Operating Systems

CS219 + CS236

Spring 2025-26

Puru

www.cse.iitb.ac.in/~puru

www.cse.iitb.ac.in/synerg

Who is who?

Puru

Varsha

Puru is absconding!

Varsha has kindly agreed to substitute for first-class

People

- Instructor
  - **Puru** (office: SIA 304, KR Bldg)
    www.cse.iitb.ac.in/~puru
  - office hours
    walk-in
    wed: 2.30 pm - 3.30 pm

- TAs and Course Logistics support
  - Debojeet, Revathy, Monil, Prince, Soham, Soham, Ameer, Rajas, Abhishek
    Arghyadip, Gaurav, Mithilkumar, Pratik Tadvi, Devang, Yashvardhan
    Vishwajit, Kaushal, Awez
  - Firuza

Course Structure and Logistics

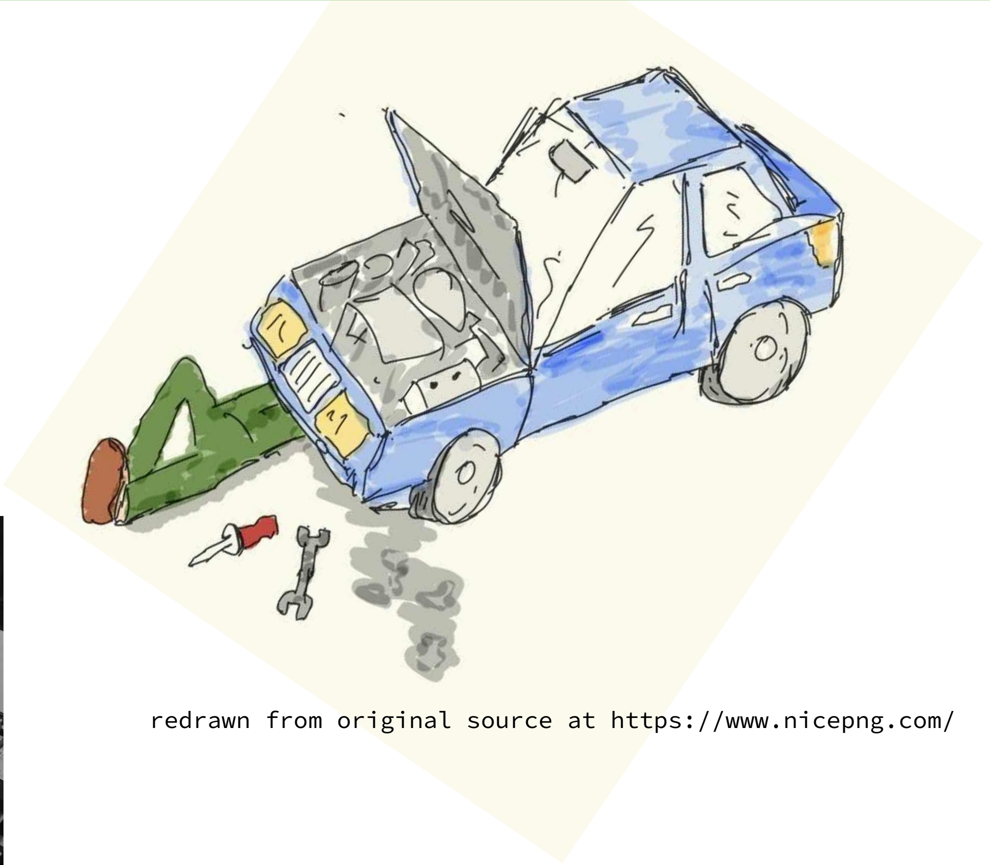
- Theory
  - Slot 2 (Mon: 930am, Tue: 1030am, Thu: 1130am)
- Lab
  - Slot L3 (Thursday: 2pm - 5pm)

