ECE 471 – Embedded Systems
Lecture 22

Vince Weaver

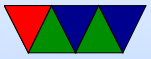
http://www.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

24 November 2015

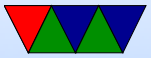
Announcements

- Project groups status report due:
  1. One-sentence summary of project
  2. What hardware you plan to use for input/output and board?
  3. Have you acquired/tested the hardware? Does it work?
  4. Are you willing to present on Tuesday rather than Thursday?



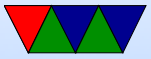
Homework 8 Review

- Some C coding mistakes
- Why need Vdd? To provide enough current for this particular chip needs extra transistor if you want parasite mode.
  You can try without Vdd but you will always read out 85C.
- Because of distance, 1-wire
- shell script



Homework 9 Review

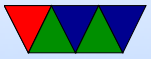
- List an *example* of poorly written embedded code.
- Why write good code?
  Cut-and-pasting, good practice, among other reasons.
- Why is touch useful? force make to rebuild
- 2038 problem
  Time in Linux is seconds since 1-1-1970. Not a problem 64-bit machines, but overflows in 2038 for 32-bit. Can



avoid with a 64-bit system or else a specially patched Linux system
discuss y2k problem

- ctime – last status (metadata) change (originally create time) things like permissions change, ownership change, rename
  mtime – last modified
  atime – last access

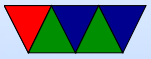
- In stat syscall. `stat` command. Why atime bad?



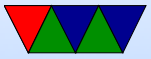
noatime, relatime

- utime() used by touch. Cannot change ctime, set to current time

- why not believe timestamp? maybe could look at ctime. also set clock back if own machine.
  HW assignment at Cornell

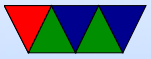


Low-Level Linux/Project digressions

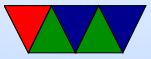


Linux and Keyboard

- 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



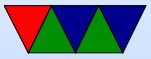
- Linux assumes “CANONICAL” input mode, i.e. like a teletype. One line at a time, blocking input, wait until enter pressed.
- You can set non-CANONICAL mode, async input, and VMIN of 1 to get reasonable input for a game. Arrow keys are reported as escape sequences (ESCAPE-[-A for up, for example).
-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



Power and Energy Concerns

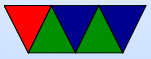
Table 1: ATLAS 300x300 DGEMM (Matrix Multiply)

Machine	Processor	Cores	Frequency	Idle	Load	Time	Total Energy
Raspberry Pi	ARM 1176	1	700MHz	3.0W	3.3W	23.5s	77.6J
Gumstix Overo	Cortex-A8	1	600Mhz	2.6W	2.9W	27.0s	78.3J
Beagleboard	Cortex-A8	1	800MHz	3.6W	4.5W	19.9s	89.5J
Pandaboard	Cortex-A9	2	900MHz	3.2W	4.2W	1.52s	6.38J
Chromebook	Cortex-A15	2	1.7GHz	5.4W	8.1W	1.39s	11.3J



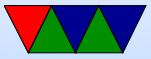
Questions

- Which machine consumes the least amount of energy? (Pandaboard)
- Which machine computes the result fastest? (Chromebook)
- Chromebook is a laptop so also includes display and wi-fi
- Consider a use case with an embedded board taking a picture once every 20 seconds and then performing a



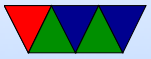
300x300 matrix multiply transform on it. Could all of the boards listed meet this deadline?
No, the Raspberry Pi and Gumstix Overo both take longer than 20s and the Beagleboard is dangerously close.

- Assume a workload where a device takes a picture once a minute then does a 300x300 matrix multiply (as seen in Table 1). The device is idle when not multiplying, but under full load when it is. Over an hour, what is the energy usage of the Chromebook? What is the energy usage of the Gumstix?



Chromebook per minute: $(1.39s \times 8.1W) + (58.61s \times 5.4W) = 327.75J$
Chromebook per hour: 327.75J * 60 = 19.7kJ

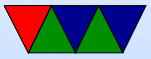
Gumstix per minute: $(27s \times 2.9W) + (33s \times 2.6W) = 164.1J$
Gumstix per hour: 164.1J * 60 = 9.8kJ



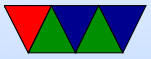
Pandaboard Power Stats

- Wattsuppro: 2.7W idle, seen up to 5W when busy
- `http://ssvb.github.com/2012/04/10/cpuburn-arm-c html`
- With Neon and CPU burn:

Idle system	550 mA	2.75W
cpuburn-neon	1130 mA	5.65W
cpuburn-1.4a (burnCortexA9.s)	1180 mA	5.90W
ssvb-cpuburn-a9.S	1640 mA	8.2W

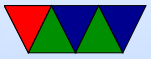


Easy ways to reduce Power Usage

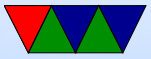


DVFS

- Voltage planes – on CMP might share voltage planes so have to scale multiple processors at a time
- DC to DC converter, programmable.
- Phase-Locked Loops. Orders of ms to change. Multiplier of some crystal frequency.
- Senger et al ISCAS 2006 lists some alternatives. Two phase locked loops? High frequency loop and have programmable divider?



- Often takes time, on order of milliseconds, to switch frequency. Switching voltage can be done with less hassle.



When can we scale CPU down?

- System idle
- System memory or I/O bound
- Poor multi-threaded code (spinning in spin locks)
- Thermal emergency
- User preference (want fans to run less)