PLD, Your One Stop Lock

Group Project

Jason Hedlund

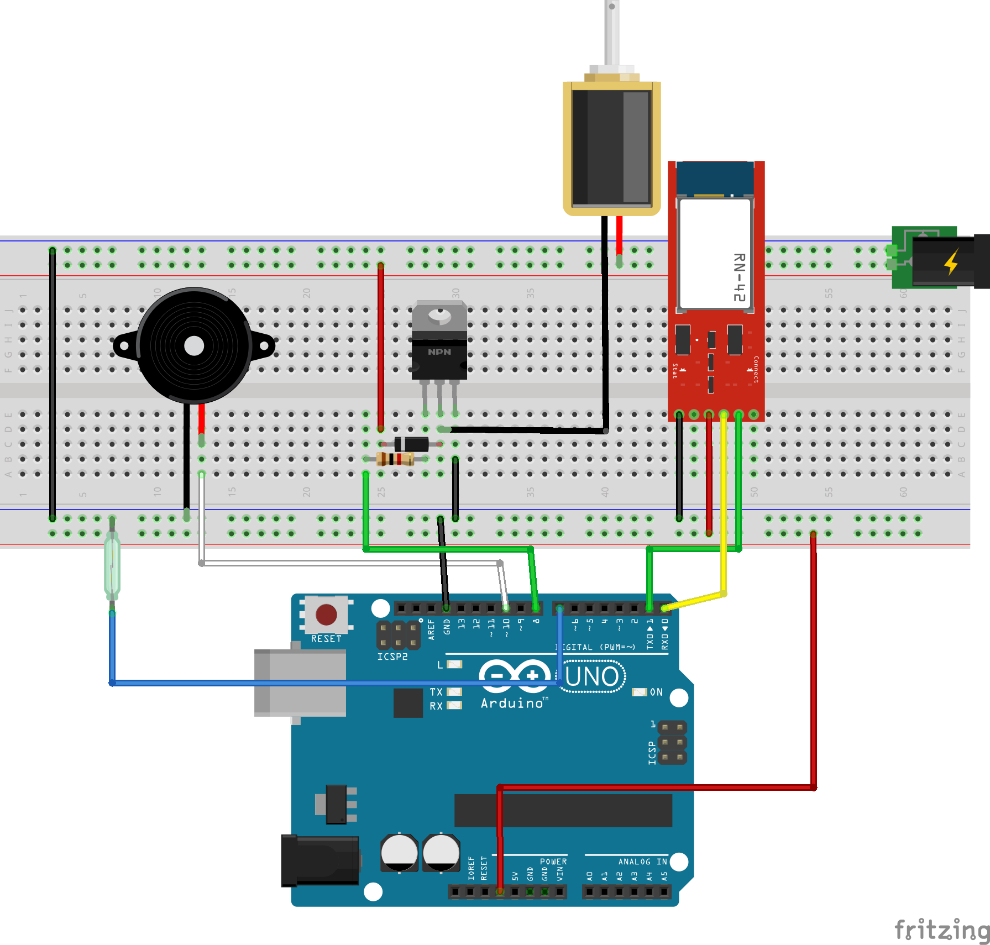
[jhedlund@hawk.iit.edu](mailto:jhedlund@hawk.iit.edu)

Mark Sherman

[msherma3@hawk.iit.edu](mailto:msherma3@hawk.iit.edu)

What Mark and I first envisioned with our device was to make a home automated system that was capable of checking the status of windows, doors, and other entry ways into your house or apartment. Among this, we wanted to be able to control and communicate to this device through a cell phone or other wireless device such as an X-bee. This sounded like a good idea at the time, but the idea was later scrapped as we felt like it was overused product in an already flooded marketplace. We were struggling with the complication of the device and how to make it into something that was more redundant and simple to use. This is how the true Pop-it, Lock-it, and Drop-it device was formed. Taking our initial idea of making a modular locking system we discussed that it would be much more feasible to just make a simple lock that could be used in a more universal setting. We stressed the idea that when people buy locks in the first place, it’s not always simply for a door or for a home, but rather items such as cabinets or tool boxes. Our spectrum had changed a little, since you can very well use PLD still on a doorway or window, but we also saw the real demand for more applicable uses. Our purpose was to fill a void in another device that had not seen much attention in the “smart” world, as we had searched google for smart locks, but had felt like they had not meet what we were trying to accomplish.

