ECE 471 – Embedded Systems
Lecture 28

Vince Weaver

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

vincent.weaver@maine.edu

16 November 2026

Announcements

- Midterm on Wednesday
- Don't forget projects
- Don't forget HW#9
- HW#7 has been graded
- HW#8 probably won't be graded before the midterm

Project Status – Due Tuesday (24th)

- A one-line statement of your project topic
- A short summary of the progress you've made so far
- List any parts you need that you don't have yet
- Important! List which day you'd like to present. (Friday the 4th, Monday 7th or Wednesday 9th vs Friday the 11th) (there will be some bonus for presenting early)

Project Status – Additional

- Identify the problem: this can be brief, the idea is you picked the topic, but can you identify the hardware/ software problem you need to solve to complete it.
  - For example: you're making a line-following robot.
  - The hardware problem to be solved would be assembling a mobile platform that can detect a line
  - The software problem to be solved is code that can read the sensor, detect the line, and provide feedback to the motors

HW#6 Review (Real Time Systems)

HW#6 – Question 2

- My results on a Pi3
- min: 2083 ns / max: 6302 ns / average: 2796 ns
- What is a safe value to use to guarantee a deadline?
  - Average + 10%?
  - Max + 10%?
  - Double?
  - Should be higher than max, but there's no real answer here, it depends a lot on what exactly you are trying to do

HW#6 – More on Safe Maximum

- Average itself not necessarily good, half the time you'd miss the deadline
- There are a lot of outliers
- Better might be to calculate standard deviation and make it 2 times stdev
- Example: "human-rating" a rocket? Sometimes in this case use 1.4x safety margin instead of 1.1x

HW#6 – Question 3 (Under Load)

- min: 2396 ns / max: 25,913,564 ns (25ms), average: 9,465,773 ns
- Much worse

HW#6 – Q4 (Under Load + chrt 70)

- Ideally
  - min: 2084 ns / max: 7917 ns / average: 2740 ns
  - Using real-time priorities can make a general purpose Linux OS give you more reasonable timings under load
- Sometimes not quite so ideal
  - Still occasional large outliers
  - Seemed to go down a lot if using `taskset -c 0` to bind to a CPU
  - Cache problem? If industrious could use `perf` to track down

HW#6 – Question 5 (Under Load 70)

- Sorry I didn't warn harder to backup your files (some people lost README)
- For me took too long didn't bother to let finish
- But if load 70 / our job bumped to 75
  min: 1980 ns / max: 6979 ns / average: 2739 ns
  back to somewhat reasonable

HW#6 – Question 6

- Why sysadmin not let anyone set priority?
- They could essentially prevent other jobs from running at all, including important system background tasks

HW#6 – Something Cool

- Standard Deviation.
- Is an occasional outlier OK?
- Within two standard deviations?

HW#6 – Question 8 Type of Real Time

- Brakes: hard real time, disaster/failure if not met in time
- Radio: soft real time. Probably still want to change station even if takes a while to do it
- Video: firm. If you miss deadline the frame decoded is useless

More on Firm Realtime

- Decoding video? What makes "firm" real-time? It's about the results being useless after the deadline missed
- A bit different than video game degraded framerate. Video game constantly monitoring delay and if can't keep up adjusts the physics engines, etc, to use different deltaT
- Video generally has fixed frame rate, not variable

More on Video Decoding

- Compressed video stream like MPG each frame depends on frame before.
- If one is late not only is it useless but the next frame possibly gets corrupted too (you've maybe seen this on TV when antenna gives out and you get weird boxy glitches all over screen)
- Can lose sound sync too
- Can actually see what happens when not hit 60fps. "stream killer" video on twitch, things like random static or lots of movement

HW#7 Review – Code

- Follow directions: temperature probe should be connected to channel 2
- Also make sure you set the single-ended bit (0x80) in addition to channel 2 (so 0xA0)
  If you forget the 0x80 you'll get weird results reading unconnected pins
- People managed merging 2 bytes into one OK
- What is the max frequency? If you set SPI frequency to 500kHz or higher gives wrong results, not sure if data sheet addresses this.

