Midfield Data Transfer System

Cal Poly Computer Engineering Capstone

The Midfielders

Adam Levasseur and Erik Miller

March 14th, 2017

System Overview

Background

The midfield antenna was designed at Stanford University under Dr. Ada Poon. It was then recreated at Cal Poly by Grad Student Danielle Nishida under Dr. Tina Smilkstein. It's original purpose was to power implanted devices and thus designed to efficiently transfer power through the body.

For our capstone project, Dr. Tina Smilkstein asked us to investigate the ability of the antenna to send data through flesh.

Design

With future integration into implants in mind, our system is designed to be modular, with SMA connections on the microcontrollers and antennas allowing them to be swapped out quickly and easily.

Our system contains two main parts: the transmitter and receiver. Both consist of an Adafruit Feather M0 RFM69HCW 433 MHz and an antenna. The transmitter consists of one Feather flashed with the transmitter sketch, and typically has the midfield antenna attached, while the receiver Feather is flashed with the receiver sketch, and typically has the loop antenna. As alluded to above, the antennas can be switched out to test other options, such as the plain wire antennas.

The intent of our system is provide a simple data pipeline through flesh. As it currently stands, the transmitter takes input via UART, translates it to SPI, then sends it out via 433 MHz RF (with the option to AES encrypt communication). The receiver takes in the 433 MHz signal, translates it to SPI, then sends it to the receiving computer via UART.

The modularity of the system allows it to be used in other applications. For example, the receiving microcontroller could be swapped out for a RFM69HCW 433 MHz chip and an FPGA, which could then be programmed wirelessly with SPI from the RFM chip.

We originally intended to use Analog Devices ADF7023 RF transceiver chips to handle our RF transmissions, as they offered several powerful features like Reed Solomon Forward Error Correction (which allows for some data to be recovered if it is lost). However, they are 5 mm X 5 mm surface mount chips, which would need better more advanced soldering equipment than we normally use as well as custom PCBs. Thus, we decided to switch to off the shelf chips

We have four main antennas that we use with our system. The first two are two identical pieces of enamel coated magnet wire, 16.5 cm in length, which act a 433 MHz quarter wave antennas, both for sending and receiving. The next simple antenna is the loop antenna, a 18.8 cm piece of magnet wire, soldered in a loop to a male SMA connector and intended to work with 1.6 GHz, which we typically use as a receiver for both the wire and midfield antennas. Lastly, the midfield antenna; originally designed to send power at 1.6 GHz, this green PCB is soldered to 4 SMA male connectors, then uses a power splitter to send signal to each port of the antenna. It is commonly used for transmitting to the loop antenna, and appears to work at both 433 MHz and 1.6 GHz.

Our original research indicated that we would need to mix the 433 MHz signal from our transceiver chips up to and back down from 1.6 GHz, but when we tested, the 433 MHz transceivers sent and received signal through the ostensibly 1.6 GHz antenna without issue.

Set up

Setting up the transceiver chips

Software Setup

The software required for this system includes, in order of installation:

Arduino IDE
This software is used to interact and upload code to the Feather boards.
Download Link: https://www.arduino.cc/en/Main/Software

Arduino Additional Board Manager - Adafruit
The system uses Adafruit's Feather M0 boards to send/receive data. These chipsets require special board managing software in order to use the Arduino IDE to interact and upload code to the Feather boards.
URL: https://adafruit.github.io/arduino-board-index/package_adafruit_index.json
Board Manger Drivers: **Adafruit AVR Boards**, **Adafruit SAMD Boards**, and **Arduino Leonardo & Micro MIDI-USB**

- For further installation support, visit: https://learn.adafruit.com/adafruit-feather-m0-radio-with-rfm69-packet-radio/setup and https://learn.adafruit.com/adafruit-feather-m0-radio-with-rfm69-packet-radio/using-with-arduino-ide

Arduino Library - RFM69 by LowPowerLabs
This library is required for the Feather M0 Board in order to communicate with the on-board RFM69HCW chip.
Github Repository: https://github.com/adafruit/RFM69

- For further installation support, visit: https://www.arduino.cc/en/Guide/Libraries