Our vision was to keep the lock all around simple, but add functionality to it that only something like an Arduino could offer. The name PLD was formed during a brainstorming session when we had still had the old system in mind. Though the purpose of the device changed, we felt like the name still fit as essentially you were doing the same actions, but the sophistication of the device had narrowed as with the new plan. Since we were trying to keep the device sounding interesting, as most people would not think of a lock as a fun device, we settled on more of a comedic product name. Though the name seems to distract what the device is really about, it does encompass what we are trying to convey with the product. Pop-it, as in unlock the device; lock-it, as in close and lock the device; drop-it, as in drop your worries. We wanted to keep the name catchy, as in, even if you didn’t particularly care for the device, you would still know the name. The point being, we were aware that our product had turned into a more simplistic device in both design and usage, but we wanted to make up where we had felt that we were dragging behind.

As stated previously, our intention with PLD had skewed from what he originally been thought out to be. With that in mind, we had bought most of our parts with our previous product in mind, so we had to be improvisational. At the heart of our product we used an Arduino Uno, this would be used for all our logic and sensors to detect when and if the lock was to open. Our lock itself consisted of a solenoid and a circuit to drive that particular solenoid. We decided to go with a solenoid, because it would accomplish what we wanted with a simple back and forth motion, as in, lock or unlock, without adding more unnecessary components. At first we used a breadboard to get a feel of what to expect as far as spacing and component placement, but we did end up using a more permanent solderable project board. This was also so that we could fit the project inside of a plastic box that we rigged up to look like a small little safe for demonstration purposes. We also wanted to be able to show that it was capable of being stuffed inside a small enclosure, and made to your liking. Our circuit itself was mainly driven by a transistor that would act as a gate to allow and stop voltage from getting to the solenoid. We added a diode between the two pins of the solenoid so that the two sides would not short out on each other, such that the solenoid would not become inoperable. The solenoid itself was powered via a 12V 1A power supply reused from an old router, it would not operate on 5V since it needed more amps than a USB port could handle safely. So in summary, the Arduino would send a HIGH signal from an pin 8, pass through a 1K resistor to slow down the voltage, then travel to the collector leg of the transistor, activating the gate, which would allow the circuit to flow through from ground, which then the solenoid would unlocking. We included a small piezo speaker to go off when the device was unlocked via pin 10 of the Arduino, this way if you did not hear the pop of the lock, you would have something again for redundancy. How we actually unlocked the device is where we got a little creative. We had originally planned to use magnetic switches to check the status of doors and windows to see if they are open or locked, we had decided to reuse this type of reed switch to instead unlock the device. We decided we could hide the switch in plain sight, where you could hide the switch wherever you want inside the enclosure. The magnetic reed switch was connected via ground and pin 7 of the Arduino, what happens is that when there is no magnetic that is touching near the switch it would report as a 0 or LOW to the Arduino, but when a magnet was induced it would read as HIGH as the voltage would flow through as the switch is activated, this then would tell the Arduino to send a HIGH signal to the solenoid. Finally, the wireless connectivity we had decided upon was Bluetooth, because most if not all people has a cell phone with Bluetooth connectivity. With a simple Bluetooth application downloaded via the Android app store, we could send a signal from a Bluetooth module connected via the RX and TX on the Arduino an “ON” command, which would push the letter “A” to the Arduino. Then we simply coded it so that if the device sees an “A” to set pin 7 HIGH for the solenoid on the Arduino. Our code itself is only 35 lines long, about as simple as the hardware itself. The whole program fits into a loop that if a certain value is seen, in this case either HIGH or ‘A’, then set pin 7 HIGH and play a tone on the piezo. We added a little bit of a delay too for how quickly you can lock or unlock, because we didn’t want to overheat the transistor or power supply since the solenoid does draw that much power. 

We worked a lot on the project together, we meet a few times over the course of the 4-5 weeks we had to do the project a couple hours earlier before class and just discussing what we should do via text messaging. What can be said about what I, Jason, included solely can really only be summed up in designing parts of the circuit, troubleshooting the soldered circuit board, and making the project enclosure. The longest and most hands on part of the entire build what easily of been the project enclosure. I have a lot of parts laying around my workbench and I thought it would be simplest to just use parts that were laying around, nuts, bolts, etc. So I designed the project enclosure lock latch with some metal pipe hangers and some bolts, and drilled and screwed everything together so that it would be a little firmer and could take a little bit of abuse. I also added a hinge and handle to the enclosure so you could easily demonstrate the locking and closing of the device. It probably took me a good 6-7 hours to cut out all the holes, mount the solenoid, soldered circuit board, Arduino, and switches. If I would have had more time I would have loved to actually 3D print an enclosure and do it more right rather than such a crude prototype, but I think of it as with the Apple I, it was designed in a garage, and I was designed this in my basement. Just for kicks, I threw together a little logo in Photoshop and printed out a nice vinyl laminated sticker to throw on top of the enclosure just to give it a little more flair.

