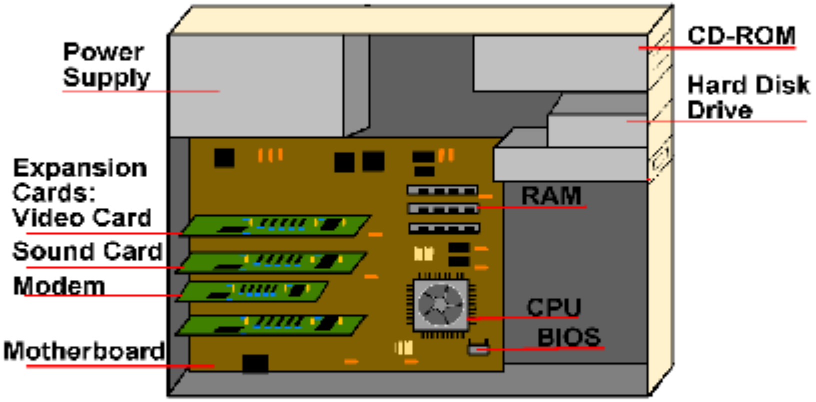
Operating Systems and Hardware Overview

Portions of this material from U. Manitoba

Components of a Computer System

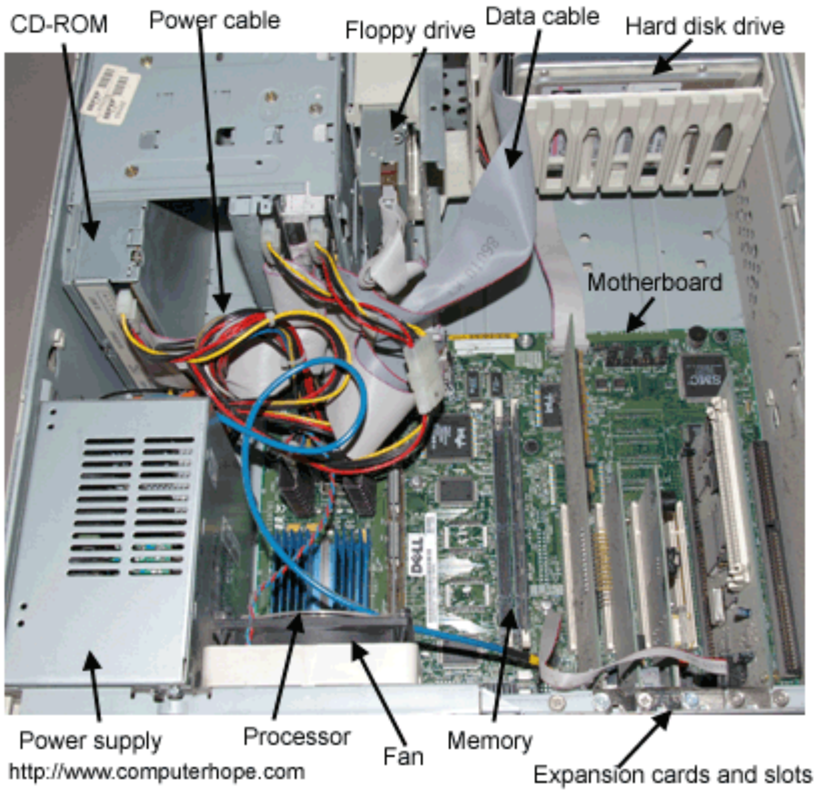
- A computer system consists of both hardware and software
- Many different types:
  - Tablets, laptops
  - Desktops, workstations
  - Embedded systems
  - Servers
  - Mainframes
  - Super Computers

Inside a Computer Case



Components Inside a Computer
(Wires not shown)

Inside of a computer case



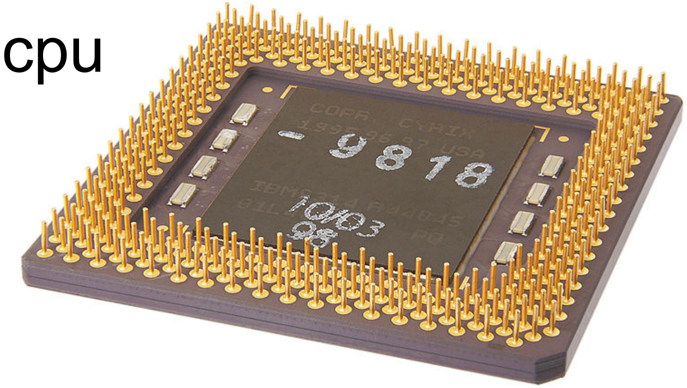
http://www.computerhope.com

Hardware Components

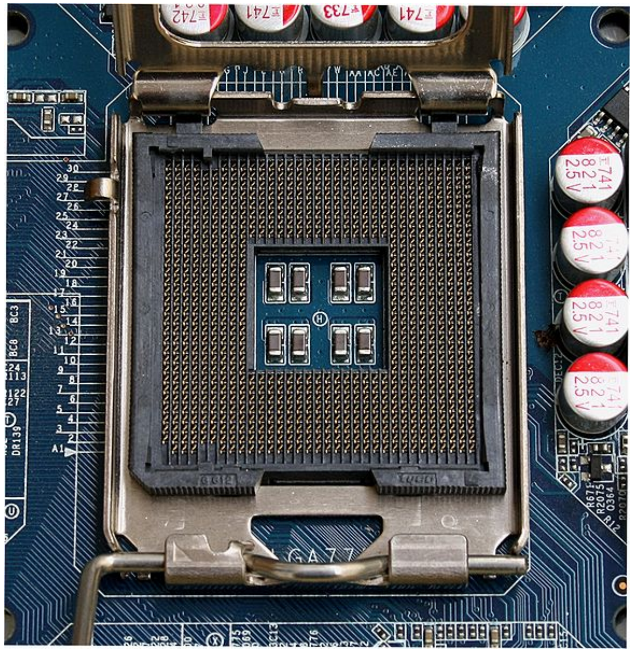
- Processor
- Main memory
- Secondary memory
  - hard disk, optical disks, tapes, …
- Input devices
  - keyboard, mouse, microphone, …
- Output devices
  - monitor, printer, speaker, …

