What is an Operating System?

January 19th, 2021
Profs. Natacha Crooks and Anthony D. Joseph
http://cs162.eecs.Berkeley.edu
Goals for Today

• What is an Operating System?
  – And – what is it not?
• What makes Operating Systems so exciting?
• Oh, and “How does this class operate?”

Greatest Artifact of Human Civilization...
Greatest Artifact of Human Civilization…

The Internet!
Running Systems at Internet Scale

Worldwide Internet Users

- ARPANet
- RFC 675
- TCP/IP
- Internet
- HTTP 0.9
- WWW

1969 1974 1990

% Population Million
Bell’s Law: New computer class every 10 years

Across Incredible Diversity

- Number crunching, Data Storage, Massive Inet Services, ML, ...
- Productivity, Interactive
- Streaming from/to the physical world

The Internet of Things!
And Range of Timescales

<table>
<thead>
<tr>
<th>Activity</th>
<th>Time (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1 cache reference</td>
<td>0.5</td>
</tr>
<tr>
<td>Branch mispredict</td>
<td>5</td>
</tr>
<tr>
<td>L2 cache reference</td>
<td>7</td>
</tr>
<tr>
<td>Mutex lock/unlock</td>
<td>25</td>
</tr>
<tr>
<td>Main memory reference</td>
<td>100</td>
</tr>
<tr>
<td>Compress 1K bytes with Zippy</td>
<td>3,000</td>
</tr>
<tr>
<td>Send 2K bytes over 1 Gbps network</td>
<td>20,000</td>
</tr>
<tr>
<td>Read 1 MB sequentially from memory</td>
<td>250,000</td>
</tr>
<tr>
<td>Round trip within same datacenter</td>
<td>500,000</td>
</tr>
<tr>
<td>Disk seek</td>
<td>10,000,000</td>
</tr>
<tr>
<td>Read 1 MB sequentially from disk</td>
<td>20,000,000</td>
</tr>
<tr>
<td>Send packet CA-&gt;Netherlands-&gt;CA</td>
<td>150,000,000</td>
</tr>
</tbody>
</table>

Jeff Dean: “Numbers Everyone Should Know”
Operating Systems are at the Heart of it All!

• Make the incredible advance in the underlying technology available to a rapidly evolving body of applications
  – Provide **consistent abstractions** to applications, even on different hardware
  – Manage **sharing of resources** among multiple applications

• The key building blocks:
  – Processes
  – Threads, Concurrency, Scheduling, Coordination
  – Address Spaces
  – Protection, Isolation, Sharing, Security
  – Communication, Protocols
  – Persistent storage, transactions, consistency, resilience
  – Interfaces to all devices
Example: What’s in a Search Query?

- Complex interaction of multiple components in multiple administrative domains
  - Systems, services, protocols, …
But: What is an operating system?
What does an Operating System do?

• Most Likely:
  – Memory Management
  – I/O Management
  – CPU Scheduling
  – Communications? (Does Email belong in OS?)
  – Multitasking/multiprogramming?

• What about?
  – File System?
  – Multimedia Support?
  – User Interface?
  – Internet Browser? 😊

• Is this only interesting to Academics??
Definition of an Operating System

• No universally accepted definition
• “Everything a vendor ships when you order an operating system” is good approximation
  – But varies wildly

• “The one program running at all times on the computer” is the kernel
  – Everything else is either a system program (ships with the operating system) or an application program
One Definition of an Operating System

- Special layer of software that provides application software access to hardware resources
  - Convenient abstraction of complex hardware devices
  - Protected access to shared resources
  - Security and authentication
  - Communication
Operating System

Switchboard Operator

Computer Operators
What makes something a **system**?

- Multiple interrelated parts
  - Each potentially interacts with the others
- Robustness requires an **engineering mindset**
  - Meticulous error handling, defending against malicious and careless users
  - Treating the computer as a concrete machine, with all of its limitations and possible failure cases

**System programming is an important part of this class!**
What is an operating system?

Illusionist
Hardware/Software Interface

What you learned in CS 61C – Machine Structures (and C)

The OS abstracts these hardware details from the application
What is an Operating System?

