The Hardware/Software Interface

CSE 351 Winter 2021

Instructor:

Mark Wyse

Teaching Assistants:

Kyrie Dowling

Catherine Guevara

Ian Hsiao

Jim Limprasert

Armin Magness

Allie Pfleger

Cosmo Wang

Ronald Widjaja

http://xkcd.com/676/

Lesson Outline

- **Course Introduction**
  - https://courses.cs.washington.edu/courses/cse351/21wi/
- Course Policies
  - Remote Instruction

Introductions: Course Staff

- Your Instructor: just call me Mark
  - (no Prof., Dr., Mr., or any other honorific, please)
  - CSE PhD student, Computer Architecture
  - Washingtonian
  - Food lover and cooking enthusiast
  - Video/board game enthusiast

- Teaching 351 for the second time
  - Instructor in Wi18
  - TA'd in Wi13, Wi14, and Su14 (Coursera offering)
  - Have also TA'd CSE 352, 467, 471, CSEP 548

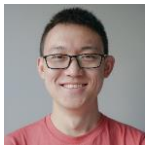
Introductions: Course Staff

- Teaching Assistants










  - Available in section, office hours, and on Ed Discussion
  - An invaluable source of information and help
    - They have been in your seat
    - They know the material and assignments well
    - They are eager to help you!

- Get to know us
  - We are all here to help you succeed!

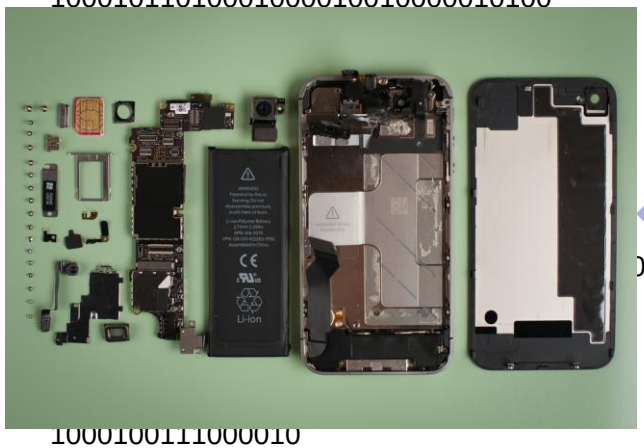
Who are You?

- ~ 105 → 125 students registered, single lecture
- CSE majors, EE majors, and more
  - Most of you will find almost everything in the course new
  - Many of you are new to CSE and/or UW (for 2020-2021)

- Get to know each other and help each other out!
  - Learning is much more fun with friends
  - Working well with others is a valuable life skill
  - Diversity of perspectives expands your horizons
  - A lot of work this quarter is expected to be done in groups

Welcome to CSE351!

1000001101111100001001000001110000000000
0111010000011000
1000101101000100001001000010100



000

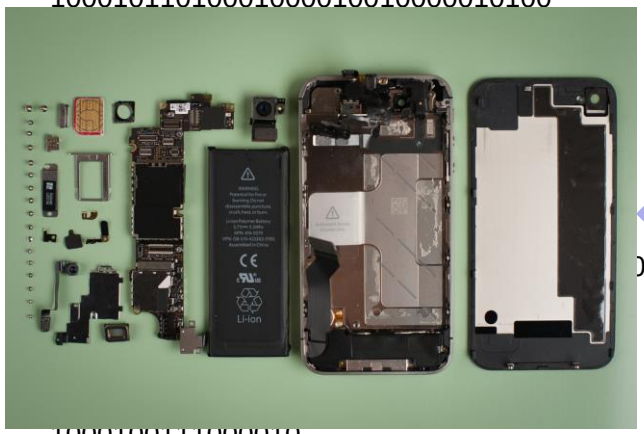
HW/SW Interface

1000100111000010
1100000111111010000011111
1111011101111100001001000011100

- Our goal is to teach you the key abstractions "under the hood"
  - How does your source code become something that your computer understands?
  - What happens as your computer is executing one or more processes?

Welcome to CSE351!

