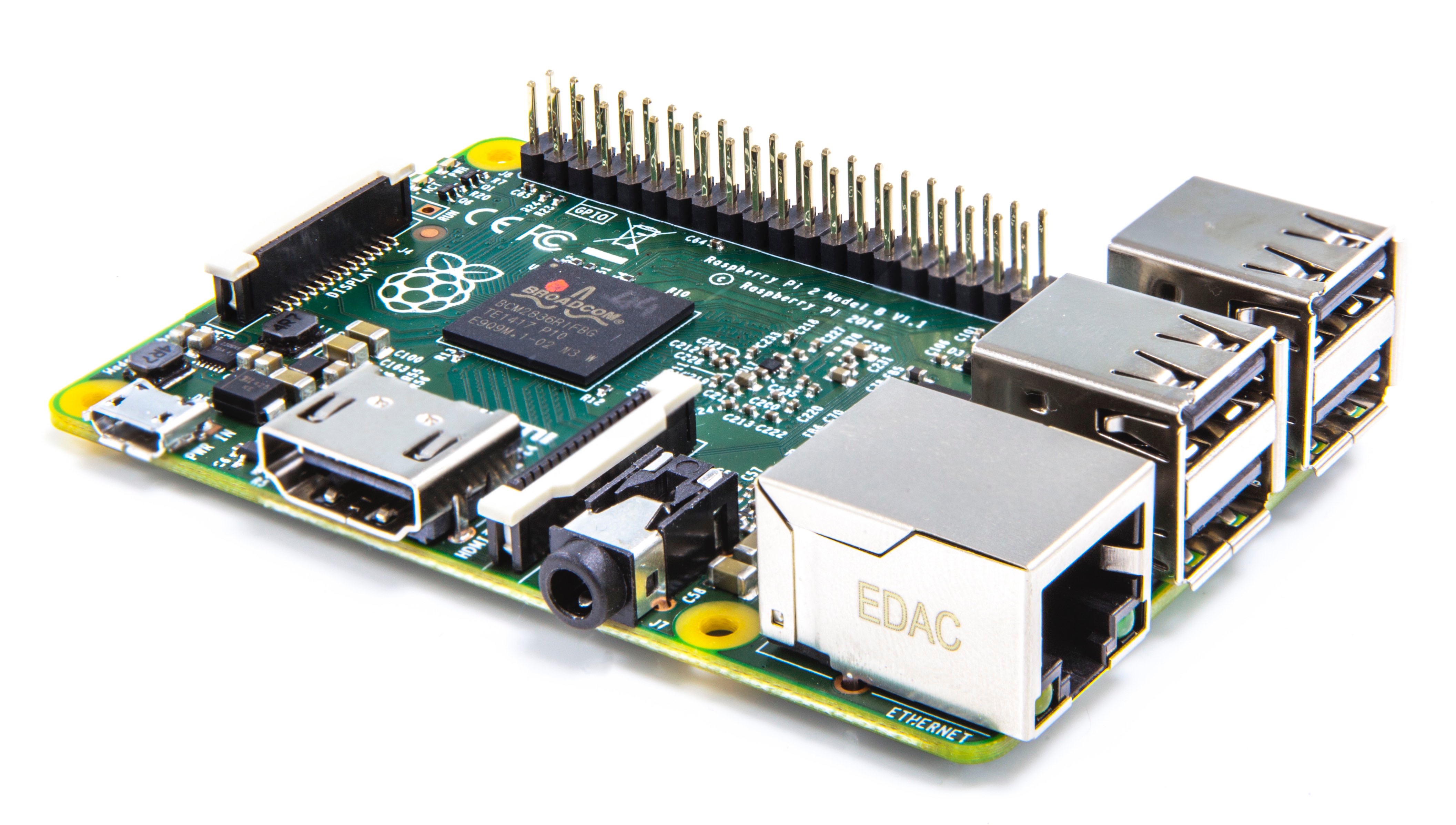
NFC Badge System for Flagstaff Medical Center

