The Automated Locker Bank

Design Document

Prepared by:

Victor Baziuk, Donald Brant, Michael Deng, John Hunter, Muthana Mohammed, Bryan Moy, Ryan Szczecinski, Carmen Tam, Stephanie Yu

Prepared for: Alden High School and Bulldog Manufacturing

Client Liaison: Donald Brant (dmbrant@buffalo.edu)

Contents

Introduction

The goal of this project is to design an automated locker system for use at Alden High School and provide manufacturing specifications to their associated student-run manufacturing business Bulldog Manufacturing. The NFC Locker Module utilizes NFC (Near-field Communication) for authentication of both registered users as well as administrative users. The NFC Tag Batch Writer has also been designed to allow for easy deployment of a list of NFC payloads. Additional design considerations include the creation of an enclosure for the NFC Locker Module that simultaneously acts to protect the module and prevent the lock mechanism from jamming. For the safety and security of the school a mechanical override system must also be incorporated to allow for the circumvention of the digital systems. Due to the potential for a large rollout of this system scalability has been a key consideration at every step of development.

A Note on Development Constraints Due to COVID-19

This recent global pandemic has impacted all things existing within its context and the development of this project is not immune to that reality. Our sudden move to distance learning has scattered our team to the winds, leaving us each to deal with our own novel stresses to this novel virus and limiting our potential for collaboration. Disruptions to distribution chains have also resulted in a hardware based project that lacked the hardware necessary to build and test our designs. Even now as the end of the development cycle approaches there are parts crucial to our design that still have yet to arrive and thus have yet to be tested. With a singular design lacking in the modularity that would allow for hardware development by multiple team members each in isolation, the parts that have been received were funneled to a single team member, creating a bottleneck for hands-on development. Additionally, the closure of the Alden High School campus has put the development and testing of an accurate, well-fitting, and reliable enclosure out of our reach. Likewise for our ability to test various power options on-site. With all this in consideration please proceed through this document with the understanding that the design described within is incomplete.

System Components

NFC Locker Module

The NFC Locker Module is the core component of this project. It provides authentication of registered and administrative NFC tags, as well as administrative functionality for registering new tags and clearing all registered tags. This system uses UID based authentication, relying on the UID value assigned to each NFC tag. These UID values are globally unique and read-only meaning that under normal circumstances cloning a tag should not be possible. In our research we have found that given the right (paid) tools, a deep understanding of NFC technology, and physical access to the desired tag, it is technically possible to clone an NFC tag down to its UID however it is also true that given the right tools, knowledge, and physical access it's also possible to pick a lock and as the current lockers rely on master keys for traditional authentication override, and our automated system relies on those same locks and keys for our mechanical override we have come to the assessment that the same method for fallback security used for the current system will work for our new system, namely hallway security cameras. Another disadvantage of using NFC UIDs for authentication is that most modern NFC capable smart devices utilize a dynamic UID system, meaning that the UID changes with each scan. The decision is then between supporting smart devices by using payload based authentication (which would make potentially unwanted tag duplication much easier) or rely on the standardized security of NFC UIDs. We chose the latter however the code for this module is implemented in a modular way that would allow for a transition to payload authentication.

As mentioned previously, the NFC Locker Module utilizes two classes of tags, registered tags (for student use) which are unique per locker and administrative tags which are globally hardcoded at installation. There are three types of administrative tag:

1. Master tag: acts as like a master key, allowing opening the lock mechanism on any locker.
2. Registrar tag: Puts the locker into registration mode, allowing for registration of a new student tag. Student tag registrations persist until cleared. Any standard fixed-UID NFC tag can be used with this system.
3. Clear tag: Clears all registered tags. Only master tags can open the locker until a new student tag is registered.

