Taylor Perry

Charleston Southern University

Senior Project Documentation

"Implementation of Student RFID Cards at CSU"

Table of Contents

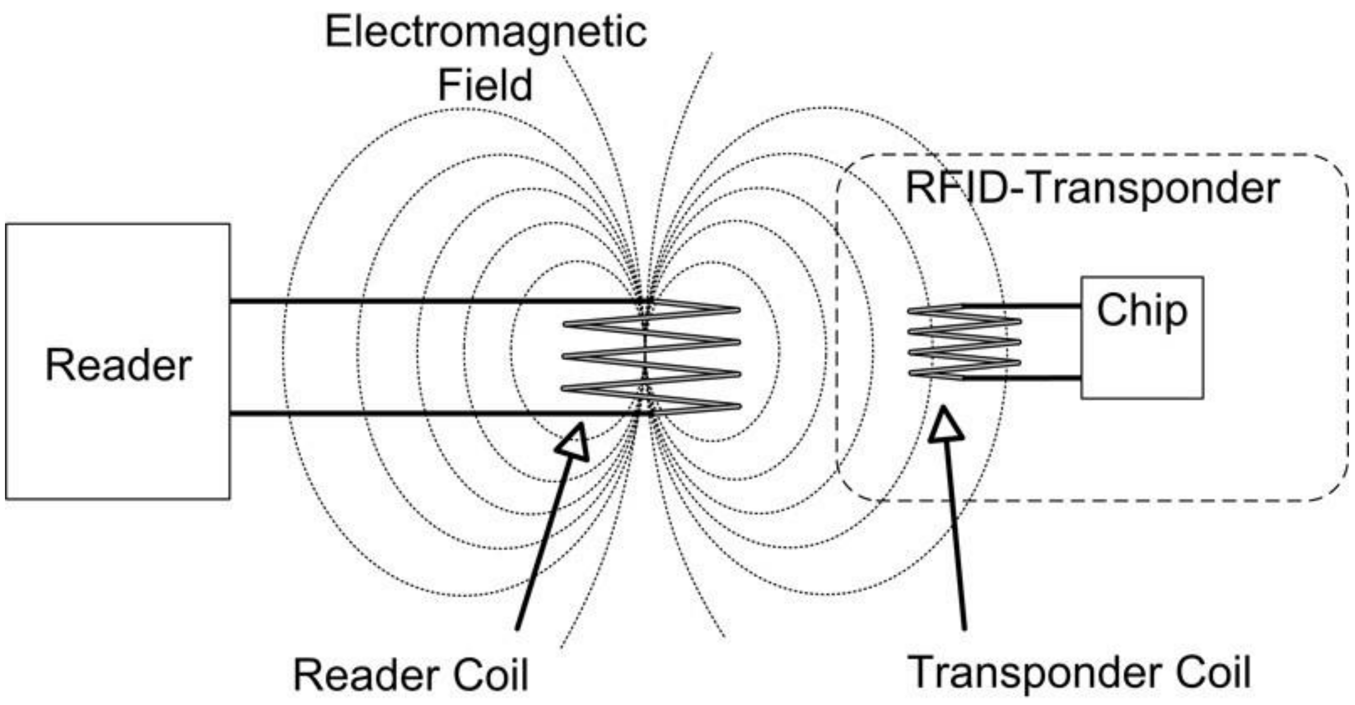
Statement of Purpose

The Idea of this project is to build and introduce RFID readers to be able to read student ID's at Charleston Southern to make certain processes easier as a student. The primary problem I wish to simplify with this project is to make the hassle of leaving Chapel. The current process is everyone leaves at once and at each door there is a scanner that reads the bar code on a students ID. It works well but unfortunately with hundreds of students leaving at once and different lighting conditions this could make scanning the bar code difficult and with hundreds of cards to scan the wasted seconds add up. With Student RFID cards all students would have to do is tap their cards on a RFID reader and they would be on their way. This gives the student more time to get lunch (which everyone is already going that way) or more time before their next class to get ready.

Research and Background

Brief History of RFID

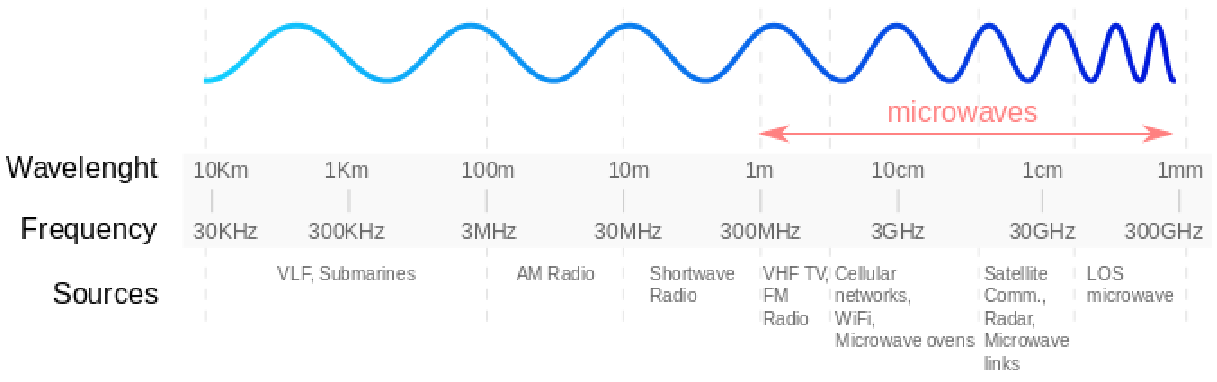
Radio Frequency Identification, also known as RFID, is a helpful technology tool used to keep track of Anything that you can think of. For RFID to work it requires a reader and a tag, where the reader sends out a radio frequency wave to activate and pick up any tags near its frequency range and gathers information on the tag. With different readers and tags it can be a puzzle sometimes to make sure that you get the right equipment for the job you want to do without going over budget getting something that you do not need.



RFID was invented by British military during World War II to help detect if incoming planes were returning British planes or an enemy attack. Before RFID, Radar was already used to detect any object using radio waves, but the British wanted to have no uncertainty that an incoming plane was their own or not. This is when inventor, Watson-Watt decided to crate the IFF (Identify Friend or Foe). Unlike todays RFID technology this was known as Active-Radio Frequency Identification where both the reader and tag had to be powered and the British plane either had a passive tag that would send a signal once the reader activated it, or there would be a active signal that would continuously send out a signal . (VIOLINO, 2005)

After the war, the technology began to find quick usage of keeping track of items remotely which was popular with stores and retail to help prevent theft where items would contain a 1 bit tag, and once the item was purchased it would change the bit and prevent it from activating the alarms. One of the first American patents was made by Mario W. Cardullo where the tag could have writable memory and was used as a keyless/wireless door lock. The newer improved RFID also helped with agricultural purposes to help keep track of livestock and also help farms keep track of medications given to animals and prevent small mistakes. (VIOLINO, 2005)

As mentioned before there are two components with RFID, the reader and the tag with each part having different technologies inside to help customize what you need the RFID to do. The RFID reader is usually the more expensive because it has to generate the signal for the reader and then convert that signal into a radio frequency wave. Depending on the distance that is needed for the reader to pick up a tag you will need to spend more money and a stronger signal strength. (AtlasRFID, n.d.) So how the readers scale is based on the radio frequency that it can register the RFID tags just like how your radio picks up a specific radio wave to listen to your favorite radio station. (AtlasRFID, n.d.)



This Photo by Unknown Author is licensed under CC BY-SA

To know what reader you need to get you first need to know what tag you need first. There are three different kinds of common tags, Low-frequency, High-frequency, and Ultra-high-frequency. The different frequency tag dictates the range and memory size that the tag can hold, where low-frequency are short read range and hold a few Kilobytes of information and often used in situations where the long wavelengths can pass through dense objects. High-frequency operates in the 3-30 MegaHertz and have longer read ranges and better memory. This is the most popular tag option given the cost of some tags often seen in clothing goods and can be used as NFC applications. Since NFC, Near Field Communication, is classified as a high-frequency category but is very different for normal use of tags that we use for the other frequencies. NFC is more common, and only used in one-to-one transactions which is very popular with cardless payments that we use on our phones and other devices. The last category of popular RFID tags would be Ultra-high-frequency which are great for logistics where you can scan a shipping container or pallet full of items and the RFID will tell you all the items that are included in the container. A great example of this is with most shipping trucks where you can track your package and rather than scanning thousands of boxes you can just scan the truck container and it updates thousands of users that their packages location.

My interest with RFID has come from my actual use of the technology. While I worked at Target I helped with fulfilling online orders and most clothing goods had RFID tags in them and using the “RFID gun” we were able to enter in the DPCI (department, class, item) number and once you get close to the item you’re looking for it will give you a meter showing if you are getting closer to the item or not. (Group, 2018)

So using this device made me curious and interested in the possibilities of RFID as a logistic tool, not knowing how widely it was already used. Another occasion of the personal use of RFID is with the MagicBands from Disneyworld. I would try to figure out on my own on how and why Disney would use these bands to store all your information on these little devices. If you haven’t been to Disneyworld (or don’t care for mega-corporations) they provide a MagicBand that is used for making payments, keyless entry into your hotel rooms and into the park, collecting pictures from rides or around the park, and more. After first setting up my RFID reader with Arduino I quickly tested my Magicband and it read as a Ultra-high-frequency tag. Even the Annual passholder card is a RFID tag and gives the link ” https://disneyworld.disney.go.com/faq/my-disney-experience/frequency-technology/ ” if you have any questions about how RFID is used at Disney. With what I now know I plan on making many more RFID projects for home life that include adding IOT, Internet Of Things, to make those projects accessible online and can open a world of possibilities.