HW#7 Review – Code

- Be sure to check for error on open(), biggest source of errors. Linux won't crash, it will happily just report errors that your code is likely ignoring.

HW#7 Review – Questions

- Disadvantage of SPI?
  - More wires
  - No standard
  - No error checking
- Advantage of SPI?
  - Lower Power
  - Full Duplex
  - No max speed
  - Simpler overall protocol

HW#7 Review – Questions

- Talking about TMP36 on end of cable, SPI *not* involved
- Voltage Drop, Noise?
- Datasheet has two options, convert to current, or an extra resistor.

HW#7 Review – Questions

- Minimum frequency of 10kHz or results invalid. Maybe cannot go this fast if bitbanging via GPIO. Also if context switch in middle, Linux not realtime?
  - If unfamiliar with bitbanging, it just means running a protocol over GPIOs via software rather than using the SoC hardware to do it

HW#7 Review – Linux "fun"

- Devices here for compatibility with ancient UNIX
  - /dev/null
  - /dev/full
  - /dev/zero, holes in files
  - /dev/random – random numbers (see next slide)
- Windows has ancient compatibility devices too, cause lots of trouble as they are invalid for use in filenames (in any directory, not just /DEV) (CON, COM, LPT)

HW#7 Aside – Random Numbers

- /dev/random is strong random numbers, /dev/urandom provides pseudo-random
- Strong random numbers needed for unpredictable crypto, ssh, tls (secure web connections)
- How do you generate random numbers?

HW#7 Aside – Random Numbers

- CPUs can provide this. Do you trust them not to have backdoored things? Do the Linux devels?
- Can generate entropy with random I/O, like mouse movements, incoming packets, etc
- Can have pseudo-random number generators (PRNG)
  Look random, but can predict next result if know seed
  Often used in less critical cases like games and such
  Also if you need reproducible random sequences (debugging)
- Pi has RNG in its SoC

Homework 8 – Code

- Error checking. Exit if cannot open. If you don't, can segfault if try to fscanf a NULL FILE*
- Returning -1 on error might be bad idea
- Compiler warning: `strncpy()` comparing if src longer than count and you don't NUL terminate
- What to report on error? What's an invalid temperature? Not just unlikely? (Below Absolute zero)
- If using streams (FILE *fff), on fopen() error it returns NULL, not -1.
- Be sure to close files, otherwise leak file descriptors Be careful if multiple exit points, must close at all (goto)
- Be careful with your 9/5 Fahrenheit conversion!

Homework 8 – Autodetecting 1-wire

- Open a directory and finding a file
- Finding a file using C is tricky `opendir()` `readdir()`, horrible interface
  Bit of a tangent on the downsides of the `readdir()` interface

HW#8 – Questions

- Why need Vdd? To provide enough current for this particular chip needs extra current if you want parasite mode.
  You can try without Vdd but you will always read out 85C.
  Manual suggests MOSFET, but apparently it's possible on Pi if use 4.7k resistor as well as "strong-pullup=y" kernel command line option.
- (have sensor 100 meters away what bus to use?)
  Because of distance, 1-wire

HW#8 – More Questions

- What is generic term for program that loads OS? bootloader (note: asked for generic, so not uboot or grub or whatever the pi specific code is)
- Why boot partition FAT32? It needs to be really simple, simple enough you possibly write it in assembly language and fit in a tiny chunk of ROM. FAT is an extremely simple filesystem from the 1970s and doesn't require many resources

HW#8 – Shell Script

- `#!/bin/sh` should be first line (magic number)
- Trouble if edit on windows, why (linefeed vs carriage return)
  shebang description
- Making executable with chmod
- Default shell, can put other things there, like python or perl, etc, even ARM emulator
- sh vs bash