


TCSS 422: OPERATING SYSTEMS

INTRODUCTION



Wes J. Lloyd
School of Engineering and Technology
University of Washington - Tacoma

September 30, 2021 TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington Tacoma

1

W Where are you joining us from? (WORLD VERSION)



Start the presentation to see live content. For screen share software, share the entire screen. Get help at polllev.com/app

2

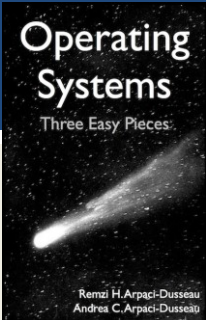
**INTRODUCTIONS: What is your name? nickname
W / alias? and list one or more areas of interest in
Computer Science:**

Start the presentation to see live content. For screen share software, share the entire screen. Get help at polllev.com/app

3

OBJECTIVES – 9/30

- **Syllabus, Course Introduction**
- C Review Survey
- Background Survey
- **Chapter 2: Operating Systems – Three Easy Pieces**
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

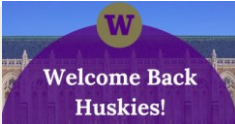


September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.4
--------------------	---	------


4

TCSS 422 – Fall 2021

- **Fall 2021:**
 - **In-person course**
 - Up to 40% of the course can be online without reclassification as a hybrid (H) or distance learning (D) course in response to covid challenges
 - All lectures streamed LIVE via Zoom and recorded for 24/7 availability
 - If not feeling well, please join Zoom live stream
 - In-class activities can be submitted asynchronously, online as needed
 - **Masks are required at all times in the classroom**
 - What is a mask?
 - <https://www.tacoma.uw.edu/sites/default/files/2021-09/fgfacecovering21.pdf>
- 19 class meetings
 - 2 Thursday holidays: Nov 11 and 25
- Midterm ~ Thursday Nov 4th
- Final exam Tuesday Dec 14th



TCSS 422
FALL 2021



L1.5

5

RESOURCES

- Textbook coupon 10% off “BUY10” until Friday at 11:59pm
- <http://www.lulu.com/shop/remzi-arpaci-dusseau-and-andrea-arpaci-dusseau/operating-systems-three-easy-pieces-softcover-version-100/paperback/product-23779877.html>

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.6
--------------------	---	------

6

TCSS422 – FALL 2021 COMPUTER OPERATING SYSTEMS

- Syllabus
- Grading
- Schedule
- Assignments

See website at:

<http://faculty.washington.edu/wlloyd/courses/tcss422>

Website also integrated into Canvas

Enables access using mobile device w/o logging into Canvas

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.7

7

TCS422 COURSE WORK

- **Assignments (45%)**
 - 4 Assignments: roughly every two weeks
 - Submit ALL programming assignments via Canvas
 - Please do not email submissions – they are prone to be lost
 - If Canvas has closed, please request it be reopened...
- **Tutorials/Quizzes/In-class activities (15%)**
 - ~ 6 - 9 total items
 - Drop lowest two
 - Variety of formats: collaborative in class (*via Zoom breakout rooms*), online, reading, tutorial
- **Exams: Midterm and Final (40%)**
 - In class on Nov 4 and Dec 14
 - Final exam is comprehensive, with emphasis on new material

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

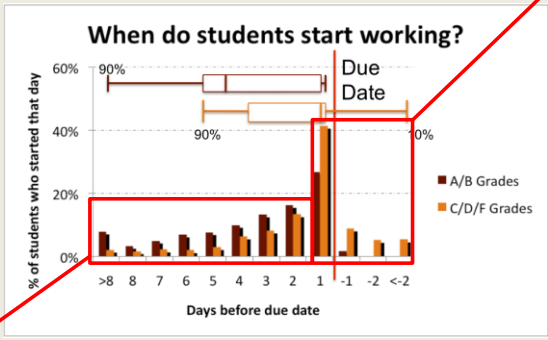
L1.8

8

TCSS 422: PROGRAM DUE DATES

- Programs - please start early:

Less than 50% chance of A/B



Better than 50% chance of A/B

From Virginia Tech Department of Computer Science - 2011

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.9
--------------------	---	------

