

CS169: OS Lab

Course Overview

Intro to Weenix and Weenix Kernel

“So. Threads in Weenix aren't *that* important,
right?”

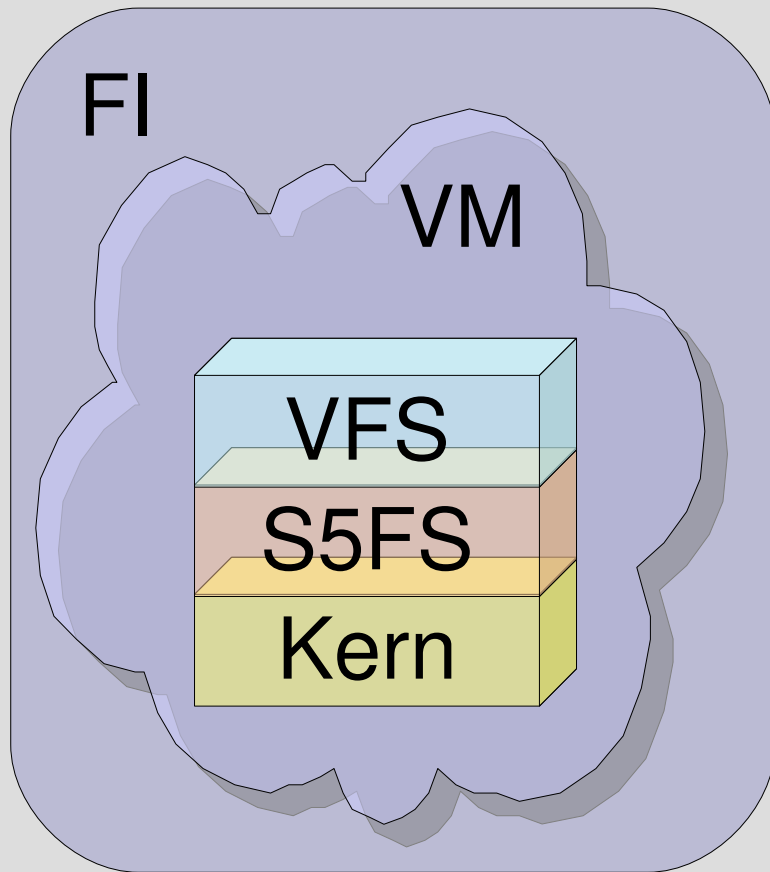
- Owen (ostrain)

The CS169 TAs

CS169: OS Lab

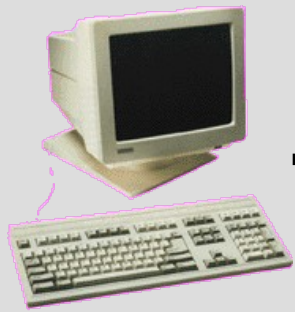
- Write an actual operating system (Weenix)
- Much time required, but very rewarding
- Can switch back to CS167
- Four projects:
 - Kern (parts 1 and 2)
 - Virtual File System
 - S5 File System
 - Virtual Memory/Final integration
- At the end, you have a real operating system
 - can run “arbitrary” C programs

Weenix



- Kern: foundation (threads and drivers)
- VFS: abstract FS
- S5FS: simple FS
- VM: virtual memory
- FI: neaten up the edges (run userland programs)

“Machine” (Really Xen)



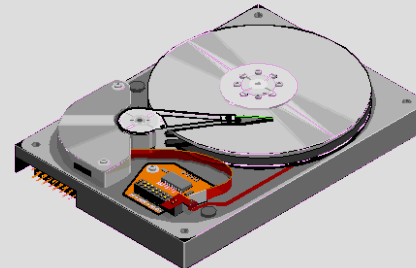
Terminal
(monitor and keyboard)



CPU



Memory



Disk

From Computer Desktop Encyclopedia
© 2005 The Computer Language Co. Inc.

Weenix



User Land!