Undergrad Research

Chris Heiser

Austin Sanders

Spring 2015

Equipment:

RC522 Tag Reader

Raspberry Pi B model

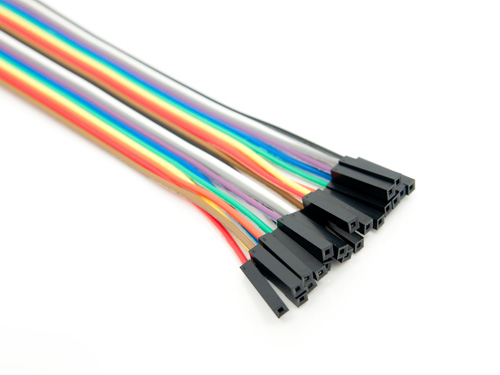


RC522 RFID Reader



Jumper Wire - F/F

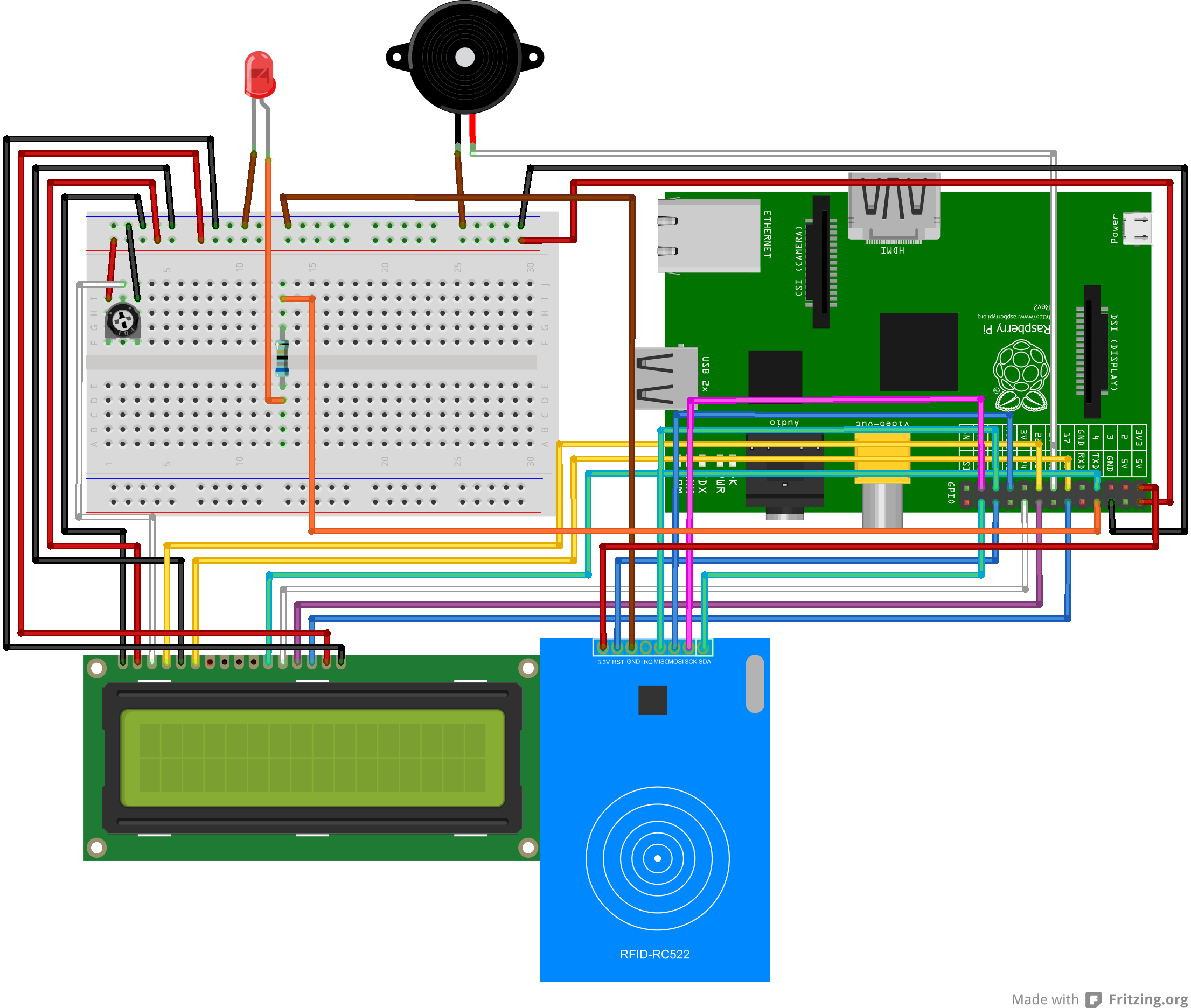
Blank NFC Tags



Technology is largely shifting towards different types of wireless communications. These systems are useful and convenient when applied to meaningful everyday tasks. For example, companies are attempting to transition customers with Near-Field Communication (NFC) enabled cell phones into paying for everyday purchases with their device. NFC is a specialized subset of the RFID technology that is of particular interest for this project. The NFC technology is a powerful, yet simple way to run an employee badge system. The simplicity, speed, and security it provides are what make it the perfect candidate for our project. An employee badge can be programmed to broadcast employee credentials to the reader. Employees would be able to use their badge with a time clock, security doors, sanitizer stations, or any other activity of interest. The cost of cards is very minimal, and the readers are reasonably priced enough to be installed in bulk around the hospital.

Radio Frequency Identification (RFID) is how devices are able to uniquely identify themselves using radio waves within a specified range. RFID is split into three categories: Low Frequency (LF), High Frequency (HF), and Ultra High Frequency (UHF). Each piece of RFID communicates on its own frequency band and has a specific purpose. The process is divided into two components: a reader and a tag. A reader can poll up to 100 meters, depending on the power source. An active reader has a large read range, while a passive reader’s range is dramatically shorter. Passive readers do not rely on their own power, and many conserve energy by broadcasting a shorter range.

NFC is largely valued for its convenience of use as well, as noted in the payment system example above. It is categorized as HF RFID that operates on the 13.56 MHz frequency band. As a subset or RFID, there is a further limitation on NFC that makes it practical for employee functions at the hospital. NFC readers have a stricter read range of less than 6 centimeters, which gives it a stronger security because the reader and tag must nearly be physically touching. In a real world situation, there should also be no reason that two tags would be in range of one reader at a time.

