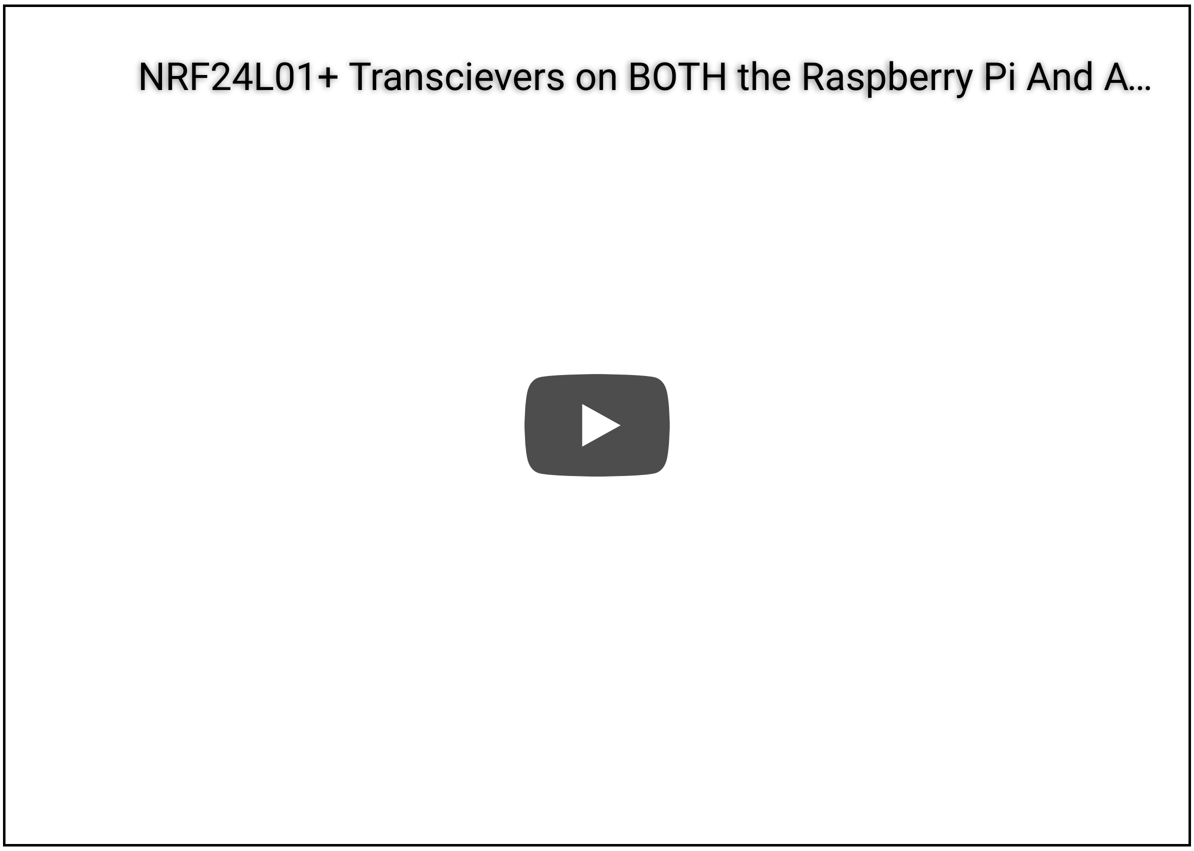
SPENCER NUSBAUM

PORTFOLIO

Home About Blog Projects Resume

Working with the NRF24L01+ Transcievers on the Raspberry Pi And Arduino

Jul, 22, 2018 Posted in Uncategorized



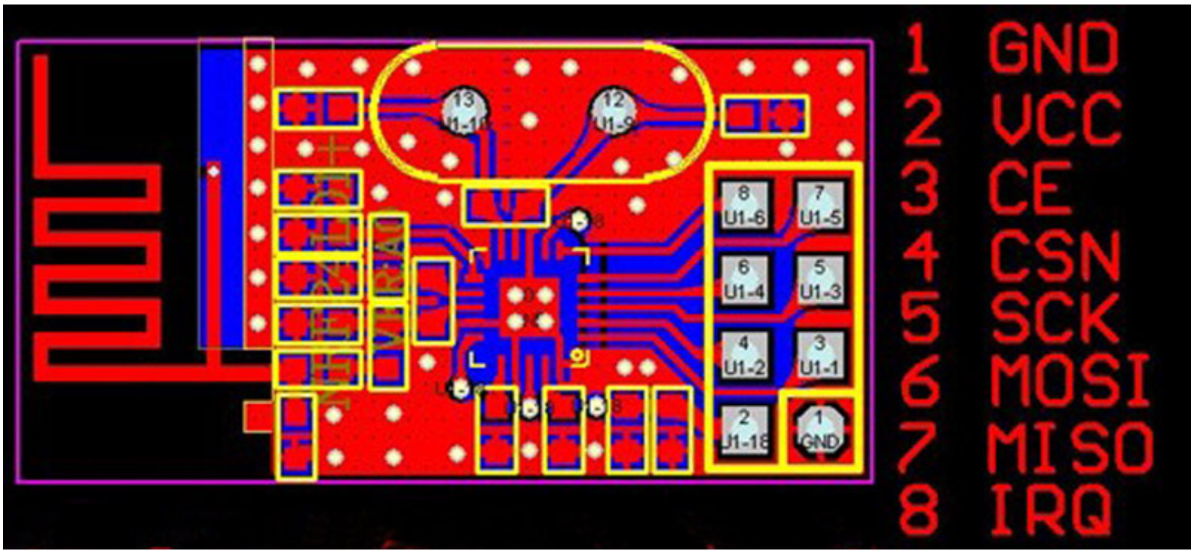
NRF24L01+ Transcievers on BOTH the Raspberry Pi And A…

After sifting through a bunch of resources for the NRF24L01+ modules, I finally came up a refined collection that