• Illusionist
  – Provide clean, easy-to-use abstractions of physical resources
    » Infinite memory, dedicated machine
    » Higher level objects: files, users, messages
    » Masking limitations, virtualization
OS Basics: Virtualizing the Machine

Process: Execution environment with restricted rights provided by OS

Thread | Address Spaces | Files | Sockets

Operating System

Compiled Program

System Libs

Hardware

Processor

PgLBI & TLB

Memory

OS Mem

I/O Ctrlr

Storage

Networks

ISA
Compiled Program’s View of the World

- **Compiled Program**
- **System Libs**
- **Operating System**
  - Process: Execution environment with restricted rights provided by OS
  - Threads
  - Address Spaces
  - Files
  - Sockets

---

**Compiled Program’s View of the World: ISA**

- Application’s “machine” is the process abstraction provided by the OS
- Each running program runs in its own process
- Processes provide nicer interfaces than raw hardware

---

**Hardware**

**I/O Ctrlr**
System Programmer’s View of the World

- **Process**: Execution environment with restricted rights provided by OS
  - Threads
  - Address Spaces
  - Files
  - Sockets

**Operating System**
- Application’s “machine” is the process abstraction provided by the OS
- Each running program runs in its own process
- Processes provide nicer interfaces than raw hardware

**Program**
```
#include <stdio.h>
int main(void) {
    printf("Hello!\n")
}
```
What’s in a Process?

A process consists of:
• Address Space
• One or more threads of control executing in that address space
• Additional system state associated with it
  – Open files
  – Open sockets (network connections)
  – …
For Example...

```
Activity Monitor (All Processes)

<table>
<thead>
<tr>
<th>Process Name</th>
<th>% CPU</th>
<th>CPU Time</th>
<th>Threads</th>
<th>Idle Wake Ups</th>
<th>PID</th>
<th>User</th>
<th>Sandbox</th>
</tr>
</thead>
<tbody>
<tr>
<td>kernel_task</td>
<td>24.0</td>
<td>28:22.53</td>
<td>237</td>
<td>12,805</td>
<td>0</td>
<td>root</td>
<td>No</td>
</tr>
<tr>
<td>VDCAssistant</td>
<td>16.3</td>
<td>0:54:41.87</td>
<td>4</td>
<td>250</td>
<td>0</td>
<td>250</td>
<td>Yes</td>
</tr>
<tr>
<td>Microsoft PowerPoint</td>
<td>7.3</td>
<td>1:08.30</td>
<td>39</td>
<td>108</td>
<td>36013</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>QuickTime Player</td>
<td>5.1</td>
<td>11:56.16</td>
<td>10</td>
<td>91</td>
<td>34450</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>Camtasia 2020</td>
<td>4.0</td>
<td>8:37.92</td>
<td>31</td>
<td>1,154</td>
<td>34483</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>Firefox</td>
<td>3.3</td>
<td>24:02:12.22</td>
<td>128</td>
<td>82</td>
<td>457</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>WindowServer</td>
<td>3.0</td>
<td>0:05:01.78</td>
<td>9</td>
<td>41</td>
<td>222</td>
<td>windowserver</td>
<td>Yes</td>
</tr>
<tr>
<td>Finder</td>
<td>2.3</td>
<td>1:14:10.04</td>
<td>9</td>
<td>7</td>
<td>470</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>coreaudiod</td>
<td>2.0</td>
<td>6:39:26.38</td>
<td>13</td>
<td>199</td>
<td>203</td>
<td>coraudiod</td>
<td>Yes</td>
</tr>
<tr>
<td>Box Sync Monitor</td>
<td>1.6</td>