TCSS 422: PROGRAMS

- Tentative - subject to change
- Assignment 0:**
Introduction to Linux, Ubuntu Virtual Machine
- Assignment 1:**
Programming with multiple processes (in C)
- Assignment 2:**
Multithreaded programming and concurrency (C or Java)
- Assignment 3:**
Kernel (real) mode programming (in C)

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.10
--------------------	---	-------

TCSS 422: PROGRAM DUE DATES

■ Programs - please start early

- Work as if deadline is several days earlier
- Allows for a “buffer” for running into unexpected problems
 - Underestimation of the task at hand
 - Allows time to seek C help from CSS lab mentors (*checking on availability for Fall 2021*)
 - If less familiar with C/pointers (TCSS 333/380), **BUDGET MORE TIME**

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.11

11

UBUNTU 20.04 – VIRTUAL MACHINE

■ Ubuntu 20.04

- Open source version of Debian-package based Linux
- Package management: “apt get” repositories
 - See: <https://packages.ubuntu.com/>

■ Ubuntu Advantages

- Enterprise Linux Distribution
- Free, widely used by developers
- Long term releases (LTS) every 2 years, good for servers
- 6-month feature releases, good for sharing new features with the community

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.12

12

UBUNTU 20.04 – VIRTUAL MACHINE INSTALLATION

- Ubuntu 20.04 on Oracle VirtualBox
- HOW-TO installation videos:
 - Windows 10
 - <https://youtu.be/x3Zpe1rIPFE>
 - Mac OS X
 - <https://youtu.be/Hzji7w8820Y>
- > AFTER VirtualBox, INSTALL THE **Guest Additions**
 - **IMPORTANT USABILITY ADD-ON:** Provides file system sharing, clipboard integration, mouse tricks
- <https://youtu.be/Kbez-XdXqrw>

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.13

13

C PROGRAMING IN TCSS 422

- Many OSes are coded primarily in C and Assembly Language
- C is a particularly useful language for working with hardware / hardware drivers and operating systems
- C allows writing programs that can directly access the computer's physical memory (in kernel/real mode) providing nearly the power and speed of assembly language
 - *But in a much easier to write high-level language*
- Ideally, all university operating system courses are taught in C/C++. Our textbook is in C/C++
 - *This quarter we will offer the option of assignment of completing assignment 2 in Java (multithreaded programming)*

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.14

14

C MENTORING

- <https://www.tacoma.uw.edu/set/students/mentors>
- School of Engineering and Technology Mentors
- Office hours held online via Zoom (in person?)
- Varied hours and availability based on mentors schedules
- Monday – Thursday: ~9:30 am – 9:00 pm
- Friday: ~ 9:30 - 3:30 pm
- Fall quarter hours will be posted once available

- Student mentors managed by SET's Monika Sobolewska

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.15
--------------------	---	-------

15

INSTRUCTOR HELP

- Office hours: tentative 5:40p TR after class
 - Additional hours based on survey results
 - Also available by appointment

- Take **ownership** of your educational outcome
 - ~10 weeks in TCSS 422 is very small relative to entire IT career
 - Make the most of this **limited** opportunity
 - Maximize your educational investment
 - ***** Ask questions in class *****
 - Also questions after class, email, Canvas discussion boards
 - Seek help using UWT resources, the Internet, YouTube videos (video.google.com) and online tutorials

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.16
--------------------	---	-------

16

CLASS PARTICIPATION

- **Questions and discussion are strongly encouraged**
 - Leverage your educational investment
 - All questions are encouraged!
 - This instructor appreciates questions at all levels
 - there is no judgement for any question
- **Daily feedback surveys**
 - How much is new vs. review?
 - Checking the pace...
 - What is unclear? It's helpful to know when topics are not clear
 - Use the survey to write questions and feedback that come to you during the lecture
- **Poll-EV**

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.17
--------------------	---	-------

17

OBJECTIVES – 9/30


- Syllabus, Course Introduction
- **C Review Survey**
- Background Survey
- **Chapter 2: Operating Systems – Three Easy Pieces**
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.18
--------------------	---	-------

18

C REVIEW SURVEY

QUIZ 0 – IN CANVAS



September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.19

19

OBJECTIVES – 9/30

- Syllabus, Course Introduction
- C Review Survey
- **Background Survey**
- Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma


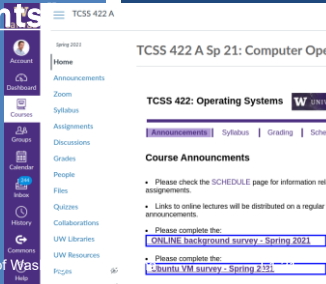
L1.20

20

PLEASE COMPLETE BACKGROUND & VM SURVEYS

SEE LINKS AT:
<http://faculty.washington.edu/wlloyd/courses/tcss422/announcements.html>

or in Canvas under “Announcements”
we will resume at ~tba pm

September 30, 2021 TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington

21

WE WILL RETURN AT 2:40PM



September 30, 2021 TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - coma L1.22

22

STUDENT BACKGROUND SURVEY

- Please complete the Student Background Survey
- <https://forms.gle/anQ2BSANhm5yQ4Wy9>

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L2.23

23

VIRTUAL MACHINE SURVEY

- Please complete the Virtual Machine Survey to request a “School of Engineering and Technology” remote hosted Ubuntu VM
- <https://forms.gle/V2sg4iW1awvhFx4W8>

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L2.24

24

OBJECTIVES – 9/30

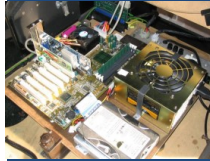
- Syllabus, Course Introduction
- C Review Survey
- Background Survey

- Chapter 2: Operating Systems – Three Easy Pieces
 - **Introduction to operating systems**
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.25
--------------------	---	-------

25

INTRODUCTION TO OPERATING SYSTEMS



September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.26
--------------------	---	-------

26

OBJECTIVES

- **Chapter 2: Operating Systems – Three Easy Pieces**
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - **THREE EASY PIECES:**
 - Virtualizing the CPU
 - Virtualizing Memory
 - Virtualizing I/O
 - Operating system design goals

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.27

27

OPERATING SYSTEMS

- Responsible for:
 - Making it easy to **run** programs
 - Allowing programs to **share** memory
 - Enabling programs to **interact** with devices

OS is in charge of making sure the system operates correctly and efficiently.

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.28

28

OBJECTIVES – 9/30

- Syllabus, Course Introduction
- C Review Survey
- Background Survey

- Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - **Management of resources**
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.29
--------------------	---	-------

29

RESOURCE MANAGEMENT

- The OS is a resource manager
- Manages CPU, disk, network I/O
- Enables many programs to
 - **Share the CPU**
 - **Share the underlying physical memory (RAM)**
 - **Share physical devices**
 - Disks
 - Network Devices
 - ...

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.30
--------------------	---	-------

30

OBJECTIVES – 9/30

- Syllabus, Course Introduction
- C Review Survey
- Background Survey

- Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - **Concepts of virtualization/abstraction**
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.31
--------------------	---	-------

31

VIRTUALIZATION

- Operating systems present **physical resources** as **virtual representations** to the programs sharing them
 - Physical resources: CPU, disk, memory, ...
 - The virtual form is “**abstract**”
 - The OS presents an illusion that each user program runs in isolation on its own hardware
 - This virtual form is general, powerful, and easy-to-use

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.32
--------------------	---	-------

32

ABSTRACTIONS

- What form of abstraction does the OS provide?
 - CPU
 - Process and/or thread
 - Memory
 - Address space
 - → large array of bytes
 - All programs see the same “size” of RAM
 - Disk
 - Files

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.33

33

WHY ABSTRACTION?

- Allow applications to reuse common facilities
- Make different devices look the same
 - Easier to write common code to use devices
 - Linux/Unix Block Devices
- Provide higher level abstractions
- More useful functionality

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L1.34

34

ABSTRACTION CHALLENGES

- What level of abstraction?
 - How much of the underlying hardware should be exposed?
 - What if **too much**?
 - What if **too little**?
- What are the correct abstractions?
 - Security concerns

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.35
--------------------	---	-------

35

OBJECTIVES – 9/30

- Syllabus, Course Introduction
- C Review Survey
- Background Survey
- Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - Operating system design goals

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.36
--------------------	---	-------

