CSM117 - Team BitBites: Candy Dispenser

Joel Park
Akshaya Subramanian
Tianhao Zhao
Franklin Zhang
Spring 2016
June 1 2016

Abstract:
In this project we will use an Android application to send commands over Bluetooth to a Raspberry Pi 3 to control a servo and dispense candy.

Background:
Bluetooth is a Frequency Hopping Spread Spectrum communication system on the 2.4GHz ISM band. It hops 1600 times a second based on a specific code sequence, giving it security and noise resistance. It is commonly used for Bluetooth keyboard/mice, Bluetooth speakers / headsets / handsfree, and smartwatches.

Motivation:
The motivation for this project is to gain an understanding of wireless networking and the process to implement this. We wanted something fun and relevant, while also being realistic in the timeframe we had. We decided to go with a Candy Dispenser because we see mobile payments on the rise often controlling hardware. This allows us to explore one possible implementation of the system.

To fulfill the requirements of this project, we had to build a mobile phone application and use a wireless network of our choice. As many of our team members were Electrical Engineering majors, we also wanted to add Hardware to the system.

Wireless Network Used:
There are a couple different networks that we could have used. We could have use Bluetooth, Wi-Fi, NFC, or some other wireless system. The Wireless system we used is constrained by the fact that it must run on a mobile device which limits us to Bluetooth Wi-Fi and NFC.

Wi-Fi: The consideration for this was a real world rollout of the system. In the real world, Wi-Fi access is not guaranteed, is very power hungry and has significant overhead which we don't want to have in a system like this. Additionally there significant complexity in setting up a internet server to handle all of the requests. Internet connected things is a significant security risk as it is very difficult to secure a hardware device against malicious attack from the open internet.

NFC: Not all phones have NFC. This makes development hard and limits our market to phones that have the NFC. The range is also very short, forcing the phone to be held directly to the tag. Lastly, the classroom lab did not have active NFC components, and getting such materials was an unnecessary cost and complexity.

Bluetooth: Bluetooth is commonly available in all modern smartphones we can address the widest market possible for those trying to use our system to dispense candy. Bluetooth is very low power, especially things like Bluetooth Low Energy (BLE) which allows the system to run isolated for long periods of time off limited power.

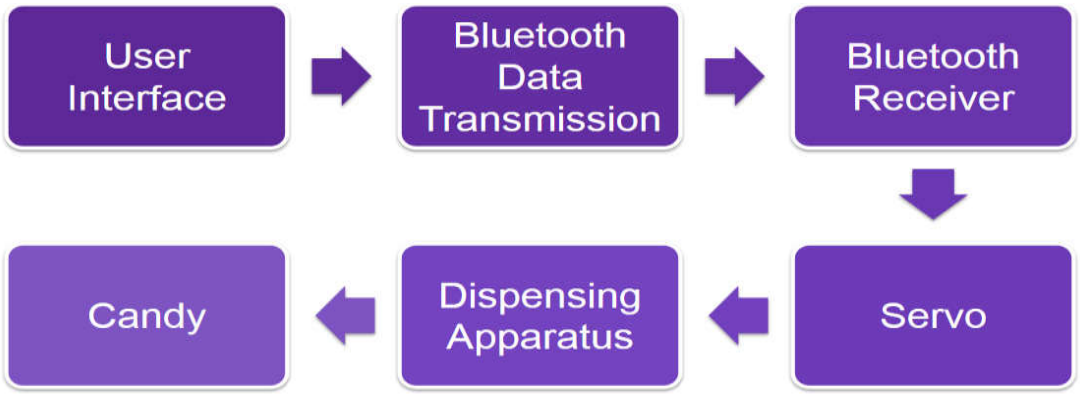
Goals:
The goal of this specific project was to build a Android application to make product selections, build a Bluetooth activated dispenser with the Raspberry Pi, and to learn about different Wireless Systems and how to implement one of these.