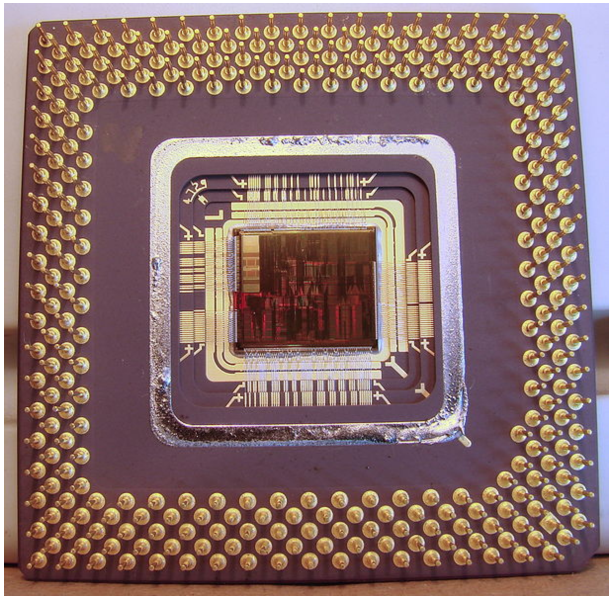
Processors

cpu

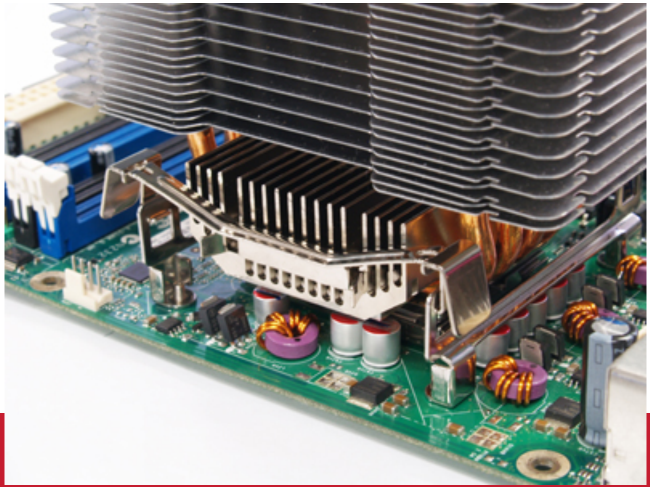


cpu socket





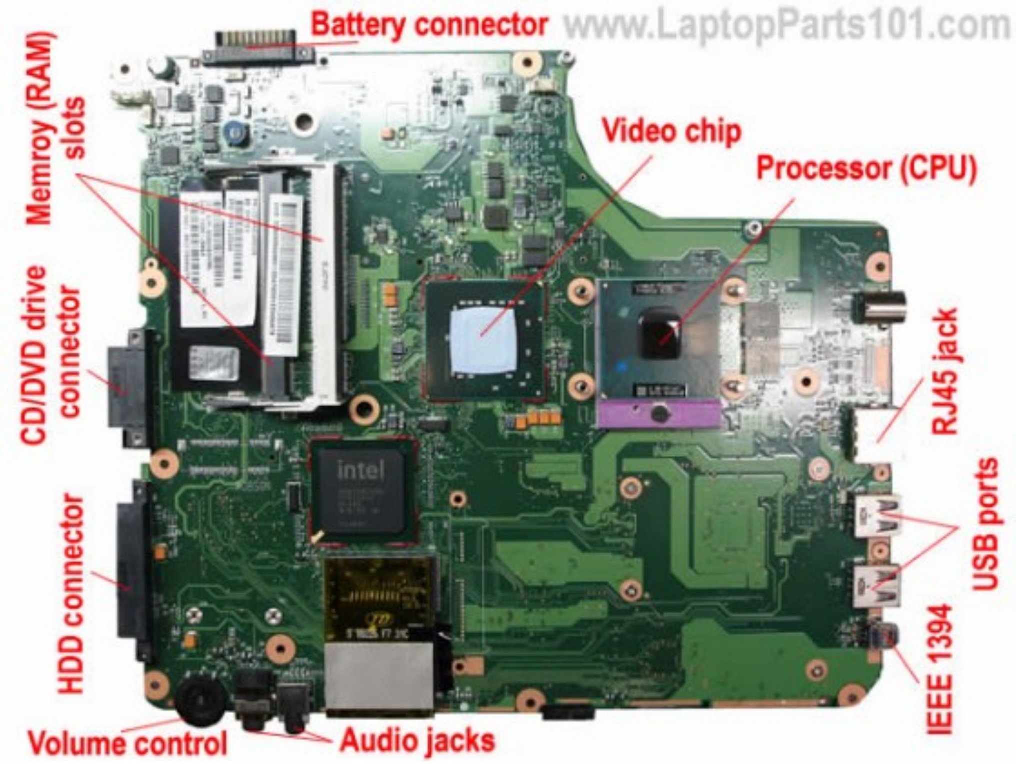
cpu with cover removed



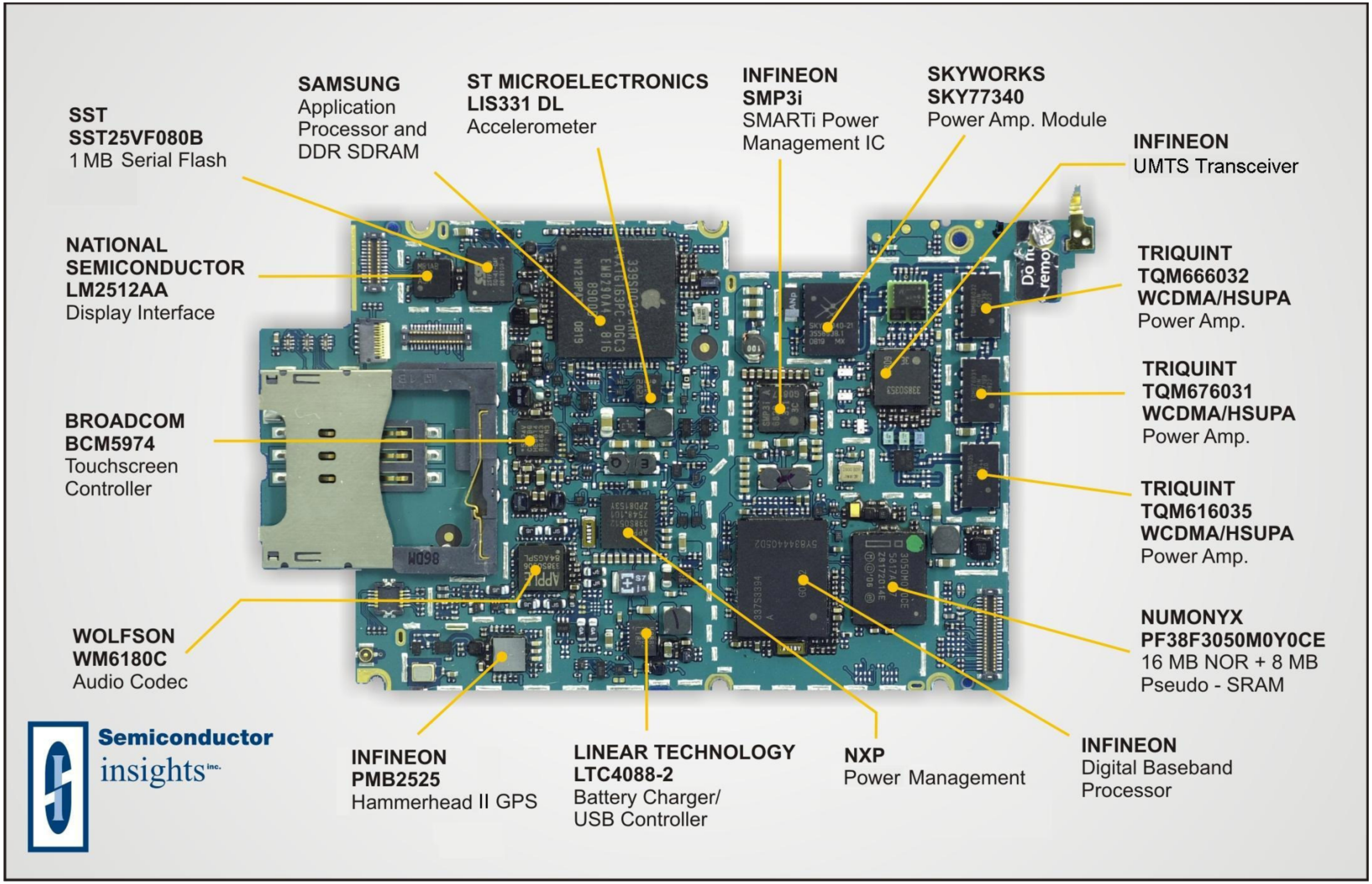
with cooler

Main Memory

Motherboard of a Laptop



Motherboard of iphone3G

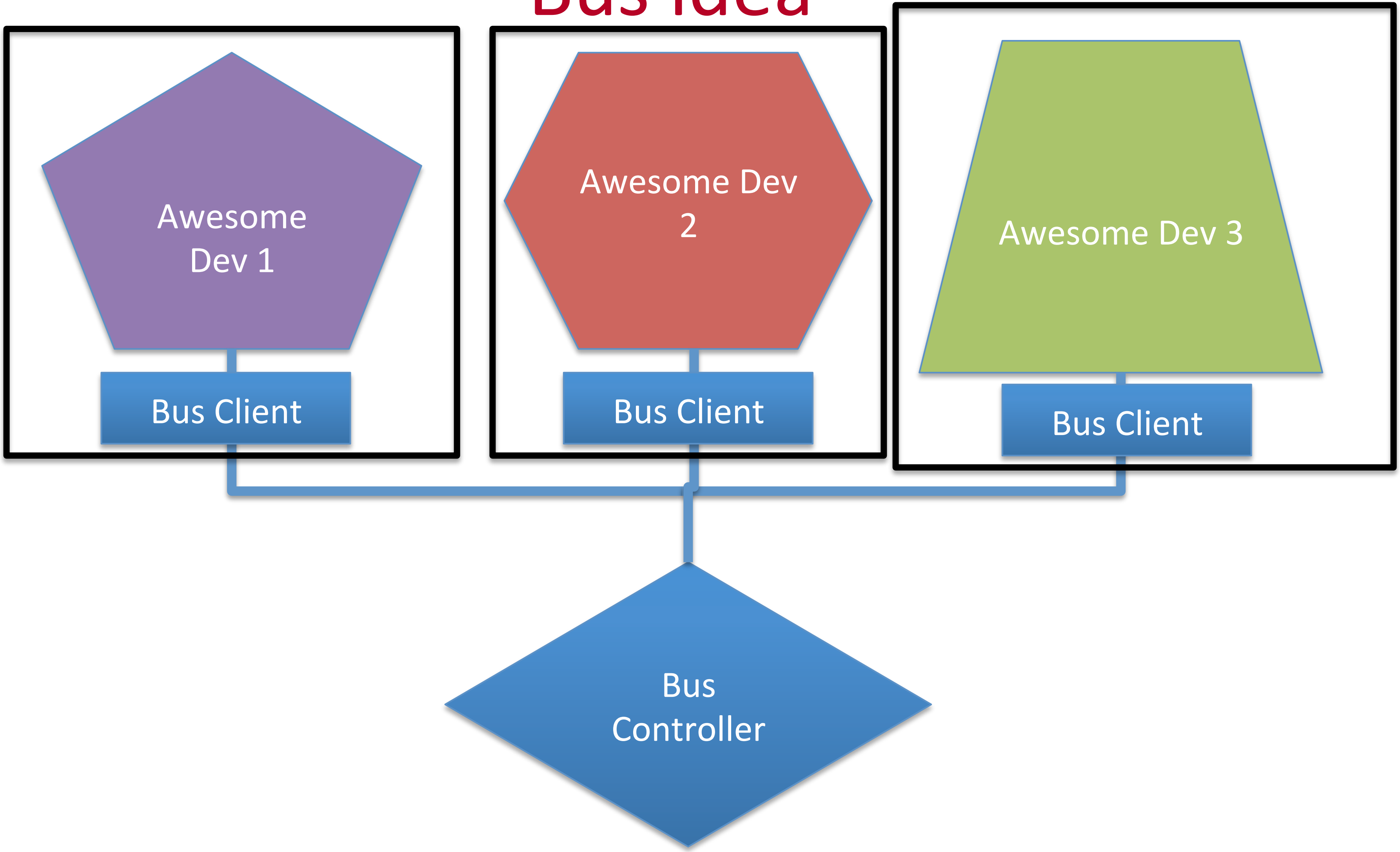


http://www.rapidrepair.com/Merchant2/merchant.mvc?Screen=cp-iphone3g

Buses

- Key abstraction for hardware
- A hardware-level messaging system for components
- A lot of EE-type specifications about clock rate, voltage, etc.
- You mostly need to know that the specs for all components have to agree

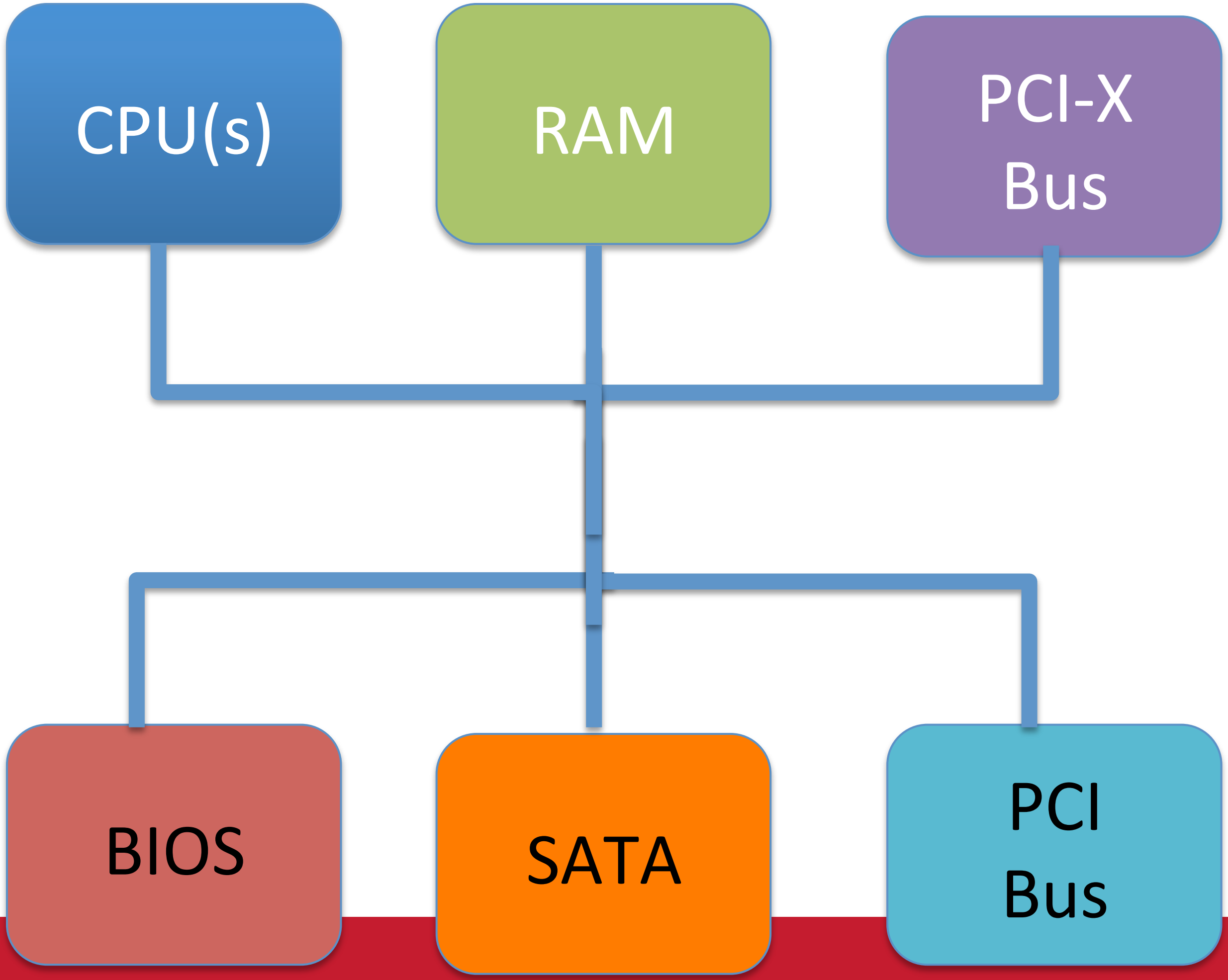
Bus idea


Awesome Dev 1
Awesome Dev 2
Awesome Dev 3
Bus Client
Bus Client
Bus Client
Bus Controller