VM

System Calls

open, close, read, write, lseek, getdents

fork, exec

yield

mmap

VFS

File system
(S5FS)

TTY driver

Line discipline

Virtual device
drivers

Process Management

Scheduler

Dispatcher

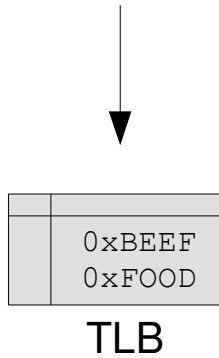
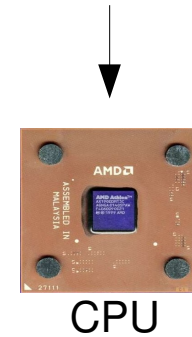
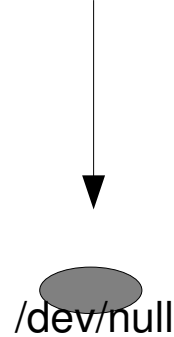
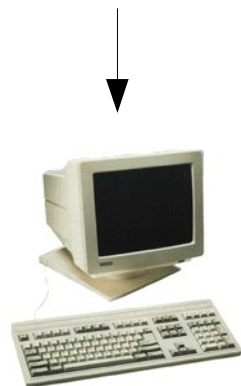
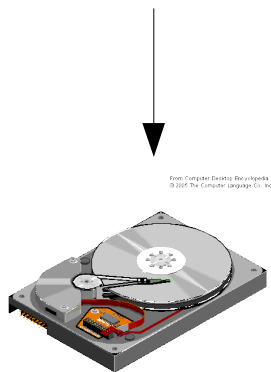
TLB mgmt

Paging

vmpage array

Disk driver

Terminal driver



Kern 1



User Land!



VM

System Calls

open, close, read, write, lseek, getdents

fork, exec

yield

mmap

VFS

File system
(S5FS)

TTY driver

Line discipline

Virtual device
drivers

Process Management

Scheduler

Dispatcher

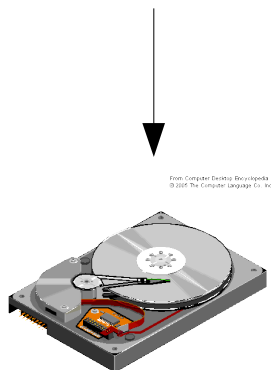
TLB mgmt

Paging

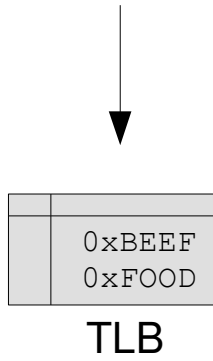
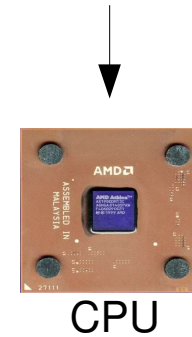
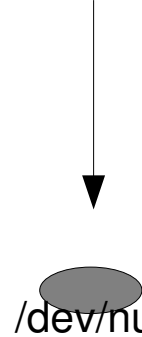
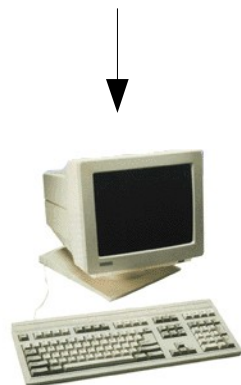
Core map