36

VIRTUALIZING THE CPU

- Each running program gets its own “virtual” representation of the CPU
- Many programs seem to run at once
- Linux: “top” command shows process list
- Windows: task manager

```
top - 18:25:07 up 450 days, 1:03, 3 users, load average: 0.32, 0.28, 0.29
tasks: 656 total, 1 running, 633 sleeping, 0 stopped, 0 zombie
CPU(s):  0.0%us,  5.0%sy,  94.9%id,  0.0%wa,  0.0%st,  0.0%ni,  0.0%ot
Mem:  7425772K total,  749408K used,  6676364 free,  54516K buffers
Swap:  2183372K total,  72224K used,  21111476K free,  528336K cached
```

PID	PPID	UID	PGID	NI	TY	SP	ST	VSZ	RSS	%CPU	%MEM	TIME	COMMAND
812	rcscljgpt	20	0	6006	276	Mon	S	152	0	0	0	4:12.19	postgres
813	rcscljgpt	20	0	6006	276	Mon	S	112	0	0	0	4:12.19	postgres
8170	rcscljgpt	20	0	6006	276	Mon	S	112	0	0	0	4:12.19	postgres
4480	aravali	20	0	15432	1712	S28	R	0	0	0	0	0:00.52	top
4206	rcscljgpt	20	0	6006	186	Mon	S	0	0	0	0	0:27.78	postgres
7880	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	4:21.48	postgres
8126	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	4:21.48	postgres
1620	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:31.79	postgres
1107	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:40.46	postgres
1970	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:46.79	postgres
1718	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:10.19	postgres
1620	rcscljgpt	20	0	6006	186	Mon	S	0	0	0	0	0:20.28	postgres
1717	rcscljgpt	20	0	6006	186	Mon	S	0	0	0	0	0:24.42	postgres
20	rcscljgpt	20	0	0	0	0	S	0	0	0	0	0:00.00	rcscljgpt
1000	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:04.54	postgres
8126	rcscljgpt	20	0	6006	186	Mon	S	0	0	0	0	0:10.19	postgres
812	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:11.73	postgres
7006	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	4:31.84	postgres
7006	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	4:32.16	postgres
8126	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	4:32.48	postgres
8126	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	4:32.48	postgres
1294	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	3:51.22	postgres
1420	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	1:36.50	postgres
1170	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:38.00	postgres
1412	rcscljgpt	20	0	6006	186	Mon	S	0	0	0	0	0:20.46	postgres
1420	rcscljgpt	20	0	6006	186	Mon	S	0	0	0	0	0:20.46	postgres
1620	rcscljgpt	20	0	28916	1526	Mon	S	0	0	0	0	1:00.42	htop
1770	rcscljgpt	20	0	24342	8234	Mon	S	0	0	0	0	0:29:52.22	Java
1076	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:22.79	postgres
1130	rcscljgpt	20	0	6006	276	Mon	S	0	0	0	0	0:22.22	postgres
1	rcscljgpt	20	0	19100	2280	Mon	S	0	0	0	0	0:01:01.00	rcscljgpt
2	rcscljgpt	20	0	0	0	0	S	0	0	0	0	0:00:00.00	rcscljgpt
4	rcscljgpt	20	0	0	0	0	S	0	0	0	0	79:08:28	rcscljgpt
3	rcscljgpt	20	0	0	0	0	S	0	0	0	0	0:00:00.00	rcscljgpt
6	rcscljgpt	20	0	0	0	0	S	0	0	0	0	1:07:42	rcscljgpt
8	rcscljgpt	20	0	0	0	0	S	0	0	0	0	1:04:10	rcscljgpt
9	rcscljgpt	20	0	0	0	0	S	0	0	0	0	0:00:00.00	rcscljgpt
10	rcscljgpt	20	0	0	0	0	S	0	0	0	0	22:22:22	rcscljgpt
10	rcscljgpt	20	0	0	0	0	S	0	0	0	0	0:00:00.00	rcscljgpt
11	rcscljgpt	20	0	0	0	0	S	0	0	0	0	1:00:01:04	rcscljgpt
12	rcscljgpt	20	0	0	0	0	S	0	0	0	0	0:00:00.00	rcscljgpt

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
 School of Engineering and Technology, University of Washington - Tacoma

L1.37

37

VIRTUALIZING THE CPU - 2

- Simple Looping C Program (simpleloop.c)