I think would be helpful to someone else getting into the world of the NRF24L01+ modules.

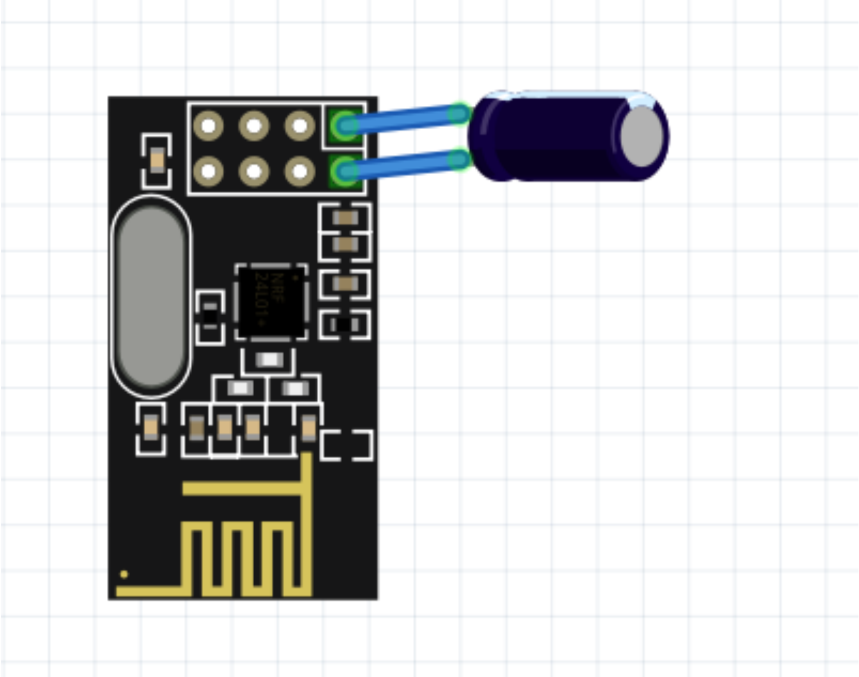
To overview, the NRF24L01+ module is a low power 2Mbps RF transceiver for the 2.4GHz ISM band, and it costs nearly $1.00 per unit. For example, they can be found on amazon, ten for $11.98 (as of 7/21/2018): https://www.amazon.com/Makerfire-Arduino-NRF24L01-Wireless-Transceiver/dp/B00O9O868G.