<td>24:08:17.77</td>
<td>1</td>
<td>10</td>
<td>887</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>Activity Monitor</td>
<td>1.2</td>
<td>0:03:21.81</td>
<td>4</td>
<td>1</td>
<td>442</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>Adobe CEF Helper (Renderer)</td>
<td>0.9</td>
<td>25:45:27.86</td>
<td>22</td>
<td>9</td>
<td>862</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>syslog</td>
<td>0.7</td>
<td>9:01:32.78</td>
<td>6</td>
<td>0</td>
<td>385</td>
<td>root</td>
<td>No</td>
</tr>
<tr>
<td>stderr</td>
<td>0.5</td>
<td>3:31:95.6</td>
<td>5</td>
<td>0</td>
<td>131</td>
<td>stderr</td>
<td>No</td>
</tr>
<tr>
<td>Creative Cloud</td>
<td>0.5</td>
<td>6:04:07.73</td>
<td>23</td>
<td>1</td>
<td>711</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>FirefoxCP Web Content</td>
<td>0.4</td>
<td>3:46:07.57</td>
<td>44</td>
<td>95</td>
<td>645</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>CalendarAgent</td>
<td>0.4</td>
<td>3:21:22.81</td>
<td>6</td>
<td>1</td>
<td>493</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>Box Sync</td>
<td>0.3</td>
<td>7:41:29.20</td>
<td>37</td>
<td>132</td>
<td>721</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>Adobe CEF Helper (GPU)</td>
<td>0.3</td>
<td>10:09:18.33</td>
<td>8</td>
<td>26</td>
<td>815</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>FirefoxCP WebExtensions</td>
<td>0.2</td>
<td>1:48:22.14</td>
<td>38</td>
<td>2</td>
<td>619</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>FirefoxCP Privileged Content</td>
<td>0.2</td>
<td>5:20.13</td>
<td>36</td>
<td>0</td>
<td>1940</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>mDNSResponder</td>
<td>0.2</td>
<td>0.04</td>
<td>3</td>
<td>0</td>
<td>36984</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>Core Sync</td>
<td>0.1</td>
<td>52:47:09</td>
<td>5</td>
<td>5</td>
<td>237</td>
<td>mDNSresponse</td>
<td>Yes</td>
</tr>
<tr>
<td>mDNSResponder</td>
<td>0.1</td>
<td>1:09:33.70</td>
<td>26</td>
<td>7</td>
<td>63810</td>
<td>adj</td>
<td>No</td>
</tr>
<tr>
<td>mDNSResponder</td>
<td>0.1</td>
<td>22:03.56</td>
<td>4</td>
<td>0</td>
<td>434</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>mds</td>
<td>0.1</td>
<td>38:35.57</td>
<td>10</td>
<td>2</td>
<td>89</td>
<td>root</td>
<td>No</td>
</tr>
<tr>
<td>mds_stores</td>
<td>0.1</td>
<td>1:08:29.25</td>
<td>5</td>
<td>1</td>
<td>362</td>
<td>root</td>
<td>Yes</td>
</tr>
<tr>
<td>logd</td>
<td>0.1</td>
<td>25:51.78</td>
<td>3</td>
<td>0</td>
<td>81</td>
<td>root</td>
<td>No</td>
</tr>
<tr>
<td>dasd</td>
<td>0.1</td>
<td>12:59.16</td>
<td>4</td>
<td>1</td>
<td>117</td>
<td>root</td>
<td>No</td>
</tr>
<tr>
<td>FirefoxCP Web Content</td>
<td>0.1</td>
<td>22:20.26</td>
<td>38</td>
<td>3</td>
<td>571</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>cpufreqd</td>
<td>0.0</td>
<td>5:11:85</td>
<td>3</td>
<td>0</td>
<td>426</td>
<td>adj</td>
<td>Yes</td>
</tr>
<tr>
<td>laud</td>
<td>0.0</td>
<td>46:40:16</td>
<td>4</td>
<td>20</td>
<td>71</td>
<td>root</td>
<td>No</td>
</tr>
<tr>
<td>FirefoxCP Web Content</td>
<td>0.0</td>
<td>1:07:11.46</td>
<td>38</td>
<td>3</td>
<td>654</td>
<td>adj</td>
<td>Yes</td>
</tr>
</tbody>
</table>
```