Arduino Library - SPI Flash by LowPowerLabs
This library is required for the Feather M0 board in order to use the SPI flash capabilities to use and flash the on-board RFM69HCW chip.
Github Repository: https://github.com/LowPowerLab/SPIFlash

- For further installation support, visit: https://www.arduino.cc/en/Guide/Libraries

Arduino-Serial - Command Line Program
The program is used to send and receive data to the Feather board's using UART. Before running the program, the files must be compiled to create an executable instance of the software.
Github Repository:
https://github.com/letmeadam/Midfield-Data-Transfer-System.git

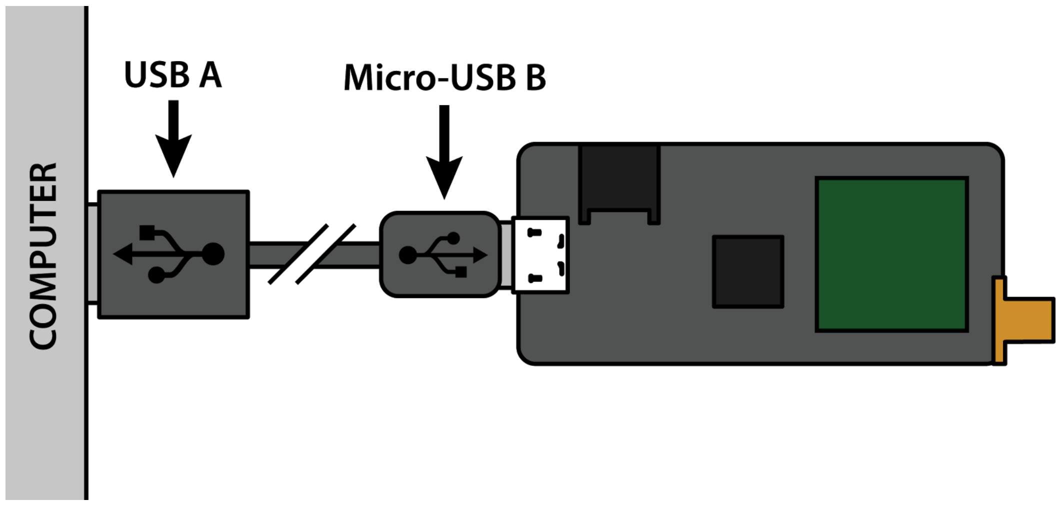
- For compiling instructions see the included "README.md"

Arduino Sketches - Transmitter/Receiver
The Arduino sketch files (.ino) are included in the previous Github Repository and are used to program the Feather M0 boards for data transfer.
Github Repository:
https://github.com/letmeadam/Midfield-Data-Transfer-System.git

- Validate the code using the ✓ button, and upload to the boards using the → button.

Connecting to Computer

Using a Male USB A to Micro-USB B cable, connect the Male USB A plug to the computer's USB A port and the Micro-USB B plug into the Micro-USB B port on the Feather M0 board. Repeat for second Feather M0 board.



Testing Connections

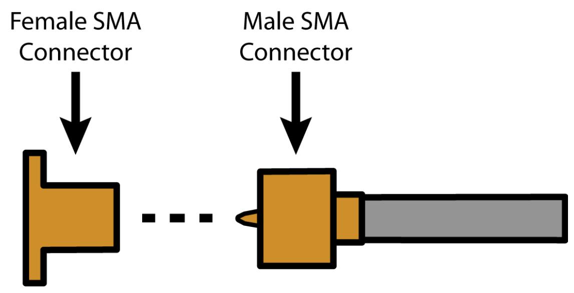
With the Arduino IDE open, observe the port menu from the upper tool bar: **Tools > Ports**. Under Windows, the port may appear in the format "COM#" (i.e. "COM3"). For MacOS and Linux, the format "*.usbmodem#" with common prefixes of "cu." and "tty." and suffixes corresponding to the usb port on the computer (i.e. "cu.usbmodem1412" and "cu.usbmodem1413"). Verify which port the Feather board is connected to by disconnecting the board, observing the Tools > Port menu and reconnecting the board to determine which port is being used.