1000001101111100001001000001110000000000
0111010000011000
1000101101000100001001000010100



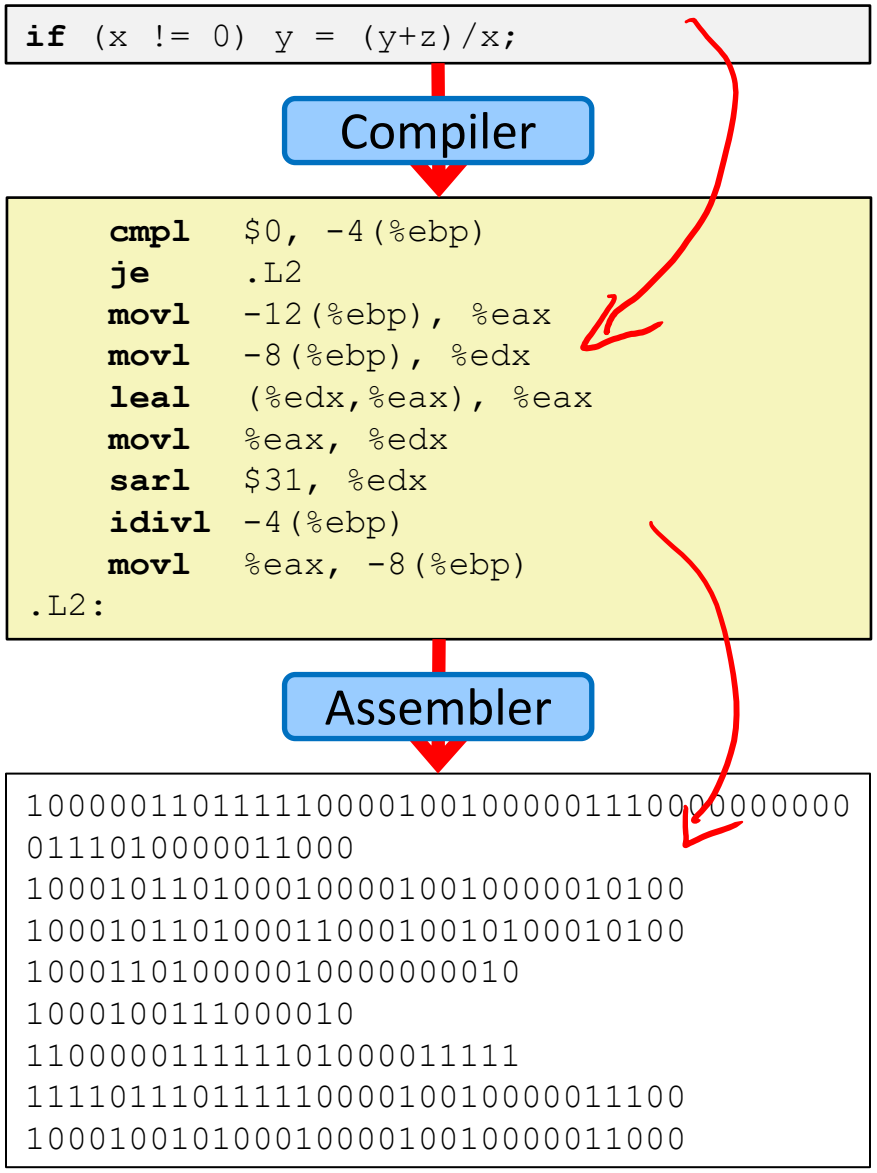
000

HW/SW Interface

1000100111000010
1100000111111010000011111
1111011101111100001001000011100

- This is an *introduction* that will:
  - Profoundly change/augment your view of computers and programs
  - Leave you impressed that computers ever work

Code in Many Forms



High Level Language (*e.g.*, C, Java)

Assembly Language

Machine Code

Roadmap

How does your source code become something that your computer understands?

C:

```
car *c = malloc(sizeof(car));
c->miles = 100;
c->gals = 17;
float mpg = get_mpg(c);
free(c);
```

Java:

```
Car c = new Car();
c.setMiles(100);
c.setGals(17);
float mpg =
    c.getMPG();
```

Assembly language:

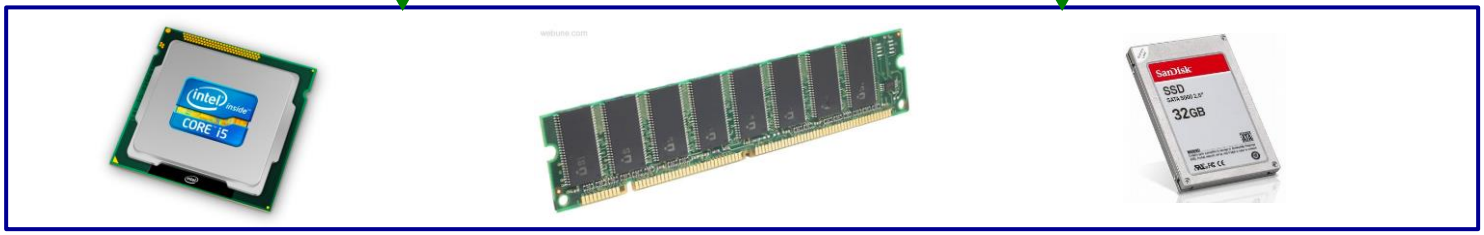
```
get_mpg:
    pushq   %rbp
    movq    %rsp, %rbp
    ...
    popq    %rbp
    ret
```