System: 4.21%
User: 3.74%
Idle: 93.08%

CPU LOAD: 2495
Threads: 483
Processes: 483
Operating System’s View of the World

- OS translates from hardware interface to application interface
- OS provides each running program with its own process

Operating System

Process 1
- System Libs
- Threads
- Address Spaces
- Files
- Sockets

Process 2
- System Libs
- Threads
- Address Spaces
- Files
- Sockets

Thread
Address Spaces
Files
Sockets

Compiler

ISA

Hardware

Processor

Memory

OS Mem

Storage

Network

I/O Ctrlr
What is an operating system?

Referee
What is an Operating System?

• Referee
  – Manage protection, isolation, and sharing of resources
    » Resource allocation and communication

• Illusionist
  – Provide clean, easy-to-use abstractions of physical resources
    » Infinite memory, dedicated machine
    » Higher level objects: files, users, messages
    » Masking limitations, virtualization
OS Basics: Running a Process

Compiled Program 1

System Libs

Process 1

Threads Address Spaces Files Sockets

Compiled Program 2

System Libs

Process 2

Threads Address Spaces Files Sockets

Operating System

Compiler

ISA

Hardwar e

Processor Memory

I/O Ctrlr

Storag e

Networ ks
OS Basics: Switching Processes
OS Basics: Switching Processes

- Process 1
  - Threads
  - Address Spaces
  - Files
  - Sockets
  - System Libs

- Process 2
  - Threads
  - Address Spaces
  - Files
  - Sockets
  - System Libs

- Compiled Program 1
- Compiled Program 2

- Operating System

- Hardware
  - Processor
  - Memory
  - Storage
  - Networks
  - I/O Controller
  - PgTables & TLB

- Compiler
- ISA

- Process 1
- Process 2
- Compiled Program 1
  - System Libs
- Compiled Program 2
  - System Libs

- Compiled Program 1
- Compiled Program 2
- System Libs
OS Basics: Switching Processes

Compiled Program 1
- System Libs
- Process 1
  - Threads
  - Address Spaces
  - Files
  - Sockets

Compiled Program 2
- System Libs
- Process 2
  - Threads
  - Address Spaces
  - Files
  - Sockets

Operating System

Hardware
- Processor
- Memory
- Storage
- Network

I/O Controller

Compiler

ISA

Compiler
OS Basics: Switching Processes

- Compiled Program 1
  - System Libs
  - Process 1
    - Threads
    - Address Spaces
    - Files
    - Sockets

- Compiled Program 2
  - System Libs
  - Process 2
    - Threads
    - Address Spaces
    - Files
    - Sockets

- Operating System
  - Processor
  - Memory
  - Storage
  - Networks
  - I/O Controller

- Hardware
  - ISA

- Compiler

- OS Basics: Switching Processes

1/19/21

Crooks & Joseph CS162 © UCB Spring 2021  Lec 1.35
OS Basics: Protection

Compiled Program 1
- System Libs

Process 1
- Threads
- Address Spaces
- Files
- Sockets

Compiled Program 2
- System Libs

Process 2
- Threads
- Address Spaces
- Files
- Sockets

Operating System

Compiler

Hardwar e

Processor

Memory

I/O Ctrlr

PGTE & TLB

Networks

Storage

Files

Sockets

System Libs

Compiled Program 1

Compiled Program 2
OS Basics: Protection

Compiled Program 1
System Libs

Process 1

Operating System

Compiler

Hardware

Processor
Memory

I/O Ctrlr

PgTbl & TLB

Storage

Networ ks

Threads Address Spaces Files Sockets Threads Address Spaces Files Sockets

Process 2

Compiled Program 2
System Libs

Segmentation fault (core dumped)

Compiled Program 1
System Libs

Process 1

Operating System

Compiler

Hardware

Processor
Memory

I/O Ctrlr

PgTbl & TLB

Storage

Networ ks

Threads Address Spaces Files Sockets Threads Address Spaces Files Sockets

Process 2

Compiled Program 2
System Libs

Segmentation fault (core dumped)
OS Basics: Protection

- **OS isolates processes from each other**
- **OS isolates itself from other processes**
- ... even though they are actually running on the same hardware!
What is an operating system?

Glue
What is an Operating System?

• Referee
  – Manage protection, isolation, and sharing of resources
    » Resource allocation and communication

• Illusionist
  – Provide clean, easy-to-use abstractions of physical resources
    » Infinite memory, dedicated machine
    » Higher level objects: files, users, messages
    » Masking limitations, virtualization

• Glue
  – Common services
    » Storage, Window system, Networking
    » Sharing, Authorization
    » Look and feel
• OS provides common services in the form of I/O
OS Basics: Look and Feel

Process: Execution environment with restricted rights provided by OS

Compiled Program
System Libs

Operating System

Hardware

Compiled Program
System Libs

Process: Execution environment with restricted rights provided by OS

Operating System

Hardware

Compiler

Crooks & Joseph CS162 © UCB Spring 2021
Lec 1.43
Process: Execution environment with restricted rights provided by OS

Compiled Program

System Libs

Operating System

Threads
Address Spaces
Files
Sockets
Windows

Network Manager

Power Manager

Compiled Program

System Libs

Processor

Memory

OS Mem

Storag e

Networks

Displays

Batter y

PGTbl & TLB

I/O Ctrlr

Compiler

OS Basics: Background Management
What is an Operating System?

- **Referee**
  - Manage protection, isolation, and sharing of resources
    » Resource allocation and communication

- **Illusionist**
  - Provide clean, easy-to-use abstractions of physical resources
    » Infinite memory, dedicated machine
    » Higher level objects: files, users, messages
    » Masking limitations, virtualization

- **Glue**
  - Common services
    » Storage, Window system, Networking
    » Sharing, Authorization
    » Look and feel
Why take CS162?