Bibliography

AtlasRFID. (n.d.). *What is RFID? | The beginner's Guide to RFID Systems*. Retrieved from AtlasRFIDstore: https://www.atlasrfidstore.com/rfid-beginners-guide/#rfidreaders

Group, R. L. (2018, November 12). *Comparing different types of RFID tags*. Retrieved from Resource Label Group: https://www.resourcelabel.com/comparing-different-types-of-rfid-tags/

VIOLINO, B. (2005, Jan 16). *The History of RFID Technology*. Retrieved from RFID Journal: https://www.rfidjournal.com/the-history-of-rfid-technology

Project Languages, Software, Hardware

Adruino Language, software, and hardware.

- Software
    - https://www.Arduino .cc/en/Guide
        - Download the Arduino Desktop IDE
        - Once you have that you need the following libraries
            - Go to Tools > Manage Libraries >
                - LiquidCrystal(by Arduino)
                - SD (by Arduino)
                - MRFC522 (by GithubCommunity)
                - uRTCLib (by Naguissa)
        - One you have those downloaded you can check to see if your Arduino is being read by going to Tools > Processor, and making sure that you have the correct unit and make sure that the COM is present as well, it should say COM2 or COM3
- Language: C++
    - Arduino IDE is C++ Native.
- Hardware:
    - Arduino UNO/NANO
    - RC522 RFID reader
    - Micro-card Reader
    - RTC 3231 module
    - Active buzzer
    - 220 Ω resister
    - 2k Ω resister
    - Red led
    - 25 male-female wires
    - 6 male-male wires
    - 1 or more RFID tags

Visual Studio Code

- Software
    - Any version go to https://code.visualstudio.com/
        - Download the appropriate version for your OS. I use the Windows version for my main project but for the practice environment I used the linux x64
        - Once downloaded you will need to download certain extensions that can compile c++
            - On the far right click on "extensions(ctrl+shift+X)" > search for c/c++ by microsoft and install that.
            - Then download the files from the folder containing the VSC Code.
        - Make sure that you make a folder on your desktop called "Sr.Project" and inside
            - `"C:\\Users\\'user'\\Desktop\\Sr. Project\\product.csv"`

Project Requirements

1. Requires a computer/laptop with a USB port
2. Arduino hardware listed in Project Languages, Software, Hardware

Project Implementation Description and Explanation

To make sur that I had all the requirements for this project I created a Ubuntu Linux virtual box to test everything. Starting on a clean slate and recording all what I need is what I think is the best way to make sure that I don't miss a step or miss any important library needed for anyone to just pick up the project.

To set up the Ubuntu environment I used this link to help set up :

- https://brb.nci.nih.gov/seqtools/installUbuntu.html

And to get the ubuntu ISO: I used Ubuntu 20.10 environment

- https://ubuntu.com/download/desktop

Once set up I can download the specific software that I used to write the code.

Arduino setup and requirements.

- https://www.Arduino .cc/en/Guide
- Download the Arduino Desktop IDE
- Once you have that you need the following libraries
    - Go to Tools > Manage Libraries >
        - LiquidCrystal(by Arduino)
        - SD (by Arduino)
        - MRFC522 (by GithubCommunity)
        - uRTCLib (by Naguissa)
- One you have those downloaded you can check to see if your Arduino is being read by going to
    - Tools > Processor, and making sure that you have the correct unit and make sure that the COM is present as well, it should say COM2 or COM3

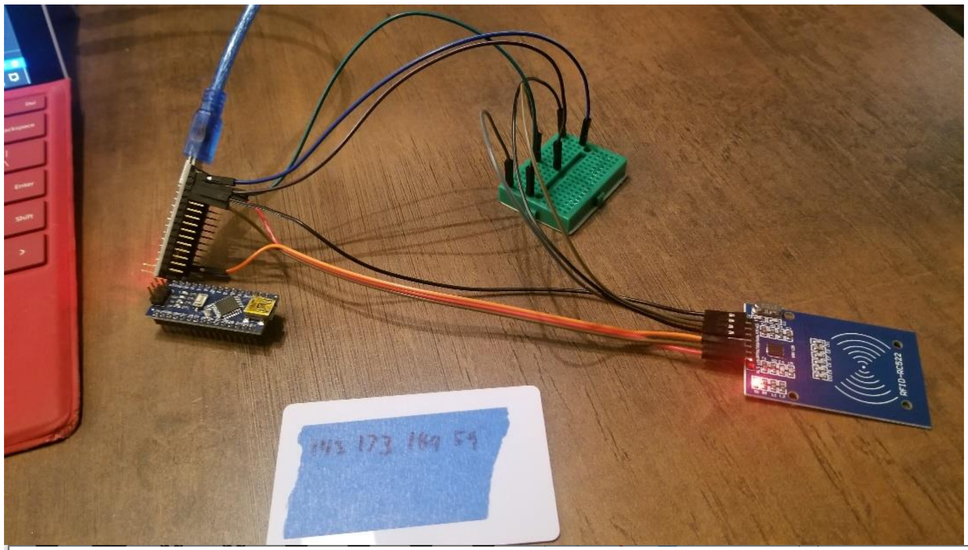
Visual Studio Code

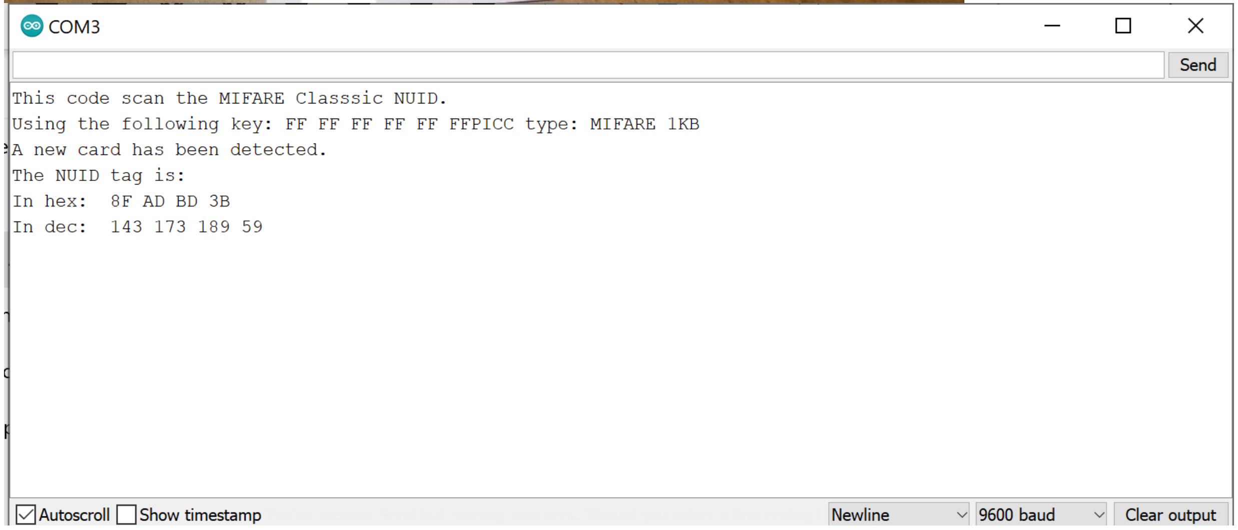
- Any version go to https://code.visualstudio.com/
    - Download the appropriate version for your OS. I use the Windows version for my main project but for the practice environment I used the linux x64
- Once downloaded you will need to download certain extensions that can compile c++
    - On the far right click on "extensions(ctrl+shift+X)" > search for c/c++ by microsoft and install that.
    - Then download the files from the folder containing the VSC Code.
- Make sure that you make a folder on your desktop called "Sr.Project" and inside
    - `"C:\\Users\\'user'\\Desktop\\Sr. Project\\product.csv"`

1. Step one: Test each module to make sure that it is working.
    a. RFID module: Once you download the MFRC522 library there will be an example folder under "File > Examples>MFRC522 > READNUID" and that can test
        i. Hook up the pins from the RFID to the correct digital pin on the Arduino (nano/uno)

RFID PIN	Arduino Pin
SDA	Pin 10
SCK	Pin 13
MOSI	Pin 11
MISO	Pin 12
IRQ	Ignore
GND	GND
RST	Pin 9
3.3v	3.3v

i. The SCK, MOSI, and MISO are common pins in input modules(RFID and SD) reader so those will be shared