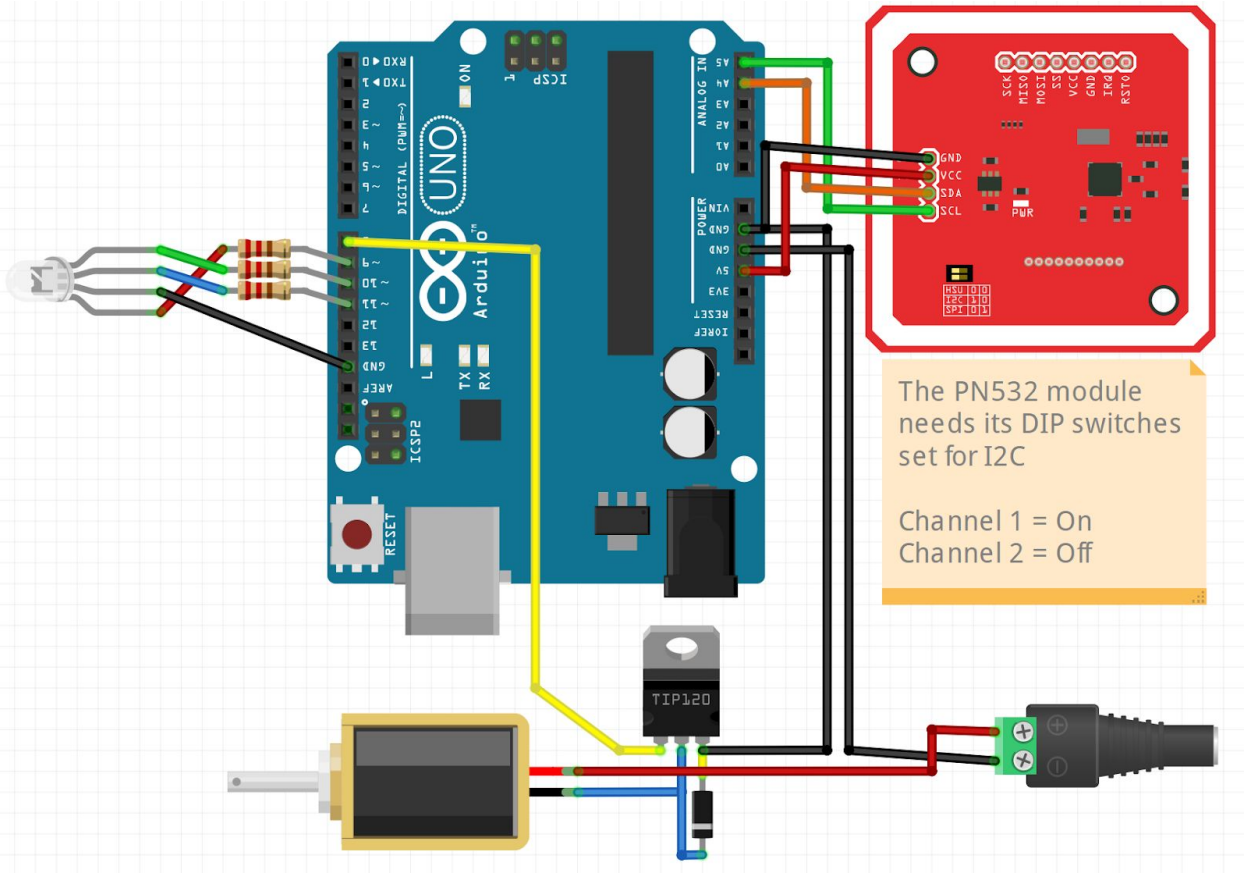
The use of administrative tags in this way eliminates the need for additional input channels (such as buttons), reducing the hardware complexity and cost.

We've also attempted to minimize the number of output channels as well, using a single RGB LED to convey the system state as well as a 12V latch/lock solenoid used as the locker's digital locking mechanism.

RGB LED Indicator Key

Red (on startup)	System initializing.
Off	Passively scanning for tags.
Orange	Processing a scanned tag.
Green	Registration mode. Awaiting tag to register.
Blue	Processing tag for registration.
4 Green Blinks	Successful registration.
3 Blue Blinks	Unsuccessful registration: attempt to register a recognized tag (admin or previously registered).
4 Blue Blinks	Unsuccessful registration: attempt to use or register a tag with a UID longer than the supported UID length.
Pink	Clearing all registered tags.
2 White Blinks	Unrecognized tag (not registered or admin).

