Raspberry Pi: Electronic Any Surface Drum-kit

CPE 555: Real-Time Embedded Systems

Stevens Institute of Technology

May 8, 2015

Jorge Rojas

Dr. Hong Man

I. Abstract:

A drum kit is not only one of the most expensive instruments, but it also is one of the hardest to transport. This becomes even a bigger ordeal when one wants to add variety of sounds with different cymbals, toms, or snares that occupy more space and costs more money. Through the use of a Raspberry Pi, some RGB (Red-Green-Blue) LEDs, piezo elements, and an ADS1115 analog-to-digital converter, it is possible to make a cheaper, and more portable, electronic drum-kit. After successfully connecting the devices together, with a modular git repository that makes it easy to set up any Internet-connected raspberry pi, it was possible to get readings from the vibration sensors, output different sounds, and turn on different color LEDs based on which piezo element gets triggered. Nonetheless, a future functionality that could be added is using multi-threading within the python code to make independent processes for each channel of the analog-to-digital converter (ADC), allowing the user to make harmonies with the different drum sounds.

II. Bill of Materials (BOM):

1. Micro-SD card with at least 4-GB memory
2. Raspberry Pi 2 model B
3. Edimax 150Mbps Wireless IEEE802.11b/g/n nano USB Adapter
4. Adafruits ADS1115 4 channel ADC with programmable gain amplifier
5. Four 3W RGB Color High Power LED Light Emitter 4 pins (20mm)
6. Four Piezo elements
7. Two breadboards
8. Ten female-to-female wire headers and 20+ jumper wires
9. Four 1-Mohm resistors