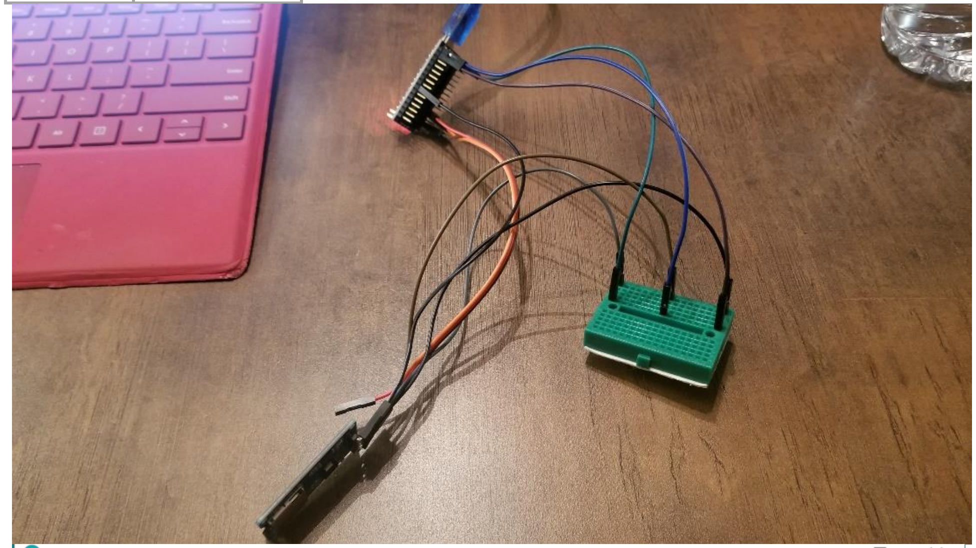


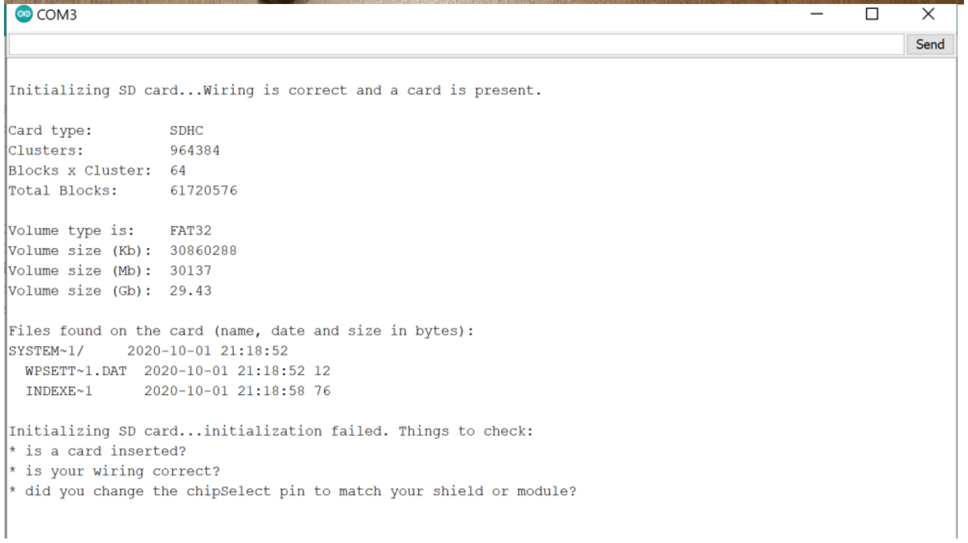
```
This code scan the MIFARE Classsic NUID.
Using the following key: FF FF FF FF FF FFPICC type: MIFARE 1KB
A new card has been detected.
The NUID tag is:
In hex:  8F AD BD 3B
In dec:  143 173 189 59
```

b. Test the Micro SD reader, the example library is already included when you download the Arduino IDE and can be found under "File > Examples > SD > CARDINFO" just to test to see if the SD reader can pick up the SD card.

   i. The hook up should look like this

SD Pin	Arduino Pin
CS	Pin 4
SCK	Pin 13
MOSI	Pin 11
MISO	Pin 12
GDN	GND
VCC	5v

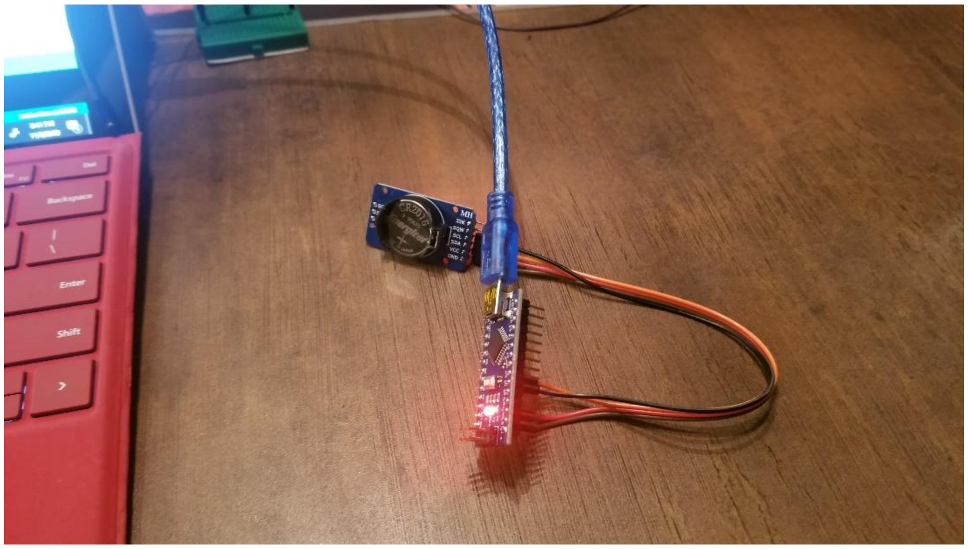




```
Initializing SD card...Wiring is correct and a card is present.

Card type:         SDHC
Clusters:          964384
Blocks x Cluster:  64
Total Blocks:      61720576

Volume type is:    FAT32
Volume size (Kb):  30860288
Volume size (Mb):  30137
Volume size (Gb):  29.43

Files found on the card (name, date and size in bytes):
SYSTEM~1/     2020-10-01 21:18:52
  WPSETT~1.DAT  2020-10-01 21:18:52 12
  INDEXE~1      2020-10-01 21:18:58 76

Initializing SD card...initialization failed. Things to check:
* is a card inserted?
* is your wiring correct?
* did you change the chipSelect pin to match your shield or module?
```

c. Then test the RTC using the example found by downloading the DS3231 library and under "File > Examples > DS3231 > now " you can test the time

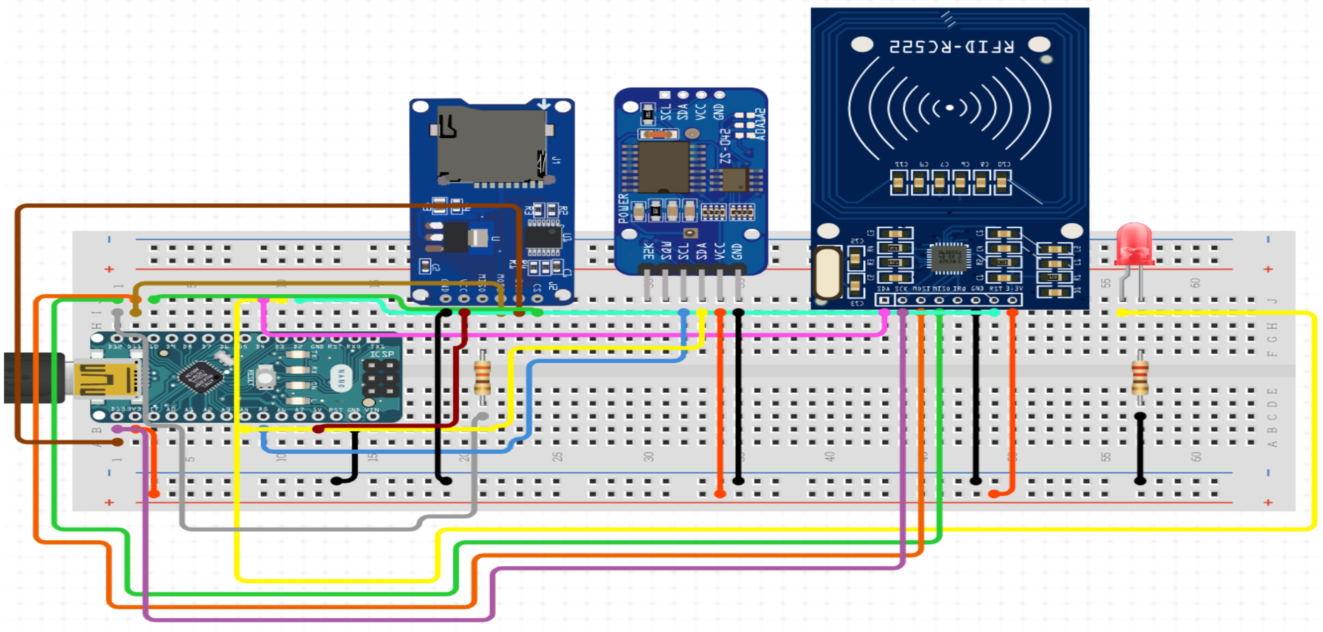
RTC pins	Arduino Pins
SCL	Pin A5
SDA	Pin A4
VCC	5v
GND	GND



Now that you have made sure that all individual modules work you can now combine them all.
I will only use the bread board for the LED light, Arudino pins 13,12, and 11. the rest are not shared and can be hooked up to the Arduino directly.

I used this graph to help map where my wires need to be hooked up. This was my personal preference and what I liked since most times I had exactly what I needed and where to hook it up.