• Some of you will actually design and build operating systems or components of them.
  – Perhaps more now than ever
• Many of you will create systems that utilize the core concepts in operating systems.
  – Whether you build software or hardware
  – The concepts and design patterns appear at many levels
• All of you will build applications, etc. that utilize operating systems
  – The better you understand their design and implementation, the better use you’ll make of them.
Intros - Natacha Crooks

• Assistant Professor in the Data Science and Foundation Group, EECS
  – PhD from UT Austin in 2019
  – Started in 2020, no physical office yet …

• Research Areas
  – Distributed Systems, Databases, and Privacy-Preserving Systems
Intros - Anthony D. Joseph

• Chancellor’s Professor in Electrical Engineering and Computer Science
  – 465 Soda Hall (RISE Lab), http://www.cs.berkeley.edu/~adj/
  – Campus Cyber-Risk Responsible Executive, co-chair EECS dept IT committee

• Research areas:
  – Modin (drop-in Pandas replacement), Fog Robotics (edge computing), Cancer Genomics/Precision Medicine (ADAM/Apache Spark), Secure Machine Learning (SecML), DETER security testbed
  – Previous: Cloud computing (Apache Mesos), Peer-to-Peer networking (Tapestry), Mobile computing, Wireless/Cellular networking

• Outside Activities
  – Big Data and Apache Spark BerkeleyX MOOCs (’15/’16 >240k students with >11% finishing)
  – Unite Genomics co-founder (focused on rare diseases treatments)
CS162 TAs: Sections TBA

Akshat Gokhale  
(Head TA)

Alina Dan

William Hsu

Eleanor Cawthon

Kevin Svetlitski

Allan Yu
Enrollment

- Class has 310 limit
  - Unable to make class larger
- This is an Early Drop Deadline course (January 29th)
  - If you are not serious about taking, please drop early
  - Department will continue to admit students as other students drop
  - Really hard to drop afterwards!
    » Don’t forget to keep up with work if you are still on the waitlist!
- We are serious about requiring CAMERAs
  - Zoom proctoring of exams
  - Required sections, design reviews, interactions with your group (screen shots!)
  - This is part of keeping people from falling off into /dev/null in cyberspace!
- On the waitlist/Concurrent enrollment ??
  - Unfortunately, we maxed out sections and TA Support
  - If people drop, we can move others off waitlist
  - Concurrent enrollment is after the waitlist
CS162 in the age of COVID-19
CS162 in the age of COVID-19

• Well, things are considerably different this term!
  – Many lessons learned from the Spring/Summer/Fall semesters
  – Everything is remote – all term!

• Most important thing: People, Interactions, Collaboration
  – How do we recover collaboration without direct interaction?
  – Remember group meetings?

• Must Work to bring everyone along (virtually)!
  – Cameras are essential components of this class
    » Must have a camera and plan to turn it on
    » Will need it for exams, discussion sections, design reviews, OH
  – Need to bring back personal interaction – even if it is virtual
    » Humans not good at interacting text-only
    » Virtual coffee hours with your group (camera turned on!)
  – Required attendance at: Discussion sections, Design Reviews
    » With camera turned on!
Infrastructure, Textbook & Readings

- **Infrastructure**
  - Website: [http://cs162.eecs.berkeley.edu](http://cs162.eecs.berkeley.edu)
  - Piazza: [https://piazza.com/berkeley/spring2021/cs162](https://piazza.com/berkeley/spring2021/cs162)
  - Pre-recorded Lectures, Some Live Lectures (recordings available next day)

  - Suggested readings posted along with lectures
  - Try to keep up with material in book as well as lectures

- **Supplementary Material**
  - Operating Systems: Three Easy Pieces, by Remzi and Andrea Arpaci-Dusseau, available for free online
  - Linux Kernel Development, 3rd edition, by Robert Love

- **Online supplements**
  - See course website
  - Includes Appendices, sample problems, etc.
  - Networking, Databases, Software Engineering, Security
  - Some Research Papers!
Syllabus

• OS Concepts: How to Navigate as a Systems Programmer!
  – Process, I/O, Networks and Virtual Machines
• Concurrency
  – Threads, scheduling, locks, deadlock, scalability, fairness
• Address Space
  – Virtual memory, address translation, protection, sharing
• File Systems
  – I/O devices, file objects, storage, naming, caching, performance, paging, transactions, databases
• Distributed Systems
  – Protocols, N-Tiers, RPC, NFS, DHTs, Consistency, Scalability, multicast
• Reliability & Security
  – Fault tolerance, protection, security
• Cloud Infrastructure
Learning by Doing