- Moodle/Piazza for interactions/discussion forum
- Moodle for lab submissions

- Open for (3rd-year) UG CSE students
- **Not open for non-CSE students**

Goals of the course

- Why OS?
- What is OS?
- Design principles and OS primitives
- Implementation details and hands-on engagement
- **Be the OS!**


redrawn from original source at https://www.nicepng.com/

Topics

- **CPU virtualization**
  - The process abstraction
  - Limited direct execution
  - Processes and Process Management
  - CPU Scheduling
- **Memory virtualization**
  - The address space abstraction
  - Virtual memory, Address Translation
- **Concurrency**
  - Synchronization primitives
  - Threads and multi-threaded operations
- **File systems**
  - The file abstraction
  - Metadata management, Caching, IO Scheduling
  - Consistency, Journaling & Transactions
- **System virtualization**
  - Virtual machines
  - Containers/cgroups
- **IO (??)**
  - Network stack processing
  - RPC
- **Security (??)**
  - OS primitives

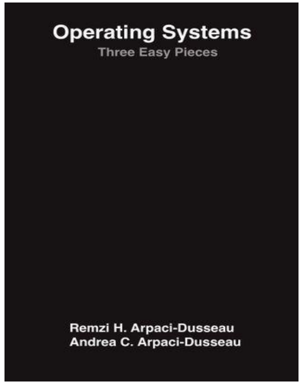
Textbooks/Material

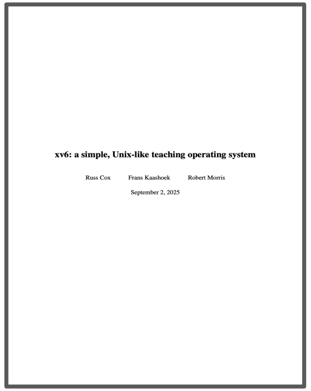
Operating Systems: Three Easy Pieces
by Remzi and Andrea Arpaci-Dusseau

http://pages.cs.wisc.edu/~remzi/OSTEP

xv6
a simple Unix-like teaching operating system

https://pdos.csail.mit.edu/6.828/2025/xv6.html




xv6: a simple, Unix-like teaching operating system

Russ Cox Frans Kaashoek Robert Morris

September 2, 2025

Think OS A Brief Introduction to Operating Systems, Allen B. Downey

The Design of the UNIX Operating System, Maurice J. Bach

Operating System Concepts, by Silberschatz, Galvin and Gagne

Dive Into Systems, Suzanne J. Matthews, Tia Newhall, Kevin C. Webb

Computer Systems: A Programmer's Perspective, Randal E. Bryant and David R. O'Hallaron

Course components

- In-class teaching
  - www.cse.iitb.ac.in/~puru/courses/spring2025-26
    All content — class notes, labs, references
  - **No slides!**
  - Attendance is required. DX grade policy will be informed soon.

- Labs (~8)
  - Linux based tools
  - Programming in C
  - xv6 internals

- Quizzes (2)
- Lab Quizzes (4)
- Mid-semester exam
- End-semester exam

Assessment

- CS219 (Theory)
  - 2-3 scheduled quizzes (15%-20%)
  - In-class ***surprise*** quizzes (~5%)
  - Mid term exam (30-35%)
  - End term exam (40-45%)

- CS236 (Lab)
  - 4 lab quizzes (20%-30% each)

Labs — `think.discuss.code.repeat`

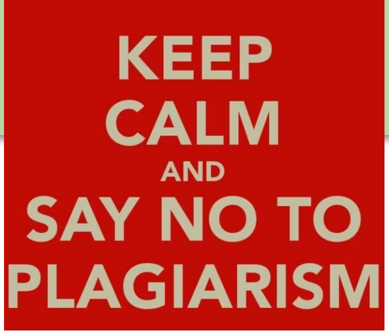
Systems is nothing without hands-on!