Arduino Pin	RFID	Micro SD	RTC	LED
D13	SCK	SCK		
D12	MISO ->330 resister ->	MISO		
D11	MOSI	MOSI		
D10	SDA			
D9	RST			
D4		SDA		
D3				LED +(long end)
A4			SDA	X
A5			SCL	
3v	3v			
5V		VCC	VCC	
GND	G1	H1	I1	



C++ CODE

```
#include <iostream>
#include <vector>
#include <string>
```

```
#include <fstream>
#include <list>

using namespace std;

int main() {
    string tag;
    string Name;
    string ID;
    string RFID;

    struct dbIDs{
        string ID;
        string RFID;
    };


//list with RFID tags from scanner
    list<string> tagList = list<string>();

    //list with studnet database  info (name, ID , RFID)
    list<dbIDs> idlist = list<dbIDs>();

    ifstream sdfile("C:\\Users\\Pyroc\\Desktop\\RFIDTAG.txt");
    //sdfile.open();
    if(sdfile.is_open()){
        while(getline(sdfile, tag)){

            //putting tag values into a list
            tagList.push_front(tag);
            ////test to see if tags are going into tagList.
            //cout << tagList.front() << endl;
        }
        sdfile.close();


    }else cout << "unable to open file. "<< endl;




    ifstream inFile;
    inFile.open("C:\\Users\\Pyroc\\Desktop\\Sr. Project\\database.csv");
    //store values into a list1
    if(inFile.is_open()){
        while(!inFile.eof()){
            getline (inFile, Name,',' );

            getline (inFile, ID,',' );  // i NEED TO COMPARE THE LINE AND RFID VALUE TO
OUTPUT THE ID VALUE

            getline (inFile, RFID );

            //creating a new object that contains the student ID and RFID values
            dbIDs newID;

            //new values added to the list from database.
            newID.ID = ID;
            newID.RFID = RFID;

            idlist.push_front(newID);

            // cout << idlist.front().ID << endl;
            // cout << idlist.front().RFID << endl;
        }
    }
    inFile.close();

    ofstream product("C:\\Users\\Pyroc\\Desktop\\Sr. Project\\product.csv");

    //compare the tags in the list "taglist" to see if they are in the
    //the student database list "idlist"

    //open a file to write the iter->ID values cause thats ulimate goal
    while(!tagList.empty()){

        std::list<dbIDs>::iterator iter;

        for(iter = idlist.begin(); iter != idlist.end(); iter++){

            if(tagList.front() == iter->RFID){

                cout << iter->ID << endl;
                product << iter->ID << "," << endl;
                break;

            }
            //cout << iter->RFID << endl;
        }
        //cout << endl << tagList.front() << endl;
        tagList.pop_front();

    }
    product.close();
    cout << "this is the end";
}
```

Arduino Code

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(3, 2, A0, A1, A2, A3);


#include <MFRC522.h> // for the RFID

#include <SPI.h> // for the RFID and SD card module

#include <SD.h> // for the SD card

#include <RTClib.h> // for the RTC


// define pins for RFID

#define CS_RFID 10

#define RST_RFID 9

// define select pin for SD card module

#define CS_SD 4
```

```
// Create a file to store the data

File myFile;


// Instance of the class for RFID

MFRC522 rfid(CS_RFID, RST_RFID);


// Variable to hold the tag's UID

String uidString;


// Instance of the class for RTC

RTC_DS1307 rtc;


// Pins for LEDs and buzzer

const int redLED = 6;

const int greenLED = 7;

const int buzzer = 5;


void setup() {


  // Set LEDs and buzzer as outputs

  pinMode(redLED, OUTPUT);

  pinMode(greenLED, OUTPUT);

  pinMode(buzzer, OUTPUT);


  // Init Serial port

  Serial.begin(9600);

  lcd.begin(16,2);
```

```
while(!Serial); //

// Init SPI bus
SPI.begin();
// Init MFRC522
rfid.PCD_Init();

// Setup for the SD card
Serial.print("Initializing SD card...");
lcd.print("Initializing ");
lcd.setCursor(0, 1);
lcd.print("SD card...");
//delay(3000);
lcd.clear();
if(!SD.begin(CS_SD)) {
  Serial.println("initialization failed!");
  lcd.print("Initializing ");
  lcd.setCursor(0, 1);
  lcd.print("failed!");
  return;
}
Serial.println("initialization done.");
lcd.print("Initialization ");
lcd.setCursor(0, 1);
lcd.print("Done...");

// Setup for the RTC
```

```
  if(!rtc.begin()) {
    Serial.println("Couldn't find RTC");
    lcd.clear();
    lcd.print("Couldn't find RTC");
    while(1);
  }
  else {
    // following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
  }
  if(!rtc.isrunning()) {
    Serial.println("RTC is NOT running!");
    lcd.clear();
    lcd.print("RTC Not Running!");
  }
}

void loop() {
  //look for new cards
  if(rfid.PICC_IsNewCardPresent()) {
    readRFID();
    RFIDtagList();


  }
  delay(0);
}
```

```
void readRFID() {
  rfid.PICC_ReadCardSerial();
  lcd.clear();
  Serial.print("Tag UID: ");
  lcd.print("Tag UID: ");
  //Since a RFID tag has 4 seperated values you need to run this to make sure all the information is read
  uidString = String(rfid.uid.uidByte[0]) + " " + String(rfid.uid.uidByte[1]) + " " +
    String(rfid.uid.uidByte[2]) + " " + String(rfid.uid.uidByte[3]);
  Serial.println(uidString);
  lcd.setCursor(0, 1);
  lcd.print(uidString);

  // Sound the buzzer when a card is read
  tone(buzzer, 2000);
  delay(20);
  noTone(buzzer);
   delay(20);
}

void RFIDtagList() {
  // Enables SD card chip select pin
  digitalWrite(CS_SD,LOW);

  // Open file
  myFile=SD.open("RFIDTAG", FILE_WRITE);

  // If the file opened ok, write to it
```

```
 if (myFile) {
   Serial.println("File opened ok");
   lcd.clear();
   lcd.print("File opened ok");
   //delay(2000);
   myFile.print(uidString);
   myFile.print(", ");

  // For future improvements where it would be important to record the time of the scan

//    // Save time on SD card
//records the time of the scan and prints to serial monitor(only when hooked up to the computer)
 DateTime now = rtc.now();

 // Print time on Serial monitor
   Serial.print(now.year(), DEC);
   Serial.print('/');
   Serial.print(now.month(), DEC);
   Serial.print('/');
   Serial.print(now.day(), DEC);
   Serial.print(' ');
   Serial.print(now.hour(), DEC);
   Serial.print(':');
   Serial.println(now.minute(), DEC);
   Serial.println("sucessfully written on SD card");
```

```
    lcd.clear();
    lcd.print(now.year(), DEC);
    lcd.print(':');
    lcd.print(now.month(), DEC);
    lcd.print(':');
    lcd.print(now.day(), DEC);
    lcd.print(' ');
    lcd.setCursor(11, 0);
    lcd.print(now.hour(), DEC);
    lcd.print(':');
    lcd.print(now.minute(), DEC);
    lcd.setCursor(0, 1);
    lcd.print("Written on SD...");
   // delay(2000);

    myFile.close();