NFC readers are slightly more complex, and need some setup to work properly. Our project is using a Raspberry Pi to control an RC522 RFID reader. The Raspberry Pi is running the latest version of the Rasbian software. (Command line or GUI version) The RC522 has a specific connection configuration, which uses female-to-female jumper wire to connect the pins from the Raspberry Pi to the RC522. The NFC tags that we chose to develop with have a 46-byte capacity, which means we could encode a short URL onto the cards that would communicate with the hospital database.

**Fig. 1**

Wiring diagram that shows how the RC522 reader (blue) is connected to the Raspberry Pi GPIO port (green) through a series of jumper wire.

The FMC badge system currently uses plastic badges that have a magnetic stripe that is programmed with employee identification information. When swiped through door readers and time clocks, it relays the information about the employee, which is handled by the system. There is not currently any advanced badge technology in the hospital and any changes would need new equipment. As mentioned before, the equipment is fairly inexpensive and a process could be refined to make the equipment set up simple.

We first needed to get all the connections and software set up before proceeding with testing a NFC tag. Use the jumper cables to wire the RC522 to the Raspberry Pi. Install MySQL onto the Pi. There are many guides that will explain how to use the command line to install and set up a local or remote MySQL database. Here is one example:

<http://www.raspberry-projects.com/pi/software_utilities/mysql>

Our project accomplished reading NFC tags in two different ways. It can read tags using a Python script, and also using a C script. First, there is a SQL file that will create the necessary tables in the MySQL database. Next, executing the attendance.py script will open the RC522 for reading tags. The script will take the tag ID and a timestamp, and insert it into the MySQL database.