Disk driver

Terminal driver



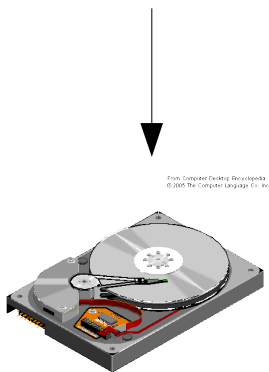
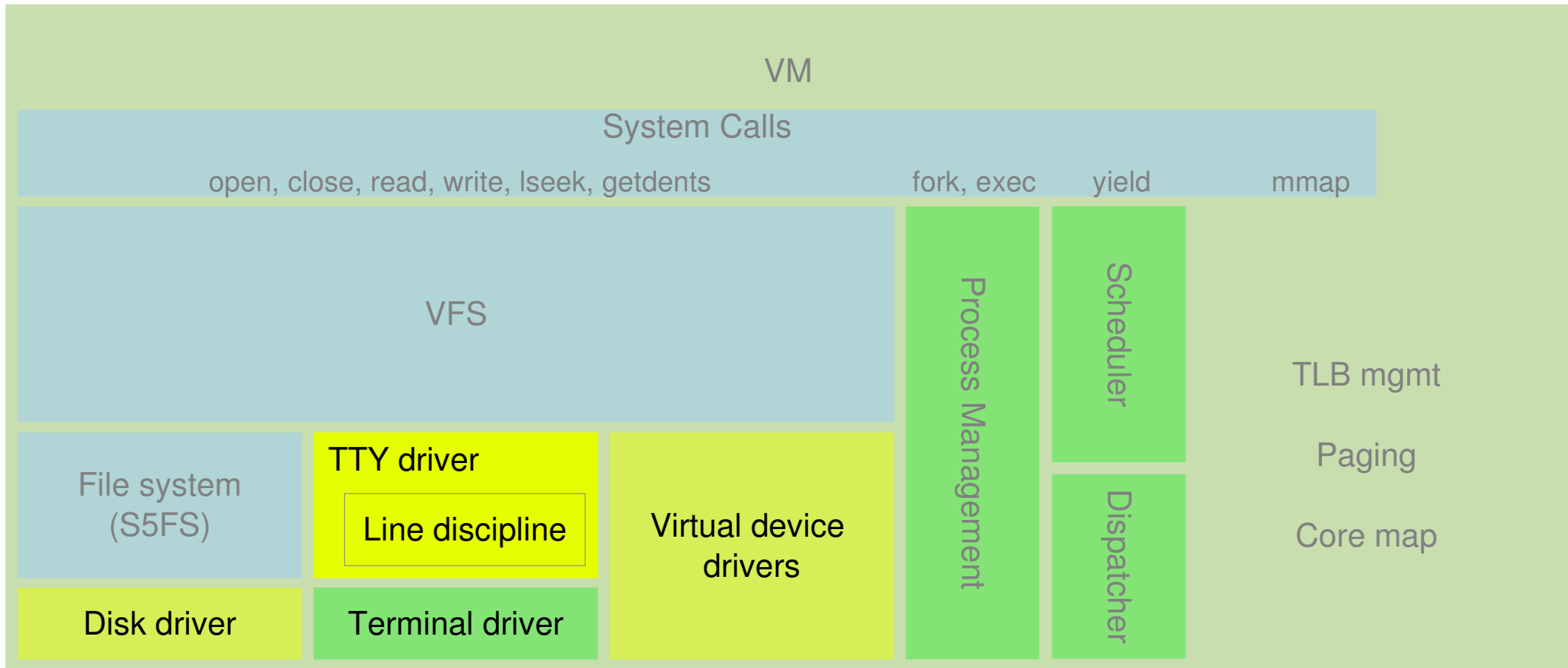
From Computer Exotics Backups
© 2004 The Computer Language Co., Inc.



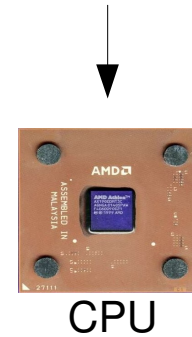
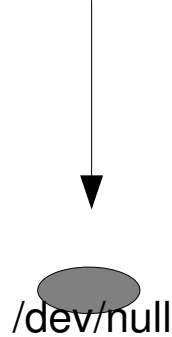
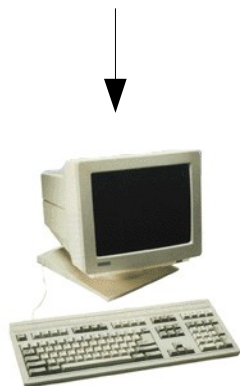
Kern 2



User Land!



