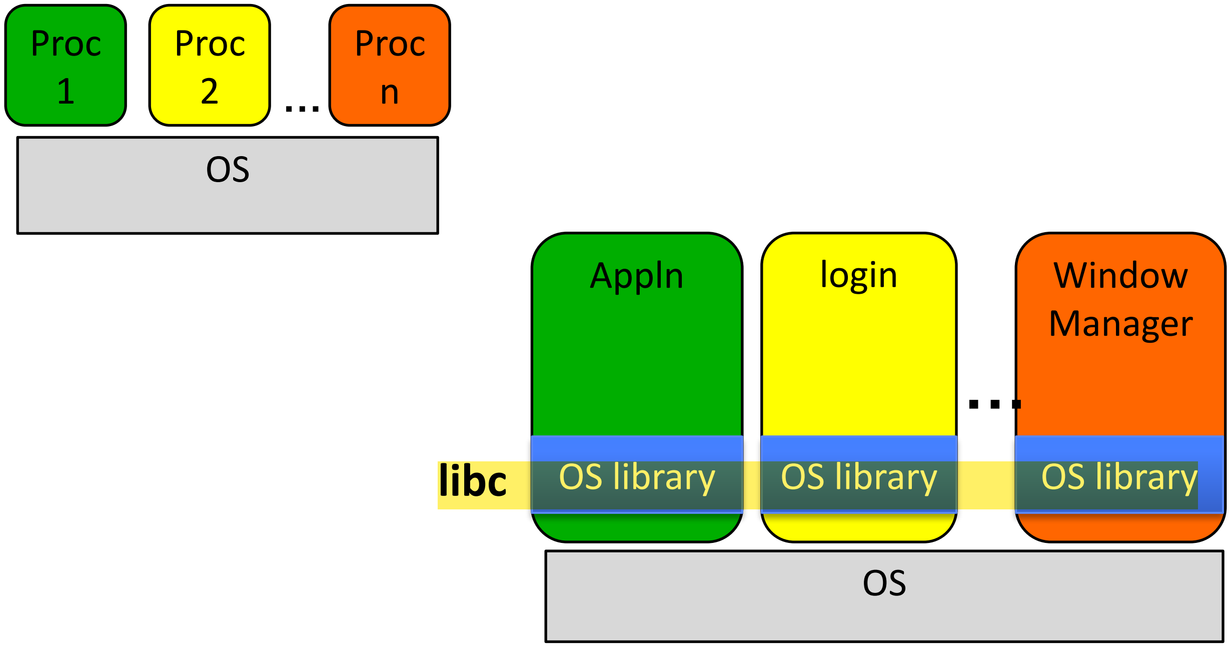
System Calls ("Syscalls")

"But, I've never seen a syscall!"

- OS library issues system call
- Language runtime uses OS library…

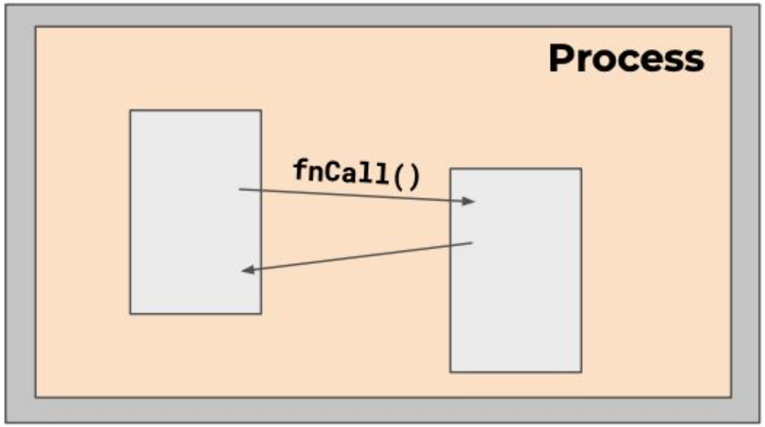


OS Library Issues Syscalls

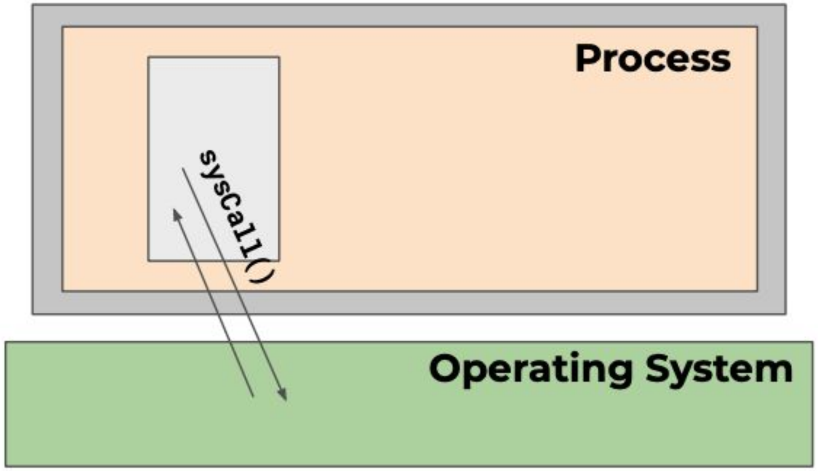


Function Calls vs System Calls

Function Call:



System Call



- Caller and callee in the same process
  - Same user
  - Same “domain of trust”

- OS is trusted; User process is not
- OS code runs privileged with complete access to all system resources

System Calls

- API : Application Programming Interface
  - Function available to write user programs
- System calls are the interface of the OS for
  - Processes
    - Creating, existing, waiting, and terminating
  - Memory
    - Allocation and deallocation
  - Files and Folders
    - Opening, reading, writing, closing
  - Inter Process Communication

- We will study process system calls in later lectures

Lecture Summary

- Program in execution is a process
- A process in execution moves from one state to another
  - Running, Ready, Waiting states
- An important OS data-structure for multiprogramming/context switching is Process Control Block (PCB)
  - Each process has a PCB which contains all information about the process
- There are many queues of PCBs which OS Kernel uses for scheduling
  - Ready Queue
  - Device Queue
  - Event Queue ..