ECE 471 – Embedded Systems
Lecture 17

Vince Weaver

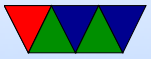
http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

9 October 2020

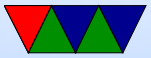
Announcements

- HW#5 was due
- HW#6 will be posted, due in *two* weeks (there might be a slight delay in getting it posted)
- Midterm is a week from Friday, the 16th
- Review on Wed
- No class Monday

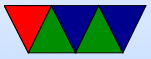


Homework #4 Error Checking

- What do you do if there's an error?
- Ignore it? Why could that be bad?
- Retry until it succeeds?
- Print an error message and continue?
  Can you continue?
  What if continuing with a bad file descriptor breaks things?
  What if printing too many error messages fills up a log, swamps the screen, hides other errors?

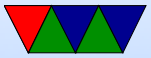


- Good error message
  Can't be confused with valid input (airlock)
  If displayed to user, make it easy to understand
- Print an error message and exit?
  What if it's a critical system?
- Crashing is almost never the right answer.

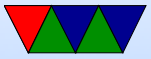


Homework #4 Permissions

- We haven't really discussed Linux permissions
- List file, "user" "group" "all"
- `drwxr-xr-x`
- Often in octal, 777 means everyone access
- Devices under /dev or /sysfs might be set to only root or superuser
- Traditional UNIX /dev you can set with chown (to set user/group) or chmod (to set permissions)
- Group under /etc/group, so gpio group



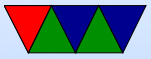
- Having to manually set permissions a pain. Program called udev that does it automatically when a device driver is configured. It might take a few ms to notice
- Why is it better than using "sudo"? Why might I as grader not want to run your code using "sudo" if I can avoid it?
- How to set up sudo? /etc/sudoers file



Homework #4 Review

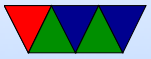
- Blink frequency. Remember, 1Hz is 500ms on / 500ms off

  not 500us, not 1s
- Still grading the switch part.



Homework #4 Question

- 5.a Why usleep? Less resources (not busy sleeping), cross-platform (not speed-of-machine-dependent), compiler won't remove, other things can run, power saving.
  Be careful saying accuracy! usleep() guarantees a minimum time delay, but it is best effort how long the delay actually is. So if you really need *exact* time delays you probably want some other interface.
- 5.b Layer of abstraction. In this case, not having

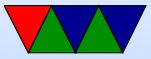


to bitbang the interface or know low-level addresses, portability among machines.
ability to run WiringPi is not a benefit

- 6.a Machines from dmesg: 2020: Pi4 (7) Pi3 1.2 (4) pi3b+ (6) dmesg a good place to find error messages, etc.
- 6.b Kernel versions. Current Linus kernel (upstream) is 5.9 /5.9-rc8
  Uname syscall, what the parts mean

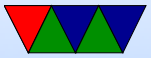
```
Linux linpack-test 4.14.50-v7+ #1122 SMP Tue Jun 19 12:26:26 BST 2018 armv7l GNU/Linux\\
Linux orvavista 4.5.0-2-amd64 #1 SMP Debian 4.5.5-1 (2016-05-29) x86_64 GNU/Linux\\
```

2020: 4.14.79 (1) 4.19.50 (1) 4.19.66 (3) 4.19.75 (1)



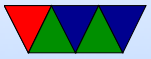
4.19.97 (3) 5.4.51 (7)

- 6.c. Disk space. Why `-h`? Human readable. what does that mean? Why is it not the default? At least Linux defaults to 1kB blocks (UNIX was 512) Lots of large disks.

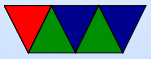


HW#4 – Debouncing

- Tricky as we are detecting levels not edges here
- Reading and only reporting if you say have 3 in a row of save val
- Reading, sleeping a bit, then report the value after has settled
- Just sleeping a long time after any change? If a short glitch happens this might misreport.
- Sleep too long, might miss events
- Debounce if using interrupt-driven code

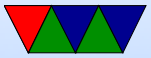


In that case debouncing might be to ignore repeated changes if they happen too close together



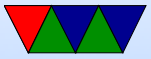
How a Program is Loaded on Linux

- Kernel Boots
- `init` started (systemd)
- `init` calls `fork()`
- child calls `exec()`
- Kernel checks if valid ELF. Passes to loader
- Loader loads it. Clears out BSS. Sets up stack. Jumps



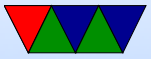
to entry address (specified by executable)

- Program runs until complete.
- Parent process returned to if waiting. Otherwise, init.



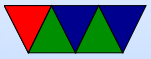
Viewing Processes

- You can use `top` to see what processes are currently running
- Also `ps` but that's a bit harder to use.

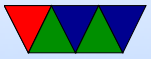


Homework 6 – Background

- You have two weeks to do this one
- Handout should cover most of it
- bit-banging i2c
- Why not bitbang everything? A pain. Hardware does it for you. Hardware even does more, can often buffer or DMA, timing more exact.
- Why might you want to bitbang i2c? Only have one i2c bus? Or no i2c bus, only GPIOs? kernel has bitbang driver

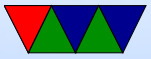


- Tell my boring frontpath i2c-bitbang story

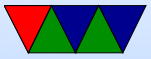


Homework 6 – Implementation

- Use the gpio interface to drive the SDA and SCL lines to manually run the 4x7 LED display
- Still easier than full bitbang, where you'd have to write to various i/o addresses
- A lot of the code is provided for you, follow the directions
- How do you set SDA low?
  Set output to '0'
- How do you set SDA high?
  Open collector, need to let it float, not be driven 1



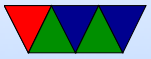
The Linux interface lets you select a line to be open-drain and when you do that, when you output a 1 it knows to switch the pin to “input” which lets the line float



Homework 6 – Multiple Files

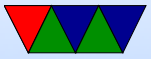
- This homework has the compiler compiling multiple small files and then linking them together.
- Why do this? Easier to edit smaller files, easier if collaborating with others, lets you share code without cut and pasting
- `gcc -c file.c` creates `file.o`
- Link a number of files together

  `gcc -o executable file1.o file2.o file3.o`
- This is how you create libraries (static just a matter of



`ar`, dynamic are a bit more complicated)

- All global functions/vars are exported, unless you declare them `static`
- How do you know how to call functions in other files? Need to pre-declare prototype so the compiler knows how to set up the registers before calling. Traditionally with C this is done in a .h header file
- Include files with `#include "file.h"`. You may also have seen angle brackets, what is the difference?
- By default all functions/global vars are exported. How can you specify to only be visible in own file? Use the



`static` keyword.