Hardware



The Fritzing file for this diagram can be found in our GitHub Repository

Please note: the above wiring diagram is accurate to our test platform with the exception of the solenoid components. Due to parts ordering and delivery issues our team did not yet receive the solenoid. The solenoid logic has been tested to work (with an LED stand in) but the solenoid wiring above is based on the wiring diagram from the former CSE 453 Alden Middle School Locker project design document and could not be independently verified by our team for accuracy and functionality.

Parts List

- Arduino Uno R3
- Common Anode RGB LED
- 3 x 220 Ohm Resistors
- Elechouse PN532 NFC RFID Module V3
- 12V Latch/Lock Solenoid
- Diode Rectifier - 1A, 400V (1N4004)
- TIP120 NPN Darlington Bipolar Power Transistor
- 12V AC/DC PSU

Software

The provided NFC Locker Module software is written in C++ for deployment on an Arduino Uno (or equivalent). In the source code (available on our GitHub repository) you'll find descriptive comments to help with understanding the code's functionality as well as its deployment.

Please note that for deployment the byte arrays masterIDs, registrarIDs, and clearIDs must be populated with the hex values of their associated tags. The NDEF Library we utilized for this project provides example code (specifically ReadTag) which can be run to print the UIDs of tags to the serial console. This is the method we utilized to find the hex values needed to register our administrative tags. Due to the way the admin IDs are stored ALL admin tags must conform to the adminUIDLength.

Functions

- **setup**: Standard Arduino boot function. Here the PN532 NFC module, RGB LED, and serial port (for use in debugging) are initialized.
- **loop**: Standard Arduino 'main' function, ran after setup. This is where the main logic of the system is laid out. At the start of each loop the LED is turned off to indicate that the system is entering its passive scanning state and there's a delay to allow enough time to pass for any previously scan tags to be removed from the scanner to prevent double scanning. When a tag is detected it is then processed. First its UID length is checked to see if it's compatible with the system (a value which can be adjusted before deployment). If it is then a check is performed to see if the tag is a registered tag, if it is the solenoid is engaged and the locker can be opened, otherwise the processing continues to see if the tag is administrative. If the tag is a master tag the solenoid engages and the locker is opened. If the tag is a clear tag all registered users are cleared from the system (the EEPROM memory is wiped). If the tag is a registrar tag then registration mode is entered. When a tag is scanned in registration mode a very similar round of checks is made on that tag to prevent duplicate registration or registration of an administrative tag. Only unrecognized tags with legal UID lengths can be registered. Going back a step, if the tag isn't a registrar tag or any other admin tag and it isn't a registered user, then it's an unrecognized tag and it gets ignored.
- **idListCheck**: Perform a byte-wise comparison to see if the provided UID (byte array) is a member of the provided list of IDs. This is used to check a UID against the hardcoded lists of administrative IDs.
- **idRegCheck**: Perform a byte-wise comparison to see if the provided UID (byte array) is stored in the EEPROM as a registered ID.
- **registerTag**: Write the provided UID to the EEPROM and increase the count of registered tags
- **setLed**: Set the RGB LED to the provided Red, Green, and Blue values (integers between 0 and 255)

- **ledBlink**: Blink the RGB LED at the specified speed the specified number of times with the provided Red, Green, and Blue values

