ECE 471 – Embedded Systems
Lecture 1

Vince Weaver

https://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

9:00am, Barrows 119

31 August 2026

Announcements

- If you're interested in ECE574 Cluster Computing there's still time to sign up
- Meets at 11am
- Can take 500 level courses as a senior, but need approval before you can get added to the course

Welcome to ECE471!

We're going to learn all about embedded systems!

https://web.eece.maine.edu/~vweaver/classes/ece471_2026f/

Syllabus – Instructor Info

- I'm Professor Weaver
- Go over syllabus
- QR-Code: Should you trust it?
- Office is 203 Barrows
- Tentative Office hours 1:30pm-2:30pm Tues/Thurs. Also feel free to stop by if door open
- Lecture notes will be posted to website usually within a day or so
  (Now Title II compliant!)

Pre-reqs / Requirements

- ECE271 or equivalent experience
  - Mostly C programming
  - A tiny bit of ARM assembly language
  - Some previous Linux knowledge helps
  - I'll review a lot of this as we go
- No textbook (Raspberry Pi instead).

Syllabus – Hardware

- You will need a Raspberry Pi. More on that later. The Pi is for homeworks, you do not need to bring it to class.
- You will need a breadboard. I assume you already have one. If not let me know.
- I will provide some jumper wires (that you can keep)
- I will loan out other devices/sensors that you will need to return at the end of the semester (please remember as it counts toward your participation grade)

Syllabus – grading – Homeworks

- Homeworks, 50%: 10 or 11 total, lowest dropped.
  - Most will involve the Raspberry Pi.
  - Generally will be due on Friday afternoon. Will have a week to do them.
  - Submission by e-mail, grades sent in response to that e-mail, if you don't like that let me know.
  - Will send e-mail when assignment posted on website.
  - Will reply with grades. No Brightspace.

Syllabus – grading – Tests

- Midterms, two, 15% total
  Tentatively 17 October and 18 November
- Final, 10%, Wednesday 16 December

Syllabus – grading – The Rest

- Project, 20%:
  - Involves using what you learned to do a small embedded project, with a final writeup and demo the last week of classes.
  - Can work in group.
  - Doesn't have to be a Pi or written in C.
  - More details as we get closer.
- Class participation, 5%
  Part of this is returning borrowed items at end.

Syllabus – Late Work / Regrade

- Late work penalty. I will consider late work, but best to turn it in by the deadline even if it isn't completely finished.
- Make regrade requests via e-mail.

Homework Help

- I'll be glad to help if you get really stuck on homeworks
- Often the easiest way to do this is send me your code, as I can run it through the compiler and test it.
- Describing your issue or sending me a screenshot might not be enough and I'll probably ask you to send your code

Pandemic Policy

- Follow UMaine Guidance
- If you test positive for Covid/Mpox/Bird-Flu/etc please don't come to class. Let me know and we can make sure you get the work done
- If you are sick for any reason but still coming to class I encourage you to wear a mask

Syllabus – Academic Honesty

- This has been a problem in the past!
- Do not copy code from other students, either current or from previous years.
- Asking help from the professor/TA is fine
- Asking for general help, or discussing with classmates is fine
- Even having someone look over your code to help find a problem is fine

Syllabus – Academic Honesty – What not to do

- Do not share your code, even if the person requesting claims they will just use it as a reference. In my experience it's too tempting and the person will "accidentally" submit it as their own.
- Do not share code even if it's after the submission deadline as students will submit late work.
- Just don't copy someone else's code and submit it as your own. This includes cut-and-paste or retyping
- If this happens, the person turning in copied code will get a zero, but so will the person providing the code

Syllabus – Academic Honesty – AI/Internet

- Also don't copy code off the internet (again, looking for advice online is fine, but copying code directly is not)
- Don't use AI tools that do the homework for you! (Like Microsoft/Github Co-pilot/ChatGPT)

AI Policy

- Please do not use AI at all for this class
- Especially do not use it on the homeworks, both for coding but also short-answer questions

Why not AI?