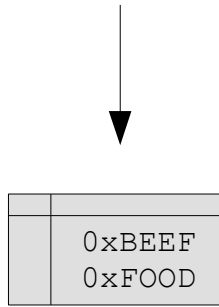
From Computer Devices: Beavertails
© 2000 The Computer Language Co., Inc.



CPU



Memory



0xBEEF
0xF00D

TLB

VFS



User Land!



VM

System Calls

open, close, read, write, lseek, getdents

fork, exec

yield

mmap

VFS

File system (S5FS)

File system (testfs)

TTY driver

Line discipline

Virtual device drivers

Process Management

Scheduler

Dispatcher

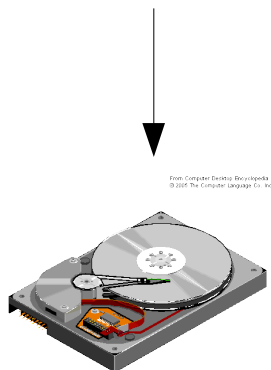
TLB mgmt

Paging

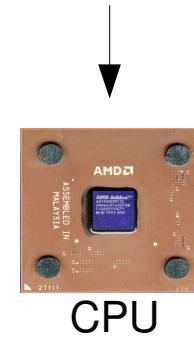
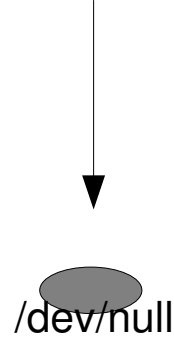
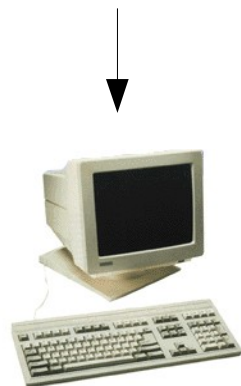
Core map

Disk driver

Terminal driver



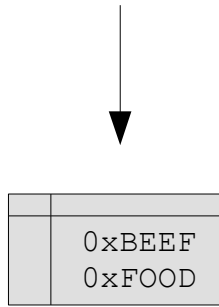
From Computer Exotics Backups
© 2006 The Computer Language Co., Inc.



CPU



Memory



0xBEEF
0xFOOD

TLB

S5FS



User Land!



VM

System Calls

open, close, read, write, lseek, getdents

fork, exec

yield

mmap

VFS

File system
(S5FS)

TTY driver

Line discipline

Virtual device
drivers

Process Management

Scheduler

Dispatcher

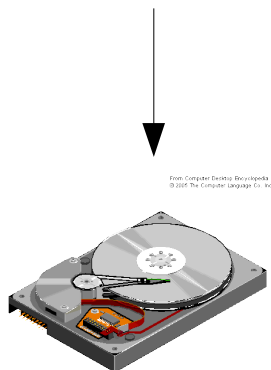
TLB mgmt

Paging

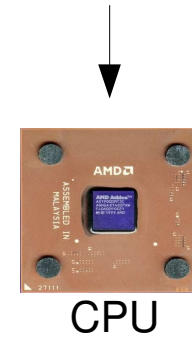
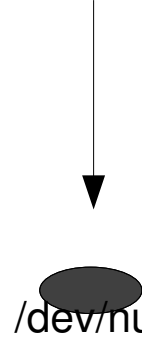
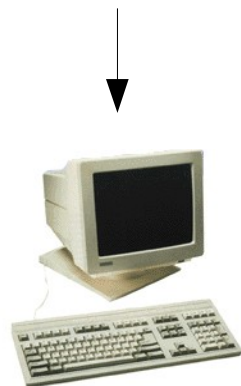
Core map

Disk driver

Terminal driver



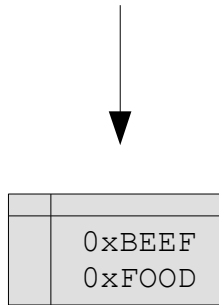
From Computer Exotics Backups
© 2006 The Computer Language Co., Inc.



CPU



Memory



0xBEEF
0xFOOD

TLB

VM/FI



User Land!