This was only accomplished on a local database. We attempted to use a remote database through the NAU TUND server, but because of the CAS system used by NAU we were unable to successfully open a connection to the database.

**Python**

Use this command in the command line to start the program:

sudo python attendance.py

**Below is a list of the files used in the python execution, and a short description of their function.**

**Scripts:**

* **Mysql\_conn\_test.py**
  + **Insert the connection settings for the database that will be used into this script, and test if the connection to the database is successful.**
* **Mysql.sql**
  + **Sql script is used to create the necessary tables in the MySQL database to correspond to the NFC reader output data.**
* **Attendance.py**
  + **Main script that ties all the other scripts together. Initializes the GPIO port and keeps an open connection that listens for a read on the NFC reader. Script continues to run until there is a keyboard interrupt that exits the program.**
* **Nfc.py** 
  + **Initializes the NFC reader and keeps a constant connection that will scan for any incoming NFC tags.**
* **Mysql.py**
  + **Initializes the connection to the MySQL database, and allows for data to be inserted on the corresponding table.**
* **MFRC522.py**
  + **Low-level code that interacts between the Raspberry Pi and the NFC reader using the GPIO connection.**

**C**

Use the following series of commands in the command line to start the program:

sudo python attendance.py

gcc config.c rfid.c rc522.c main.c –o attendance –lbcm2835

sudo cp RC522.conf /etc

sudo ./attendance –d (The –d is only necessary to print additional information to the screen and may not be necessary in practical applications)

After starting the program, enter a location name and press enter.

The C implementation does not provide functionality for creating or editing databases or tables. The only functionality provided by the C files is reading data from an NFC card and inputting data into a predefined database.

**Below is a list of the files used in the C execution and a short description of their function.**

* **Config.c**
  + **Allows for interfacing with a configuration file**
* **Main.c**
  + **Links together all parts of the program and creates a single, runnable executable**
* **RC522.c**
  + **Provides low level interaction between the RPi and NFC reader via GPIO connection**
* **RFID.c**
  + **Provides low level functions specific to the RFID reader.**

**Feasibility, Recommendations, and Practical Application:**

Using NFC readers is a feasible approach to creating an attendance system. This is substantiated by the following points:

* An attendance-tracking station can be built for under one-hundred dollars
* With proper instruction, implementing a Raspberry Pi based NFC attendance system is fairly simple
* NFC readers have a low bit error rate (they are accurate)
* The developed system can be scaled very easily

To put this system into practice, it is recommended that users employ the Python scripts. There is no noticeable performance difference between the Python and C versions of the system, but Python is traditionally much more readable. Additionally, the python scripts do not require compilation- so they are not only more readable, but they are easier to work with.

To create a working system, the only things that need to be changed within the code are the names of the databases, tables, and fields into which the information should be inserted. It is recommended that further work be conducted to improve the system by adding audible or visible feedback when a card is read. This is due to the fact that cards are occasionally not read even when placed in close proximity to the scanner.**References**

<http://blog.atlasrfidstore.com/rfid-vs-nfc>

<http://en.wikipedia.org/wiki/Near_field_communication>

<http://www.raspberrypi.org/wp-content/uploads/2015/01/Pi2ModB1GB_-comp.jpeg>

<http://www.instructables.com/files/orig/FBB/V9TI/HPT91MG3/FBBV9TIHPT91MG3.png>

<http://www.identivenfc.com/media/catalog/product/cache/1/image/500x500/9df78eab33525d08d6e5fb8d27136e95/n/f/nfc_cards_blank.png>

<http://miniimg.rightinthebox.com/images/384x384/201311/anoopp1385538290576.jpg>

<http://www.seeedstudio.com/depot/images/20pin300mm_LRG.jpg>