  }
  else {

    Serial.println("error opening data.txt");
    lcd.clear();
    lcd.print("error opening data.txt");
  }
  // Disables SD card chip select pin
  digitalWrite(CS_SD,HIGH);
}
```

Test plan and results

Test	Expected outcome	Actual Outcome
If Duplicate RFID cards were scanned what would happen?	The Reader holds the most current cards value to make sure that it isn't accidentally scanned twice, but if the person were to scan go to the back of the line then scan again it would duplicate the RFID value. The .CPP code should only produce one Student ID even if multiple RFID scans were made.	It is possible to have a duplicate values, but only if the student gets back in line to scan their RFID again.
How many values can the .CPP file handle before it slows down or crashes	In excel where the database is I can easily create a list of fake values and go up to around 300 (expected chapel turn out) to see if any problems occur. I think the program will run with no problems.	Using 380 fake RFID values there was no difference in completion time.
Using an Arduino Nano instead of a Arduino UNO	The only difference between the two devices is that the Nano is 1/4 the size and does not have a additional power source. I expect that My project can run on a smaller Arduino helping the space the project takes up	I did not get this to work with the Arduino Nano.
Will the project run if not connected to a computer first	If I were to save the last ran program on the Arduino I should be able to instead plug into the computer plug the Arduino into a power bank and have it run like normal.	The Project will save the most recently ran project. Hitting the reset button can make sure a new start is officially made.
Can the name of the generated csv file reflect the time that the csv file is created?	The .cpp file will have the final product named after the time it was compiled that way multiple scanners do not overwrite data and lose Student ID values. The only problem is that the .cpp file reads in a specific file, so if I were to experiment on how the .cpp file is to open the RFID tag list I could have multiple files on the Arduino . (This is where the RTC module helps)	I could not find a way to successfully name the .csv product file as the time and date but I can see that as a future enhancement.
When the ID is scanned, does the output of the Arduino file reflect the ID number that was scanned?	The RFID tag (student RFID) itself will not hold any information about the student. This is for security purposes if the student lost their id its easier to change one value (RFID number) than to change all of the students information(Name, Student ID, RFID number)	Putting data on the RFID is possible but for this project I did not choose to utilize that for this project. Can be a Future enhancement
When your program reads in the output from the Arduino code, does it correctly extract the student ID that corresponds to the RFID code	The only value that the Arudino code gives is the list of RFID Values. It's the purpose of the C++ code to take the RFID values and compare them to the excel database, then gives the Student Id.	Yes, the only way to make sure of this is to have the student database side by side to compare the values but

		the right student ID values are present with RFID valuesT
When the ID is scanned very quickly, does the ID number get scanned completely or does the scan fail	Because the RFID number is so small it only takes a fraction of a second and the main issue is if you don’t ‘tap’ your card or get close enough. The buzzer should go off if your card was read.	The only problem with this is if the student does not “tap” the card to the reader. Since the RFID reader is at 13 Mhz this gives it a decent read speed, and since the

		values are so small (15 Bytes) *tapping the RFID card to the reader isn't required but the space for the RFID card to be read is only a few centimeters distance.
If the ID scan fails, what does your design say will happen? Include a test case to verify that the outcome is what you expect.	If your card is not read, then you RFID value is not read. Just like with system now if you bar code on your student ID was not scanned properly you will have to email your advisor. You know the card was scan because the Buzzer goes off.	If the correct card is scanned and close enough to the reader to be scanned then the will not be missed.
What happens if RFID tag is scanned that is not in the student database.	If the RFID scanned is not in the database, then in the C++ code it will not fetch a Student ID. It will also not prevent the rest of the RFID values from being converted to Student ID's.	If the RFID value is not in the database then the C++ code will iterate the database and then see that the value is not there and then move on to the next value.

