ECE471: Embedded Systems – Final Project

Due: Friday, 20 December 2024

Overview:

- Design an embedded system that does something interesting. This is very open-ended, but some guidelines are below.

Guidelines:

- You may work either alone or in groups of two or three. If you work in a group your end project will have higher expectations.
- You may use any embedded board or microcontroller for this project. I may not be able to provide a full amount of help though if you use something other than a Raspberry Pi.
- You may use any programming language you like, but again if it's not in C or Assembler I might not be able to provide a full range of help.
- You may use code/libraries found online as long as licensing allows it and you properly document it. There must be *some* original code done by your group. You cannot just stitch together code found online.
- Your board will have to take input from a user, and display some manner of output. Both of these need to go through one of the low-level hardware interfaces discussed in class (i2c, SPI, 1-wire, GPIO, USB, A/D, PWM, audio in/out, HDMI, etc.)
- Your project may *complement* one being done in another class. However your project must be a new implementation, you cannot just turn in previously done work. Specifically, you cannot turn in work you did that was graded as part of your senior project or ECE498.

Part 1: Topic Selection (due 8 November 2024) (5pts)

Each group should send a brief e-mail describing your project topic and listing group members.

Part 2: Progress Report (due 25 November 2025) (10pts)

A brief status update detailing progress your group has made. This is primarily to make sure your project is on track to be finished in time; if things are not going well the topic can be adjusted.

Send this report by e-mail. Only one submission is needed per group.

1. State in one sentence a summary of your project.
2. Describe the hardware that you will be using: the embedded board, the input device, and the output device.
3. Have you acquired and tested the hardware mentioned? Are you on track for being finished on time?
4. Will you be willing to volunteer to present early (prev Friday, Monday or Wednesday) rather than the last day?
5. You can submit the status update by e-mail.

Part 3: In-Class Presentation 6, 9, 11 & 13 December 2024 (40pts)

- You will have 7 minutes to present. Plan for 5 minutes of showing off the device and presenting plus 2 minutes for questions. Points will be taken off for going over.
- You may present slides using the projector if you want, but that's not strictly necessary.
- Your presentation should have at least the following information. Feel free to include more.
  - Brief overview of what your device does and how it works.
  - A summary of the hardware being used, including the embedded board
    - Describe the input hardware and how you connect to it
    - Describe the output hardware and how you connect to it
    - Hardware limits: describe any power concerns
  - A summary of the software being used
    - A summary of the operating system (if applicable) and the programming language you used, and why.
    - Software limits: describe any real time constraints, and code density concerns
  - A summary of higher-level concerns
    - List any computer security concerns with using your device
    - Discuss if there are any ethical implications of your device (can it harm users? does it violate user privacy?)
  - Challenges/Future Work
    - list any challenges you had getting things working.
    - list any future work, things you might add if you had more time.
  - Leave time to do a brief demo of the hardware. if possible

Part 4: Project Writeup, Due 20 December 2024 (45pts)

This will be a short paper (at least 6 pages, but you can include pictures, diagrams, etc.) that must contain all of the following:

1. Introduction: What the device is and high level overview of what it does. Also be sure to make clear what is actually working in your implementation (as opposed to things you wanted to get work but for various reasons did not).
2. Hardware
   (a) Embedded Board Description: Describe the hardware, CPU (architecture, type, speed), RAM, and I/O. Also describe the operating system or other software (kernel version, etc.)
   (b) Input device description: Describe the device you are interfacing with, how you access it in software, and document the protocol you use to communicate with it.

(c) Output Device description: same as for the input device.

(d) Links to any data sheets for hardware you used, as well as schematics for any circuits you designed yourself.

(e) Power Consumption: Explain any energy or power concerns with your application, and how you could optimize it to use less power.

3. Software

(a) Programming Language: Which one did you use? Why? Briefly explain the tradeoffs between the language you chose and doing the same in assembly language. (If your project is in assembly language, then explain the tradeoffs versus C).

(b) Operating system: does your project use an operating system? Explain why or why not.

(c) Real Time: Does your device have real time constraints? What would happen if your code encountered an unexpectedly large delay?

(d) Code Density: was code density a concern?

(e) If you use code not written by your group (code found online, libraries, etc) explain what the extra code does, and how your code interfaces with it. Explain how much of the code is original to your group.

4. Security / Ethics

(a) Security: Describe any computer security issues there might be with your device (can it be exploited?) If you say there are no security issues, make sure you explain why.

(b) Ethics: discuss if there are any ethical implications of your device (can it harm users? does it violate user privacy?)

5. Related Work

(a) Has anyone done a project like this before?

(b) How does your project compare to existing similar projects?

6. Conclusion

(a) If you worked in a group: List who worked on what part.

(b) Challenges: List any challenges you had, and if things didn't work, explain why.

(c) Future Work: List any improvements you might make if you had more time and resources to work on the project.