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <sys/time.h>
4  #include <assert.h>
5  #include "common.h"
6
7  int
8  main(int argc, char *argv[])
9  {
10     if (argc != 2) {
11         fprintf(stderr, "usage: cpu <string>\n");
12         exit(1);
13     }
14     char *str = argv[1];
15     while (1) {
16         Spin(1); // Repeatedly checks the time and
17                 // returns once it has run for a second
18         printf("%s\n", str);
19     }
20     return 0;
    
```

September 30, 2021

TCSS422: Operating Systems [Fall 2021]
 School of Engineering and Technology, University of Washington - Tacoma

L1.38

38

VIRTUALIZING THE CPU - 3

```
prompt> gcc -o cpu cpu.c -Wall
prompt> ./cpu "A"
A
A
A
^C
prompt>
```

- **simpleloop.c**
- **Runs forever, must Ctrl-C to halt...**

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.39
--------------------	---	-------

39

VIRTUALIZATION THE CPU - 4

```
prompt> ./cpu A & ; ./cpu B & ; ./cpu C & ; ./cpu D &
[1] 7353
[2] 7354
[3] 7355
[4] 7356
A
B
D
C
A
B
D
C
A
C
B
D
...
```

Even though we have only one processor, all four instances of our program seem to be running at the same time!

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.40
--------------------	---	-------

40

OBJECTIVES – 9/30

- Syllabus, Course Introduction
 - C Review Survey
 - Background Survey

 - Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - **Three Easy Pieces: CPU, Memory, I/O**
 - Concurrency
 - Operating system design goals
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L1.41 |
|--------------------|---|-------|

41

VIRTUALIZING MEMORY

- Computer memory is treated as a large array of bytes
 - Programs store all data in this large array
 - **Read memory (load)**
 - Specify an address to read data from
 - **Write memory (store)**
 - Specify data to write to an address
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L1.42 |
|--------------------|---|-------|

42

VIRTUALIZING MEMORY - 2

■ Program to read/write memory: (**mem.c**) (from ch. 2 pgs. 5-6)

```
1  #include <unistd.h>
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include "common.h"
5
6  int
7  main(int argc, char *argv[])
8  {
9      int *p = malloc(sizeof(int)); // a1: allocate some
                                   // memory
10     assert(p != NULL);
11     printf("(%d) address of p: %08x\n",
12           getpid(), (unsigned) p); // a2: print out the
                                   // address of the memmory
13     *p = 0; // a3: put zero into the first slot of the memory
14     while (1) {
15         Spin(1);
16         *p = *p + 1;
17         printf("(%d) p: %d\n", getpid(), *p); // a4
18     }
19     return 0;
20 }
```

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.43
--------------------	---	-------

43

VIRTUALIZING MEMORY - 3

■ Output of **mem.c** (example from ch. 2 pgs. 5-6)

```
prompt> ./mem
(2134) memory address of p: 00200000
(2134) p: 1
(2134) p: 2
(2134) p: 3
(2134) p: 4
(2134) p: 5
^C
```

- int value stored at virtual address 00200000
- program increments int value pointed to by p

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.44
--------------------	---	-------

44

VIRTUALIZING MEMORY - 4

- Multiple instances of **mem.c**

This example no longer works as advertised !
Ubuntu has been updated.
The ptr location is no longer identical. This was considered a security issue.

```
prompt> ./mem & ./mem &
[1] 24113
[2] 24114
(24113) memory address of p: 00200000
(24114) memory address of p: 00200000
(24113) p: 1
(24114) p: 1
(24114) p: 2
(24113) p: 2
(24113) p: 3
(24114) p: 3
...
```

- IN THE BOOK: (int*)p appears to have the same memory location **00200000**
- Why does modifying the value of *p in program #1 (PID 24113), not interfere with the value of *p in program #2 (PID 24114) ?
 - The OS has “virtualized” memory, and provides a “virtual” address

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.45
--------------------	---	-------

45

VIRTUAL MEMORY

- Key take-aways:
 - Each process (program) has its own **virtual address space**
 - The OS maps virtual **address spaces** onto **physical memory**
 - A memory reference from one process can not affect the address space of others.
 - **Isolation**
 - Physical memory, a shared resource, is managed by the OS

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.46
--------------------	---	-------

46

OBJECTIVES – 9/30

- Syllabus, Course Introduction
 - C Review Survey
 - Background Survey

 - Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - **Three Easy Pieces: CPU, Memory, I/O**
 - Concurrency
 - Operating system design goals
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L1.47 |
|--------------------|---|-------|

47

PERSISTENCE

- DRAM: Dynamic Random Access Memory: DIMMs/SIMMs
 - Stores data while power is present
 - When power is lost, data is lost (*volatile*)

 - Operating System helps “persist” data more permanently
 - I/O device(s): hard disk drive (HDD), solid state drive (SSD)
 - File system(s): “catalog” data for storage and retrieval
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L2.48 |
|--------------------|---|-------|

48

PERSISTENCE - 2