Designing the circuit was relatively easy, there was a simple guide on Adafruit for what type of transistor and diode you would need, and the relative layout of where you could connect the Arduino. Though it lacked a lot of technical knowhow to set it up in layman’s terms, so I did have to consult to google quite a bit, to make sure I wouldn’t end up frying any of the components. Everything was working just fine on our breadboard when we had finally managed to throw everything together for a final test before transferring components onto the solderable board. After Mark and I had worked on the soldered board in class, it was very finicky and it was not quite working right. So I took it upon myself to take it home to troubleshoot, since I have a lot of tools at my house and more space to work with. The problem was the lock was not going off, instead the speaker would chime and that was it, so initially we thought it to be a bad solder joint or that we had simply miswired the board itself. After going over each component for around 45 minutes I had noticed that if you had touched the Bluetooth board to the circuit board itself, the lock would actually fire! I thought at first it must be a cracked solder joint on the board itself that was reconnecting with the stress of the board, but what it happened to be was that the Bluetooth was bridging the Arduino ground to the main ground of the board. Simply, we had forgotten to ground the Arduino to the main ground of the board, but the Bluetooth was grounded to it, so we simply added a jumper wire between where the Bluetooth had been grounded by the Arduino and the main ground. After I had applied the fix we had no more issues regarding this and we were on full schedule to present the project.

The biggest challenge that Jason and I encountered with our final project was deciding on what exactly we wanted the project to be. We both were going toward something in the home automation sector, but didn’t want to do something that has already been done a million times.

We ended up settling on a smart locking system as a results of not wanting to build a locking system that could only be used for doors. The components themselves would not change much, we would just be using them in a slightly different way, more modular way.

Another setback that we encountered was the delayed arrival in a couple of key parts, such as the transistors, that were crucial to allowing things such as the solenoid lock to even work. Once all the parts arrived we spent some time figuring out the most efficient way to trigger the lock to open, with the least amount of code. We then set the lock up to open with the activation of the reed switch, this allowed us to test the most basic of functionality. The last step in the hardware side of things was implementing a more secure input device to trigger the solenoid to unlock the lock.

We also were experimenting a bit to figure out the best communication device to use between the Arduino controller and the input device, we went over the possibilities of using Xbees, GSM, and several different Bluetooth modules. Any of those modules would have worked, but the thing we kept noticing was that a lot of them were far more complex than what was required for our implementation. All we were really needing to do, was to send a message from the input device directly to the Arduino, no bells or whistles necessary for this component.

Since simplicity was one of the main things driving our project we decided to go with the Bluetooth RN-42 module from Parallax, this module offered us a basic TX and RX port used to send serial communication, which was all that was needed. This module in specific was a bit more expensive, at $59, than some of the other modules, but it was available from Fry’s Electronics and we did not want to wait for the shipping. The module itself is essentially just a RN-42 Bluetooth chip mounted to a breakout board with pins and a voltage regulator, which is great for testing purposes, but not necessary for function. The same exact chip can be found only without the breakout for anywhere from $5-$15, making it cheaper than any of the others.

Upon finding the best communication interface we had to do a bit of research to find the best way to connect it to the Arduino and connect to the Bluetooth itself. There was very little documentation on this specific module, so it basically ended up with us piecing together what we could find in a few places to make it work for us.

Once the Bluetooth adapter was correctly configured and connected, it was time to find a cell phone app that would allow us to control the system. After various trials with around 10 different app, we once again were able to find the one that was not overly complex for our project, and free, it was called Arduino Bluetooth Control for Android OS. This app would connect to a paired Bluetooth device and send basic serial output to the Arduino, allowing for us to trigger the lock when the corresponding serial input was received. Most of the apps were written to communicate with the Bluetooth Low Energy interface, which was not what we had, so this limited us, but still worked just fine.

Once the Bluetooth was all set, we adjusted the code to allow for the lock to disengage when the correct input was received over serial to the Arduino, as well as when the reed switch is engaged. This allowed for the reed switch to be a fail-safe in the event that the user doesn’t have their phone, or they do but the battery dies.

After getting all of the components working via the breadboard we moved it all over to a soldered board that allowed for us to tidy things up and gave the project a smaller foot print. At this point Jason worked to build a basic project box to demonstrate our project on presentation day. This allowed us to show the project working and the features it held.

In the future Jason and I are looking into creating a PCB of our project to give it an even smaller foot print, we won’t even need the full Arduino Uno at that point, for we can just use the ATMEGA chip on the PCB. We both have plans to use this project for our own personal uses for security, and are very much looking forward to the final results.