7. Appendix

(a) The source code (this can be submitted as a separate file, does not have to be included in the report).

(b) **OPTIONAL** Make a short web-site or YouTube video describing your project. Get it posted on an embedded projects website (`hackaday.com` or similar). No extra points for this, just bragging rights.

You can e-mail your final report to me. pdf or word document is fine, the code should be attached too.

Hardware Ideas:

- Displays:
  - The LED display from class
  - i2c 8x8 LED grid (some available)
  - i2c LED bargraph (1 available)
  - i2c 4-character alphanumeric display (1 available)
  - spi 0.96" 96x64 color OLED display
  - NOKIA LCD display
  - i2c 128x32 monochrome OLED display
  - Red/Black/White 1.5" ePaper display
  - Pimo-roni LED shim 28-LED GPIO
  - Pimo-roni button shim (5 buttons and RGB LED) GPIO
  - 8x16 LCD display (1 available)
  - 8x16 Organic LED display (1 available)
  - 16x2 RGB LCD display (1 available)
  - i2c/SPI adapter (LCD backpack) for the above displays
  - Inky pHAT 3-color e-paper display
- Sensors
  - Weather
    - SPI BMP280 Temp/Pressure/Altitude sensor
    - i2c BMP085 Temp/Pressure/Altitude sensor
    - i2c MPL115A2 Temperature/pressure sensor
    - i2c TMP006 infrared temperature sensor (probably broken)
    - i2c HTU21D temperature/humidity sensor
    - i2c Si7021 temperature/humidity sensor
    - i2c AM2320 temperature/humidity sensor
  - Gas
    - MiCS5524 Carbon Monoxide, Alcohol and VOC sensor
    - i2c SGP30 air quality sensor
    - i2c/SPI BMP280 Barometric Pressure Sensor
  - Light/Color
    - i2c AS7262 color sensor (2 available)
    - i2c APDS9960 Light/RGB/Gesture sensor
    - i2c UV SI1145 UV sensor

        * i2c TCS34725 RGB Sensor
        * i2c TSL2561 Luminosity sensor
    - Distance
        * A/D distance sensor (1 available)
        * i2c VCNL4010 proximity/light sensor
        * i2c VL53L0X distance sensor
    - Current / Power
        * i2c INA219 high side current sensor
        * serial USB power gauge
    - Position / Navigation
        * i2c accelerometer MMA8451
        * accelerometer ADXL335 (analog)
        * i2c HMC5683L triple-axis magnetometer (compass)
        * serial fw5632 GPS receiver
    - Water
        * Water level sensor
    - Sound In
        * Silicon MEMS microphone
        * i2s microphone
        * MAX4466 microphone
- ADC
    - i2c ADS1115 16-bit ADC with amplifier
    - i2c ADS1015 12-bit ADC
- Motion
    - Motor, lego-compatible
- Sound Out / DAC
    - i2s Stereo DAC UDA1334A
    - i2c DAC MCP4725
    - “Pirate Audio” Hat i2s/dac/speaker/display
- Storage
    - SD card breakout
    - i2c NRAM/FRAM breakout
- Identification
    - Fingerprint Sensor

- I/O
  - 1-wire DS2413 two GPIO expander
  - i2c AW9523 GPIO expander
  - USB micro breakout
- Embedded boards
  - Trinket 5V 16MHz
  - Trinket 5V
  - Trinket M0
  - Gemma M0
  - ESP2866 Breakout
  - TS-7600 embedded board
  - Adafruit KB2040 board
  - Pi Pico W
- Custom interfaces
  - OBDII (car telemetry) to Bluetooth adapter
  - PS/2 keyboard adapter
  - Wii nunchuck / breadboard adapter
  - Wii controller (bluetooth)
- Other hardware you can obtain on your own. Some ideas:
  - USB keyboard/mouse/disk
  - Interface to SD card over SPI?
  - Devices hooked to GPIOs (LEDs ? Motors?)
  - USB or rasp-pi web-cam
  - i2c Wii Nunchuck (I have breadboard adapters, you'd have to provide your own nunchuck)
  - Bluetooth devices

A few Project Ideas:

- A list of projects done in previous years can be found here: `https://web.eece.maine.edu/~vweaver/classes/ece471_final_projects/`
- Alarm Clock: set time with buttons, play wakeup sound/music over audio out
- Some manner of robot.
- Wii Nunchuck (i2c accelerometer). Show orientation on LED display? Make a simple game? Log acceleration to disk?

- Connect devices to Pi3 via bluetooth (phone, Wii controller, etc). Note: bluetooth is a pain to get working.
- Weather/Temperature Display that remembers high/low temperatures
- Audio in on the sound input driving some sort of audio visualization on LED display
- Some sort of video game utilizing LED display
- Color sensing candy sorter
- Car or bike computer
- Hooking old PS/2 style keyboard to Pi using GPIO interface
- Measure the power consumption of Pi doing various things, optimize for low-power usage.