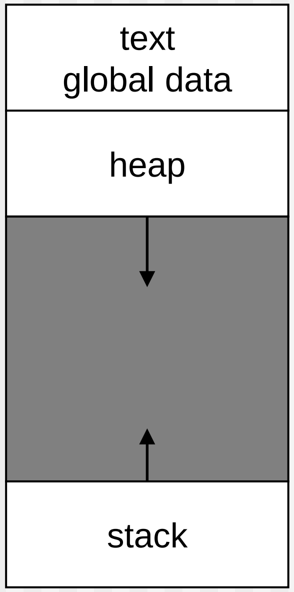
Lab : Process Management

What is a process?

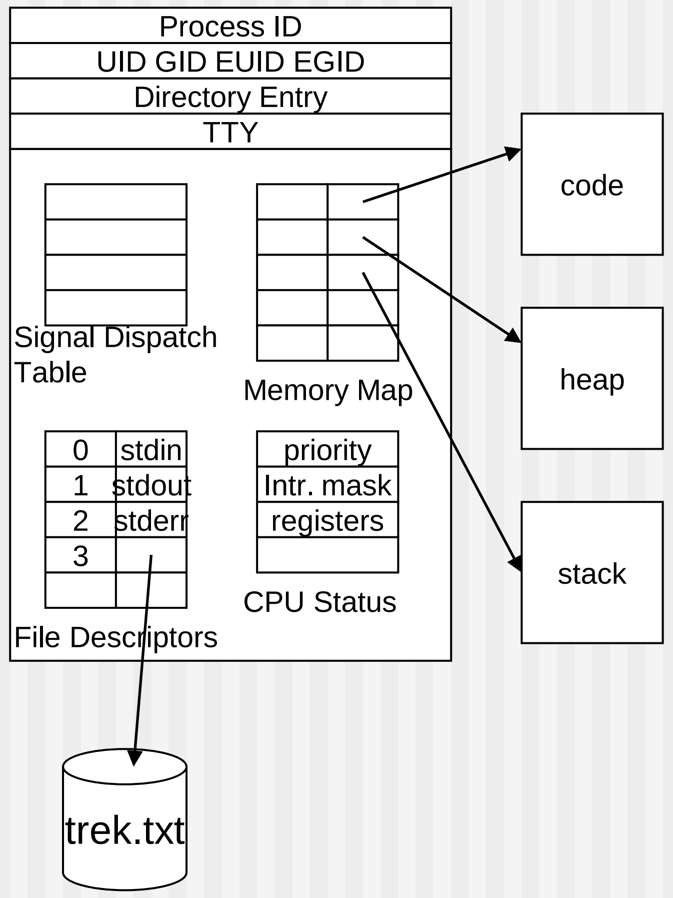
- A process is an instance of a computer program being executed using code and instructions
- Each process uses system resources like CPU or RAM to complete the specific tasks

Process Concept

- Process – a program in execution; process execution must progress in sequential fashion.
- Textbook uses the terms *job* and *process* almost interchangeably.
- A process includes:
  - Program counter
  - Stack (local variables)
  - Data section (global data)
  - Text (code)
  - Heap (dynamic data)
  - Files (cin, cout, cerr, other file descriptors)