NFC Tag Batch Writer

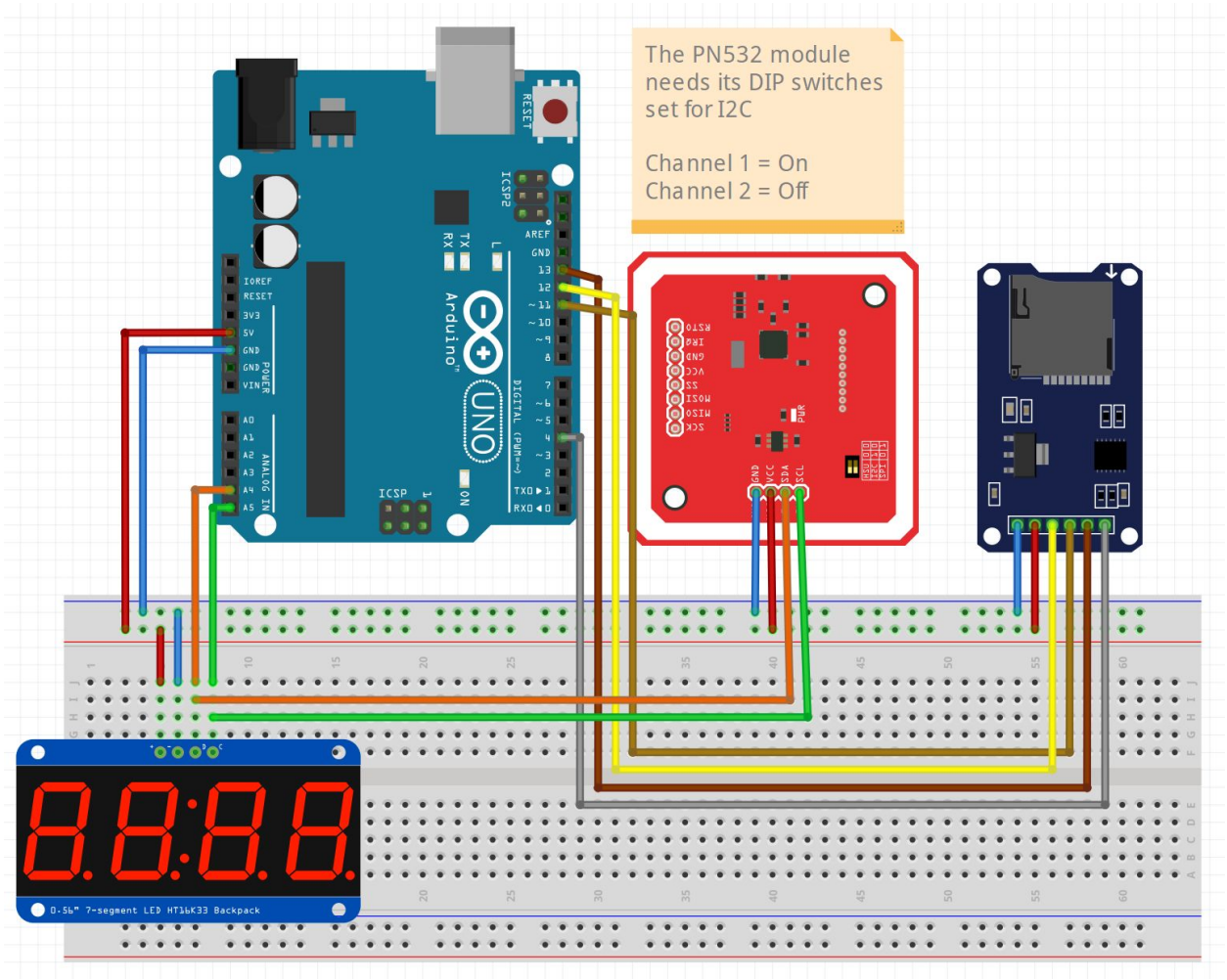
The NFC Tag Batch Writer is an auxiliary utility we've developed to allow for a series of NFC payload writes based on the contents of a file (keys.csv) stored on a MicroSD card. This module queues up the first cell in each row (displaying the associated row number on the seven segment display) and waits for a successful write of that cell's contents to an NFC tag before proceeding to queue up the next value. This utility is useful for and developed during a time of exploration of an NFC payload based authentication system (as opposed to the current UID based authentication system).

This system's seven segment display is also utilized to display status codes as described below.

Status Codes

E101	SD Error: Failed to initialize SD card.
E102	File Error: keys.csv not found.
E103	File Error: keys.csv couldn't be opened.
E104	File Error: keys.csv is empty.
Done	All tags written successfully.

Hardware



The Fritzing file for this diagram can be found in our GitHub Repository

Parts List

- Arduino Uno R3
- Elechouse PN532 NFC RFID Module V3
- Adafruit 0.56” 7-segment LED HT16K33 Backpack Display
- MicroSD 6-pin SPI Driver Module

Software

The provided NFC Tag Batch Writer software is written in C++ for deployment on an Arduino Uno (or equivalent). In the source code (available on our GitHub repository) you’ll find descriptive comments to help with understanding the code’s functionality as well as its deployment.