• Individual Homeworks (2 weeks) - preliminary
  – 0. Tools & Environment, Autograding, recall C, executable
  – 1. Lists in C
  – 2. BYOS – build your own shell
  – 3. Memory allocation (MALLOC)
  – 4. Memory management
  – 5. Sockets & Threads in HTTP server

• Three (and ½) Group Projects
  – 0. Getting Started (Individual, before you have a group)
  – 1. User-programs (exec & syscall)
  – 2. Threads & Scheduling
  – 3. File Systems
Group Projects

• Project teams have 4 members!
  – Never 5, 3 requires serious justification
  – Must work in groups in “the real world”
  – Same section (at least same TA)

• Communication and cooperation will be essential
  – Regular meetings WITH CAMERA TURNED ON!
    » Extra credit for screen shots of all of you together in zoom with
      camera enabled
  – Design Documents
  – Slack/Messenger/whatever doesn’t replace face-to-face!

• Everyone should do work and have clear responsibilities
  – You will evaluate your teammates at the end of each project
  – Dividing up by Task is the worst approach. Work as a team.

• Communicate with supervisor (TAs)
  – What is the team’s plan?
  – What is each member’s responsibility?
  – Short progress reports are required
  – Design Documents: High-level description for a manager!
Getting started

• Time-zone Survey!
  – We need to know where you are so that we can plan section/midterm times
• Start homework 0 and Project 0 right away (hopefully Today!)
  – Github account
  – Registration survey
  – Vagrant virtualbox – VM environment for the course
    » Consistent, managed environment on your machine
  – Get familiar with all the cs162 tools
  – Submit to autograder via git
• Sections on Friday – attend any section you want
  – We’ll assign permanent sections after forming project groups
  – Section attendance will be mandatory after we form groups
  – These section times will be adjusted after we have a better idea where people are
Preparing Yourself for this Class

• The projects will require you to be very comfortable with programming and debugging C
  – Pointers (including function pointers, void*)
  – Memory Management (malloc, free, stack vs heap)
  – Debugging with GDB
• You will be working on a larger, more sophisticated code base than anything you've likely seen in 61C!
• Review Session on C/C++:
  – Monday, 1/25, Time and logistics TBA
• "Resources" page on course website
  – Ebooks on “git” and “C”
• C programming reference (still in beta):
  – https://cs162.eecs.berkeley.edu/ladder/
• First two sections are also dedicated to programming and debugging review:
  – Attend ANY sections in first two weeks
Grading (Tentative breakdown)

- 36% three midterms (12% each)
  - Thursday, 2/18, TBA, time set after survey
  - Thursday, 3/18, TBA, time set after survey
  - Thursday, 4/29, TBA, time set after survey
  - These will be ZOOM-Proctored. Camera REQUIRED.
- 36% projects
- 18% homework
- 10% participation (Sections, Lecture, …)
- No final exam
- Projects
  - Initial design document, Design review, Code, Final design
  - Submission via git push triggers autograder
Personal Integrity

• UCB Academic Honor Code: "As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others."

https://asuc.org/honor-code-landing/
CS 162 Collaboration Policy

- Explaining a concept to someone in another group
- Discussing algorithms/testing strategies with other groups
- Discussing debugging approaches with other groups
- Searching online for generic algorithms (e.g., hash table)

- Sharing code or test cases with another group
- Copying OR reading another group’s code or test cases
- Copying OR reading online code or test cases from prior years
- Helping someone in another group to debug their code

• We compare all project submissions against prior year submissions and online solutions and will take actions (described on the course overview page) against offenders
• Don’t put a friend in a bad position by asking for help that they shouldn’t give!
What makes Operating Systems Exciting and Challenging?
Societal Scale Information Systems
(Or the “Internet of Things”?)

- The world is a large distributed system
  - Microprocessors in everything
  - Vast infrastructure behind them

MEMS for Sensor Nets

Internet Connectivity

Scalable, Reliable, Secure Services

Databases
Information Collection
Remote Storage
Online Games
Commerce
...

Gigabit Ethernet
Massive Cluster
Clusters
Technology Trends: Moore’s Law

Gordon Moore (co-founder of Intel) predicted in 1965 that the transistor density of semiconductor chips would double roughly every 18 months.