- You'll note that I'm not a huge fan of AI
- Makes me unusual as it's the current fad
- You're here to become an expert on embedded systems
- AI can be subtly wrong, and you can only catch it if you actually know what's going on
- I do my best to give feedback on your code, waste of time if I'm grading some AI

Syllabus – Boilerplate

- Go over boilerplate

Raspberry Pi

- We will be using Raspberry Pi devices for the homeworks
- Luckily the parts shortage seems to have finally eased `https://rpilocator.com/`
- Please don't pay more than $40 or so for a Pi!
  Due to recent RAM hikes the high-end models go for hundreds of dollars these days. We don't need that for this class.
- See note on course website about what you need
- If you can't get a Pi before HW#2, let me know and I have some really old ones I can loan out

Raspberry Pi – Models

- Pi Foundation constantly changing up the Pi lineup
  Thanks to Linux we can use most of the various models with some cautions
- See the website for a longer writeup on this:

  `https://web.eece.maine.edu/~vweaver/classes/ece471_2026f/rasp-pi.html`

Raspberry Pi – Good for this Class

- Model 3B/3B+ (probably best)
- Model 4B (faster, but often cost more and have more annoying accessories)
- Model 5 (fastest, but expensive and also annoying accessories)
- Model 2B/1B (older and harder to find, also no wifi)
- Model 2A/3A/4A – these work for the class but lack wired Ethernet
- Model 400/500 – a Model 4/5 built into a keyboard

Raspberry Pi – OK for this Class

- Pi Zero/Zero-2/Zero-W/Zero-2W
  These have limited or no networking, the header isn't soldered on, and have other limitations. They will work though.

Raspberry Pi – Please Avoid

- Compute Modules CM1/CM2/CM3/CM4/CM5
  Technically they would but you'd need hundreds of dollars of additional backplanes
- Pi Pico – has a very different chip and won't run Linux

Additional Parts – SD Card

- You will also need an SD card (8GB or bigger)
- Really old Pis take the wide ones, newer the micro ones.
  Usually not a problem as they tend to come with those adapters.
- You will want to install Linux (I tend to use Raspbian AKA Raspberry Pi Linux), getting a card pre-installed with Raspbian can save an hour or so of writing the SD card. (We will discuss this more in HW#2)

Additional Parts – Power Adapter

- You will need an adapter and/or cable
- Pi3 and older you need a USB-micro charger, generally 1A for older and 2A for newer. You can often get away with just a USB-micro cable plugged into a USB port of desktop/laptop
- If you are providing inadequate power the Pi GPU will put a little sad lightning bolt on the corner of the screen
- Pi4/Pi5 you need a USB-C adapter and there can be issues using ones that aren't the official one

Pi Optional Accessories – Hardware

It can be fun to accessorize, but the stuff on the previous page is all you need. Below are some *optional* extras.

- A case can be useful, if only to avoid accidentally shorting out things. Many people get by just fine without one.
- A dedicated GPIO connector to breadboard adapter
- Heatsinks on modern pis?

Pi Optional Accessories – Connectivity

- HDMI adapter to connect to a screen
  (note pre-3 regular HDMI is fine, pi-zero I think use mini and pi-4/pi-5 use micro)
- USB keyboard/mouse if using it like a desktop
- USB serial if you're hardcore
- Ethernet cable if connecting to a wired network (or direct to laptop)

Embedded Systems

What are some embedded systems?

Seemingly everything has a computer in it these days. IoT.

- Cellphone (though lines blurring, general purpose)
- Vehicles (Cars/Airplanes)
- Appliances (TVs, Washers, Microwaves)
- Medical Equipment
- Industrial/Factory
- Space Probes
- Video Games?

Characteristics of an Embedded System

- **Embedded**
  Inside of/controlling something larger (like a car, washing machine, etc).
  (note: being a board inside a case doesn't necessarily count)
- **Fixed-purpose**
  Why? You can optimize.
  For cost, power, size, reliability, performance.
- **Resource constrained**
  Small CPU, Memory, Disk, I/O, Bandwidth

Characteristics of an Embedded System Continu

- **Lots of I/O**
  For reading sensors (input) or controlling hardware (output)
- **Often real-time constraints**
  Want I/O to happen in guaranteed timeframe.
  Be careful with this one, there's a strict definition of this we won't discuss until later.

What Size CPU/Memory?

- Anything from 8-bit/tiny RAM to 32-bit 1GHz 1GB
- Performance has greatly improved over the years. ARM Cortex A9 in an iPad2 scores same on Linpack as an early Cray supercomputer

Type			Speed	RAM	Disk	GPU
Intel	Xeon	64-bit	4GHz	16GB	1TB	Nvidia
ARM	A53	64-bit(?)	1GHz	1GB	8GB	VC4
ARM	M0	32-bit	32MHz	16kB	128kB	none
MOS	6502	8-bit	1MHz	64kB	140kB	none

Discussion

- What concerns might you have when designing an embedded system?
  Security is a big one these days
- What language might you write your code in?
  C is still popular despite security issues.