VM

System Calls

open, close, read, write, lseek, getdents

fork, exec

yield

mmap

VFS

Process Management

Scheduler

Dispatcher

TLB mgmt

Paging

Core map

File system (S5FS)

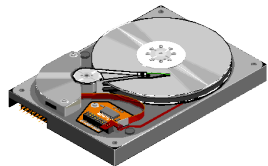
TTY driver

Line discipline

Virtual device drivers

Disk driver

Terminal driver



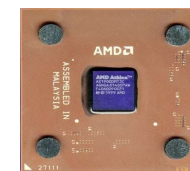
From Computer Exotics Backups
© 2004 The Computer Language Co., Inc.



/dev/null



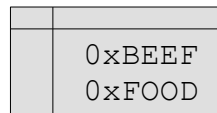
/dev/zero



CPU



Memory



TLB

Kernel

- Kern 1: threads
 - threads and processes
 - mutexes
 - “scheduler”
 - switch()
- Kern 2: device drivers
 - tty and line discipline
 - hard disk
 - virtual memory devices (/dev/zero, /dev/null)

Kern 1: Threads and Procs

- Read the Hackers guide
- Read the comments
- Good luck

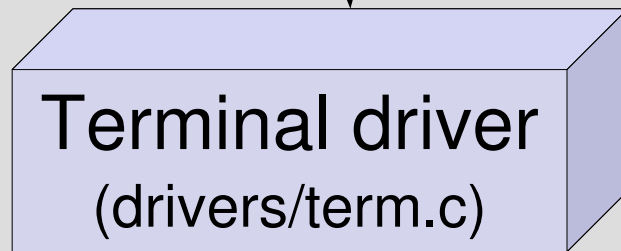
Kern 2: Terminal driver

- A **character** device
 - Receive characters from the keyboard
 - Echo characters back to the terminal
 - Write characters to the terminal
- We give you the actual terminal driver.
- You write the **line discipline**:
 - higher-level abstraction (uses the driver)
 - provides line-oriented semantics for **read**
 - handles backspace, simple editing, etc.
- You write the **tty driver**.

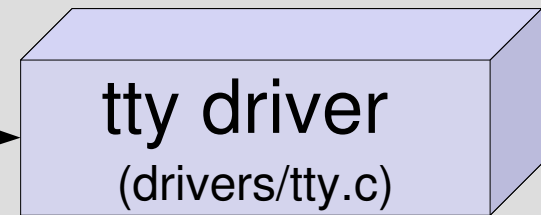
Terminal: entering characters



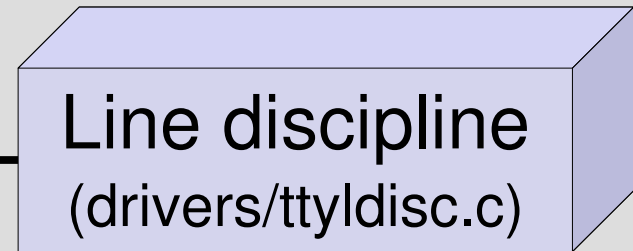
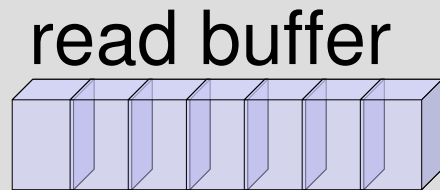
hardware
interrupt