2X transistors/Chip Every 1.5 years
Called “Moore’s Law”

Microprocessors have become smaller, denser, and more powerful.
Big Challenge: Slowdown in Joy’s law of Performance

- VAX : 25%/year 1978 to 1986
- RISC + x86 : 52%/year 1986 to 2002
- RISC + x86 : ??%/year 2002 to present

⇒ Sea change in chip design: multiple "cores" or processors per chip

Another Challenge: Power Density

• Moore’s law extrapolation
  – Potential power density reaching amazing levels!
• Flip side: battery life very important
  – Moore’s law yielded more functionality at equivalent (or less) total energy consumption
ManyCore Chips: The future arrived in 2007

• Intel 80-core multicore chip (Feb 2007)
  – 80 simple cores
  – Two FP-engines / core
  – Mesh-like network
  – 100 million transistors
  – 65nm feature size
• Intel Single-Chip Cloud Computer (August 2010)
  – 24 “tiles” with two cores/tile
  – 24-router mesh network
  – 4 DDR3 memory controllers
  – Hardware support for message-passing

• How to program these?
  – Use 2 CPUs for video/audio
  – Use 1 for word processor, 1 for browser
  – 76 for virus checking???
• Parallelism must be exploited at all levels
• Amazon X1 instances (2016)
  – 128 virtual cores, 2 TB RAM
But then Moore’s Law Ended…

- Moore’s Law has (officially) ended -- Feb 2016
  - No longer getting 2 x transistors/chip every 18 months…
  - or even every 24 months
- May have only 2-3 smallest geometry fabrication plants left:
  - Intel and Samsung and/or TSMC
- Vendors moving to 3D stacked chips
  - More layers in old geometries
Storage Capacity is Still Growing!
Society is Increasingly Connected…
Network Capacity Still Increasing

In 2011, smartphone shipments exceeded PC shipments!

2011 shipments:
- 487M smartphones
- 414M PC clients
  - 210M notebooks
  - 112M desktops
  - 63M tablets
- 25M smart TVs

4 billion phones in the world smartphones over next few years

Then…
People-to-Computer Ratio Over Time

Bell’s Law: new computer class per 10 years

- Mainframe
- Mini
- Workstation
- PC
- Laptop
- PDA
- Cell
- Mote!

Computers Per Person

1:1
10^3:1
1:10^3
1:10^6

The Internet of Things!

Number crunching, Data Storage, Massive Inet Services, ML, ...
Productivity, Interactive
Streaming from/to the physical world
What is an Operating System?
What is an Operating System Again?

• **Referee**
  - Manage sharing of resources, Protection, Isolation
    » Resource allocation, isolation, communication

• **Illusionist**
  - Provide clean, easy to use abstractions of physical resources
    » Infinite memory, dedicated machine
    » Higher level objects: files, users, messages
    » Masking limitations, virtualization

**Glue**
- Common services
  » Storage, Window system, Networking
  » Sharing, Authorization
  » Look and feel
Challenge: Complexity

- Applications consisting of...
  - ... a variety of software modules that ...
  - ... run on a variety of devices (machines) that
    » ... implement different hardware architectures
    » ... run competing applications
    » ... fail in unexpected ways
    » ... can be under a variety of attacks

- Not feasible to test software for all possible environments and combinations of components and devices
  - The question is not whether there are bugs but how serious are the bugs!

- Up to 28 Cores, 56 Threads
  - 694 mm\(^2\) die size (estimated)
- Many different instructions
  - Security, Graphics
- Caches on chip:
  - L2: 28 MiB
  - Shared L3: 38.5 MiB (non-inclusive)
  - Directory-based cache coherence
- Network:
  - On-chip Mesh Interconnect
  - Fast off-chip network directly supports 8-chips connected
- DRAM/chips
  - Up to 1.5 TiB
  - DDR4 memory
HW Functionality comes with great complexity!

