ECE 471 – Embedded Systems
Lecture 18

Vince Weaver

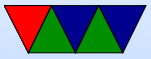
http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

12 October 2022

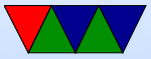
Announcements

- Midterm on Friday, the 14th
- Am trying to grade all homeworks by then



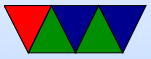
Project Preview

- Can work in groups
- Embedded system (any type, not just Pi)
  Pi Pico, Beagleboard, Orange Pi, 271 STM boards, TS-7600, etc.
- Written in any language (asm, C, python, C++, Java, Rust, etc.)
- Do some manner of input and some manner of output using the various capabilities we discussed
- I have a large amount of i2c, spi, and other devices that



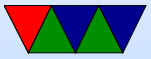
you can borrow if you want to try anything interesting.

- Past projects: games, robots, weather stations, motor controllers, music visualization, etc.
- Will be a final writeup, and then a short minute presentation and demo in front of the class during last week of classes.



HW#5 – Code Notes – Datasheet

- What does 'X' mean in this context? (don't care)
- Bits 15-8 was confusing, it's because we can ignore bits 7-0 (the i2c address and r/w) as Linux sends those for us



HW#5 – Code Notes – Constants

- Enabling oscillator. If want value 2 in top 4 bits, 1 in bottom 4? Just use `0x21`?

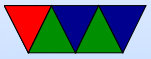
  `(0x2<<4)|(0x1)`

- Can we use hex or binary notation?

  The shifts make it more explicit what's going on, compiler will optimize for you

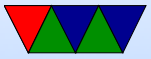
- "Magic Constants", you might instead want to do something like

```
#define HT16K33_OSCILLATOR_ON (0x2<<4)|(0x1)  // p42 of datasheet
buffer[0]=HT16K33_OSCILLATOR_ON;
```



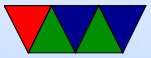
HW#5 Review – Questions – Pi Boot

- Raspberry Pi boot is odd: GPU does it.
  Why? Originally the chip was designed to be mostly GPU.
- sd-card is mildly unusual but not as unusual as GPU



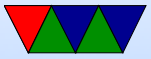
HW#5 Review – Questions – Bootloader

- Program that loads kernel and jumps to it is called the bootloader
- Not start.elf or GRUB (those are specific bootloaders)
- Not an init script (those run after the kernel is running)
- Not the boot firmware (this often loads the bootloader. In some cases firmware can act as a bootloader, but in that case it is a bootloader)

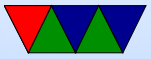


HW#5 Review – Questions – Fat32

- A full description of filesystems is a bit beyond this class
- Fat32 is a specific, simple, filesystem with roots going back to the 1970s via MS-DOS and ran on computers with less than 16kB of RAM
- The primary reason it is used by boot firmware is because the code is simple and can be easily coded in a small amount of C/asm code and can be used for early boot
- Not necessarily written in Assembly
- It's not default Linux filesystem (default Linux fs is

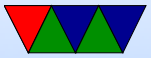


something more complex like ext4 or btrfs)



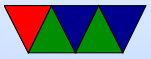
HW#5 Review – Questions – i2c Reserved Address

- Skipped i2c – those addresses are reserved.
- For various things, not just "future purposes"
- What happens if you have a device living at address 0x0? Would it work?



HW#5 Review – Linux

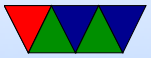
- wc, diff, piping
- You may have seen this all before in ECE331
- diff – used when making patches, also git diff
  Ask for wc -l which just shows lines. Can also show words, chars



i2c Reserved Addresses Reminder

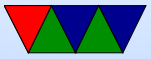
Address	R/W Bit	Description
000 0000	0	General call address
000 0000	1	START byte (helps make polling cheaper)
000 0001	X	CBUS address
000 0010	X	Reserved for different bus format
000 0011	X	Reserved for future purposes
000 01XX	X	Hs-mode master code
111 10XX	X	10-bit slave addressing
111 11XX	X	Reserved for future purposes

10-bit addresses work by using special address above with first 2 bits + R/W, then sending an additional byte with the lower 8 bits.



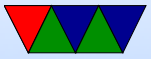
Midterm Notes

- The midterm will be in-person during class time
- Closed book/notes but you are allowed one page (8.5"x11") full of notes if you want

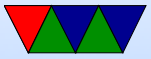


Midterm Content

- Be sure you know the characteristics of an embedded system, and can make an argument about whether a system is one or not.
  - Inside of something (embedded)
  - Fixed-purpose
  - Resource constrained
  - Sensor I/O
  - Real time constraints (if you use this, be sure you can explain)



- Benefits/downsides of using an operating system on an embedded device
  - Benefits: "Layer of Abstraction"
  - Downsides: overhead, timing
- C code
  - Have you look at some code and know what it is doing
  - Fill in missing comments
  - Look at code and find bugs
  - Mostly know what file I/O, loops, usleep, open/ioctl (things we've done in the homeworks)
- Code Density



- Why is dense code good in embedded systems?
- Know why ARM introduced THUMB/THUMB2

- GPIO & i2c
  - Know some of its limitations (speeds, length of wires, number of wires, etc)
  - Don't need to know the raw protocol
  - Know the Linux interface (open, ioctl, write) and be familiar with how those system calls work