Challenges Overcome

1. Database: Microsoft Azure database that I was working with ran out on the free trial before I could really experiment on it.
    a. Trying to find a free/reliable way to make a database to send and receive data.
        i. Currently trying Visual Studio to make a database.
    b. Since I want to make things run smoothly, I wanted to use all Microsoft and Visual Studio was a free option for this project.

This problem was solved by using a excel to hold all the information. Using excel as a steppingstone I am sure I can experiment more with different options and if it's worth trying to pay for different services.

2. Portability: Trying to find a way to collect data while the rfid is not hooked up to a computer.
    a. Using the Serial port reader "PLX-DAQ" seems to work perfect for showing data being printed to a excel file.
        i. Downside is that it cannot take that much data and has the Arduino hooked up to the computer and excel at the same time. Making it not portable.

This was a very early problem I had. This was during the design phase where I research different ways of recording RFID values. This was a program I saw that would only record 10 values, which would not be practical for this project.

3. Broken parts.
    a. So far I've got a few broken pins stuck in the Arduino that may have caused some more problems but just temp, setbacks.
4. Most efficient way yet practical.
    a. Not many resources to look up problems I'm having.
5. (8/31/20) Using the Ethernet/SD Sheild attachment seems to have made my previous code not to work. I cant seem to read any RFID card now.

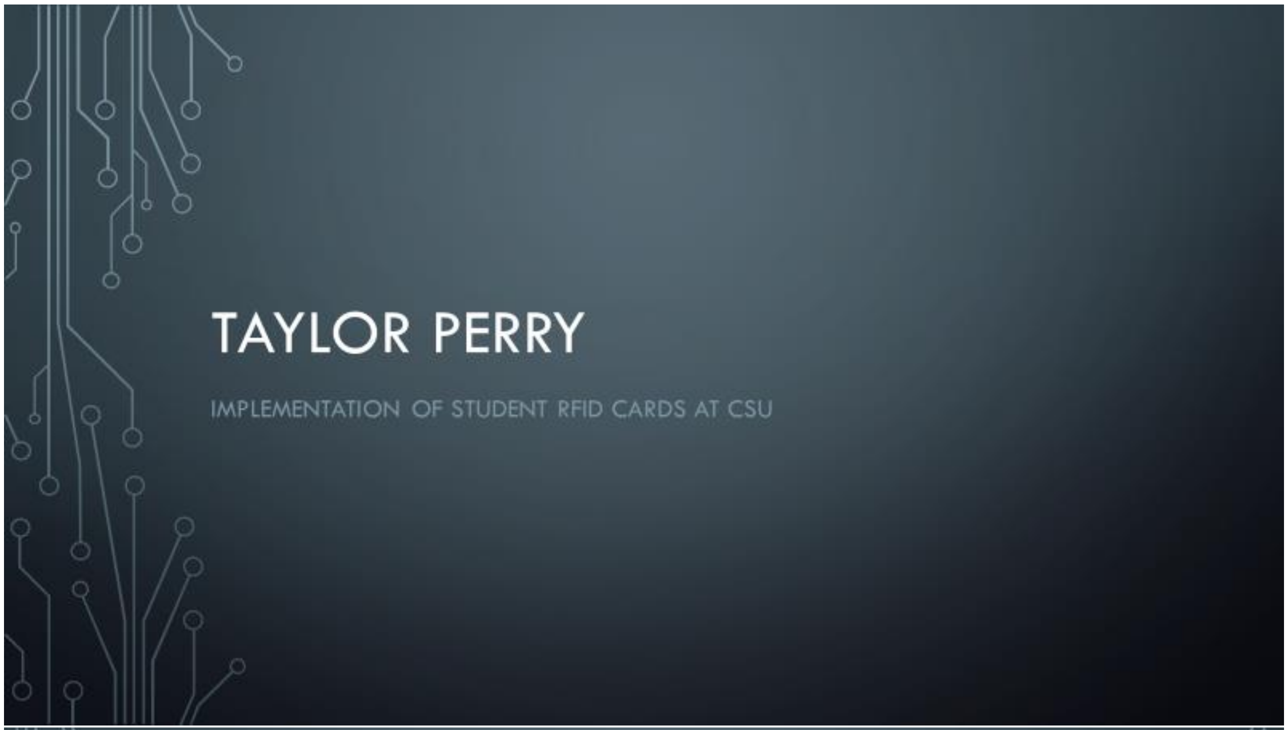
    This was an attachment that I thought I could use instead of the Micro-card reader. It would help with organization by having one less module hooked up to wires. I could not find much information about having two slave devices(RFID reader and Micro-Card reader) until much later in the project. This could be a future enhancement.

6. For the longest time I couldn’t get the RFID and SD reader to work together and I knew it was the power difference but I didn’t know how to solve it correctly. I eventually figured out I needed to add a resister to make sure that the 5v SD reader was not shorting out the 3v RFID reader.

Future Enhancements

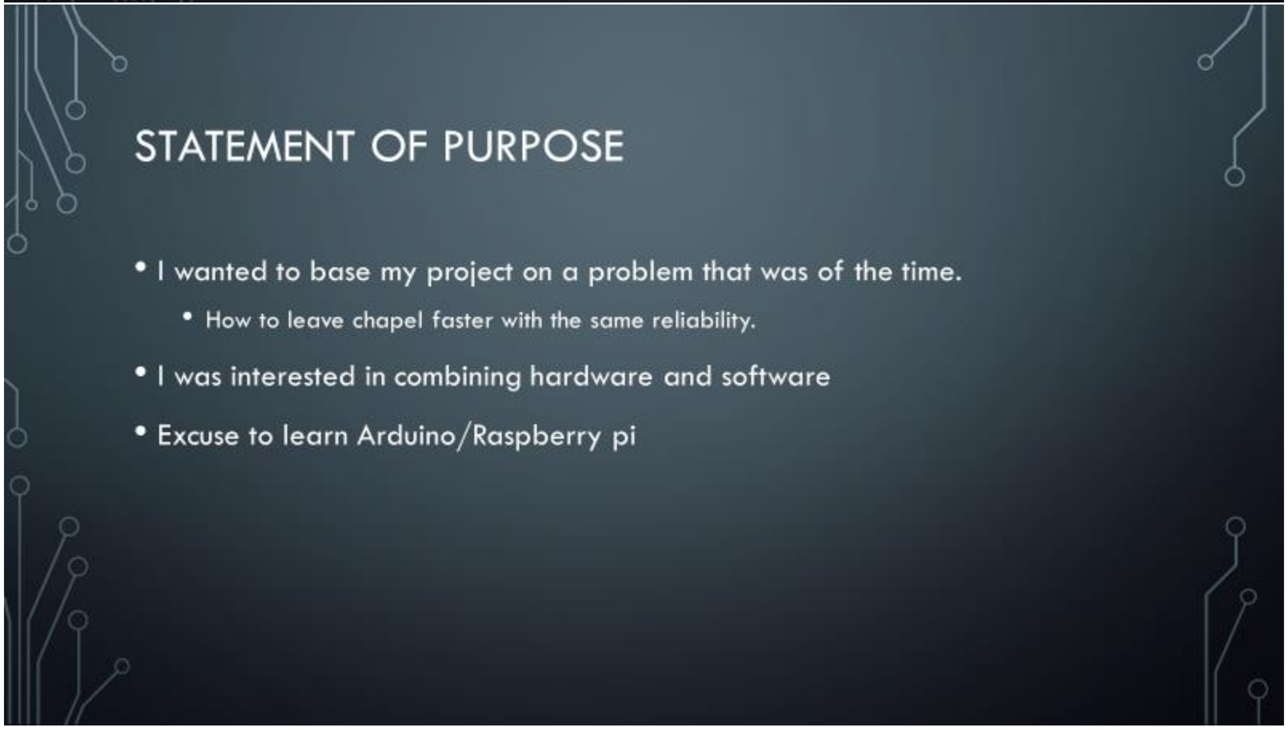
For future enhancements I would like to have this give direct results by replacing the Micro SD reader module with an IOT device that way the results can be uploaded through the internet and just all be read into one computer. I would also like to improve the functionality of the project to have just one RFID Tag to do multiple things, like get you into the nurse lounge if you are a nursing student or get you into the graphics design computer room if that is your major/minor. This could help security around campus allowing students into designated buildings without the use of security and usability of campus resources.

Power Point Slides



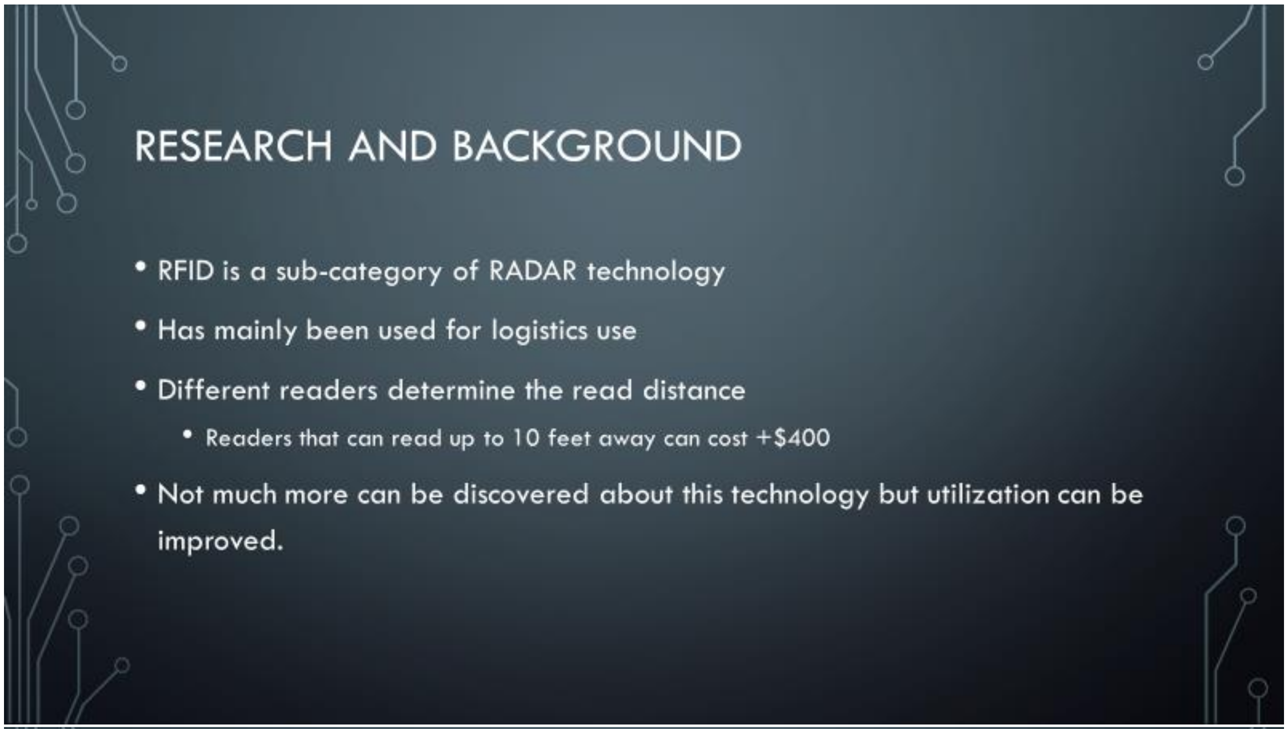
TAYLOR PERRY

IMPLEMENTATION OF STUDENT RFID CARDS AT CSU



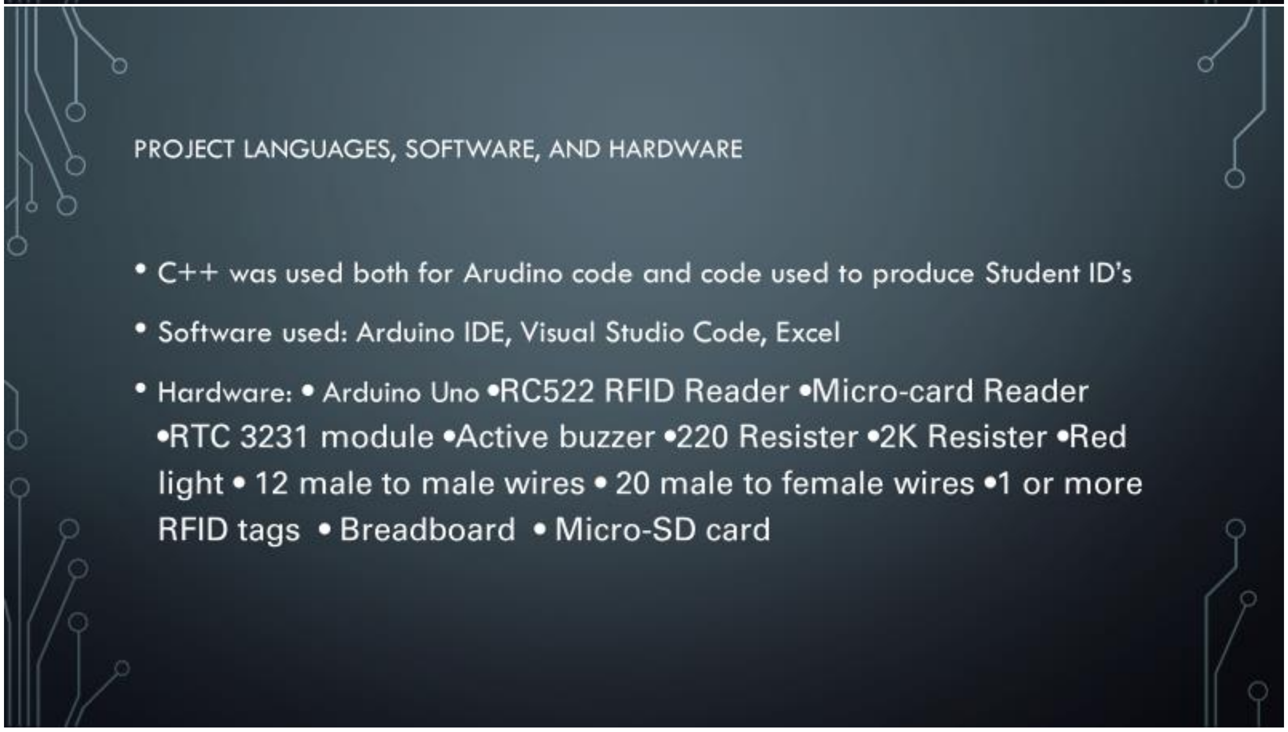
STATEMENT OF PURPOSE

- I wanted to base my project on a problem that was of the time.
  - How to leave chapel faster with the same reliability.
- I was interested in combining hardware and software
- Excuse to learn Arduino/Raspberry pi



RESEARCH AND BACKGROUND

- RFID is a sub-category of RADAR technology
- Has mainly been used for logistics use
- Different readers determine the read distance
    - Readers that can read up to 10 feet away can cost +$400
- Not much more can be discovered about this technology but utilization can be improved.

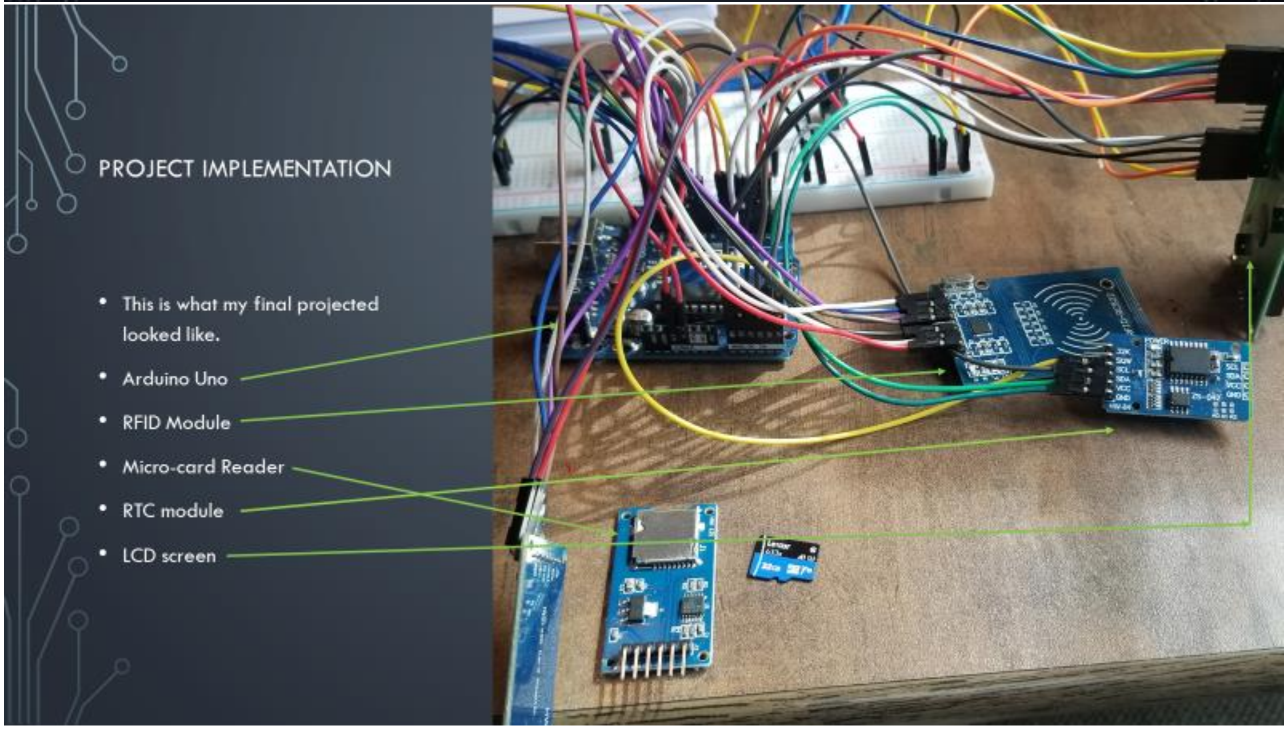


PROJECT LANGUAGES, SOFTWARE, AND HARDWARE

- C++ was used both for Arudino code and code used to produce Student ID's
- Software used: Arduino IDE, Visual Studio Code, Excel
- Hardware: • Arduino Uno •RC522 RFID Reader •Micro-card Reader •RTC 3231 module •Active buzzer •220 Resister •2K Resister •Red light • 12 male to male wires • 20 male to female wires •1 or more RFID tags • Breadboard • Micro-SD card

PROJECT REQUIREMENTS

1. A working computer that has a USB connection.
2. The software and hardware mentioned in Project software

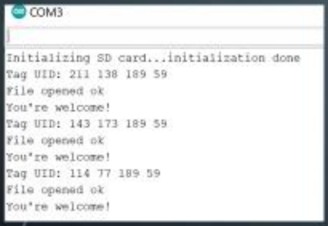


PROJECT IMPLEMENTATION

- This is what my final projected looked like.
- Arduino Uno
- RFID Module
- Micro-card Reader
- RTC module
- LCD screen

IMPLEMENTATION CONT.

Fig.1