Machine code:

```
0111010000011000
100011010000010000000010
1000100111000010
11000001111110100011111
```

OS:

Computer system:



- Memory & data
- Integers & floats
- x86 assembly
- Procedures & stacks
- Executables
- Arrays & structs
- Memory & caches
- Processes
- Virtual memory
- Memory allocation
- Java vs. C

Roadmap

What happens as your computer is executing one or more processes?

C:

```
car *c = malloc(sizeof(car));
c->miles = 100;
c->gals = 17;
float mpg = get_mpg(c);
free(c);
```

Java:

```
Car c = new Car();
c.setMiles(100);
c.setGals(17);
float mpg =
    c.getMPG();
```

Assembly language:

```
get_mpg:
    pushq   %rbp
    movq    %rsp, %rbp
    ...
    popq    %rbp
    ret
```

Machine code:

```
0111010000011000
100011010000010000000010
1000100111000010
11000001111101000011111
```

OS:

Computer system:

- Memory & data
- Integers & floats
- x86 assembly
- Procedures & stacks
- Executables
- Arrays & structs
- Memory & caches
- Processes
- Virtual memory
- Memory allocation
- Java vs. C

Some fun topics that we will touch on

- Which of the following seems the most interesting to you? (vote on Ed Lessons)

a) What is a GFLOP and why is it used in computer benchmarks?
b) How and why does running many programs for a long time eat into your memory (RAM)?
c) What is stack overflow and how does it happen?
d) Why does your computer slow down when you run out of *disk* space?
e) What was the flaw behind the original Internet worm, the Heartbleed bug, and the Cloudbleed bug?
f) What is the meaning behind the different CPU specifications? (e.g. # of cores, # and size of cache, supported memory types)

Lesson Outline

- Course Introduction
  - https://courses.cs.washington.edu/courses/cse351/21wi/
- **Course Policies**
  - Remote Instruction

Bookmarks

- Website: https://courses.cs.washington.edu/courses/cse351/21wi/
  - Schedule, policies, materials, videos, assignments, etc.
- Discussion: https://us.edstem.org/courses/3020/discussion/
  - Announcements made here
  - Ask and answer questions – staff will monitor and contribute
- Lessons: https://us.edstem.org/courses/3020/lessons/
- Gradescope: https://www.gradescope.com/courses/208347
  - Lab submissions
- Canvas: https://canvas.uw.edu/courses/1432121
  - Calendar (zoom links), gradebook

My Goals for the Course (and you)

- Create a welcoming, inclusive, and supportive learning environment for everyone
  - concerns? *Reach out!* (staff, anonymous, CSE advising, etc.)
- Enhance your understanding of fundamental topics related to the HW/SW interface
- Every student takes away one useful concept, technique, idea, etc.
- 3 Simple rules for our course:
  - Respect one another
  - Ask questions
  - Have fun!

351 Remote Instruction

- All meetings (lecture, section, office hours) via **Zoom**
  - Sign in to washington.zoom.us with "Login with SSO"
  - Find Zoom URLs via Canvas Calendar
  - Lectures and section presentations will be recorded
  - If your connection supports it, we would love to see your face (video on), but not required – highly recommend turning on camera during small group work and office hours
  - Watch your audio – background noise can be disruptive
- Practice task: turn on your mic  and say "hello!"
- Practice task: turn on your camera  and wave!
  - You may turn off your camera afterwards

351 Remote Instruction

- This class is highly *conceptual* and meant to be digested gradually
  - **Readings** (and quizzes) need to be done before lecture
  - **Lectures** will include practice problems and homework time
  - **Homework** following most lectures, due 1-2 lectures later
- Questions
  - Before and during lecture, there will be a monitored lecture discussion post
  - During group work, questions should be asked verbally or via chat (TAs will circulate through groups)
- Practice task: open the Group Chat  and type your favorite ice cream flavor