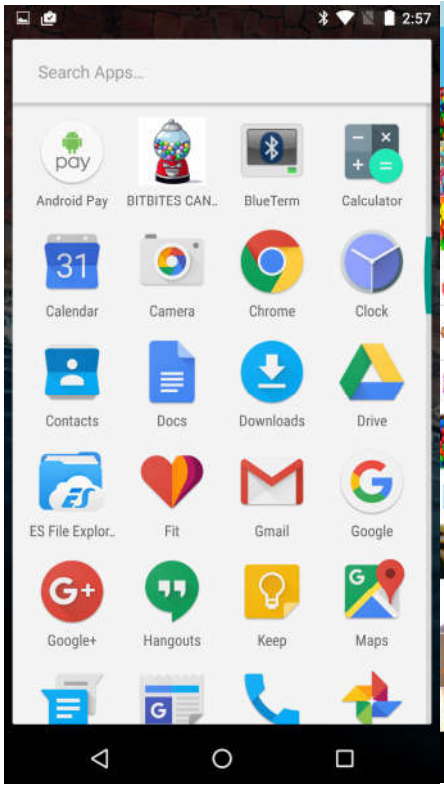
Architecture Details:
To gain a better understanding of the direct programming systems, our project was developed on an Android Studio for the Android system. We used the Bluetooth Network to communicate with our hardware which was a Raspberry Pi. The Raspberry Pi acted as both the Bluetooth receiver and the controller for the servo. The materials used, in addition to the Raspberry Pi, Android Phone, Servo, physical structure components and candy which was in our case Hershey's chocolate. Testing was done on an Samsung S7 and a Nexus 6 which we had received as part of the class for testing. We used Blueterm which allowed us to connect to the Raspberry Pi and send it arbitrary commands. This allowed us to test the Raspberry Pi receiving program to see if it was doing the correct things while the Android program was under development.

The first step to make a purchase was that the user would open up the Big Bites Candy Dispenser application. In the user interface they would be able to make their selection of candy. Once they selected the candy and made a purchase, the app would send Bluetooth data transmission which to the Raspberry Pi which acted as a Bluetooth receiver once. Once it was received, the Pi would verify the commands it then would make the appropriate pulse width modulation changes to change the angle of the servo. This would open the flap mechanism, thus activating the dispensing apparatus. This would release the candy which the buyer purchased.

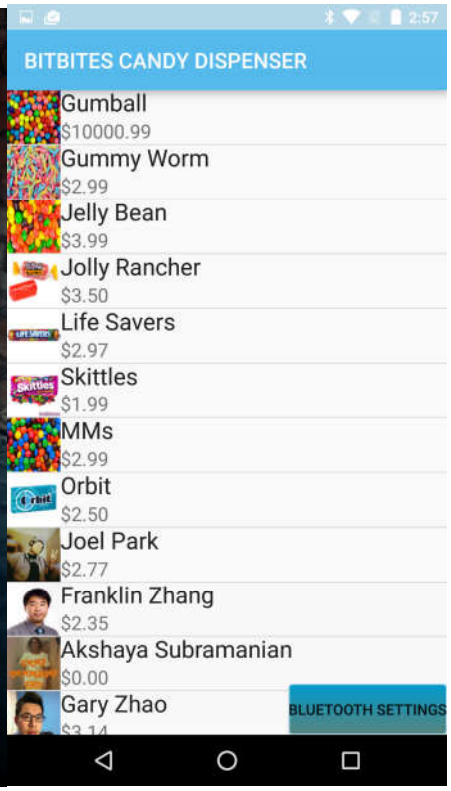
From the android phone we were able to verify the presence of a Bluetooth adapter, change the adapter state and connect and send data over Bluetooth in the following screenshots. Due to physical constraints, we did not actually have multiple candies.