tty_receive_char



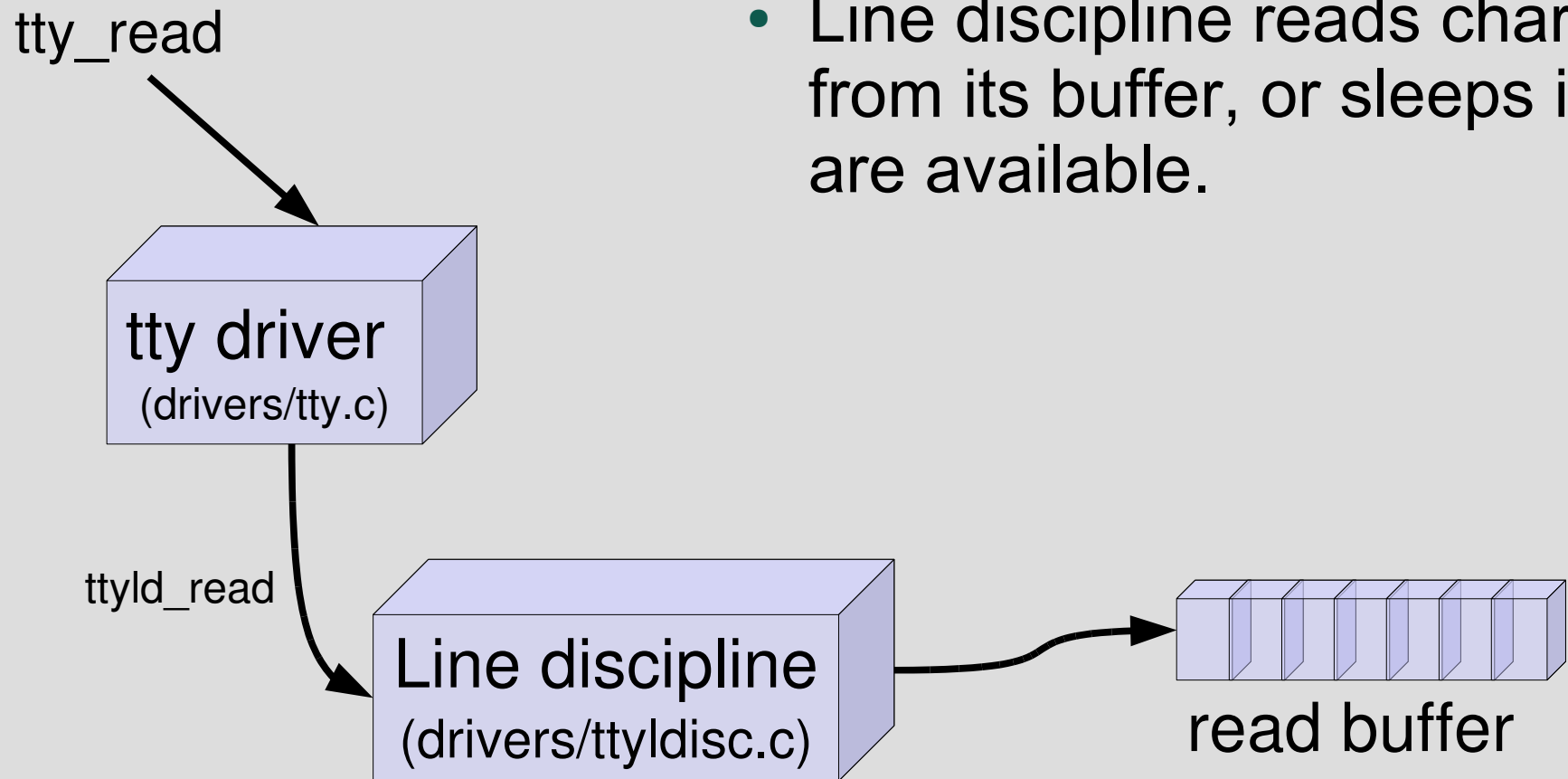
ttyld_receive_char



- Either the line discipline or the terminal driver can be replaced (since they don't deal with each other directly)