Labs are an opportunity for hands-on based learning

In each lab —

- A lab statement with tutorial + exercises
- Interaction with TAs, instructor and friends for co-learning
- Material for labquiz
- Extra practice questions
- **Attendance mandatory!**

Plagiarism policy



Encouraged (during labs)

- Explaining concepts to others
- Discussing algorithms/testing strategies
- Helping debug code
- Searching online for generic algorithms/solutions

Unacceptable (during labquiz)

- Sharing code/answers
- Copying OR reading another students code/answers
- Copying online code or material from prior years OR from the Internet (even reading and typing it yourself is plagiarism!)
- **will result in a report to DADAC + a zero on lab assignment/quiz/exam**

image source: https://www.royalcontentresearch.co.in/

The GenAI face-off

Computer Science and Engineering ≠ Prompt engineering

Is GenAI the dark side?
no easy answer, use your own judgement and tread carefully!

GenAI the genie

prompt, prompt, prompt away
the machine wants to play

answers questions, quick and free
tempts to eat from the forbidden tree

cognitive augmentation can be a win-win
cognitive handover is not your twin

prompt, prompt, prompt away
do not lose your way!

Ready, Set, OS!

Introduction to Operating Systems

CS219/CS236

Spring 2025-26

Puru

www.cse.iitb.ac.in/~puru

www.cse.iitb.ac.in/synerg

Components of a computer system

Hardware — CPU, memory, IO devices, storage …

User software/Applications — applications, databases, platforms, libraries …

System software — ??

System software

Tools/programs that enable development of user software and facilitate access to hardware resources

Examples — compilers, device drivers, linkers, parsers, **operating systems**, …

What is an OS? … some definitions

What is an OS? … some definitions

Manager of all programs

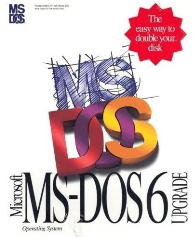
Resource manager of all hardware resources

Arbitrator/orchestrator for user programs and hardware

Security and isolation provider for programs

Software required to execute programs

Enabler of generalized/mass computing



Typical OS actions/requirements

Typical OS actions/requirements

sharing/dividing/multiplexing hardware resources
- memory, cpu, n/w and disk bandwidth

starting/stopping/resuming programs (for execution)

IO handling
- network packet reception, reading and writing to storage, kbd, …

persistence
- what you (name and write) is what you get!

uniform interfaces

isolation and robust execution

OS magic in action

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

```
$ ./cpu "A" && ./cpu "B"
A
B
B
A
A
^C
$
```

cpu.c (source: section 2.1 of [ostep])

```
int
main(int argc, char *argv[])
{
    if (argc != 2) {
        fprintf(stderr, "usage: cpu <string>\n");
        exit(1);
    }
    char *str = argv[1];
    while (1) {
        Spin(1);
        printf("%s\n", str);
    }
    return 0;
}
```

Magic details

process and process management

scheduling policy

context switch

system calls

signals, interrupts

virtualization of the CPU …

Some more OS magic

mem.c (source: section 2.3 of [ostep])

```
int
main(int argc, char *argv[])
{
    int *p = malloc(sizeof(int));                          // a1
    assert(p != NULL);
    printf("(%d) address pointed to by p: %p\n",
           getpid(), p);                                   // a2
    *p = 0;                                                // a3
    while (1) {
        Spin(1);
        *p = *p + 1;
        printf("(%d) p: %d\n", getpid(), *p);             // a4
    }
    return 0;
}
```

```
$  ./mem & ./mem &
[1] 24113
[2] 24114
(24113) address pointed to by p:
0x200000
(24114) address pointed to by p:
0x200000
(24113) p: 1
(24114) p: 1
(24114) p: 2
(24113) p: 2
(24113) p: 3
(24114) p: 3
(24113) p: 4
(24114) p: 4
```

Magic details

address space

page tables

virtualization of memory

Why an OS?