Bus Intuition

- For a given device type, the bus client speaks the protocol over the wire
- Encapsulates low-level device implementation

A logical view of hardware



North Bridge (Fast devices: e.g., GPU)

South Bridge ("Slow" Devices: e.g., Disk, USB, Most network)

Buses galore

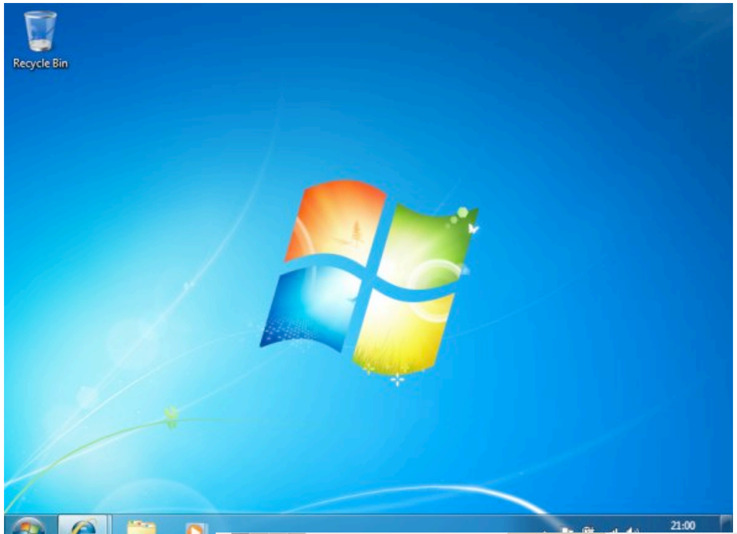
- Memory Bus: connects Northbridge components (CPU, RAM, GPU, Southbridge controller)
- Disk buses:
  - Controller on the motherboard
    - Often on south bridge
  - Speaks to disks
    - SATA, IDE, SCSI, etc
- USB, Firewire, etc. are all bus protocols

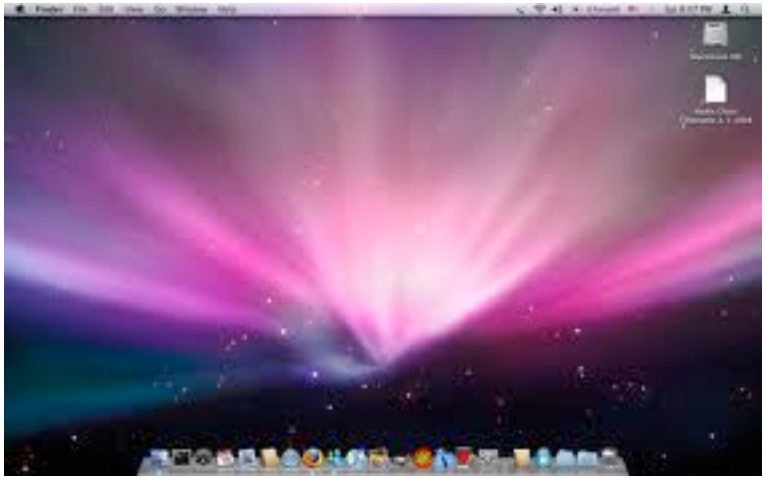
Fewer Bridges

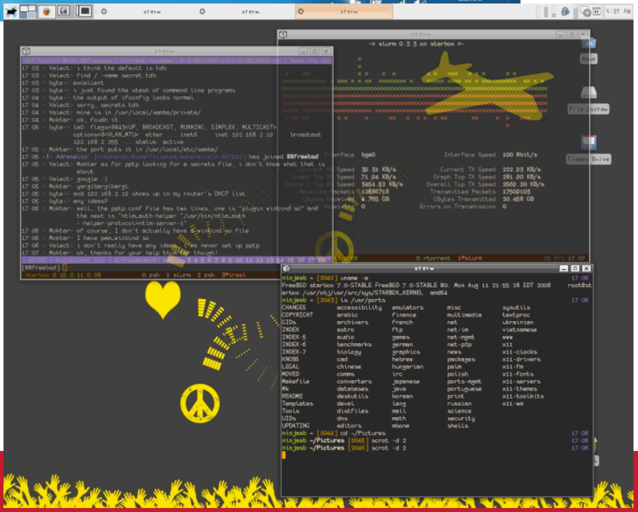
- Newer system organizations are moving more devices to the North bridge, and consolidating more things on the CPU itself.