A majority of the information you will need can be retrieved from here, from the Optimized High Speed NRF24L01+ Driver Class Documentation v1.0:
http://tmrh20.github.io/RF24/ (Home Page)
http://tmrh20.github.io/RF24/classRF24.html (Class Reference)

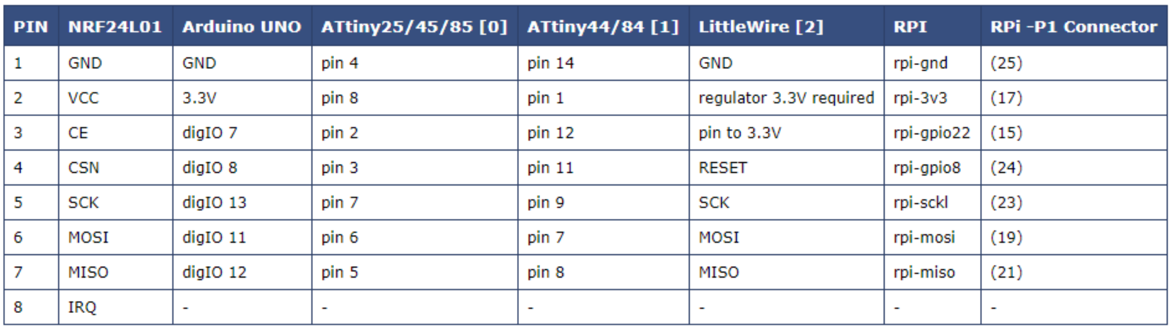
The NRF24L01/NRF24L01+ Module and pinout look as follows:



Power problems that occur with the module can be resolved by adding a 10uF capacitor to pins 1 and 2 (the capacitor can be soldered directly on the module if you wish), and this eliminates transceiver communication problems.