III. Hardware:

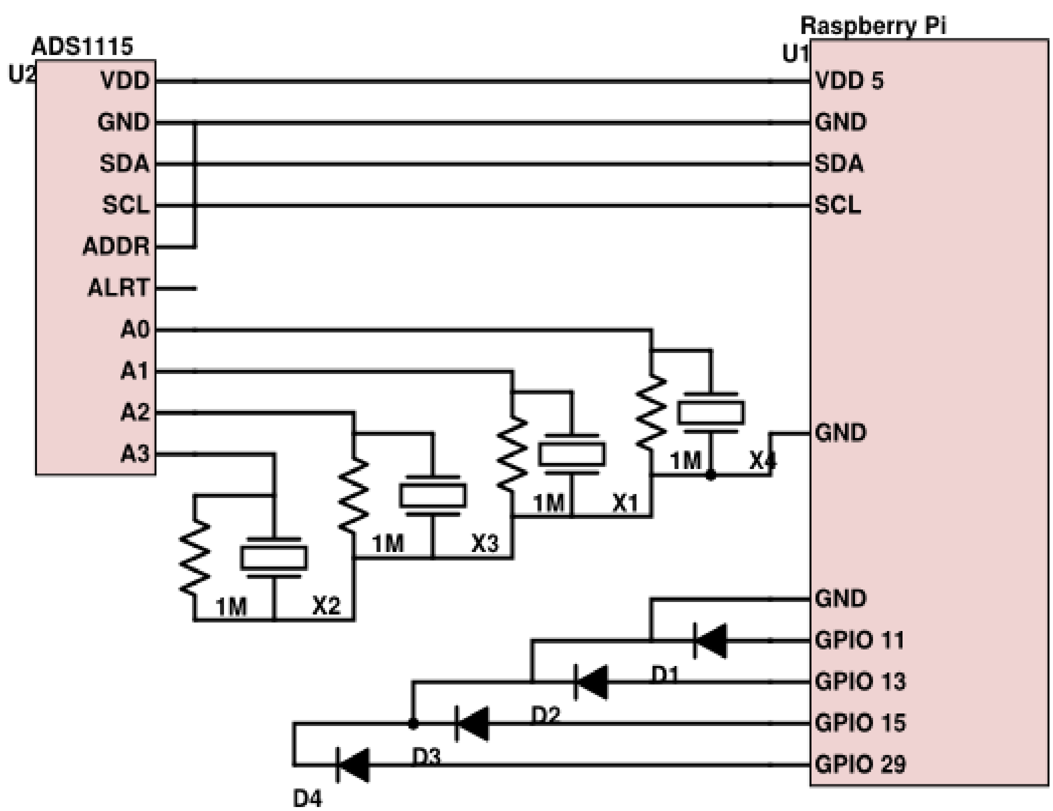

Illustration 1: E-drum Hardware Schematic

From the illustration above, it is important to note the separate ground pins for the LEDs, piezo elements, and the ADS1115 ADC. The ADC becomes necessary from the use of the Raspberry Pi, since it doesn't come with its internal converters, which complicate the circuitry a little more. Nonetheless, said device communicates with the Raspberry Pi through an I2C bus, which brings the need to enable such protocol in the micro-controller. Another detail worth noting are the LEDs. In the schematic, the RGB LEDs are viewed as individual diodes when in fact they are an array of three diodes, one for each color (red, blue, and green); however, for the first three only one of them is used. For diode number 4, it would be necessary to turn on both the blue and red LEDs to generate a purple color and differentiate from the other sounds. Currently, diode 1 is red in color, diode 2 is blue, diode 3 is green, and diode 4 would be purple (combining red and blue).

Finalized product is seen in the illustrations below:

IV. Software:

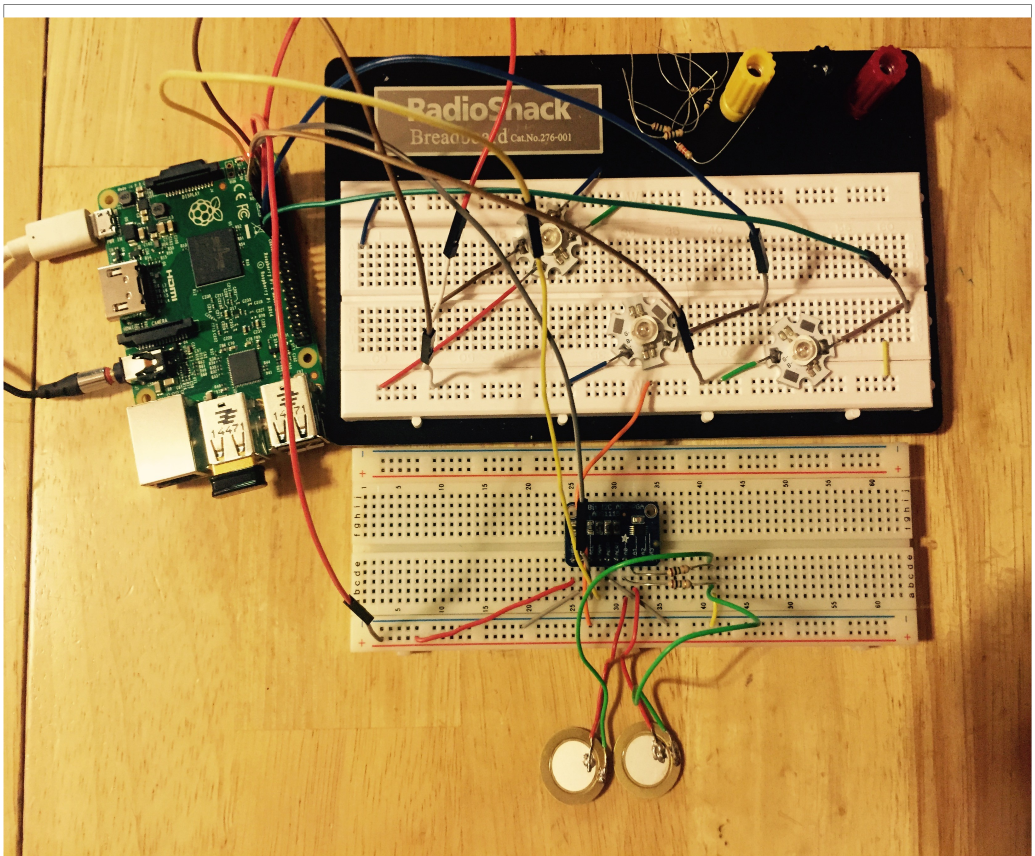

Illustration 2: Actual prototype Hardware

The Raspberry Pi doesn't ready to plug into this product out-of-the-box, thus there is software that needs to be downloaded. Nevertheless, before progressing any further into the setup, it is ideal to have a Linux system such as Ubuntu, since this micro-controller has Linux running at its core. Moving forward, to setup the Pi the SD card needs to be booted with the NOOBS Raspbian operating system, which allows the user to use the GUI interface. To set up your SD card for this feature, see references [1].

Once installed, a mouse and keyboard attached to the Pi would allow you to boot up the GUI, and with the Edimax Wi-Fi addapter, connect the device to the nearest hot-spot. After a connection is complete, python 2.7 with GPIO needs to be downloaded. Also, the Adafruit library for ADS1x15 allows for easy communication between the pi and the sensors with a couple lines of code [2]. However, this library is included in the GitHub repository, which gives more setup instructions needed to be able to run the code. Finally, enabling the micro-controller for I2C is crucial for the successful implementation of the electronic drum-kit [3].

Since Python is an interpreted language, there is no need to compile code, which makes programming and execution really simple, since there is no need for a separate compiler. Thus, with all the hardware mounted and the Pi ready to run the code, open a terminal in the GUI or establish a secure shell connection with the Raspberry Pi and navigate into the directory where the main program lives. Once there, run *sudo pytho*n cpe555_edrum.py *and the code should start running.*

V. Results:

A successful prototype was developed to work as a proof of concept. Nonetheless proper acquisition of the analog sensor readings was successful, as seen in illustration 3. Also, the appropriate sound is correctly heard on the headphone jack output when the sensor is triggered altogether with the correct LED mapped to a certain drum, which is seen in illustration 4. The only drawback is the strict sequential execution of the code, which creates a delay and halts the polling of the sensors during a short period of time. Also, it prohibits harmonizing different sounds to create more elaborate rythms.