Setting up the antennas

Connecting Antennas

The Feather boards have been outfitted with Female SMA connectors and require Male SMA connectors to attach the desired antenna. The Midfield and loop antennas currently also have female SMA connectors, so you may need to acquire male to male SMA connectors, either from the Microwave lab or from an electronics supplier like Mouser or Digikey.

Also, the board that the antenna is connected to may make a difference. The most common setup is to attach the midfield antenna to the board flashed with the transmitter code and the loop antenna to the board flashed with the receiver code. The user has the option of keeping track of which board is flashed with which firmware, opening up a serial monitor to see which firmware a board is running (the transmitter board will display transmitting, and the receiving board will display receiving), or simple flashing the boards with the appropriate firmware before use.



Increasing/Decreasing Frequency

Increasing and Decreasing the frequency from 433 MHz to any other frequency requires a frequency mixer to modulate the 433 MHz signal. The included frequency mixers are Mini-Circuit's ZX05-42MH-S+ frequency mixers which can operate between 5 and 4200 MHz. The operating 433 MHz signal generated from the transceivers can act as one source connected to the Radio Frequency (RF) port while the other source can be generated from a Frequency Synthesizer to the Local Oscillator (LO) port. The output signal will be generated on the Intermediate Frequency (IF) port as a combination of:

LO + RF = IF or **LO - RF = IF**

See more resources in **Appendix A.**

Operation

As mentioned in the connecting antennas section, ensure that the Feathers are associated with the correct roles, as any UART data sent to a board flashed for receiving will be ignored and nothing will happen. Again, either label the boards, check the serial monitor for the roles, or reflash the boards before use.

The following steps leverage a program we improved upon called Arduino-Serial. This is a command line program developed on MacOS, and will likely work on Linux (though it has not been tested). It may work on Windows 10, but we offer no guarantees with regards to that.

Transmitting a file

Any text sent via UART (at the correct baud rate) to a Feather flashed with the transmitting code will be converted to SPI, then to RF and sent out the attached antenna. In order to make the sending of files easier, we have added a feature to Arduino-Serial to take a file and send it over UART.

The specific command is:

```
./arduino-serial -b 115200 -p /dev/cu.usbmodem1412 -f input.txt
```

To break it down: "./arduino-serial" executes the program, the "-b" flag sets the baud rate (it is set to 115200, which is the rate set on the Feather), the "-p" flag sets the port the data is sent to (this must be set according to the serial port that the feather is attached to; see the Testing Connections section), and the "-f" flag identifies the file to be sent.

Receiving a file

Any RF packets addressed (and potentially correctly encrypted) to the Feather flashed with the receiving code will be converted to SPI, then sent out on the USB UART connection. Arduino-Serial also now supports collecting data and outputting it to a file.

The specific command is:

```
./arduino-serial -b 115200 -p /dev/cu.usbmodem1413 -v output.txt
```

To break it down: "./arduino-serial" executes the program, the "-b" flag again sets the baud rate (it is set to 115200, which is the rate set on the Feather), the "-p" flag again sets the port the data is sent to (this must be set according to the serial port that the feather is attached to; see the Testing Connections section), and the "-v" flag sets the name of the file that will be output to (if the file already exists, it will be overwritten).

Note: "./arduino-serial -h" will display the help prompt

Troubleshooting

No Data/Signal Received

- Ensure that all SMA connections are tight and electronically connected using a continuity sensor.
- Confirm that both Feather M0 boards are connected and programmed as one transmitter and one receiver. Use either the Serial Monitor provided by the Arduino IDE or by using the screen command.

  i.e. screen /dev/cu.usbmodem1412

Data Received is Corrupted

- Check the transmitting and receiving baud rates on both the computers and their corresponding Feather boards. The Arduino sketches are set to use a default 115200 baud rate.