```
COM3

Initializing SD card...initialization done
Tag UID: 211 138 189 59
File opened ok
You're welcome!
Tag UID: 143 173 189 59
File opened ok
You're welcome!
Tag UID: 114 77 189 59
File opened ok
You're welcome!
```

Fig.2

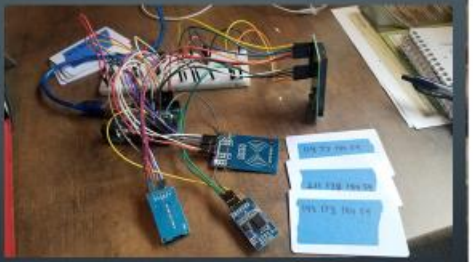
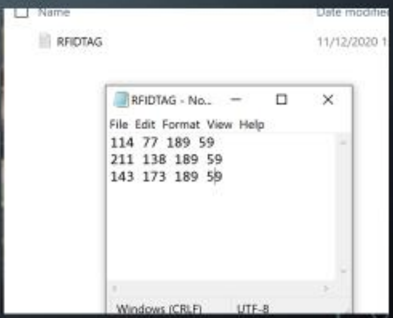


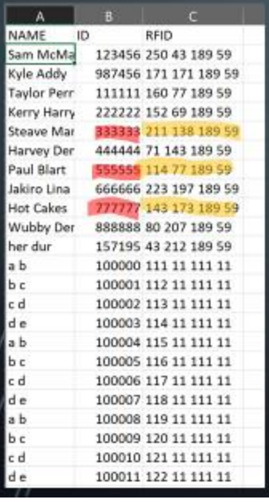
Fig.3



```
Name            Date modified
RFIDTAG         11/12/2020 1

RFIDTAG - No...
File  Edit  Format  View  Help
114 77 189 59
211 138 189 59
143 173 189 59

Windows (CRLF)    UTF-8
```

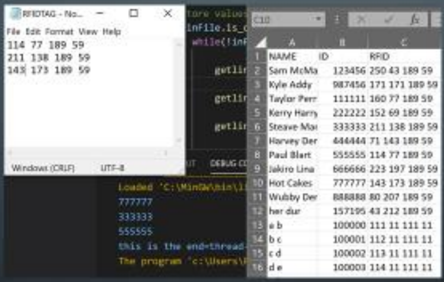
IMPLEMENTATION CONT.

Fig.4



A	B	C
NAME	ID	RFID
Sam McMa	123456	250 43 189 59
Kyle Addy	987456	171 171 189 59
Taylor Perr	111111	160 77 189 59
Kerry Harry	222222	152 69 189 59
Steave Mar	333333	211 138 189 59
Harvey Der	444444	71 143 189 59
Paul Blart	555555	114 77 189 59
Jakiro Lina	666666	223 197 189 59
Hot Cakes	777777	143 173 189 59
Wubby Der	888888	80 207 189 59
her dur	157195	43 212 189 59
a b	100000	111 11 111 11
b c	100001	112 11 111 11
c d	100002	113 11 111 11
d e	100003	114 11 111 11
a b	100004	115 11 111 11
b c	100005	116 11 111 11
c d	100006	117 11 111 11
d e	100007	118 11 111 11
a b	100008	119 11 111 11
b c	100009	120 11 111 11
c d	100010	121 11 111 11
d e	100011	122 11 111 11

Fig.5

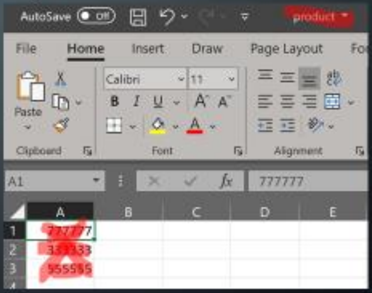


```
RFIDTAG - No...
File  Edit  Format  View  Help
114 77 189 59
211 138 189 59
143 173 189 59

Windows (CRLF)    UTF-8
```

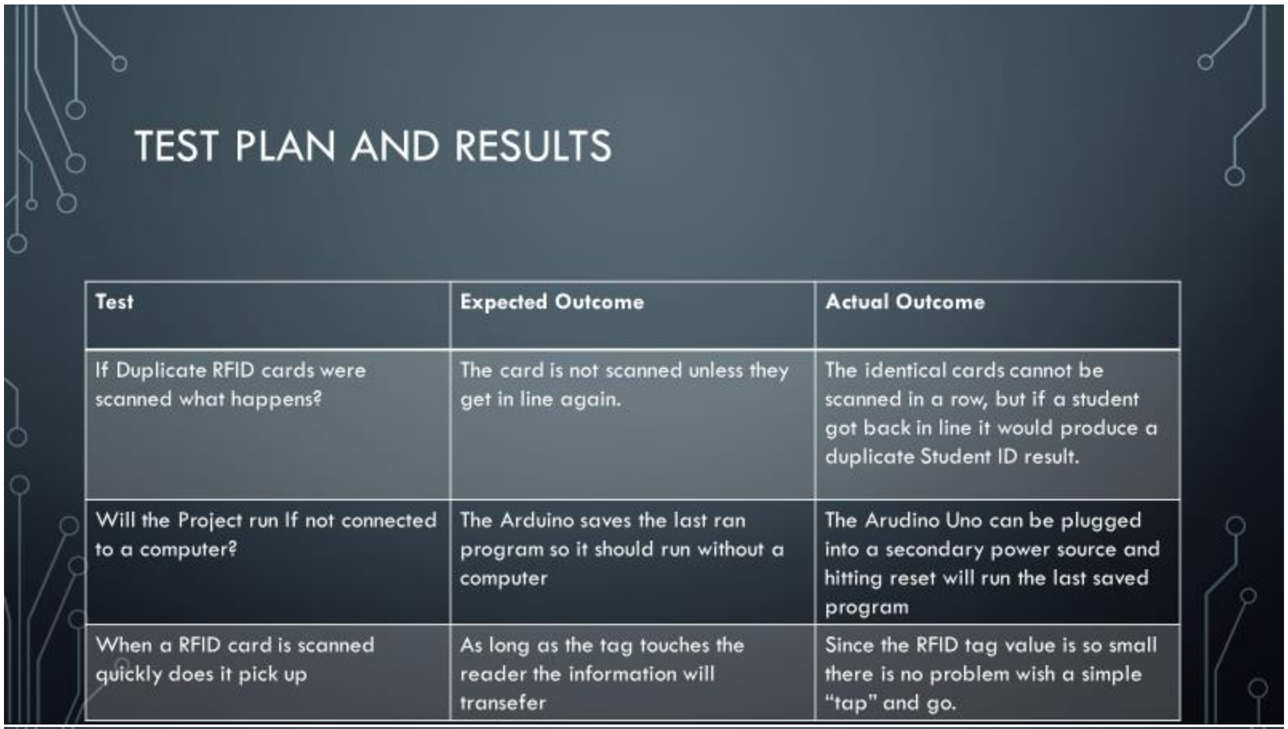
```
tore value
inFile.is_
 while(!in
     getli
     getli
     getli