```
1  #include <stdio.h>
2  #include <unistd.h>
3  #include <assert.h>
4  #include <fcntl.h>
5  #include <sys/types.h>
6
7  int
8  main(int argc, char *argv[])
9  {
10     int fd = open("/tmp/file", O_WRONLY | O_CREAT
11                | O_TRUNC, S_IRWXU);
12     assert(fd > -1);
13     int rc = write(fd, "hello world\n", 13);
14     assert(rc == 13);
15     close(fd);
16     return 0;
}
```

- `open()`, `write()`, `close()`: OS **system calls** for device I/O
- Note: man page for `open()`, `write()` requires page number: "man 2 `open`", "man 2 `write`", "man `close`"

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L2.49
--------------------	---	-------

49

PERSISTENCE - 3

- To write to disk, OS must:
 - Determine where on disk data should reside
 - Perform sys calls to perform I/O:
 - Read/write to file system (*inode record*)
 - Read/write data to file
- OS provides fault tolerance for system crashes
 - Journaling: Record disk operations in a journal for replay
 - Copy-on-write: replicate shared data across multiple disks - see *ZFS filesystem*
 - Carefully order writes on disk (*especially spindle drives*)

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L2.50
--------------------	---	-------

50

OBJECTIVES – 9/30

- Syllabus, Course Introduction
- C Review Survey
- Background Survey

- Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - **Concurrency**
 - Operating system design goals

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.51
--------------------	---	-------

51

CONCURRENCY

Linux htop (Ubuntu)

The htop screenshot shows a list of processes. The process with PID 17759 is highlighted in yellow, showing it is using 1.4% of the CPU. The command for this process is 'htop'.

Windows 10 Task Manager

The Windows Task Manager screenshot shows the Performance tab with CPU usage at 9%. The system is identified as '11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz'.

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.52
--------------------	---	-------

52

CONCURRENCY

- Linux: 179 processes, 1089 threads (**htop**)
 - Windows 10: 364 processes, 6011 threads (task mgr)

 - OSes appear to run many programs at once, juggling them

 - Modern **multi-threaded** programs feature concurrent threads and processes

 - **What is a key difference between a process and a thread?**
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L1.53 |
|--------------------|---|-------|

53

CONCURRENCY - 2

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include "common.h"
4
5  volatile int counter = 0;
6  int loops;
7
8  void
9
10
11
12
13
14 }
15 ...
```

Not the same as Java volatile:
Provides a compiler hint than an object may change value unexpectedly (in this case by a separate thread) so aggressive optimization must be avoided.

pthread.c

Listing continues ...

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.54
--------------------	---	-------

54

CONCURRENCY - 3

```

16     int
17     main(int argc, char *argv[])
18     {
19         if (argc != 2) {
20             fprintf(stderr, "usage: threads <value>\n");
21             exit(1);
22         }
23         loops = atoi(argv[1]);
24         pthread_t p1, p2;
25         printf("Initial value : %d\n", counter);
26
27         Pthread_create(&p1, NULL, worker, NULL);
28         Pthread_create(&p2, NULL, worker, NULL);
29         Pthread_join(p1, NULL);
30         Pthread_join(p2, NULL);
31         printf("Final value : %d\n", counter);
32         return 0;
33     }
    
```

pthread.c

- Program creates two threads
- Check documentation: “man pthread_create”
- worker() method counts from 0 to argv[1] (loop)

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.55
--------------------	---	-------

55

Linux
“man”
page

example

PTHREAD_CREATE(3)
Linux Programmer's Manual
PTHREAD_CREATE(3)

NAME top

pthread_create - create a new thread

SYNOPSIS top

```
#include <pthread.h>

int pthread_create(pthread_t *thread, const pthread_attr_t *attr,
void *(*start_routine) (void *), void *arg);
```

Compile and link with `-pthread`.