351 Remote Instruction

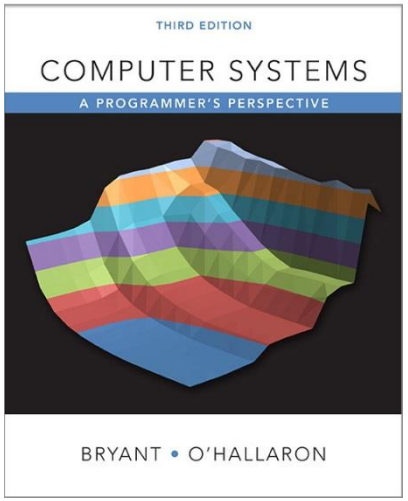
- Group work will be emphasized in this class
  - Lecture and section will use breakout rooms – you will get the most out of it if you actively participate!
  - Many assignments allow collaboration – talking to classmates will help you synthesize concepts and terminology
  - Self-select into small groups on Canvas (https://canvas.uw.edu/courses/1432121/groups#tab-114464)
    - If you are not part of a group on Canvas, then I will randomly assign you to one each lecture
    - You can freely change groups throughout the quarter
  - Responsibility for learning falls on *you*

351 Remote Instruction

- Extenuating circumstances
  - Students (and staff) are in an extremely varied set of circumstances currently and we are cognizant of that
  - For formal accommodations, go through Disability Resources for Students (DRS)
  - We will try to be accommodating otherwise, but the earlier you reach out, the better
- Don't suffer in silence – talk to a staff member!

Reference Material

- *Computer Systems: A Programmer's Perspective*
  - Randal E. Bryant and David R. O'Hallaron
  - Website: http://csapp.cs.cmu.edu
    - see site for changes and errata
  - North American 3rd edition
  - Optional, additional readings



- A good C book (or online reference)
  - *The C Programming Language* (Kernighan and Ritchie)
  - *C: A Reference Manual* (Harbison and Steele)
  - https://www.cplusplus.com

Workload

- **Readings** (~5%) 
  - one attempt per question (completeness)
- **Homework** (~15-20%) 
  - unlimited submission attempts (autograded correctness)
- **Labs** (~35-40%) 
  - last submission graded (correctness)
- **Section Worksheets** (~5%) 
  - single submission (completeness)
- **Study Guides** (~30%) 
  - Cumulative learning assessments
- **EPA:** Effort, Participation, and Altruism (~5%)
  - engagement in course and discussion, in-lesson polls

Lab Collaboration and Academic Integrity

- All submissions are expected to be yours and yours alone
- You are encouraged to discuss your assignments with other students (*ideas*), but we expect that what you turn in is yours
- It is NOT acceptable to copy solutions from other students or to copy (or start your) solutions from the Web (including Github)
- Our goal is that *YOU* learn the material

EPA

- Encourage class-wide learning!
- Effort
  - Attending office hours, completing all assignments
  - Keeping up with Ed Discussion activity
- Participation
  - Making the class more interactive
  - Lecture question voting
- Altruism
  - Helping others in section, office hours, and on Ed Discussion

To-Do List

- Admin
  - Explore/read website *thoroughly*
  - Check that you can access Ed Discussion and Lessons
  - **Get your machine set up to access the CSE Linux environment (CSE VM or attu) *as soon as possible***
  - Optionally (but encouraged!), sign up for CSE 391
- Assignments
  - Pre-Course Survey and hw0 due Wednesday (1/6)
  - hw1 and Lab 0 due Friday (1/8)

Tips for Success in 351

- Read the syllabus/policies page on the website!
- Attend all lessons and sections
  - Engage and ask questions during class time
- Learn by doing
  - Can answer many questions by writing small programs
- Visit Ed Discussion often
  - Ask questions and try to answer fellow students' questions
- Go to office hours
  - Even if you don't have specific questions in mind
- Find a study and homework group
- Start assignments early
- Don't be afraid to ask questions

Unix Tools

- Consider taking CSE 391 Unix Tools, 1 credit
  - Useful skills to know and relevant to this course
  - Available to all CSE majors and anyone **registered** in CSE351
  - If you are interested in taking, attend the first lecture!!

Due Dates and Late Work Policy

- **Homework, Readings, Section Worksheets**
  - No late days/submissions.
- **Labs**
  - **5 late day tokens**
  - May use up to **2** tokens per Lab
  - Lab submission closes 2 late days past due date
  - Late day = 24 hours, except weekends (Saturday + Sunday) count as a single late day
  - After all 5 tokens have been used, late labs incur 20% penalty per late day
- **Exceptions and extensions:**
  - Contact the course staff to discuss, but do not wait until the due date to reach out!