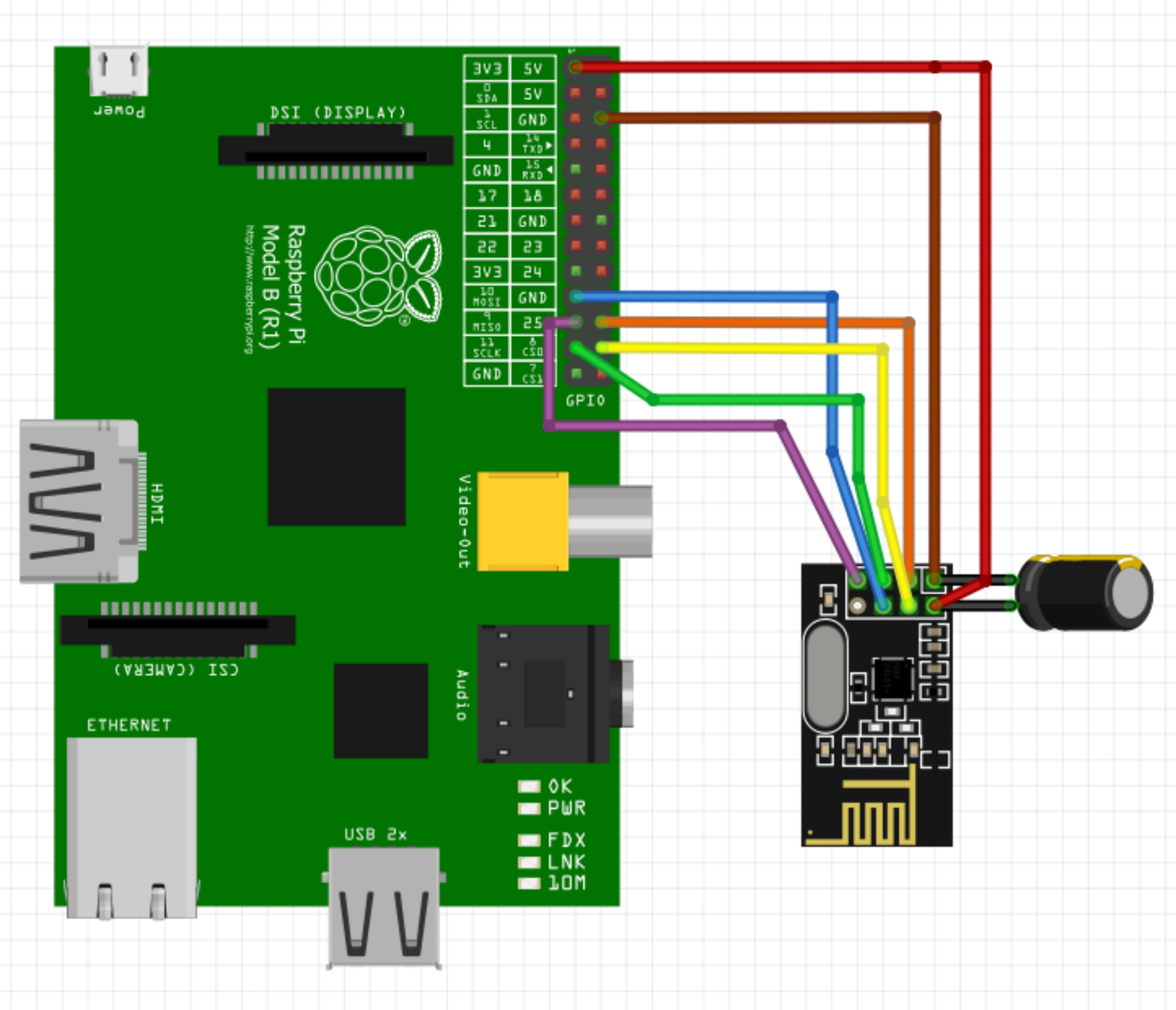


Wiring info can be found off this table below, found from http://tmrh20.github.io/RF24/ (7/21/2018).