DESCRIPTION top

The `pthread_create()` function starts a new thread in the calling process. The new thread starts execution by invoking `start_routine()`; `arg` is passed as the sole argument of `start_routine()`.

The new thread terminates in one of the following ways:

- * It calls `pthread_exit(3)`, specifying an exit status value that is available to another thread in the same process that calls `pthread_join(3)`.
- * It returns from `start_routine()`. This is equivalent to calling `pthread_exit(3)` with the value supplied in the `return` statement.
- * It is canceled (see `pthread_cancel(3)`).
- * Any of the threads in the process calls `exit(3)`, or the main thread performs a return from `main()`. This causes the termination of all threads in the process.

The `attr` argument points to a `pthread_attr_t` structure whose contents are used at thread creation time to determine attributes for the new thread; this structure is initialized using `pthread_attr_init(3)` and related functions. If `attr` is NULL, then the thread is created with default attributes.

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.56
--------------------	---	-------

56

CONCURRENCY - 4

- Command line parameter `argv[1]` provides loop length
- Defines number of times the shared counter is incremented
- Loops: 1000

```
prompt> gcc -o pthread pthread.c -Wall -pthread
prompt> ./pthread 1000
Initial value : 0
Final value : 2000
```

- Loops 100000

```
prompt> ./pthread 100000
Initial value : 0
Final value : 143012 // huh??
prompt> ./pthread 100000
Initial value : 0
Final value : 137298 // what ???
```



57

CONCURRENCY - 5

- When loop value is large why do we not achieve 200,000 ?
- C code is translated to (3) assembly code operations
 1. Load counter variable into register
 2. Increment it
 3. Store the register value back in memory
- These instructions happen concurrently and VERY FAST
- (P1 || P2) write incremented register values back to memory, While (P1 || P2) read same memory
- Memory access here is **unsynchronized (non-atomic)**
- *Some of the increments are lost*

58

W To perform parallel work, a single process may:

A	B	C	D
Launch multiple threads to execute code in parallel while sharing global data in memory	Launch multiple processes to execute code in parallel while sharing global data in memory	Both A and B	None of the above

Start the presentation to see live content. For screen share software, share the entire screen. Get help at pollev.com/app L2.59

59

PARALLEL PROGRAMMING

- To perform parallel work, a single process may:
- A. Launch multiple threads to execute code in parallel while sharing global data in memory
- B. Launch multiple processes to execute code in parallel without sharing global data in memory
- C. Both A and B
- D. None of the above

September 30, 2021	TCSS422: Operating Systems [Fall 2021] School of Engineering and Technology, University of Washington - Tacoma	L1.60
--------------------	---	-------

60

OBJECTIVES – 9/30

- Syllabus, Course Introduction
 - C Review Survey
 - Background Survey

 - Chapter 2: Operating Systems – Three Easy Pieces
 - Introduction to operating systems
 - Management of resources
 - Concepts of virtualization/abstraction
 - Three Easy Pieces: CPU, Memory, I/O
 - Concurrency
 - **Operating system design goals**
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L1.61 |
|--------------------|---|-------|

61

SUMMARY: OPERATING SYSTEM DESIGN GOALS

- **ABSTRACTING THE HARDWARE**
 - Makes programming code easier to write
 - Automate sharing resources – save programmer burden

 - **PROVIDE HIGH PERFORMANCE**
 - Minimize overhead from OS abstraction (Virtualization of CPU, RAM, I/O)
 - Share resources fairly
 - Attempt to tradeoff performance vs. fairness → consider priority

 - **PROVIDE ISOLATION**
 - User programs can't interfere with each other's virtual machines, the underlying OS, or the sharing of resources
- | | | |
|--------------------|---|-------|
| September 30, 2021 | TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma | L2.62 |
|--------------------|---|-------|

62

SUMMARY: OPERATING SYSTEM DESIGN GOALS - 2

■ RELIABILITY

- OS must not crash, 24/7 Up-time
- Poor user programs must not bring down the system:

Blue Screen

■ Other Issues:

- Energy-efficiency
- Security (of data)
- Cloud: Virtual Machines



September 30, 2021

TCSS422: Operating Systems [Fall 2021]
School of Engineering and Technology, University of Washington - Tacoma

L2.63

63

QUESTIONS



64

QUESTIONS



65