The Raspberry Pi program was coded in Python, using data gathered from internet sources. Its purpose is to open the serial port, read data, verify its correctness, and then activate the servos.



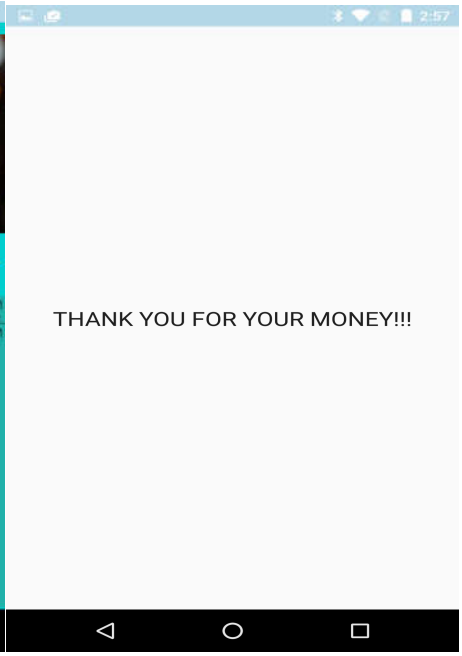


Homescreen

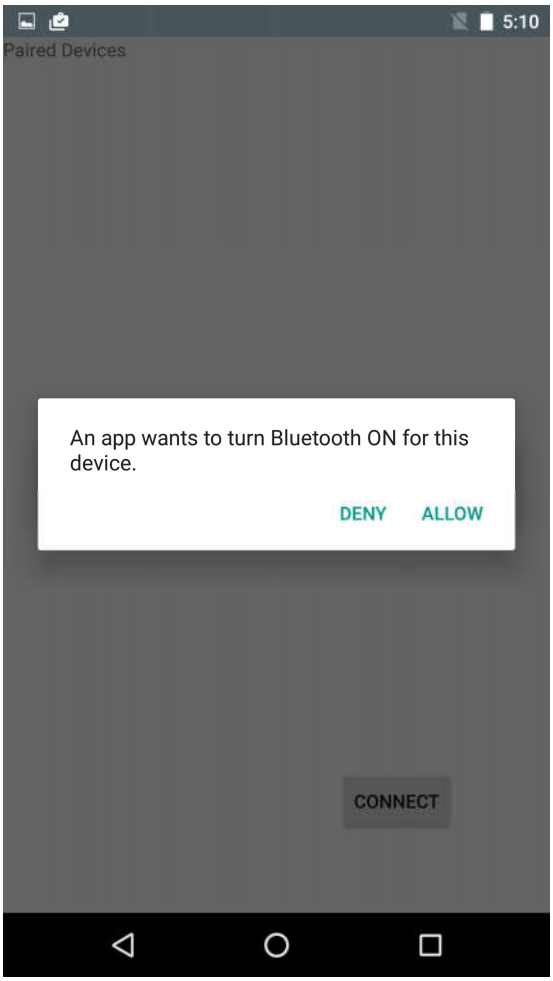


Application

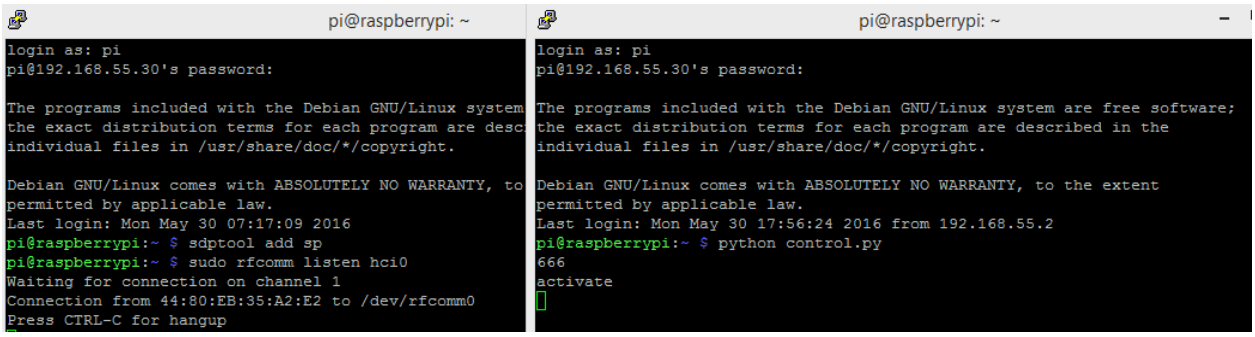
Purchase



Transaction Complete



Bluetooth Control



Raspberry Pi Terminal

Demo can be found at https://www.youtube.com/watch?v=sStwDj6lKY8

Work Division

Joel Park worked on the Android application development. He created the Graphic User Interface

Franklin Zhang which works on Android application development. He specifically focused on the Bluetooth side of the application, along with fixing the Raspberry Bluetooth and pairing process. He also helped with parts selection for the project.

Tianhao Zhao worked on the Raspberry Pi where he worked on receiving signals from the application and also activating the servo by changing the PWM.

Akshaya Subramanian worked on the physical structure and building the dispensing mechanism. She also put a significant part of the slideshow together.

Difficulties:

There are many difficulties with a project like this. On the software side, Android Studio has a very steep learning curve. As its extremely powerful, there are many different options, from what type of emulation to do, to different xml app bindings. We wanted to learn how the app actually worked, so we didn't use an online code generator.

Bluetooth is extremely complex, and sometimes finicky. Implementing Bluetooth yourself is a complex operation as there's many different commands required to first check for the presence of a Bluetooth adapter along with verifying its state. Changing state and pairing and sending commands is additional complexity. While testing, we used Blueterm to send commands. This works, however it is actually keyboard dependant. We had switched away from the default keyboard due to user preferences, and that somehow wasn't recognized by the programs in use.