User workflow for work

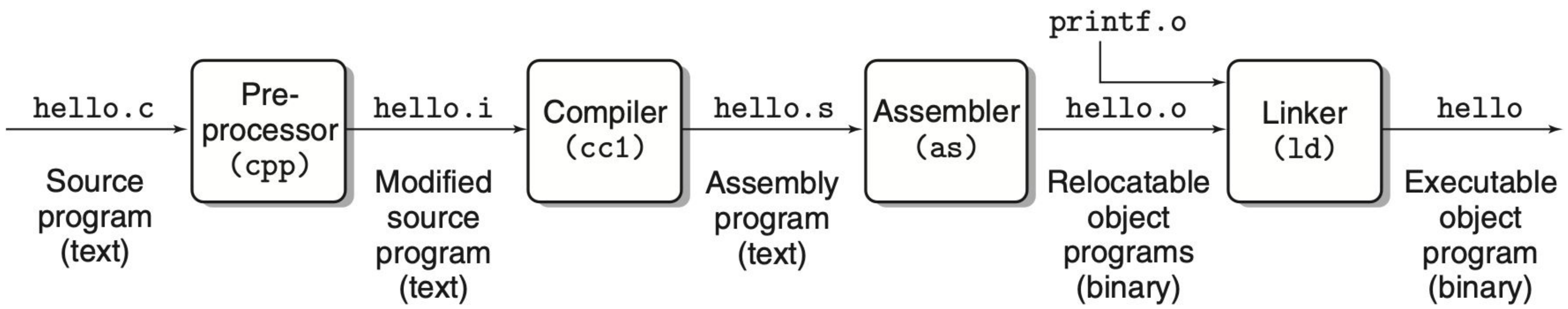


image source: Figure 1.3 of [csapp])

code.compile.link.execute.debug.repeat

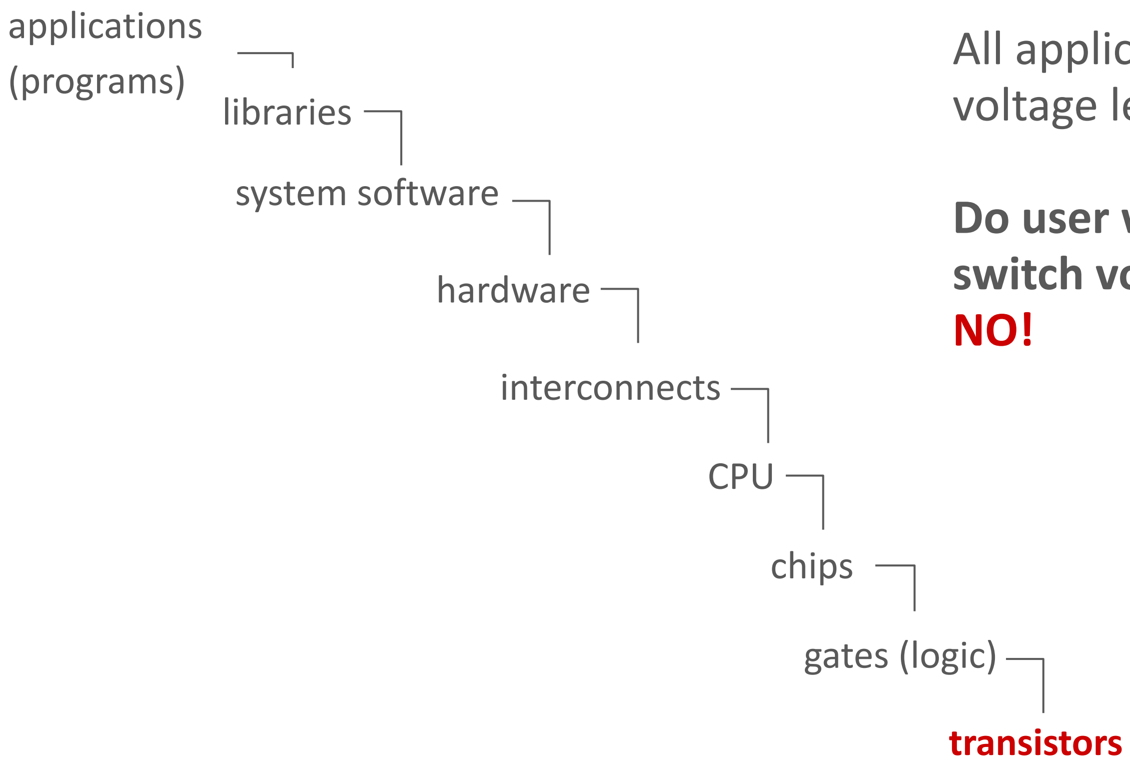
What is the executable/binary format?