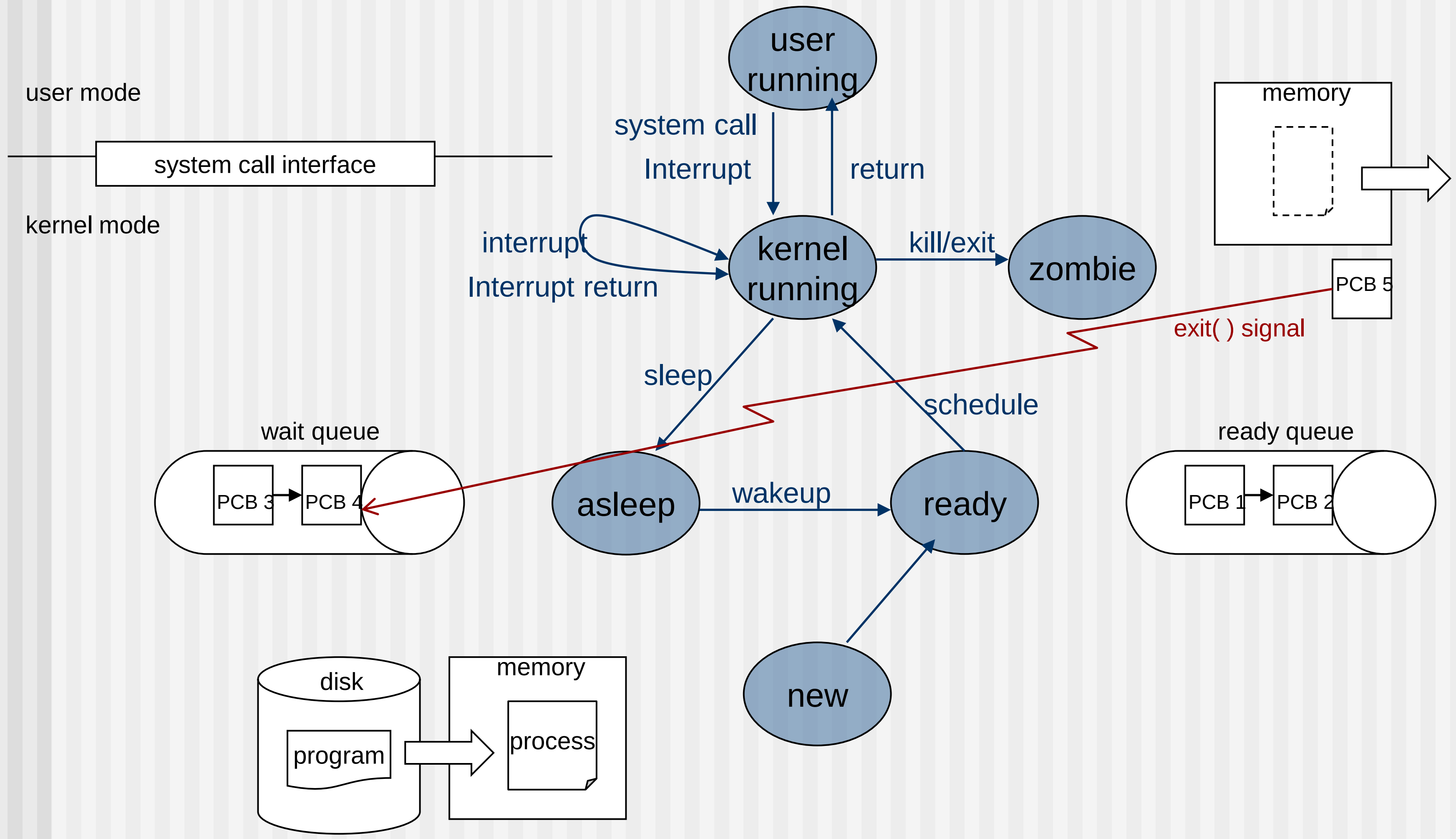


Process Control Block



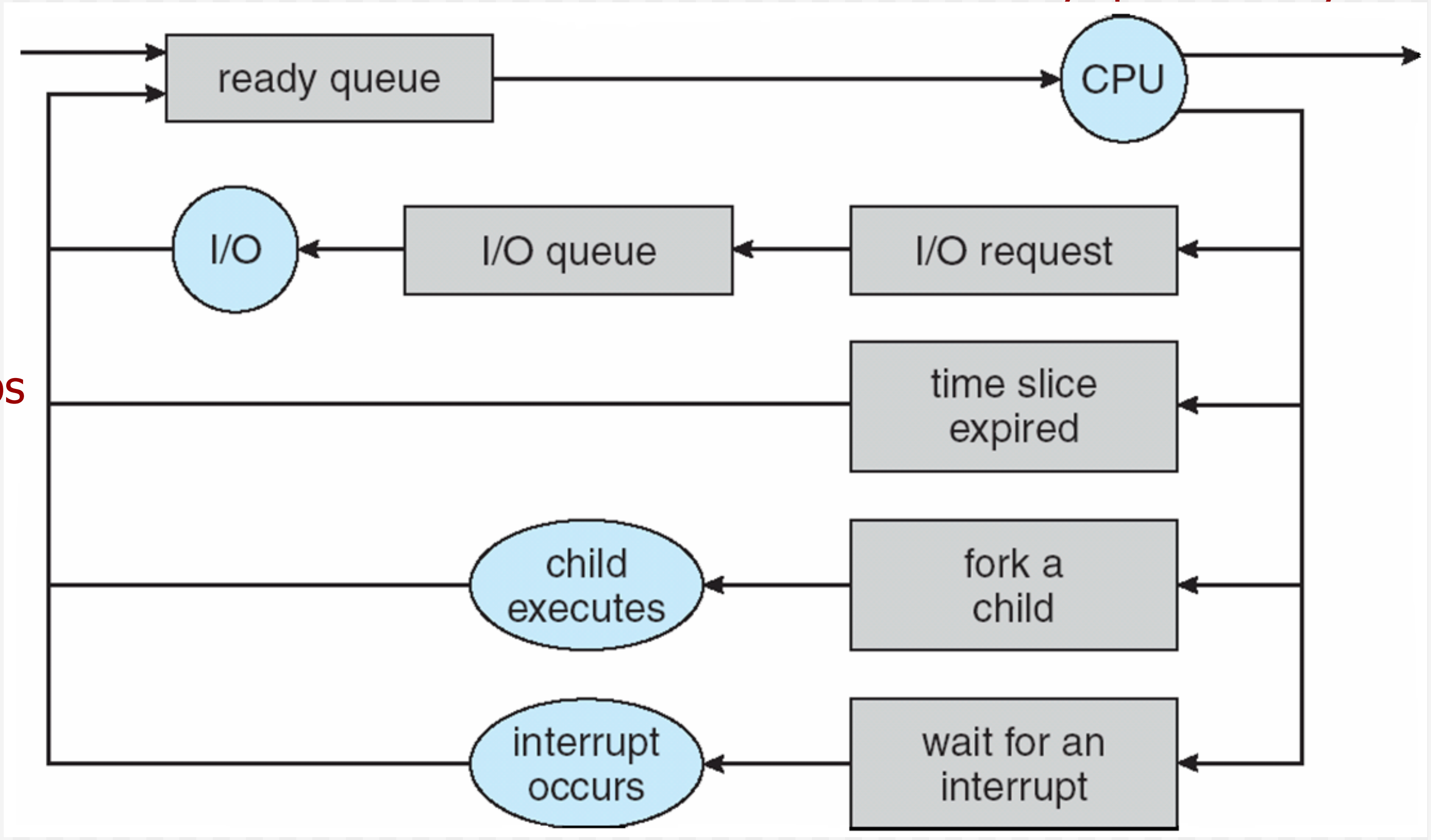
- Process ID
- CPU status
- Memory limits
- List of open files

Process Status



Process Scheduling

Short-term scheduler: picks up a process from ready queue every 100ms



Long-term scheduler: swaps I/O waiting processes in and out of memory

Process Creation

- Parent process creates children processes.
- Resource sharing
  - Resource inherited by children: file descriptors, shared memory and system queues
  - Resource not inherited by children: address space
- Execution
  - Parent and children execute concurrently.
  - Parent waits by **wait** system call until children terminate.
- UNIX examples
  - **fork** system call creates new process.
  - **execlp** system call used after a **fork** to replace the process' memory space with a new program.

Backgrounrd & Foreground Processes

- A foreground process is any process which is not continuously running and it waiting on something like user input
- A background process is something that is continually running and does not require any additional input
- Can someone name examples of each?

