ECE 471 – Embedded Systems
Lecture 30

Vince Weaver

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

vincent.weaver@maine.edu

18 November 2020

Announcements

- HW#9 due Friday
- You can start turning in parts now (assuming you don't need them for your project)
  you can keep the resistors and jumper wires
  include your name if possible
- Project status report due Monday. Not long:
  - 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

- ◦ List if you're willing to present early (Friday the 4th, Monday 7th or Wednesday 9th vs Friday the 11th) (there will be some bonus for presenting early)

Midterm Review

Midterm #2 Review

- Booting on the Pi
  - What a bootloader does
  - Why Pi is unusual
- Real Time
  - Definitions
  - Is this hard, soft, firm
- i2c/SPI/1-wire
  - Know the tradeoffs between i2c, SPI, 1-wire
  - Be able to follow the C code for them

- Security
  - Buffer overrun, why it is bad
- Coding Practices
  - Be aware of the case studies we suggested
  - Know of some of the recommended ways to write safer C code

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
- 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!
- Finding a file using C. opendir() readdir(), horrible 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.
- Because of distance, 1-wire

- 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

Starting Programs at Boot

- init process starts first
- Traditionally would start various shell scripts under /etc (the name and order of these can vary a lot)
- Possibly with advent of systemd this will change
- Currently you can still put things you want to run at start in `/etc/rc.local`

Wii Nunchuck

- Fairly easy to interface
- Put onto i2c bus. Device 0x52
- Send handshake to initialize. Use longer one (0xf0/0x55/0xfb/0x00) not the simpler one you might find(0x40/0x00). This works on generic nunchucks and possibly also disables encryption of results.
- To get values, send 0x00, usleep a certain amount, and read 6 bytes. This includes joy-x, joy-x, accelerometer

x/y/z and c and z button data. More info can be found online.

byte0 = joy-x, byte1 = joy-y, byte2 = top8 acc x, byte3 = top8 acc y, byte4 = top8 acc z, byte 5 is bottom 2 z,y,x then button c and z (inverted?)

Linux and Keyboard

- Old ps/2 keyboard just a matrix of keys, controlled by a small embedded processor.
  Communication via a serial bus. Returns "keycodes" when keypress and release and a few others.
- Many modern keyboards are USB, which requires full USB stack. To get around needing this overhead (for BIOS etc) support bit-bang mode. OS usually has abstraction layer that supports USB keyboards same as old-style

- Linux assumes "CANONICAL" input mode, i.e. like a teletype. One line at a time, blocking input, wait until enter pressed.
- You can set non-CANONICAL mode, async input, and VMIN of 1 to get reasonable input for a game. Arrow keys are reported as escape sequences (ESCAPE-[-A for up, for example).
- Even lower-level you can access "RAW" mode which gives raw keycode events, etc.
- See the `tcgetattr()` and `tcsetattr()` system calls
- There are libraries like ncurses that abstract this a bit.

Also GUI and game libraries (SDL).

Faking Linux Input Events

- How to insert input events into Linux, i.e. have a software program fake keyboard/mouse/joystick events.
- Linux supports a "uinput" kernel driver that lets you create user input.
- There is some info on a library that makes this easier here: `http://tjjr.fi/sw/libsuinput/`
- It has examples for keyboard and mouse. Joystick should be possible but there's no sample code provided.
- Python wrappers seem to exist too.

Camera Port

- The SoC has dedicated hardware for driving cameras
- 5megapixel, CSI port (Camera Serial Interface) plus i2c bus to command it.
- Can read data in parallel, directly, without needing USB overhead.
- These chips often used in cell-phones, so makes sense to have support for camera-phone without extra chip being needed.

Touchscreen Display Port

UART – serial port

- Note: Asynchronous, no clock (unlike USART)
  how do both sides agree on speed?
- Often useful on embedded boards and old systems, might be only way to reliably connect
- RS-232, originally for teletypes
- 3-15V high, -3 to -15V low
- start/stop bits, parity, bit-size
- Hardware vs Software flow control
- Speeds 300bps - 115000bps and beyond

- 50feet (15m) w/o special cables
- 3-pin version (transmit, receive ground). Also 5-pin HW flow control (CTS/RTS). Can have 2-pin version if only want to transmit
- These days often hook up USB connector
- What does 9600N81 mean?

Pi Serial Ports

- Raspberry Pi has two serial ports, good one and lousy one
  They switched them up with Pi3
- Pi does TTL (5v/0) not RS232
- Does support HW flow control, but need to activate those pins custom, is a bit complicated
- Use TTL to USB serial converter usually.
  Tell story of the prolific bricking the firmware?

Pi SMI

- `https://iosoft.blog/2020/07/16/raspberry-pi-smi`
- Secondary Memory Interface
- Available on Pis
- Allows creating wide parallel bus out of GPIOs
- Not well documented

Bluetooth

- Basic unit: piconet, master node and up to seven *active* slave nodes within 10m
- Many can exist in an area, and can be connected by a bridge. Connected piconets are called a scatternet
- There can also be up to 255 “parked” nodes in a picnoet
- When parked, can only respond to activation on beacon
- Hold and siff?
- Slaves designed to be cheap, so dumb. Master is smart and runs them. slave/slave communication not possible

- Master broadcasts clock 312.5us. Master transmits in even, slave in odd.
- Radio layer – 2.4GHz, 10 meters. 79 channels of 1MHz.
- pairing
- Bluetooth V1.1 has 13 different application protocols.
- Bluetooth 4.0 (Bluetooth Low Energy) (2010)
  - 25Mbps/200 feet
  - Entirely new stack, designed for low power rapid setup links
  - Not backwards compatible, but same frequency range
  - New profiles

- Linux interface: depends on type. Filetransfer/obex. Audio (looks like an audio driver) network device, serial device

Bluetooth and Linux

- Two competing stacks, BlueZ and Affix

```
sudo bluetoothctl
[sudo] password for vince:
[NEW] Controller B8:27:EB:52:DD:E8 linpack-test
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# scan on
Discovery started
[CHG] Controller B8:27:EB:52:DD:E8 Discovering:
[NEW] Device AC:37:43:89:4C:02 HTC BS 02CA47
[NEW] Device AC:37:43:89:2F:86 HTC BS 86B06E
[CHG] Device AC:37:43:89:2F:86 RSSI: -90
[bluetooth]# scan on
Failed to start discovery: org.bluez.Error.InPr
[bluetooth]# connect AC:37:43:89:4C:02
```

- obexpushd. Appears as serial port?