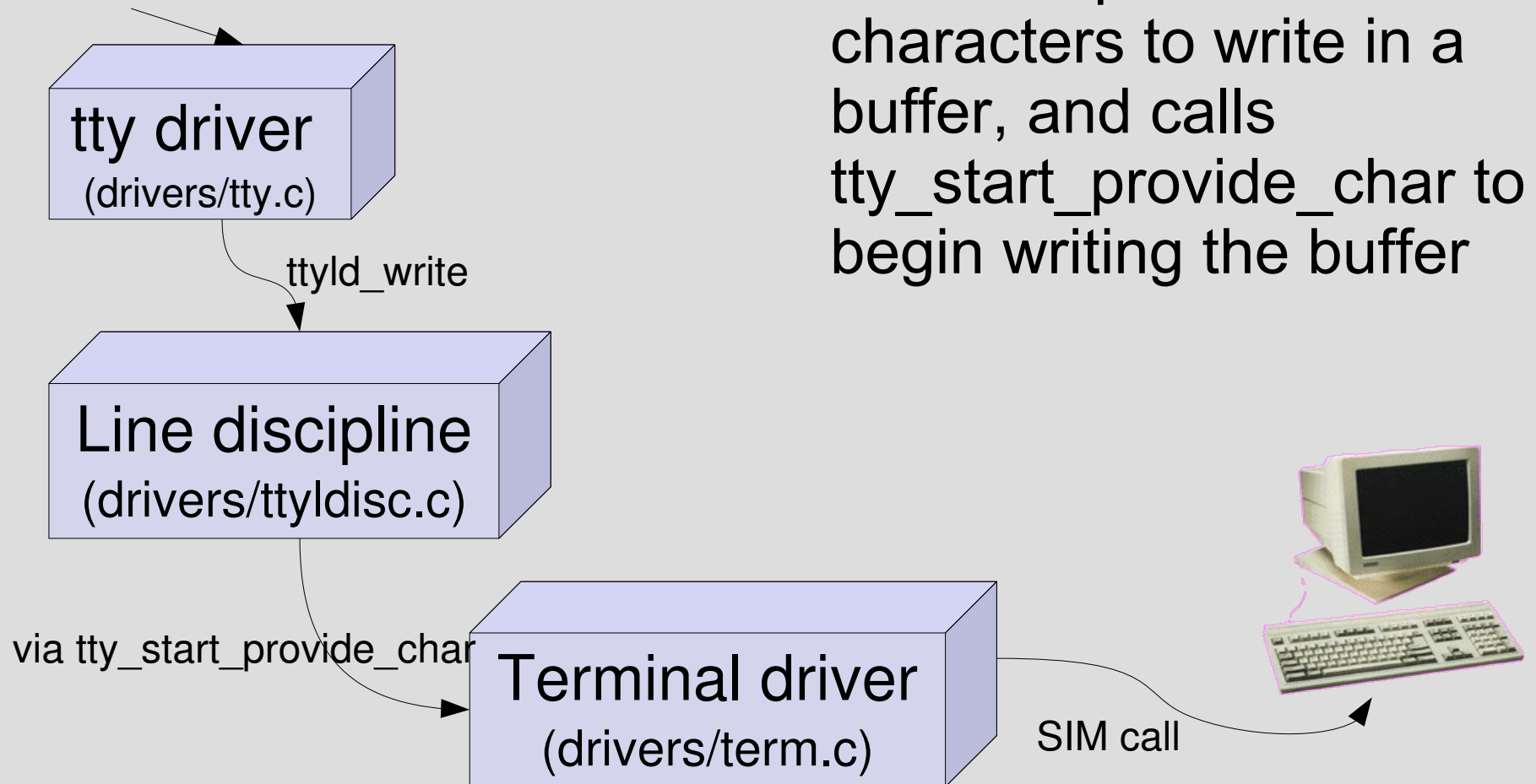
Terminal: read

- Line discipline reads characters from its buffer, or sleeps if none are available.



Terminal: write

tty_write



- Line discipline stores the characters to write in a buffer, and calls `tty_start_provide_char` to begin writing the buffer

Disk driver and virtual devices

- A **block** device supporting *seek*, *read*, *write*
- I/O uses device registers (DMA)
 - See CS167 slides and Simulator manual
- Virtual (memory) devices are mostly trivial:
 - **/dev/zero**: data source
 - read-only
 - always reads as many NULL bytes as requested
 - **/dev/null**: data sink
 - write-only
 - always accepts all data and does nothing

Getting started

- To do:
 - Email cs169tas with mentor TA preferences
- Remember:
 - Once assigned, email mentor TA with questions
 - Think before you code
 - **Start early** on this and all projects
 - **Test thoroughly** from the start
 - Check out the Wiki!!!