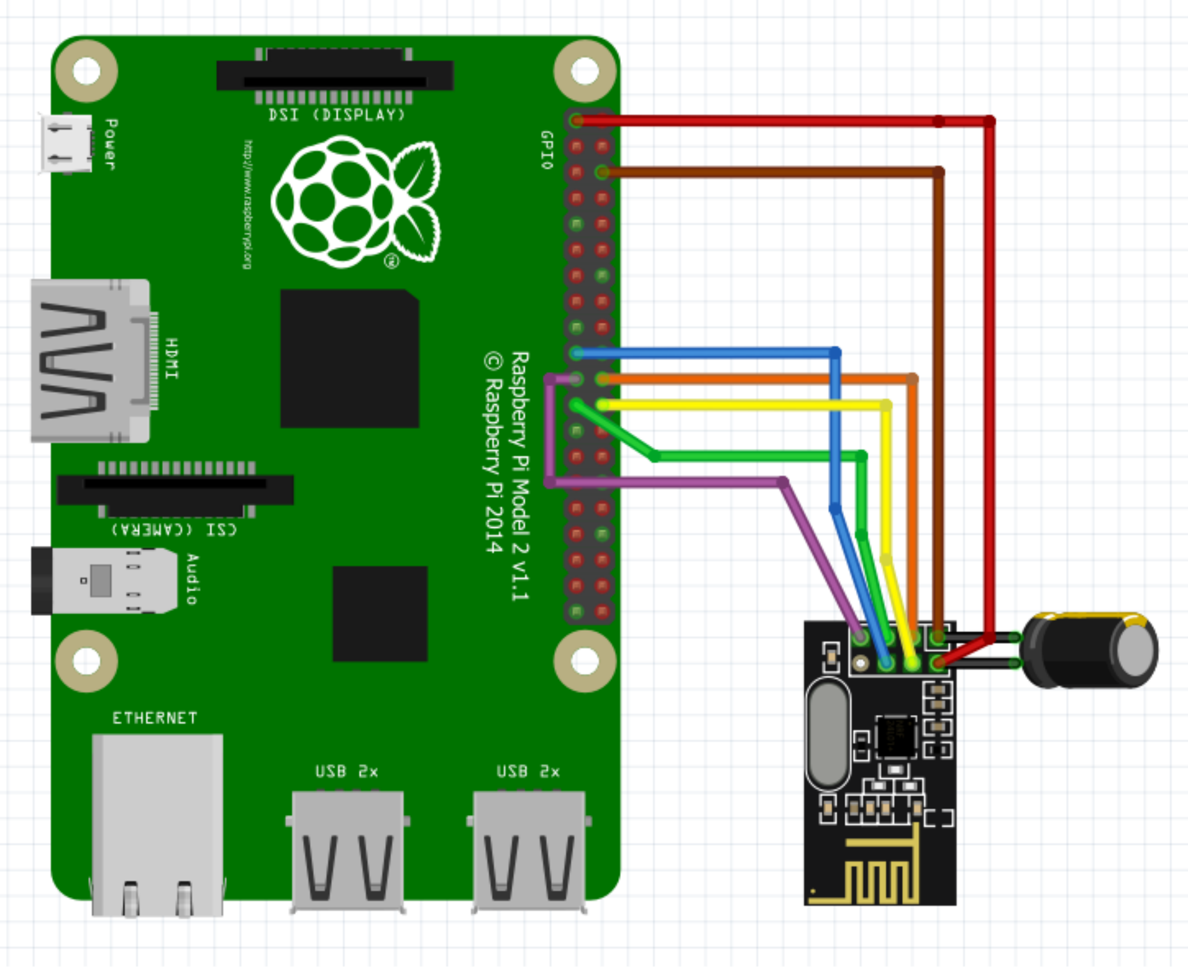


PIN	NRF24L01	Arduino UNO	ATtiny25/45/85 [0]	ATtiny44/84 [1]	LittleWire [2]	RPI	RPi -P1 Connector
1	GND	GND	pin 4	pin 14	GND	rpi-gnd	(25)
2	VCC	3.3V	pin 8	pin 1	regulator 3.3V required	rpi-3v3	(17)
3	CE	digIO 7	pin 2	pin 12	pin to 3.3V	rpi-gpio22	(15)
4	CSN	digIO 8	pin 3	pin 11	RESET	rpi-gpio8	(24)
5	SCK	digIO 13	pin 7	pin 9	SCK	rpi-sckl	(23)
6	MOSI	digIO 11	pin 6	pin 7	MOSI	rpi-mosi	(19)
7	MISO	digIO 12	pin 5	pin 8	MISO	rpi-miso	(21)
8	IRQ	-	-	-	-	-	-