The Raspberry Pi does not actually have Bluetooth enabled with the default operating system. It must be manually enabled and the operating system updated it in order to enable Bluetooth. There isn't that much documentation on Pi Bluetooth, and the little documentation that does exist is fairly sparse on the details it provides and what we had to do a lot of for reading in order to find the true process of making it work. The GUI for Bluetooth on the Pi doesn't seem to pair with devices require a pin code, so we had to use the terminal commands to set it up. It's hard to build a python application that can directly read and open up a Bluetooth Port so the easiest way to do it was to set up another program to do that. Some final errors include typing "y" instead of "yes" when asked to pair. Anything other than "yes" is accepted as "no", thus leading to many failed authentication attempts.

On the hardware side of things, we had difficulties with the servo. We failed to realize the significance of the weight of candy, leading to frequent flapper failure. We also burned out 2 servos in the process.

Future Work:
Future works include meshing many different wireless networking options. We could have a NFC tag which provides the instructions on how to pair with our Bluetooth system. The Pi could also be connected to the internet using Wi-Fi and have the ability to report statistics to an online server, while generating sales reports and updating product stock. The android application's product selections could also be able to be updated from the web instead of being hard-coded into the application. Updates could be rolled out to all hardware in use.

Additional work needs to be done to improve the reliability and security of the system. Currently, the Bluetooth software only checks for a specific phrase, where it activates. In the future, there should be some type of checksum / hashing system to verify the authenticity and validity of such commands. The hardware needs to be improved to dispense multiple types of candy, and a whole tube of them rather than small quantities.

Conclusion:
Overall this project was a success. We built a working android application proof of concept, by showing that we were able to dispense candy. We had the android phone connected to the Raspberry Pi, paired, and sending data. We had working hardware dispensing on command, shown in the demo. We all learned a significant amount about Bluetooth implementation and had fun doing so. While undergoing some difficulties with the hardware, the project helped us get an understanding of Bluetooth and other modems of wireless technology.