Moving a Process to the Background

- When executing commands on the command line, there is usually some output that is displayed on the terminal
- If you move a process to the background, the output will not be shown

Background Process Example

- Usually, when you download a file from the command line, the status is displayed on the terminal
- To move a process to the background all you have to do is add an ampersand (&) at the end of the command
- **Wget http://releases.ubuntu.com/18.04.2/ubuntu-18.04.2-desktop-amd64.iso_ga=2.142658160.410030815.1551071806-1676866732.1550780350 &**
- Now this will be moved to the background

Moving back to the Foreground

- To move a process back to the foreground, use the following steps:
- Use the **jobs** command to identify the job number of the background process
- Then use the **fg** command to bring it back with the following syntax
- **fg [job number]**

How do Processes Actually Work?

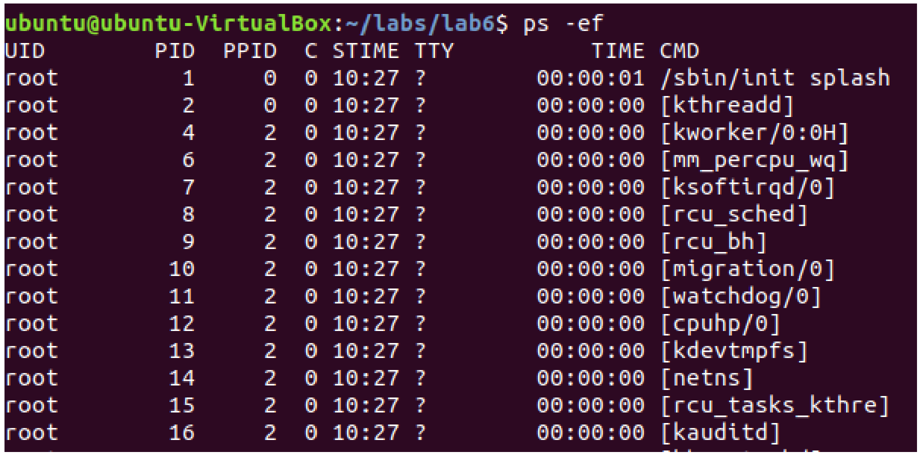
- In the Unix operating environment, processes are created by a method called “forking”
- Forking is when the OS duplicated a process
- The original process is called the parent process
- And the new process is the child process

Different Types of Processes

- There are four types of processes:
  - Running: current process that is being executed in the operating system
  - Waiting: process which is waiting for system resources to run
  - Stopped: process that is not running
  - Zombie: process whose parent processes has ended, but the child process is still in the process table