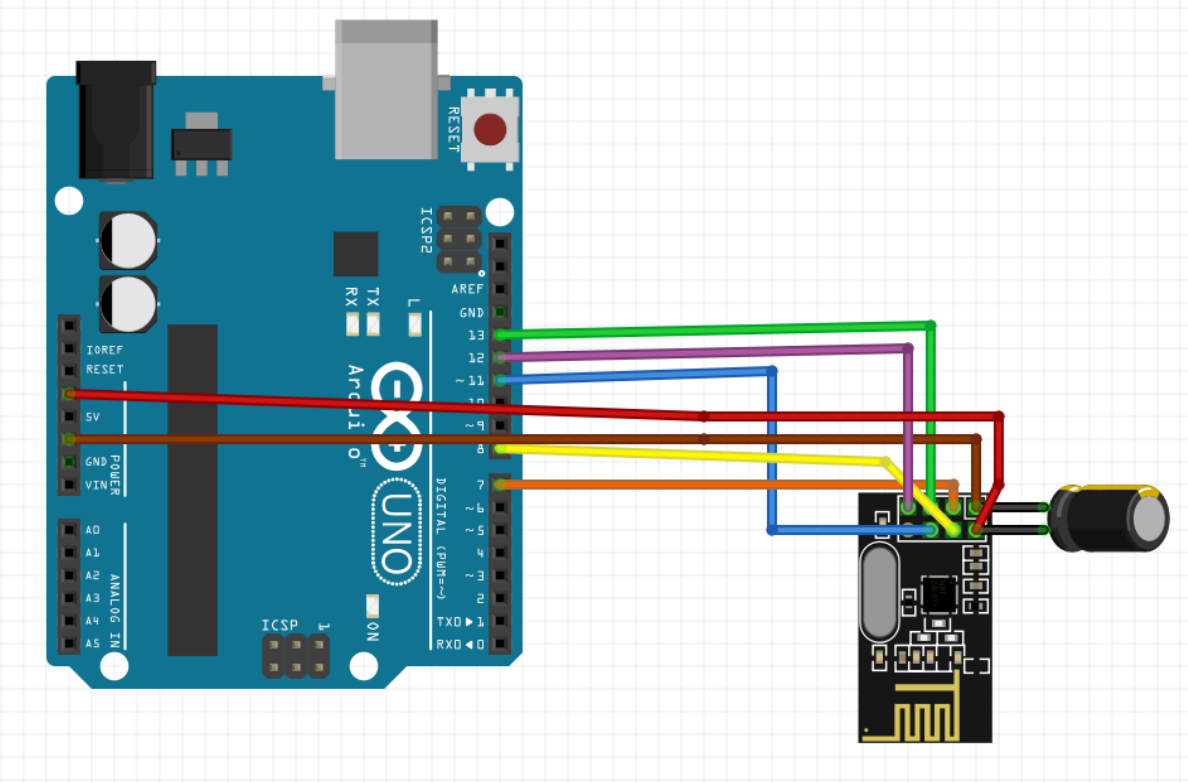
Though, it’s much easier in my opinion to look at a picture schematic of where the wires go. Here’s one for the Raspberry Pi Model B:



And here's one for the Raspberry Pi Model 2:



And here's one for the Arduino Uno:



Configuring the Raspberry Pi can be done in a few steps:

```
sudo apt-get update && sudo apt-get upgrade -y
sudo apt-get install -y git
git clone https://github.com/nRF24/RF24.git
cd RF24
sudo make install
```

Steps to configure the Raspberry Pi hosted with ❤ by **GitHub** **view raw**

Configuring the Arduino IDE is also fairly straight forward:

```
As of 7/21/2018 download RF24-master.zip from: https://github.com/nRF24/
Go to Sketch -> Include Library -> Add .ZIP Library…
Add the RF24-master.zip file
```

Steps to configure the Arduino IDE hosted with ❤ by **GitHub** **view raw**

Code samples for the Raspberry Pi are as follows:

receiver.cpp Makefile

```
######################################################################
#
# Makefile for Raspberry Pi NRF24L01/NRF24L01+ receiver
#
# Run:
#     make clean; make
#     sudo ./receiver
######################################################################
prefix := /usr/local

# The recommended compiler flags for the Raspberry Pi
CCFLAGS=-Ofast -mfpu=vfp -mfloat-abi=hard -march=armv6zk -mtune=arm117

# define all programs
PROGRAMS = receiver
SOURCES = ${PROGRAMS:=.cpp}

```

```

all: ${PROGRAMS}

${PROGRAMS}: ${SOURCES}
        g++ ${CCFLAGS} -Wall -lrf24-bcm $@.cpp -o $@

clean:
        rm -rf $(PROGRAMS)

install: all
        test -d $(prefix) || mkdir $(prefix)
        test -d $(prefix)/bin || mkdir $(prefix)/bin
        for prog in $(PROGRAMS); do \
          install -m 0755 $$prog $(prefix)/bin; \
        done

.PHONY: install