So what is an OS?

One view of an OS



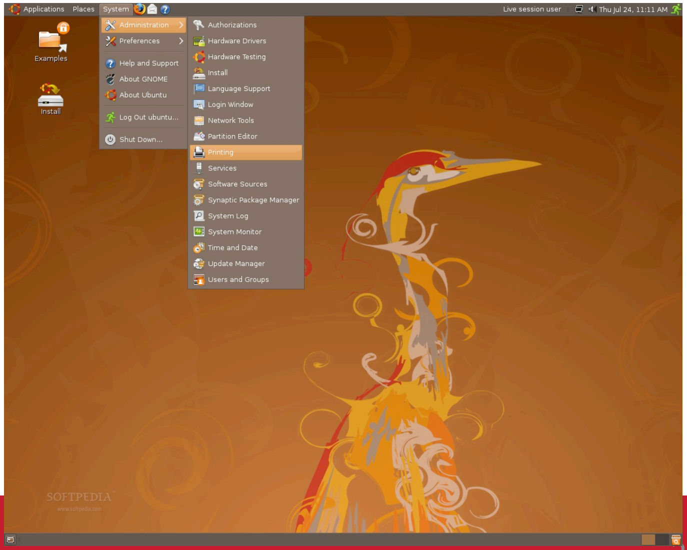




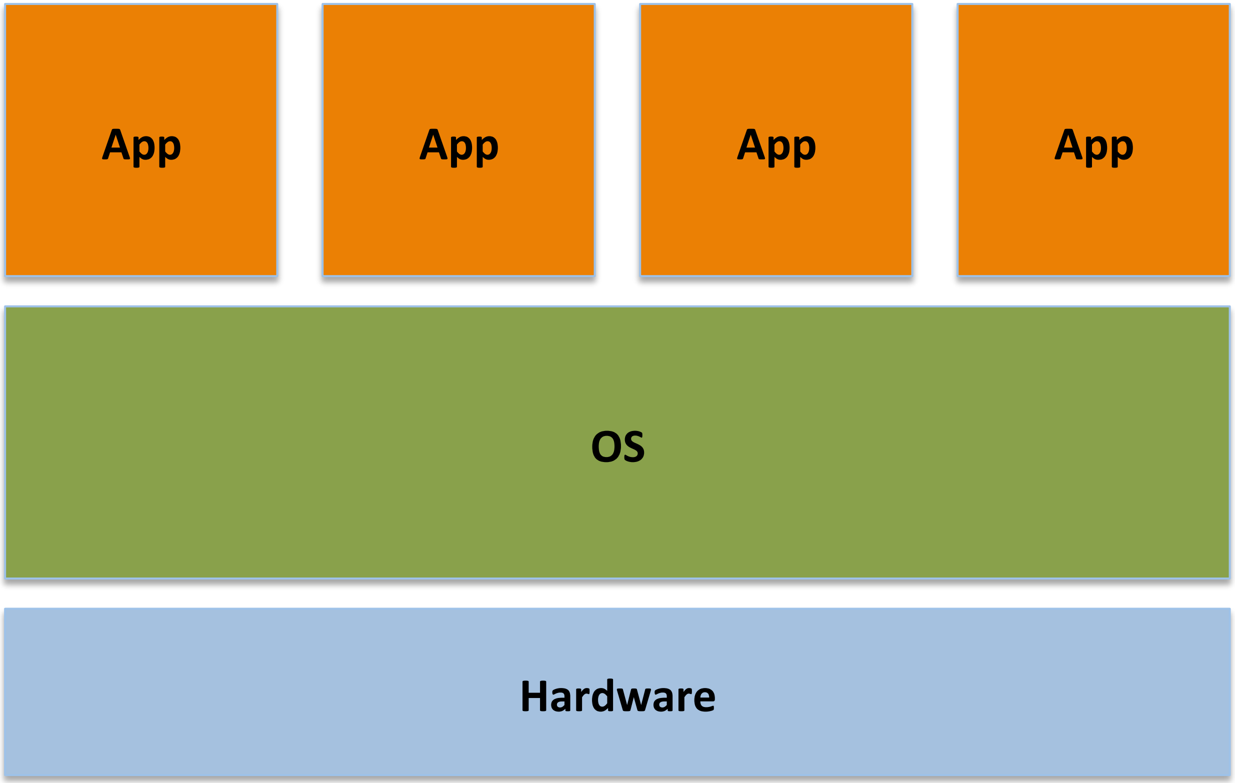
```
Off topic? #not-##freebsd | Current release: 6.3-RELEASE-p3/7.0-RELEASE-p3 | Read the doc
17:03 < Valect> i think the default is tdb
17:03 < Valect> find / -name secret.tdb
17:03 < byte-> excellent
17:03 < byte-> i just found the stash of command line programs
17:04 < byte-> the output of ifconfig looks normal
17:04 < Valect> sorry, secrets.tdb
17:04 < Valect> mine is in /usr/local/samba/private/
17:04 < Moktar> ok, foudn it
17:05 < byte-> le0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> ...
               options=8<VLAN_MTU> ether ... inet6 ... inet 192.168.2.19 ... broadcast
               192.168.2.255 ... status: active
17:05 < Moktar> the port puts it in /usr/local/etc/samba/
17:05 -!- Adrenalin` [n=Adrenali@unaffiliated/adrenalin/x-667921] has joined ##freebsd
17:05 < Valect> Moktar as for pptp looking for a secrets file, i don't know what that is
                about
17:05 < Valect> google :)
17:05 < Moktar> yarglbarglbargl
17:05 < byte-> and 192.168.2.19 shows up in my router's DHCP list
17:05 < byte-> any ideas?
17:06 < Moktar> well, the pptp.conf file has two lines, one is "plugin winbind.so" and
                the next is "ntlm_auth-helper "/usr/bin/ntlm_auth
                --helper-protocol=ntlm-server-1"
17:06 < Moktar> of course, I don't actually have a winbind.so file
17:06 < Moktar> I have pam_winbind.so
17:06 < Valect> i don't really have any ideas, i've never set up pptp
17:07 < Moktar> ok, thanks for your help thus for though!
 [17:07] [ninjaslim(+ei)] [2:FreeNode/#  [Act: 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18]
[##freebsd] 
starbox 0.15 0.11 0.08          0 zsh  1 slurm  2 zsh  3*irssi
```

```
                    -= slurm 0.3.3 on starbox =-

   Active Interface: bge0                  Interface Speed: 100 Mbit/s

  Current RX Speed: 39.31 KB/s            Current TX Speed: 222.93 KB/s
Graph Top RX Speed: 71.94 KB/s          Graph Top TX Speed: 281.90 KB/s
Overall Top RX Speed: 3464.52 KB/s    Overall Top TX Speed: 2569.20 KB/s
  Received Packets: 12686718           Transmitted Packets: 17509103
    GBytes Received: 4.765 GB          GBytes Transmitted: 20.458 GB
  Errors on Receiving: 0            Errors on Transmission: 0

15 0.08              0 rtorrent  1*slurm                    15 Fri 17:07
```

```
ninjasb ~ [2042] uname -a                                                  17:06
FreeBSD starbox 7.0-STABLE FreeBSD 7.0-STABLE #0: Mon Aug 11 21:55:28 EDT 2008     root@st
arbox:/usr/obj/usr/src/sys/STARBOX_KERNEL  amd64
ninjasb ~ [2043] ls /usr/ports                                             17:06
CHANGES         accessibility   emulators       misc            sysutils
COPYRIGHT       arabic          finance         multimedia      textproc
GIDs            archivers       french          net             ukrainian
INDEX           astro           ftp             net-im          vietnamese
INDEX-5         audio           games           net-mgmt        www
INDEX-6         benchmarks      german          net-p2p         x11
INDEX-7         biology         graphics        news            x11-clocks
KNOBS           cad             hebrew          packages        x11-drivers
LEGAL           chinese         hungarian       palm            x11-fm
MOVED           comms           irc             polish          x11-fonts
Makefile        converters      japanese        ports-mgmt      x11-servers
Mk              databases       java            portuguese      x11-themes
README          deskutils       korean          print           x11-toolkits
Templates       devel           lang            russian         x11-wm
Tools           distfiles       mail            science
UIDs            dns             math            security
UPDATING        editors         mbone           shells
ninjasb ~ [2044] cd ~/Pictures                                             17:06
ninjasb ~/Pictures [2045] scrot -d 2                                       17:06
ninjasb ~/Pictures [2046] scrot -d 2                                       17:06
```