Loaded 'C:\MinGW\bin\l
777777
333333
555555
this is the end-thread
The program 'c:\Users\
```

	A	B	C
1	NAME	ID	RFID
2	Sam McMa	123456	250 43 189 59
3	Kyle Addy	987456	171 171 189 59
4	Taylor Perr	111111	160 77 189 59
5	Kerry Harry	222222	152 69 189 59
6	Steave Mar	333333	211 138 189 59
7	Harvey Der	444444	71 143 189 59
8	Paul Blart	555555	114 77 189 59
9	Jakiro Lina	666666	223 197 189 59
10	Hot Cakes	777777	143 173 189 59
11	Wubby Der	888888	80 207 189 59
12	her dur	157195	43 212 189 59
13	a b	100000	111 11 111 11
14	b c	100001	112 11 111 11
15	c d	100002	113 11 111 11
16	d e	100003	114 11 111 11

Fig.6

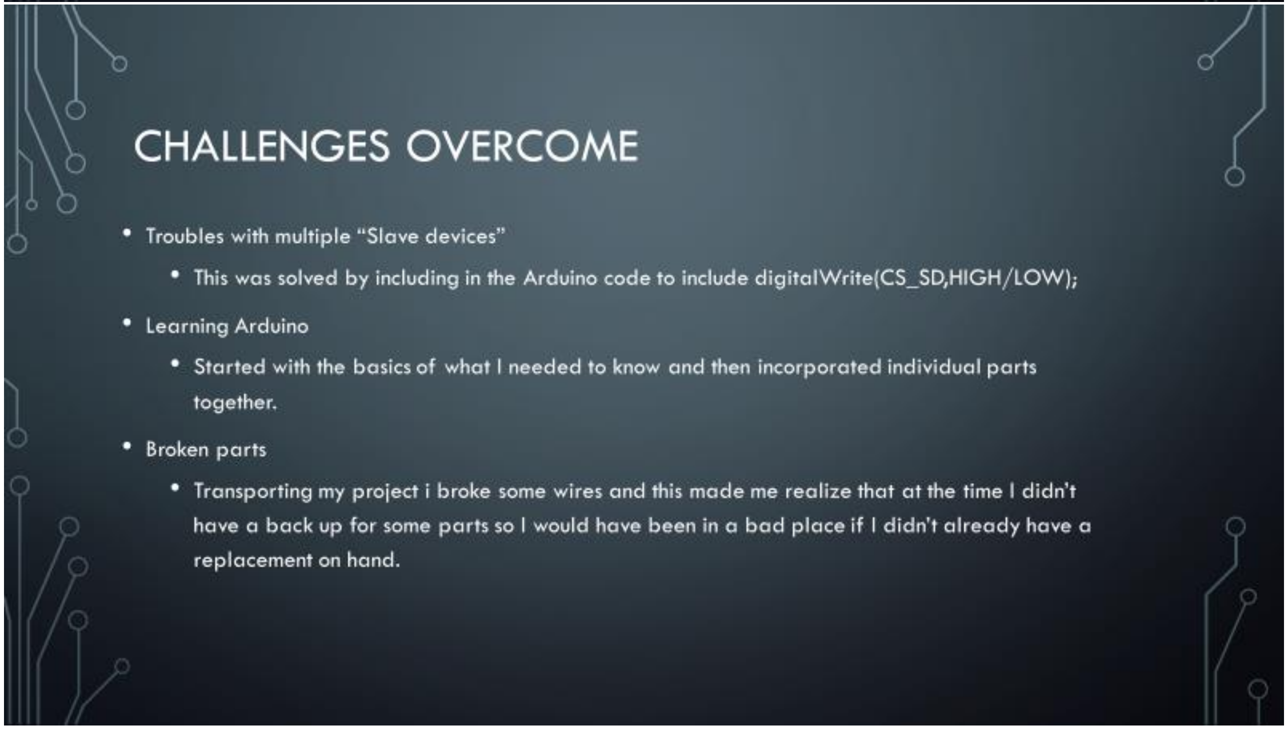


	A	B	C	D	E
1	777777				
2	333333				
3	555555				



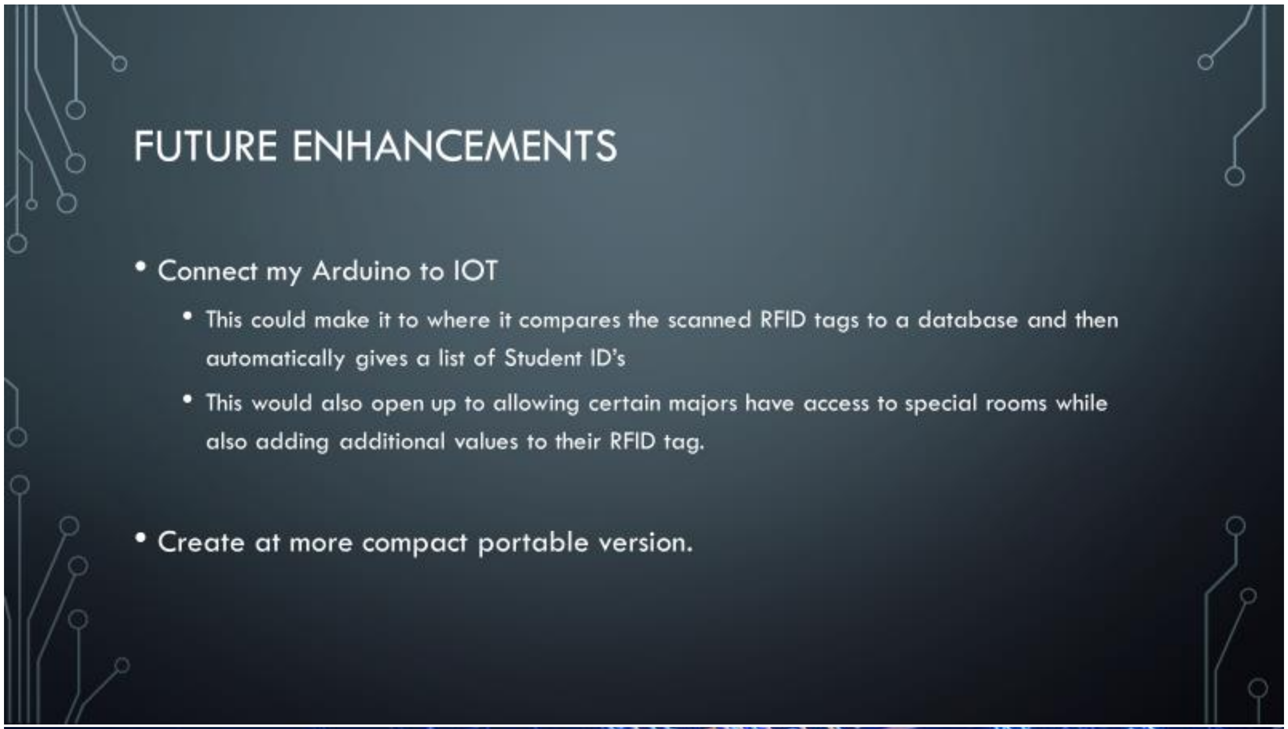
TEST PLAN AND RESULTS

Test	Expected Outcome	Actual Outcome
If Duplicate RFID cards were scanned what happens?	The card is not scanned unless they get in line again.	The identical cards cannot be scanned in a row, but if a student got back in line it would produce a duplicate Student ID result.
Will the Project run If not connected to a computer?	The Arduino saves the last ran program so it should run without a computer	The Arudino Uno can be plugged into a secondary power source and hitting reset will run the last saved program
When a RFID card is scanned quickly does it pick up	As long as the tag touches the reader the information will transefer	Since the RFID tag value is so small there is no problem wish a simple "tap" and go.



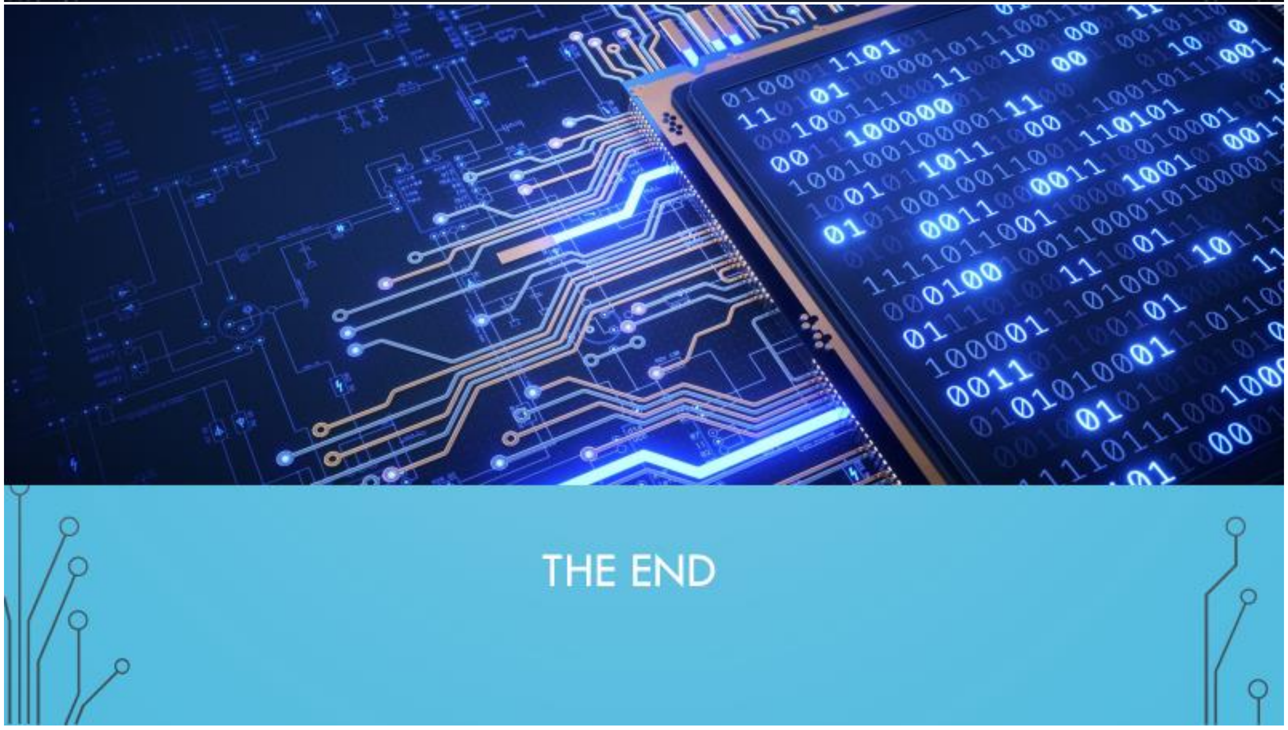
CHALLENGES OVERCOME

- Troubles with multiple "Slave devices"
  - This was solved by including in the Arduino code to include digitalWrite(CS_SD,HIGH/LOW);
- Learning Arduino
  - Started with the basics of what I needed to know and then incorporated individual parts together.
- Broken parts
  - Transporting my project i broke some wires and this made me realize that at the time I didn't have a back up for some parts so I would have been in a bad place if I didn't already have a replacement on hand.



FUTURE ENHANCEMENTS

- Connect my Arduino to IOT
  - This could make it to where it compares the scanned RFID tags to a database and then automatically gives a list of Student ID's
  - This would also open up to allowing certain majors have access to special rooms while also adding additional values to their RFID tag.
- Create at more compact portable version.



THE END