```

receiver.cpp Makefile hosted with ❤ by **GitHub** **view raw**

receiver.cpp

```
#include <iostream>
#include <RF24/RF24.h>

RF24 radio(RPI_V2_GPIO_P1_22, RPI_V2_GPIO_P1_24, BCM2835_SPI_SPEED_8MH
const uint8_t data_pipe[6] = "00001";

void setup(void) {
  radio.begin();
  radio.setRetries(15, 15);
  radio.setPALevel(RF24_PA_MAX);
  radio.openReadingPipe(1, data_pipe);
  radio.startListening();
}

int main(int argc, char** argv) {
  setup();

  while (true) {
    if (radio.available()) {
      int payload_size = radio.getDynamicPayloadSize();
      if (payload_size > 1) {
        char* payload = new char[payload_size + 1];
        radio.read(payload, payload_size);
        payload[payload_size] = '\0';
        std::cout << "Got Message: " << payload << std::endl;
      }
    }
  }
}

```

receiver.cpp hosted with ❤ by **GitHub** **view raw**

transmitter.cpp Makefile

```
######################################################################
#
# Makefile for Raspberry Pi NRF24L01/NRF24L01+ transmitter
#
# Run:
#     make clean; make
#     sudo ./transmitter
#
######################################################################
prefix := /usr/local

# The recommended compiler flags for the Raspberry Pi
CCFLAGS=-Ofast -mfpu=vfp -mfloat-abi=hard -march=armv6zk -mtune=arm117

# define all programs
PROGRAMS = transmitter
SOURCES = ${PROGRAMS:=.cpp}

all: ${PROGRAMS}

```

```
${PROGRAMS}: ${SOURCES}
        g++ ${CCFLAGS} -Wall -lrf24-bcm $@.cpp -o $@

clean:
        rm -rf $(PROGRAMS)

install: all
        test -d $(prefix) || mkdir $(prefix)
        test -d $(prefix)/bin || mkdir $(prefix)/bin
        for prog in $(PROGRAMS); do \
          install -m 0755 $$prog $(prefix)/bin; \
        done

.PHONY: install
```

transmitter.cpp Makefile hosted with ♥ by **GitHub** **view raw**

transmitter.cpp

```
#include <iostream>
#include <RF24/RF24.h>

RF24 radio(RPI_V2_GPIO_P1_22, RPI_V2_GPIO_P1_24, BCM2835_SPI_SPEED_8MH
const uint8_t data_pipe[6] = "00001";

void setup(void) {
  radio.begin();
  radio.setRetries(15, 15);
  radio.setPALevel(RF24_PA_MAX);
  radio.openWritingPipe(data_pipe);
}

int main(int argc, char** argv) {
  setup();

  if (argc != 2) {
    std::cout << "Usage: " << argv[0] << " <message to send>";
    return -1;
  }

  char* data = argv[1];
  radio.write(data, strlen(data) + 1);
  std::cout << "Data Sent" << std::endl;
}

```

transmitter.cpp hosted with ❤ by **GitHub** **view raw**

Code samples for the Arduino are as follows:

receiver

```
#include "RF24.h"
#include "printf.h"

RF24 radio(7, 8);
const byte data_pipe[6] = "00001";

void setup() {
  Serial.begin(9600);
  printf_begin();

  radio.begin();
  radio.setRetries(15, 15);
  radio.setPALevel(RF24_PA_MAX);
  radio.openReadingPipe(1, data_pipe);
  radio.startListening();
}

void loop() {
  if (radio.available())
  {
    int payload_size = radio.getDynamicPayloadSize();
    if (payload_size > 1)
    {
      char* payload = new char[payload_size + 1];
      radio.read(payload, payload_size);
      payload[payload_size] = '\0';
      printf("Got Message: %s\r\n", payload);
    }
  }
}
```

receiver hosted with ❤ by **GitHub** **view raw**

transmitter

```
#include "RF24.h"

RF24 radio(7, 8);
const byte data_pipe[6] = "00001";

void setup() {
  radio.begin();
  radio.setRetries(15, 15);
  radio.setPALevel(RF24_PA_MAX);
  radio.openWritingPipe(data_pipe);
}

void loop() {
  char data[] = "Hello world!";
  radio.write(data, strlen(data));
  delay(1000);
}
```

transmitter hosted with ❤ by **GitHub** **view raw**

Hope this helps!

Share:

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

POST COMMENT

Free WordPress Theme designed by Gavick.com
Proudly published with WordPress