Another simple view of an OS



App

App

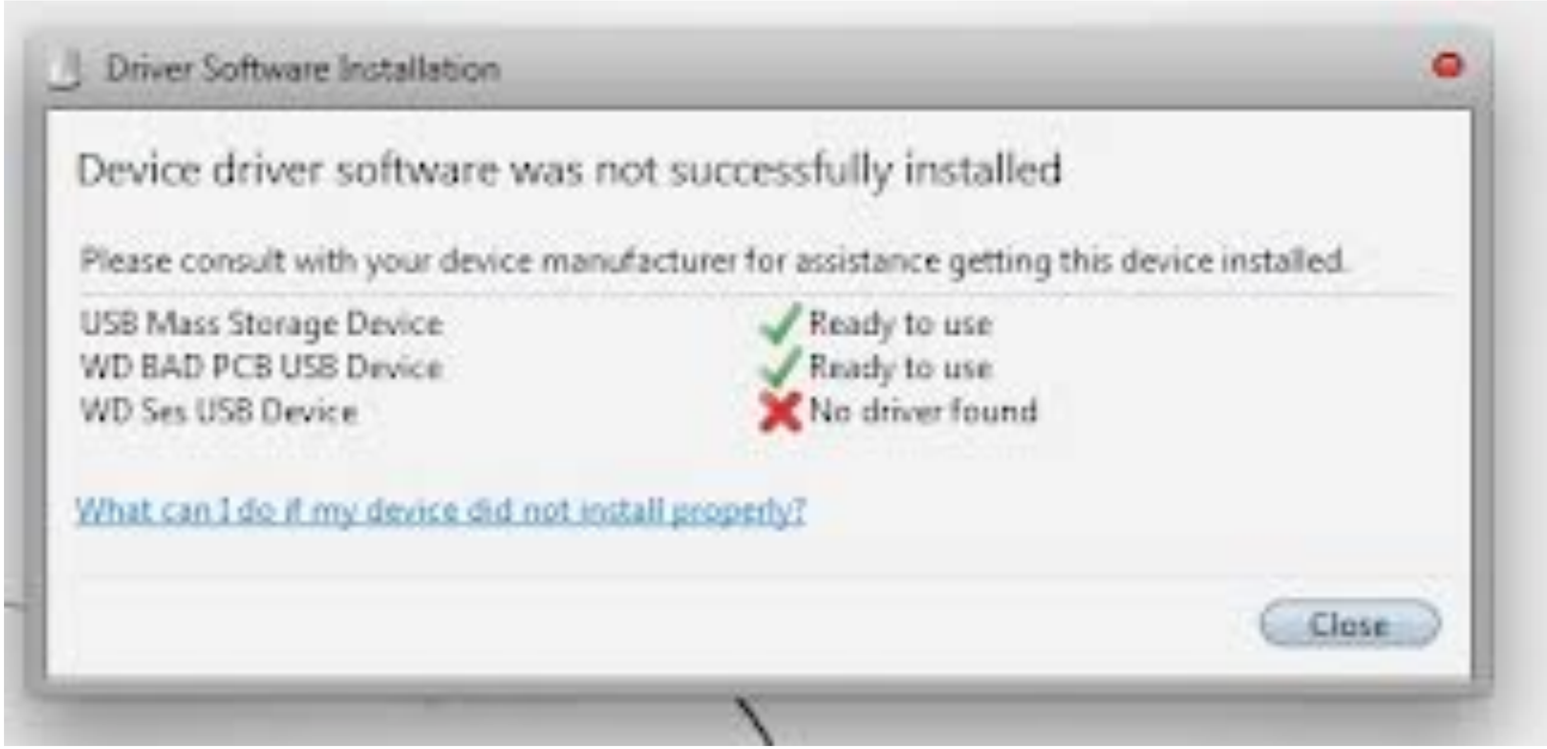
App

App

OS

Hardware

A less happy view of an OS



Driver Software Installation

Device driver software was not successfully installed

Please consult with your device manufacturer for assistance getting this device installed.

USB Mass Storage Device	✓ Ready to use
WD BAD PCB USB Device	✓ Ready to use
WD Ses USB Device	✗ No driver found

What can I do if my device did not install properly?

So which one is right?

- They all are

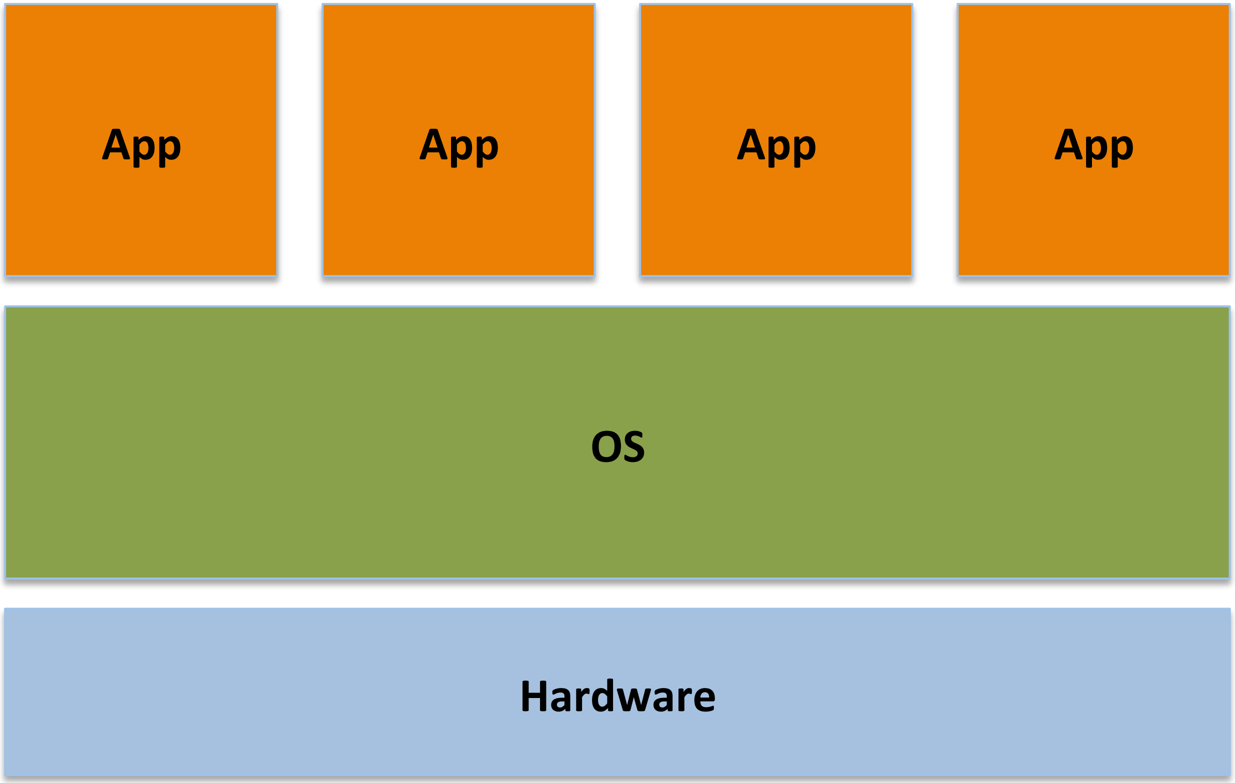
An OS serves three masters

1. Give users a desktop environment
2. Give applications a more usable abstraction of the hardware
3. Give hardware manufacturers an abstraction of the applications

Background (1)

- CPUs have 2 modes: user and supervisor
  - Sometimes more, but whatevs
- Supervisor mode:
  - Issue commands to hardware devices
  - Power off, Reboot, Suspend
  - Launch missiles, Do awesome stuff
- User mode:
  - Run other code, hardware tattles if you try anything reserved for the supervisor