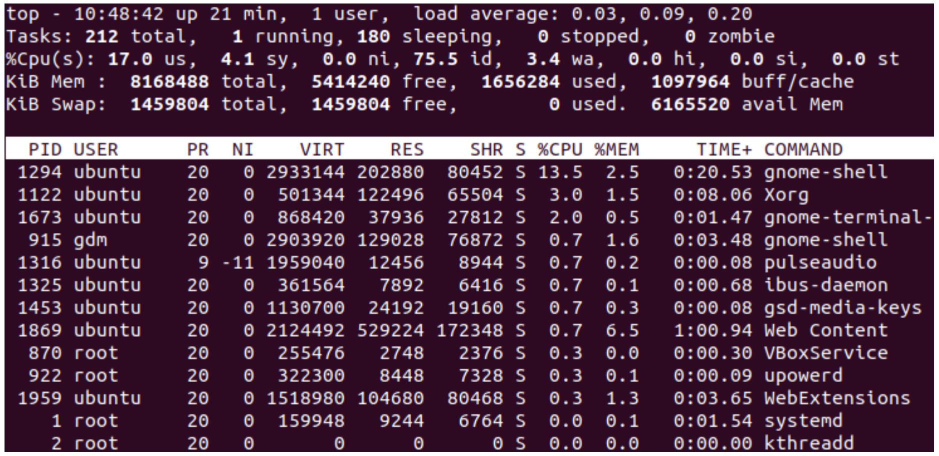
Viewing Processes

- Two commands you can use to view the process from the command line: **ps** and **top**
- To view all the processes with **ps,** use **ps -ef**



```
ubuntu@ubuntu-VirtualBox:~/labs/lab6$ ps -ef
UID          PID   PPID  C STIME TTY          TIME CMD
root           1      0  0 10:27 ?        00:00:01 /sbin/init splash
root           2      0  0 10:27 ?        00:00:00 [kthreadd]
root           4      2  0 10:27 ?        00:00:00 [kworker/0:0H]
root           6      2  0 10:27 ?        00:00:00 [mm_percpu_wq]
root           7      2  0 10:27 ?        00:00:00 [ksoftirqd/0]
root           8      2  0 10:27 ?        00:00:00 [rcu_sched]
root           9      2  0 10:27 ?        00:00:00 [rcu_bh]
root          10      2  0 10:27 ?        00:00:00 [migration/0]
root          11      2  0 10:27 ?        00:00:00 [watchdog/0]
root          12      2  0 10:27 ?        00:00:00 [cpuhp/0]
root          13      2  0 10:27 ?        00:00:00 [kdevtmpfs]
root          14      2  0 10:27 ?        00:00:00 [netns]
root          15      2  0 10:27 ?        00:00:00 [rcu_tasks_kthre]
root          16      2  0 10:27 ?        00:00:00 [kauditd]
```

ps -ef



```
top - 10:48:42 up 21 min,  1 user,  load average: 0.03, 0.09, 0.20
Tasks: 212 total,   1 running, 180 sleeping,   0 stopped,   0 zombie
%Cpu(s): 17.0 us,  4.1 sy,  0.0 ni, 75.5 id,  3.4 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  8168488 total,  5414240 free,  1656284 used,  1097964 buff/cache
KiB Swap:  1459804 total,  1459804 free,        0 used.  6165520 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1294 ubuntu    20   0 2933144 202880  80452 S 13.5  2.5   0:20.53 gnome-shell
 1122 ubuntu    20   0  501344 122496  65504 S  3.0  1.5   0:08.06 Xorg
 1673 ubuntu    20   0  868420  37936  27812 S  2.0  0.5   0:01.47 gnome-terminal-
  915 gdm       20   0 2903920 129028  76872 S  0.7  1.6   0:03.48 gnome-shell
 1316 ubuntu     9 -11 1959040  12456   8944 S  0.7  0.2   0:00.08 pulseaudio
 1325 ubuntu    20   0  361564   7892   6416 S  0.7  0.1   0:00.68 ibus-daemon
 1453 ubuntu    20   0 1130700  24192  19160 S  0.7  0.3   0:00.08 gsd-media-keys
 1869 ubuntu    20   0 2124492 529224 172348 S  0.7  6.5   1:00.94 Web Content
  870 root      20   0  255476   2748   2376 S  0.3  0.0   0:00.30 VBoxService
  922 root      20   0  322300   8448   7328 S  0.3  0.1   0:00.09 upowerd
 1959 ubuntu    20   0 1518980 104680  80468 S  0.3  1.3   0:03.65 WebExtensions
    1 root      20   0  159948   9244   6764 S  0.0  0.1   0:01.54 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
```

top

Ending a Process In Linux

- Sometimes you need to end a program or process from the command line. Use the following steps:

  1. Locate the process id [PID] of the process/program you want to kill
  2. Use the **kill** command with the following syntax: **kill [PID]**
  3. If the process is still running, do the following: **kill -9 [PID]**
  4. The -9 is a SIGKILL signal telling the process to terminate immediately

Ending All Process

- You can use the **killall** command to kill multiple processes at the same time
- Syntax: **killall [options] PIDs**
- Or you can use **pkill –u [username]** to kill all processes started by [username]