Functions

- **setup**: Standard Arduino boot function. Here the PN532 NFC module, HT16K33 7-segment display, MicroSD module, and serial port (for use in debugging) are initialized and the file keys.csv is opened and the first key (the value stored in row 1, column 1) is queued. If the file can't be open, an error code will be displayed and the program will halt.
- **loop**: Standard Arduino 'main' function, ran after setup. This is where the main logic of the system is laid out. At the start of each loop there is a delay to prevent unintentional rewrites of the previously written tag. If the previous loop ended with a successful write, then an attempt is made to queue up the next value. If the end of the file has been reached in this attempt, the 'done' status is displayed and the program halts. If a key is queued for writing the loop will continue scanning for tags until a successful write is made.

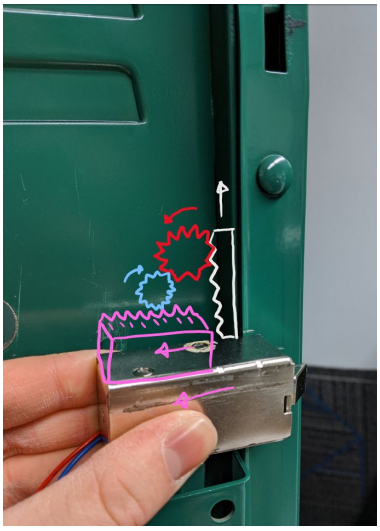
Atmega328-PU Implementations

In an effort to increase scalability of Arduino based components by reducing the size, cost, and programming time required of these components, we explored the option of using standalone Atmega328-PU chips as an alternative to using full-featured off the shelf Arduinos. Doing this would mean that each unit would be much leaner, lacking the extra LEDs, USB ports, and other ancillary components that are typically found in Arduinos. In terms of cost, the Atmega328-PU (which is the same chip that acts as the brains of the Arduino Uno) is available at almost a tenth of the cost of a genuine Arduino Uno, at prices currently ranging from $1.58 to $1.90 per unit (depending on quantity). Finally, the use of an efficient programmer, such as Adafruit's Standalone AVR Chip Programmer, would greatly accelerate the process of mass programming a fleet of devices. Despite the clear potential for this approach we were unable to make any significant steps towards hands-on development and testing of it due to prolonged delays for the delivery of crucial components such as the programmer, chips, sockets, and 16MHz crystal oscillators.

Mechanical Design

With scalability in mind our intention was to create a mechanical design that maximized use of the existing locker components while minimizing the amount of modification needed to install our NFC Locker Module. We conceptualized a solution that would only require the removal of the locker's existing latch hooks. With these removed and the solenoid would act as the only physical element holding the door closed. If we then used a simple gear system to convert the locker's latch rail's upward motion to a sideways motion for the solenoid (as shown below) we could leverage the existing combination/master lock system as our mechanical override, eliminating the need for a new keyway as was required on previous automated locker systems. Unfortunately due to a combination of part shipment delays and the COVID-19 closure

of the Alden High School campus we did not have the parts or access required to realistically flesh out this system and tailor it to our parts and the specifics of the locker latch mechanism.



Power Delivery

One of the major limitations posed by this project is power delivery. One of our requirements is that we needed to develop a power delivery system that doesn't rely directly on wall power. The existing Alden's existing automated lockers utilized expensive Dewalt electric tool batteries, a solution that wouldn't be feasible at scale given the cost of the individual batteries alone, not to mention the cost of chargers and the logistics of charging. The solution we most heavily considered was to utilize solar power, either by utilizing garden-light-sized panels per locker larger panels on top of each bank. The concept is that these solar panels would use the ambient light (fluorescent and natural) to trickle charge common rechargeable cells (perhaps 18650 or AA) and these cells would then power the NFC Locker Module(s). Unfortunately our plans to field test this concept were cut short due the closer of the Alden High School campus in the wake of COVID-19.

Resources

Our GitHub repository contains all of the source code for this project as well as the detailed Fritzing wiring diagrams (shown above) labeled with accurate part numbers and values.