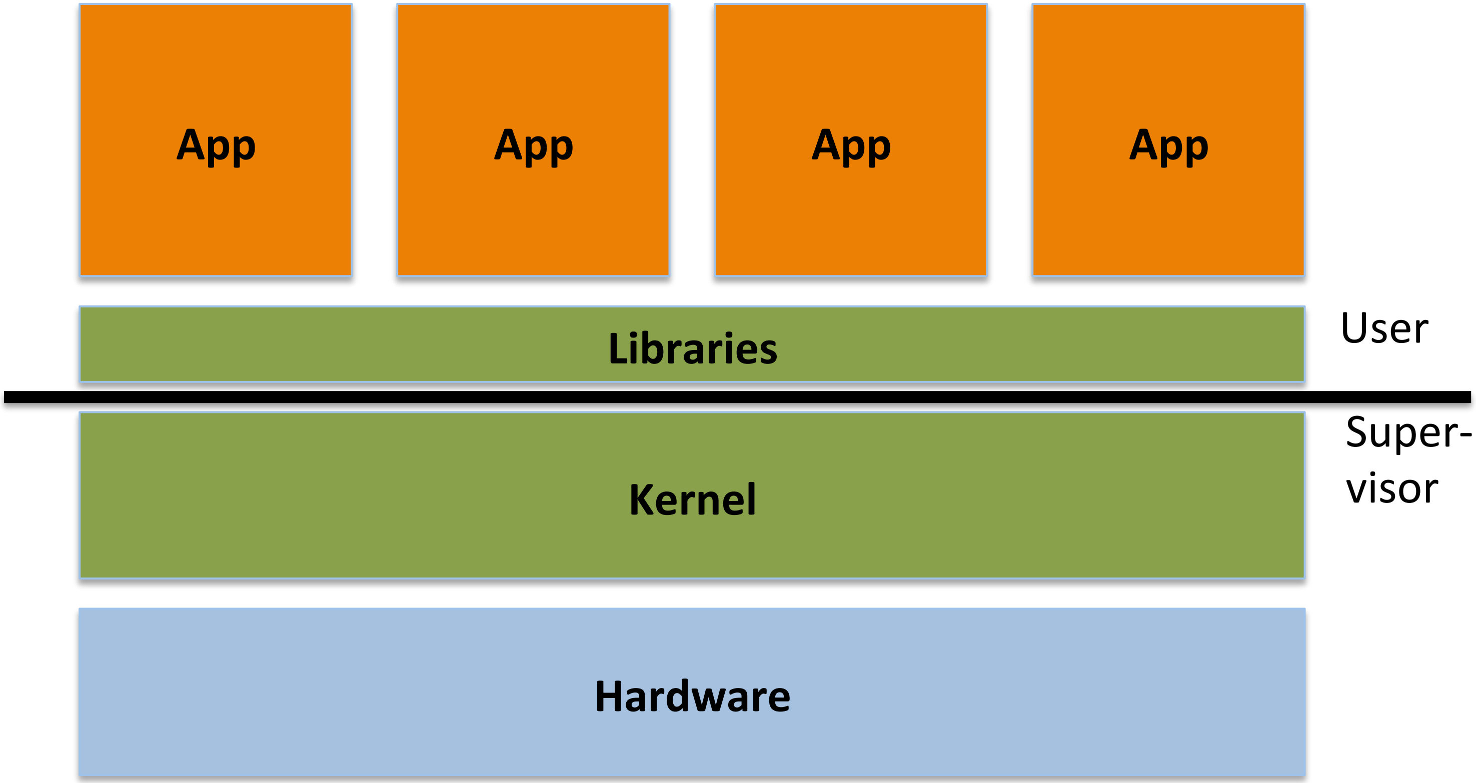
OS architecture


App App App App

OS

Hardware

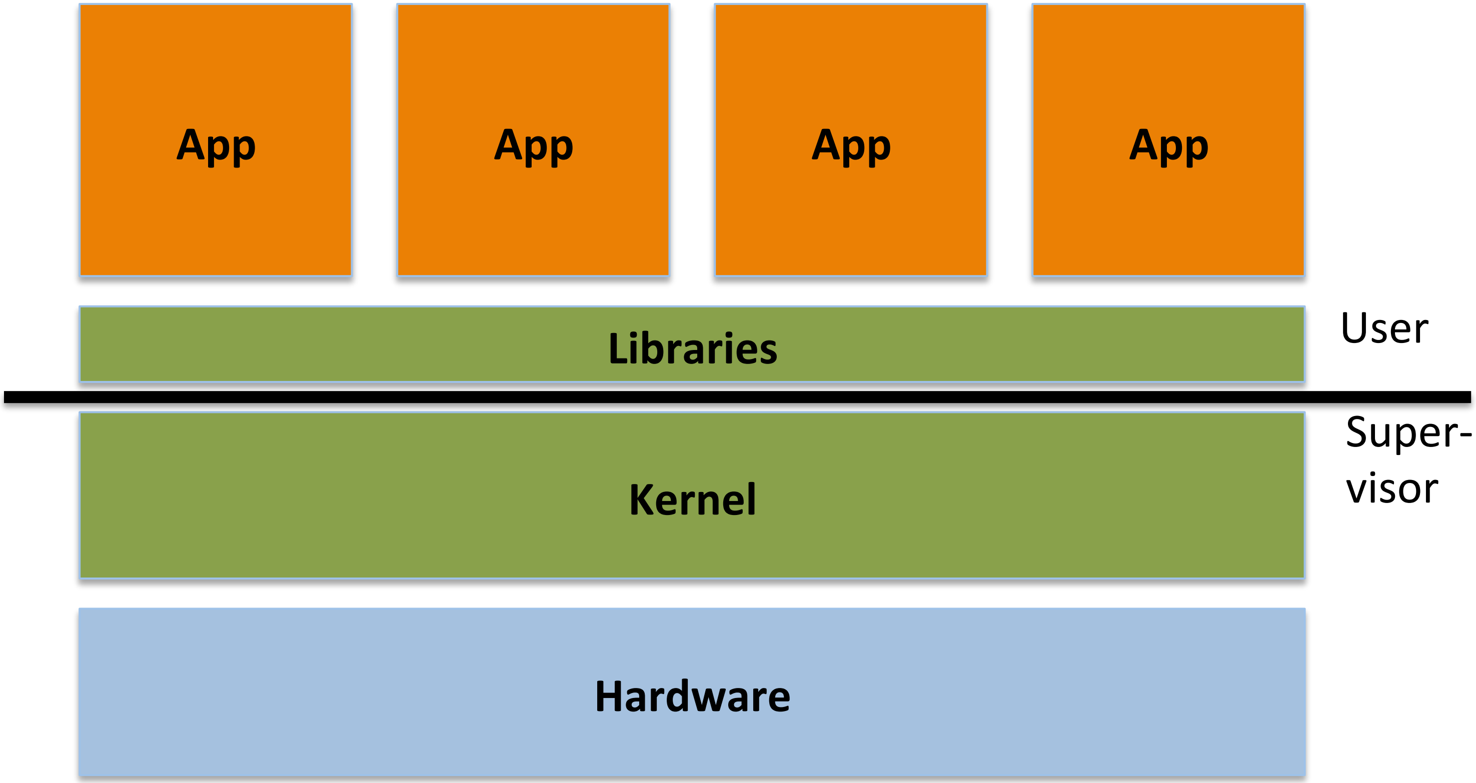
OS architecture



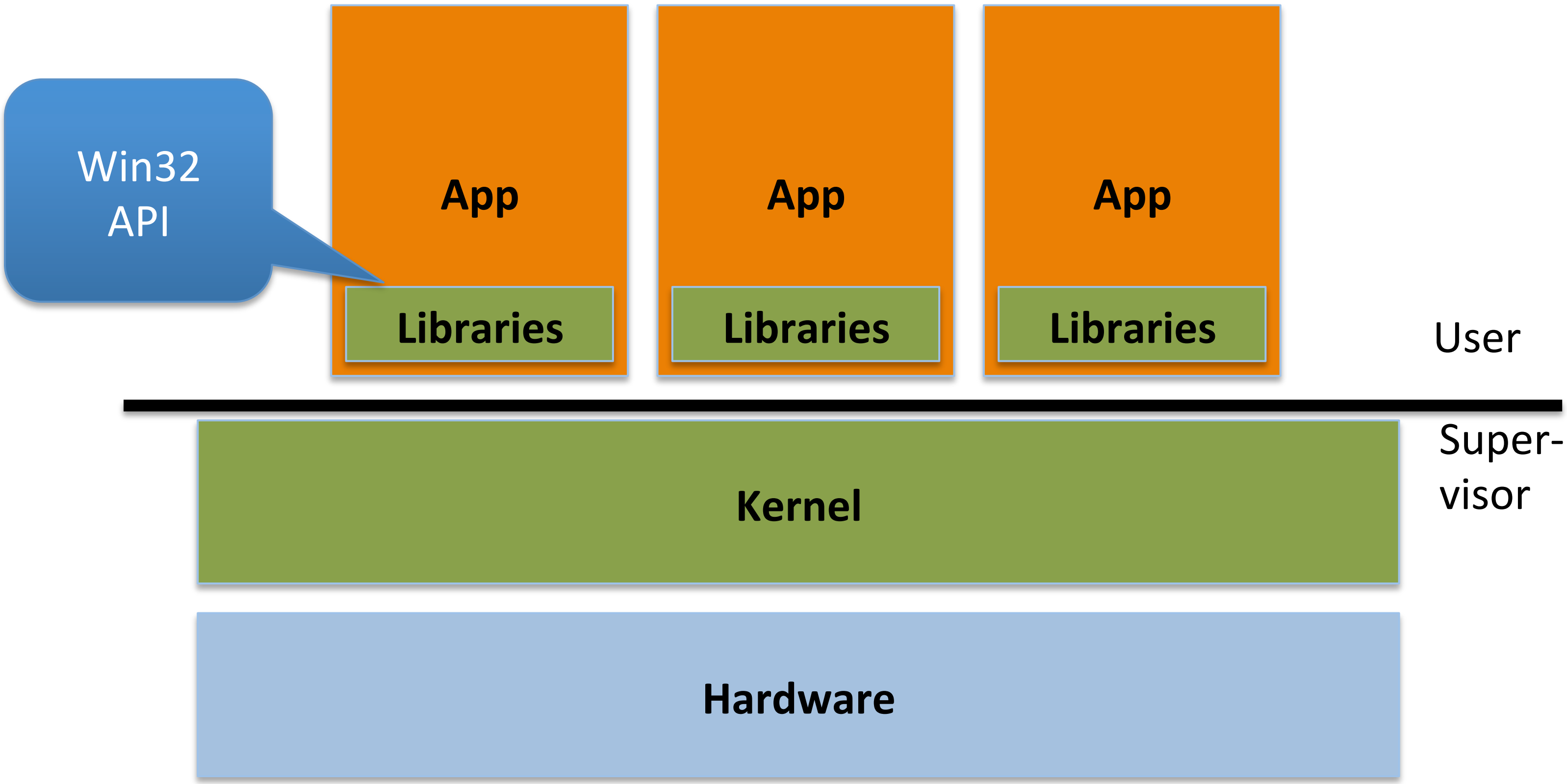
Master #2: Applications

- Application Programming Interface (API)
  - Win32 (Windows)
  - POSIX (Unix/Linux)
  - Cocoa/Cocoa Touch (Mac OS/iOS)
- Application-facing functions provided by libraries
  - Injected by the OS into each application

OS architecture


App
App
App
App
Libraries
User
Super-visor
Kernel
Hardware

OS architecture



Famous libraries, anyone?

- Windows: ntdll.dll, kernel32.dll, user32.dll, gdi32.dll
