

Introduction

Amir Masoumzadeh

CSI 333 – System Fundamentals



Aug. 22, 2023



Welcome to System Fundamentals!

Prof. Amir Masoumzadeh (amasoumzadeh@albany.edu)

- Office Hours: Tuesday/Thursday 4:30pm–5:30pm (UAB 422), or by appointment

TA Ferhat Demirkiran (fdemirkiran@albany.edu)

- Office Hours: Monday/Wednesday 2pm–3pm (TBA), or by appointment

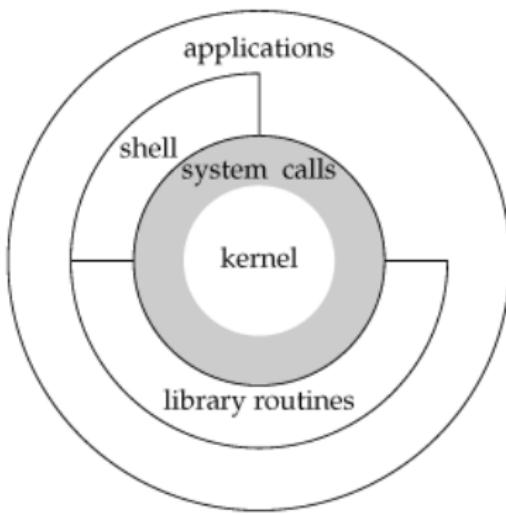


Systems software

- Primarily provide services to other software
- Often developed for performance
- Typically tied to underlying hardware
- Examples:
 - Operating systems
 - Web browsers
 - Game engines
 - Industrial control systems

Operating systems

- Written for specific (class of) hardware
- Provides services to applications such as sharing hardware resources
- Linux architecture (simplified):



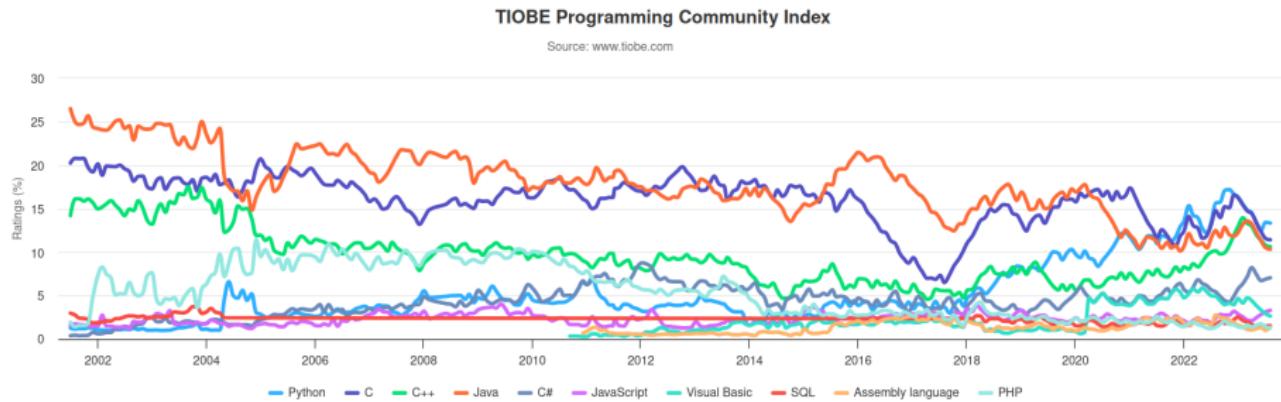


Why learn low-level programming?

- Someone needs to build system software!
 - Need to provide an interface for a hardware your company develops
 - Need the best performance possible (lowest overhead) and willing to forgo niceties of higher-level languages (like Java)
 - Need to develop for a resource-constrained environment
 - To learn how the computer really works without the fancy layers and abstractions

Why C?

- Still very popular



*TIOBE ratings are based on the number of skilled engineers world-wide, courses and third party vendors