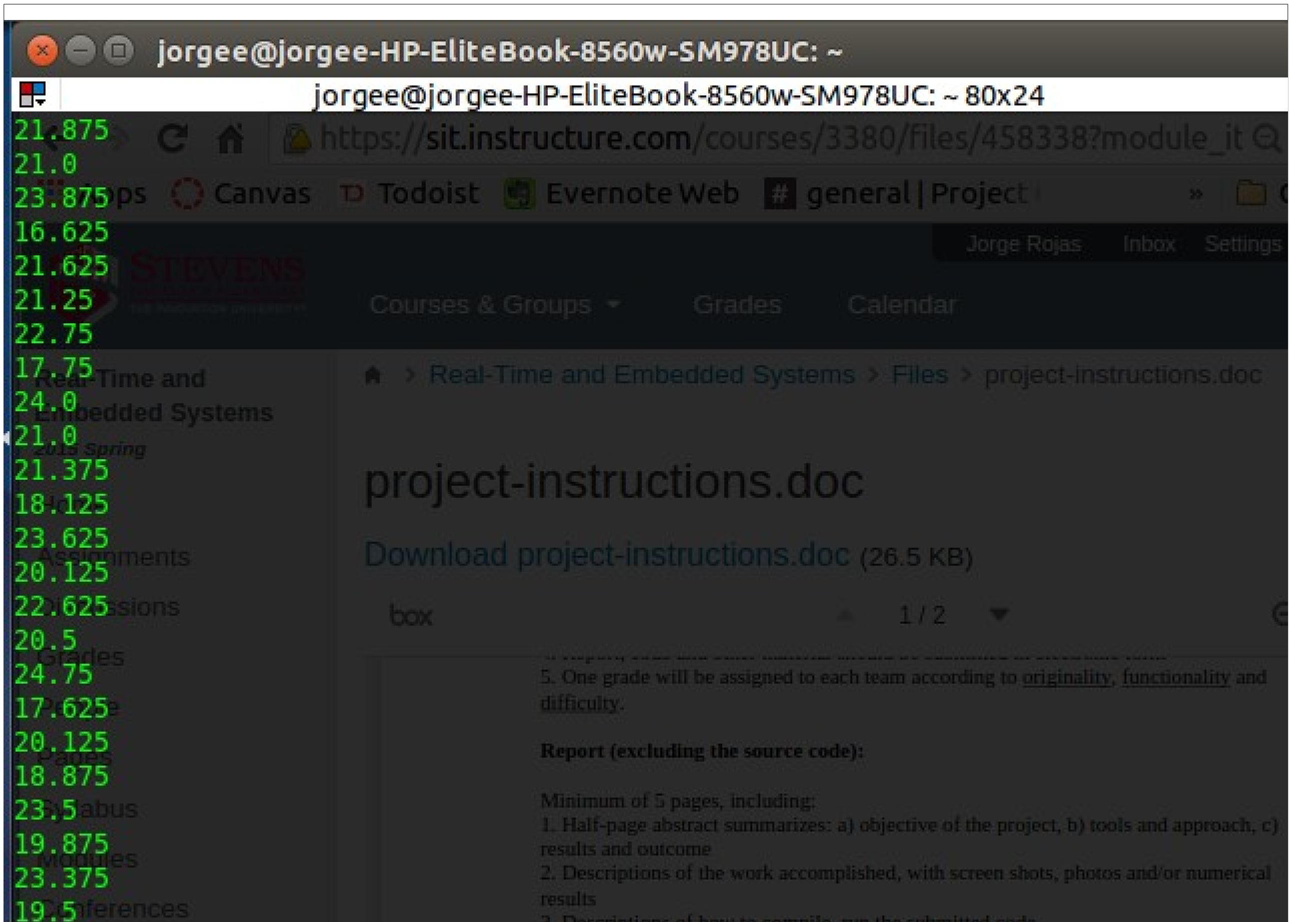

Illustration 3: ADS1x15 Digital Output

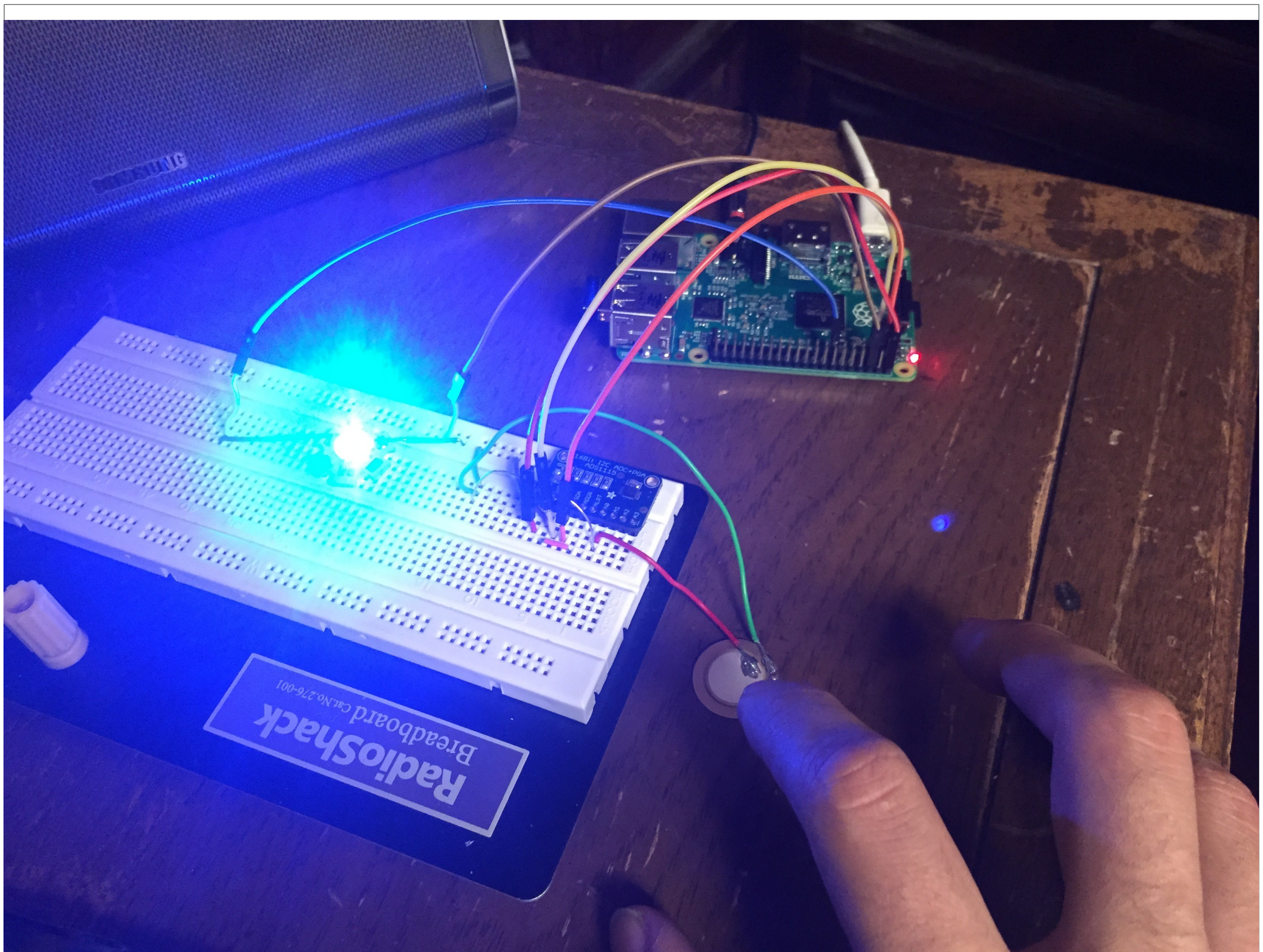

Illustration 4: Working Prototype. Activation on tap

VI. Conclusion:

There were a couple of conclusions from the development of this prototype: multi-threading will increase performance and every four more sensors included will require another ADC. Multi-threading will solve the delay issue; parallelizing each channel with a different process could allow faster performance, allowing real-time drumming a reality. The added feature would not only improve the experience, but it will also increase the value of the device as a product. On the other hand, one of the main benefits of this project is its scalability. Adding more sensors, LEDs, and sounds is extremely simple, which alludes equally to the simple drummer as much as the more experienced one. However, that comes at the cost of money and area.

Besides the using threads to parallelize the program and including more converters to include more drums. It is possible to keep an array of only 4 LEDs and use the extra pins to control the color instead of having a dedicated LED for each drum sound. This will reduce size, cost, and more space for more sensors. Regardless of all these possible features and/or improvements, the project did provide exposure to topics such as I/O, threading, scheduling, and among other topics, that really gave a practical perspective of how micro-controllers such as the Raspberry Pi are engineered for performance.

VI. References:

[1] NOOBS Installation: http://qdosmsq.dunbar-it.co.uk/blog/2013/06/noobs-for-raspberry-pi/

[2] Adafruit ADS1x15 tutorial: http://openlabtools.eng.cam.ac.uk/Resources/Datalog/RPi_ADS1115/

[3] Adafruit I2C Configuration for Raspberry Pi: https://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i2c