- Linux/Unix: libc.so, ld.so, libpthread.so, libm.so

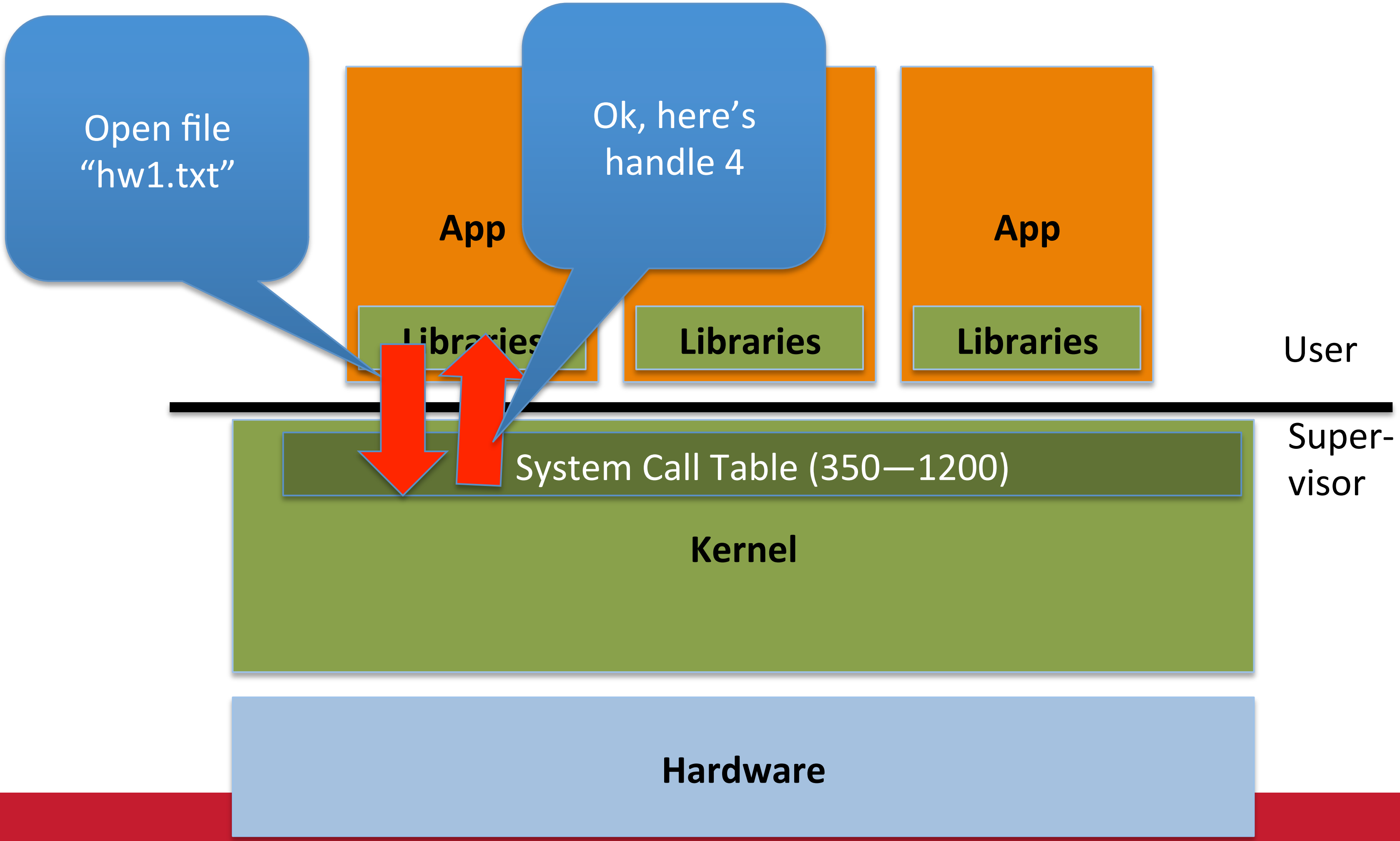
Caveat 1

- Libraries include a lot of code for common functions
  - Why bother reimplementing sqrt?
- They also give high-level abstractions of hardware
  - Files, printer, dancing Homer Simpson USB doll
- How does this work?

System Call

- Special instruction to switch from user to supervisor mode
- Transfers CPU control to the kernel
  - One of a small-ish number of well-defined functions
- How many system calls does Windows or Linux have?
  - Windows ~1200
  - Linux ~350