Learning modules

- ① Shell/Git
- ② C programming
- ③ POSIX/Linux programming
- ④ Misc. topics



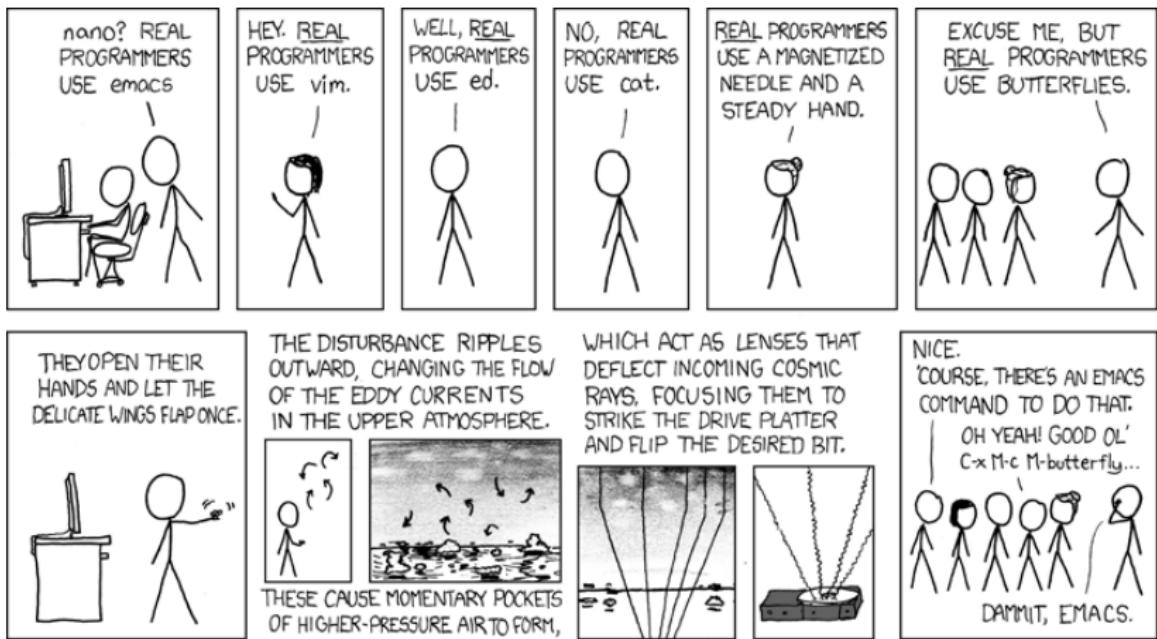
Set up your environment: shell

- You need a POSIX shell for this class
 - Your primary OS is Linux? Awesome!
 - Mac OS? Should work too
 - Windows? Windows Subsystem for Linux
 - Virtual machine (VirtualBox + Install your favorite Linux distro)
- Will be your first lab



Set up your environment: editor

- Choose your favorite editor
 - Popular editors in Linux: Emacs, Vim, nano, ...



credit: xkcd.com



Set up your environment: IDE

- Use Visual Studio Code for debugging larger C program

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows variables, local, global, and static scopes.
- Editor:** Displays the file `main.c` with the following code:

```
static int i;
int main(void) {
    volatile int local = 0;
    for(;;) {
        __asm__("nop");
        i++;
        local++;
    }
    return 0;
}
```
- Terminal:** Shows the build command and its execution:

```
FRDM-K22F_Blinky > src > C main.c > main(void)
> _FLASH
1 static int i;
2
3 int main(void) {
4     volatile int local = 0;
5     for(;;) {
6         __asm__("nop");
7         i++;
8         local++;
9     }
10    return 0;
11 }
```

```
1 0x000000b4: 82 b0      sub sp, #8
2 0x000000b6: 00 23      movs r3, #0
3 0x000000b8: 01 93      str r3, [sp, #4]
4 0x000000ba: 00 bf      nop
5 0x000000bc: 03 4a      ldr r2, [pc, #12]
6 0x000000be: 13 60      ldr r3, [r2, #0]
7 0x000000c0: 01 33      adds r3, #1
8 0x000000c2: 13 60      str r3, [r2, #0]
9 0x000000c4: 01 9b      ldr r3, [sp, #4]
10 0x000000c6: 01 33     adds r3, #1
11 0x000000c8: 01 93     str r3, [sp, #4]
12 0x000000ca: f6 e7     b.n 0x00ba <main+6>
13 0x000000cc: 7c 00     lsls r4, r7, #1
14 0x000000ce: ff 1f     subs r7, r7, #7
15
```

[100%] Built target FRDM-K22F_Blinky.elf

```
Terminal will be reused by tasks, press any key to close it.
> Executing task in Folder FRDM-K22F_Blinky: make <
```

Scanning dependencies of target FRDM-K22F_Blinky.elf
[20%] Building C object ObjekFiles/FRDM-K22F_Blinky.elf.dir/src/main.c.o
[40%] Linking C executable FRDM-K22F_Blinky.elf

Memory region	Used Size	Region Size	Xage Used
m_interrupts	1 KB	1 KB	100.00%
m_flash_config	16 B	16 B	100.00%
m_text	804 B	S23248 B	0.15%
m_data	128 B	64 KB	0.20%
m_data_2	2 KB	64 KB	3.13%

```
text   data   bss   dec   hex   filename
1836  104  2080  4020  fba  FRDM-K22F_Blinky.elf
[100%] Built target FRDM-K22F_Blinky.elf
```

Terminal will be reused by tasks, press any key to close it.



Set up your environment: git

- Git and GitHub
 - Lecture slides, assignments, questions, feedback, . . .
 - You'll practice it in the first lab
 - We need to know your github username. Let's take care of it!
 - Sign up for GitHub if you don't have an account
 - Go to course Blackboard page, navigate to Course Materials
 - Complete [GitHub Info](#)



Labs

- Schedule: Labs meet on Friday and Monday
 - Friday is first lab to cover a particular assignment. You work on first lab this Friday.
- Short self-paced exercises
- One lab assignment each week
- Should be able to finish them when attending your lab session (in most cases)



Projects

- (Mini-)Project are more significant programming assignments than labs
- About 4 of them during the semester
- You will have about 2 weeks to complete each



Submissions/Grades

- You submit your work on GitHub
 - We collect them from there at the deadline
 - More on that in the next session
- Grades will be recorded in Brightspace
 - You have 5 business days to create an issue in your assignment repository on Github if there is any issue with your grade
 - We will not re-grade after this 5-day period



Exams

- 2 exams
 - Midterm: Module 1 & 2 (Shell, Git, C programming)
 - Final: Module 3 (POSIX/Linux programming)



Policy highlights

- No tolerance for academic dishonesty: cheating, plagiarism, . . .
- No late submission (generally)
 - Passed 11:59pm on the day of the deadline and you will receive no point
 - All sort of software and hardware problems may occur
 - Plan ahead and do not leave it to last minute
- Can ask for one-time late lab submission
 - If granted, you can submit 3 days late
- Up to 10% of in-class exercises will be dropped
- 5-day period to inform us about any grading issue



How to succeed in this class

- Read ahead and exercise!
 - Assigned readings are short
 - Read them before coming to class
 - Read the freaking manual!
 - OK to research online about best practices. But refer to manuals again to fully understand what you find based on your research
- Participate in class activities
 - Bring your laptop, or team up with a friend who can bring their laptop
- Get help from us!
 - Office hours, GitHub, email, ...