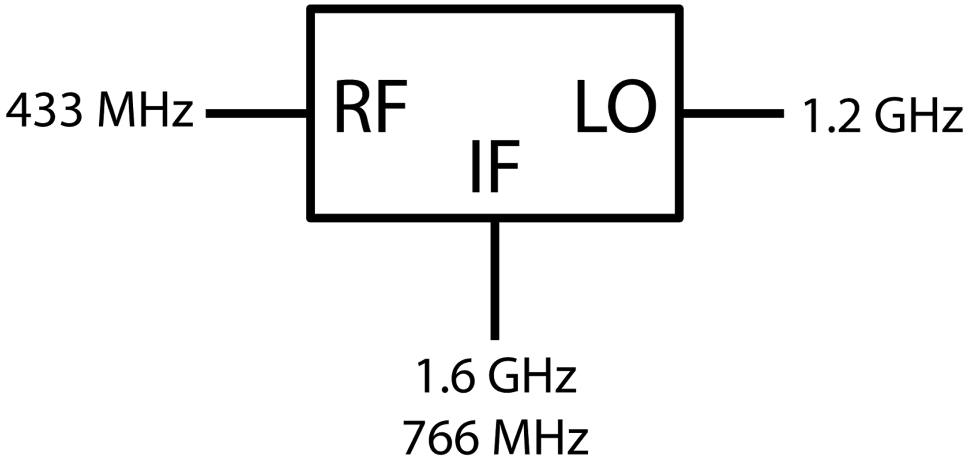
Packet Loss

- The Midfield Data Transfer System is a developing system that begins to address but does not completely send with an error-free protocol. The current system has been developed using one-way communication. Although two-way communication is possible with these Feather boards, further testing is necessary with use of the Midfield antenna.

Appendices

Appendix A: Frequency Mixing

Frequency mixing is the process of taking two RF signals and composing them into a more complex signal. In the case of this system, frequency mixing is used to increase or decrease the frequency of the data signal. The Frequency mixers included effectively use the already modulated 433 MHz signal from the Feather boards and mixes it with another desired carrier signal.



In the example set up in the figure above, two RF signals are inputted at 433 MHz (RF) and 1.2 GHz (LO). The output is a signal (IF) containing two frequencies (LO + RF and LO - RF). Technically the output signal also contains the mixing of the LO and RF signal harmonics, but for the intents and purposes of this system, only the original signals are observed.

Additional Frequency Mixing Resource:
http://www.radio-electronics.com/info/rf-technology-design/mixers/rf-mixers-mixing-basics-tutorial.php

Appendix B: Coding Acknowledgments

Arduino-Serial

The arduino-serial program is expanded from Tod Kurt's "arduino-serial-master" program. The original program by Kurt was designed to communicate to a serial port (i.e. the USB ports on a laptop). For the purposes of our designs, this program has been adapted to handle byte transmission and file I/O. Thanks to this program, our design is able to send complete files across the serial connections to the Feathers' on-board microcontrollers.

The original source code can be found at:

https://github.com/todbot/arduino-serial/

Arduino Sketches - Transmitter and Receiver

The Feather boards' programming was developed from the example Arduino sketches provided by Adafruit for the Feather M0 RFM69HCW boards. The example codes provide a template for simple message transmission and retrieval. Our design uses this general framework to constantly read/write to the connected serial port while sending/listening to the connected antenna.

The original example programs can be found at:
https://learn.adafruit.com/adafruit-feather-m0-radio-with-rfm69-packet-radio/using-the-rfm69-radio

Appendix C: Future Developments

As it is, the system is functional, but has several areas for possible improvement.

First, the transmission speed is somewhat slower than we had hoped. Improvements in transfer speed would make the system easier to use, and reduce errors incurred by movement during transmission. We suspect the main slowdown is caused by the overhead of converting from UART to SPI, then SPI back to UART.

Second, while USB out on the receiving end is good for testing data transfer integrity, it is not as useful for interfacing with implants. One interesting application could be to interface a RFM69HCW chip with an FPGA over spi, hypothetically allowing it to be programmed while implanted.

Third, while our antenna works, it's characteristics were not ideal. Redesigning it with 433 MHz in mind and integrating SMA connection pads into the PCB might significantly increase the performance.

Fourth, from the system perspective, the midfield antenna did not operate any differently than a simple quarter wave wire antenna. A project could investigate if the midfield antenna has any other desirable properties, such better transmission through large amounts (like 15 cm) of flesh, or if it heats the flesh less while transmitting than a normal antenna (which is desirable, as flesh that is heated too much dies).