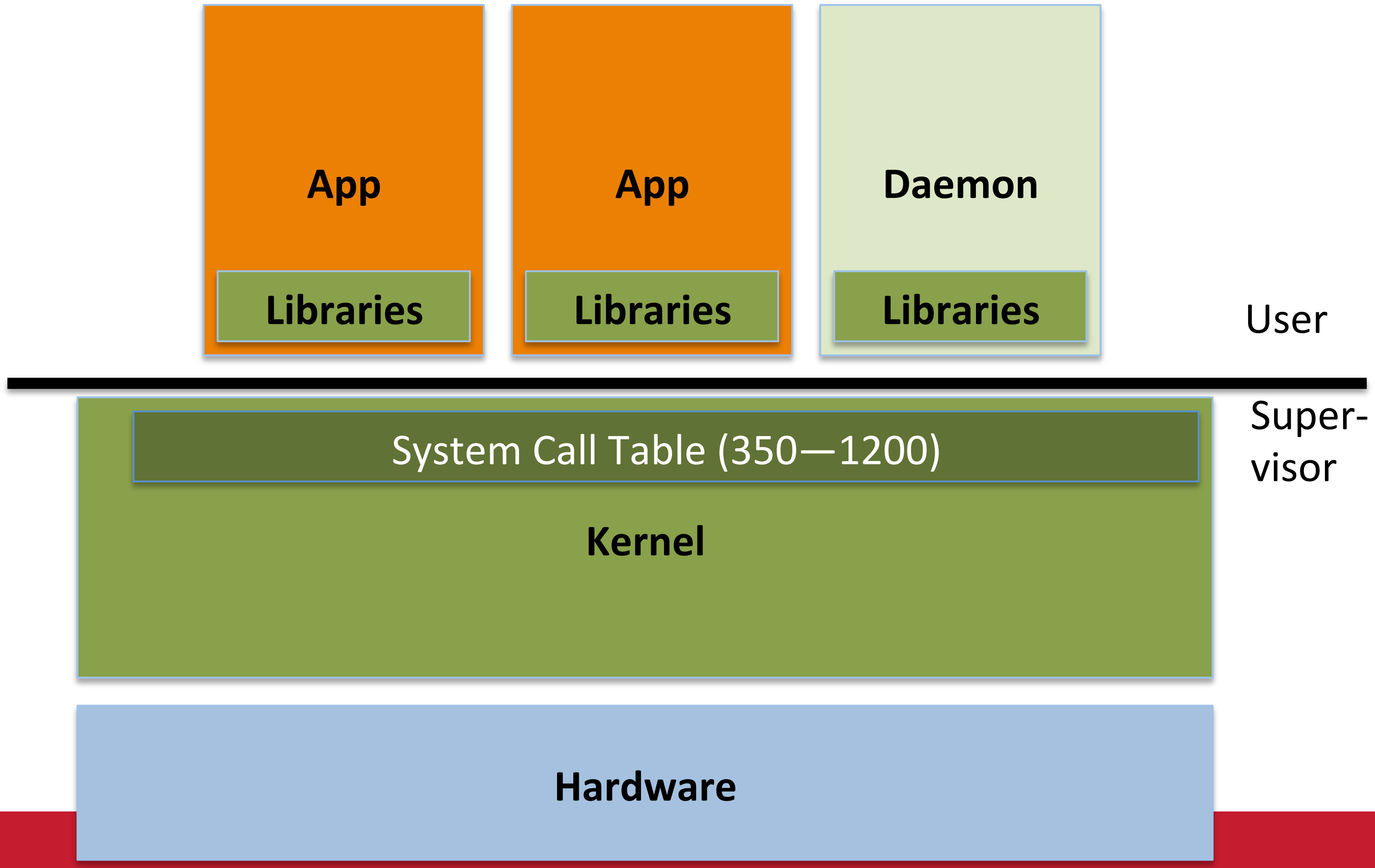
OS architecture



Caveat 2

- Some libraries also call special apps provided by the OS, called a ***daemon (or service)***
  - Communicate through kernel-provided API
- Example: Print spooler
  - App sends pdf to spooler
  - Spooler checks quotas, etc.
  - Turns pdf into printer-specific format
  - Sends reformatted document to device via OS kernel

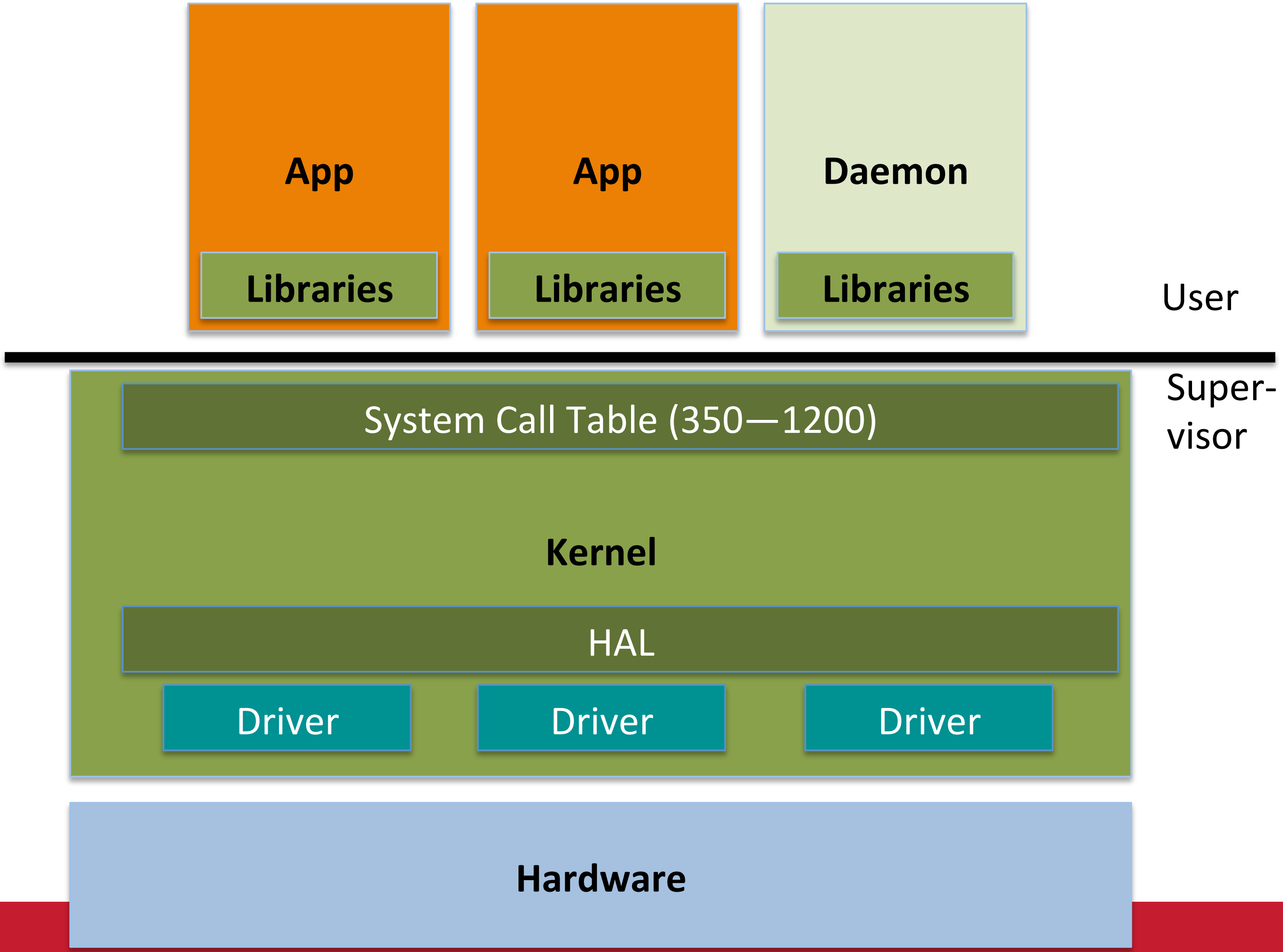
OS architecture



Master 3: Hardware

- OS kernels are programmed at a higher low level of abstraction
  - Disk blocks vs. specific types of disks
- For most types of hardware, the kernel has a “lowest common denominator” interface
  - E.g., Disks, video cards, network cards, keyboard
  - Think Java abstract class
  - Sometimes called a hardware abstraction layer (HAL)
- Each specific device (Nvidia GeForce 600) needs to implement the abstract class
  - Each implementation is called a **device driver**

OS architecture



App

Libraries

App

Libraries

Daemon

Libraries

User

Super-
visor

System Call Table (350—1200)

Kernel

HAL

Driver

Driver

Driver

Hardware

What about Master 1

- What is the desktop?
- Really just a special daemon that interacts closely with keyboard, mouse, and display drivers
  - Launches programs when you double click, etc.
  - Some program libraries call desktop daemon to render content, etc.

An OS serves three masters

1. Give users a desktop environment
   - Desktop, or window manager, or GUI
2. Give applications a more usable abstraction of the hardware
   - Libraries (+ system calls and daemons)
3. Give hardware manufacturers an abstraction of the applications
   - Device Driver API (or HAL)

So what is Linux?

- Really just an OS kernel
  - Including lots of device drivers
- Conflated with environment consisting of:
  - Linux kernel
  - Gnu libc
  - X window manager daemon
  - CUPS printer manager
  - Etc.

So what is Ubuntu? Centos?

- A **distribution:** bundles all of that stuff together
  - Pick versions that are tested to work together
  - Usually also includes a software update system

OSX vs iOS?

- Same basic kernel (a few different compile options)
- Different window manager and libraries

What is Unix?

- A very old OS (1970s), innovative, still in use
- Innovations:
  - Kernel written in C (first one not in assembly)
    - Co-designed C language with Unix
  - Several nice API abstractions
    - Fork, pipes, everything a file
- Several implementations: *BSDs, Solaris, etc.
  - Linux is a Unix-like kernel

What is POSIX?

- A standard for Unix compatibility
- Even Windows is POSIX compliant!

RTFM

- man pages are your friend!
  - (not a dating service)
  - Ex: ‘man malloc’, or ‘man 3 printf’
    - Section 3 is usually where libraries live - there is a command-line utility printf as well
- Use ‘apropos *term*’ to search for man entries about *term*
- Windows has MSDN online, which is good

Summary

- Understand basic hardware terminology
- Understand what an OS is
  - Three masters
  - Nomenclature
- Questions?