Intel Skylake-X
I/O Configuration

Proc
Caches
Memory

Busses

I/O Devices:
- Controllers
- Disks
- Displays
- Keyboards
- Networks

Intel X299 Chipset

Up to 44 x PCI Express® 3.0
Up to 24 x PCI Express® 3.0
8 x SATA Ports, eSATA; Port Disable
Up to 10 x USB 3.0 Ports 14 x USB 2.0 Ports XHCI; USB Port Disable
Integrated 10/100/1000 MAC
Intel® Ethernet Connection

Optional

Intel® High Definition Audio
Intel® Rapid Storage Technology for PCI Express® Storage
Intel® Rapid Storage Technology with RAID
Intel® Smart Connect Technology
Intel® Extreme Tuning Utility Support
Increasing Software Complexity

Millions of Lines of Code
(source https://informationisbeautiful.net/visualizations/million-lines-of-code/)

I/19/21  Crooks & Joseph CS162 © UCB Spring 2021  Lec 1.83
Example: Some Mars Rover (“Pathfinder”) Requirements

• Pathfinder hardware limitations/complexity:
  – 20Mhz processor, 128MB of DRAM, VxWorks OS
  – cameras, scientific instruments, batteries, solar panels, and locomotion equipment
  – Many independent processes work together

• Can’t hit reset button very easily!
  – Must reboot itself if necessary
  – Must always be able to receive commands from Earth

• Individual Programs must not interfere
  – Suppose the MUT (Martian Universal Translator Module) buggy
  – Better not crash antenna positioning software!

• Further, all software may crash occasionally
  – Automatic restart with diagnostics sent to Earth
  – Periodic checkpoint of results saved?

• Certain functions time critical:
  – Need to stop before hitting something
  – Must track orbit of Earth for communication

• A lot of similarity with the Internet of Things?
  – Complexity, QoS, Inaccessibility, Power limitations … ?
Questions

• Does the programmer need to write a single program that performs many independent activities?
• Does every program have to be altered for every piece of hardware?
• Does a faulty program crash everything?
• Does every program have access to all hardware?

Hopefully, no!

Operating Systems help the programmer write robust programs!
**OS Abstracts the Underlying Hardware**

- Processor → Thread
- Memory → Address Space
- Disks, SSDs, … → Files
- Networks → Sockets
- Machines → Processes

**Application**  
**Abstract Machine Interface**

**Operating System**  
**Physical Machine Interface**

**Hardware**

**OS as an Illusionist:**
- Remove software/hardware quirks (**fight complexity**)
- Optimize for convenience, utilization, reliability, … (**help the programmer**)

**For any OS area (e.g. file systems, virtual memory, networking, scheduling):**
- What hardware interface to handle? (physical reality)
- What’s software interface to provide? (nicer abstraction)
OS Protects Processes and the Kernel

- Run multiple applications and:
  - Keep them from interfering with or crashing the operating system
  - Keep them from interfering with or crashing each other
Basic Tool: Dual-Mode Operation

- Hardware provides at least two modes:
  1. Kernel Mode (or “supervisor” mode)
  2. User Mode
- Certain operations are **prohibited** when running in user mode
  - Changing the page table pointer, disabling interrupts, interacting directly with hardware, writing to kernel memory
- Carefully controlled transitions between user mode and kernel mode
  - System calls, interrupts, exceptions
UNIX System Structure

User Mode
- Applications (the users)
- Standard Libraries
  - shells and commands
  - compilers and interpreters
  - system libraries

Kernel Mode
- Kernel
  - system-call interface to the kernel
    - signals
    - terminal handling
    - character I/O system
    - terminal drivers
  - file system
    - swapping block I/O
    - system
    - disk and tape drivers
  - CPU scheduling
    - page replacement
    - demand paging
    - virtual memory

Hardware
- terminal controllers
- terminals
- device controllers
- disks and tapes
- memory controllers
- physical memory
Virtualization: Execution Environments for Systems

Additional layers of protection and isolation can help further manage complexity
What is an Operating System, … Really?

• Most Likely:
  – Memory Management
  – I/O Management
  – CPU Scheduling
  – Communications? (Does Email belong in OS?)
  – Multitasking/multiprogramming?

• What about?
  – File System?
  – Multimedia Support?
  – User Interface?
  – Internet Browser? 😊

• Is this only interesting to Academics??
Operating System Definition (Cont.)

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is a good approximation
  - But varies wildly
- “The one program running at all times on the computer” is the kernel
  - Everything else is either a system program (ships with the operating system) or an application program
“In conclusion…Operating Systems:”

• Provide convenient abstractions to handle diverse hardware
  – Convenience, protection, reliability obtained in creating the illusion

• Coordinate resources and protect users from each other
  – Using a few critical hardware mechanisms

• Simplify application development by providing standard services

• Provide fault containment, fault tolerance, and fault recovery

• CS162 combines things from many other areas of computer science:
  – Languages, data structures, hardware, and algorithms