Who understands it?

Where is it stored?

How is it used?

Zoomed in version of work in action



All application logic/work is switching voltage levels of a transistor!

Do user write programs to (explicitly) switch voltage levels?
NO!

What is the key to (computing) world peace?

What is the key to (computing) world peace?

ABSTRACTIONS!

else machines are in danger of becoming glorified paper weights

abstraction

functionality/service provided by an entity used/consumed by another entity

implementation detail not concern of users

well-defined interface to request for service

(real life) examples —

CPU abstractions

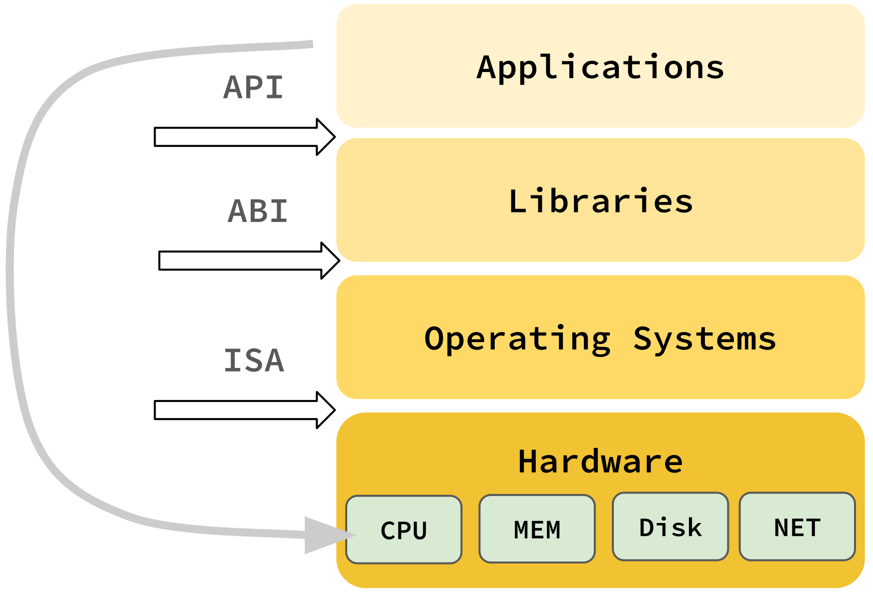
ISA — Instruction set architecture

- instructions
- registers
- memory model
- IO model
- data types
- e.g., x86, risc-v, arm, powerpc, msp430, …

CPU ***interfaces*** with hardware and **uses** hardware capabilities to **provide** the ISA abstraction

Which entity uses the ISA abstraction?

The abstractions stack



Each layer uses abstractions of lower layer to provide new abstractions

Higher-level abstractions enable agility/uniformity

Goal is to the study design and implementation details of the OS abstractions and interfaces

One-slide summary

Sending emails via flipping voltage levels of a transistor is impractical
But, we want to send 100 emails per day!

Abstractions are the secret sauce of world peace!

The CPU's ISA is an abstraction for hardware capabilities

The OS **uses** the CPU's ISA to provide its own abstractions

Homework

What are the OS abstractions?
What is the difference between API and ABI?
Chapter 2 of [ostep]