NRF24L01+RF Transceiver Quickstart Guide

The **nRF24l01+ RF Transceiver Module** is a wireless Transmitter/Receiver (transceiver) device that adds wireless data transmission to your microcontroller project using the 2.4GHz Band. Pair up 2 of these modules and you can send data back and forth with each other at range of approximately 25 to 50 meters (depending on the line of site, wall obstructions and data rate kbps selection). This tiny board can be wired up directly to the Arduino board using the SPI pins.

HARDWARE SPECIFICATIONS

- Worldwide 2.4GHz ISM band operation
- 250kbps, 1Mbps and 2Mbps on air data rates
- Ultra low power operation
- Power supply : 1.9~3.6V
- Dimension : 12x18mm

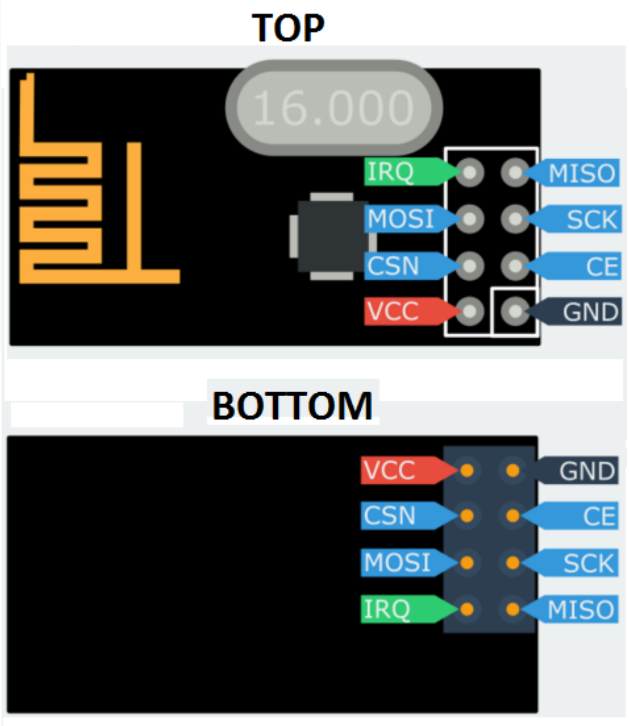
PARTS LIST

For this quickstart guide, we will need the following materials:

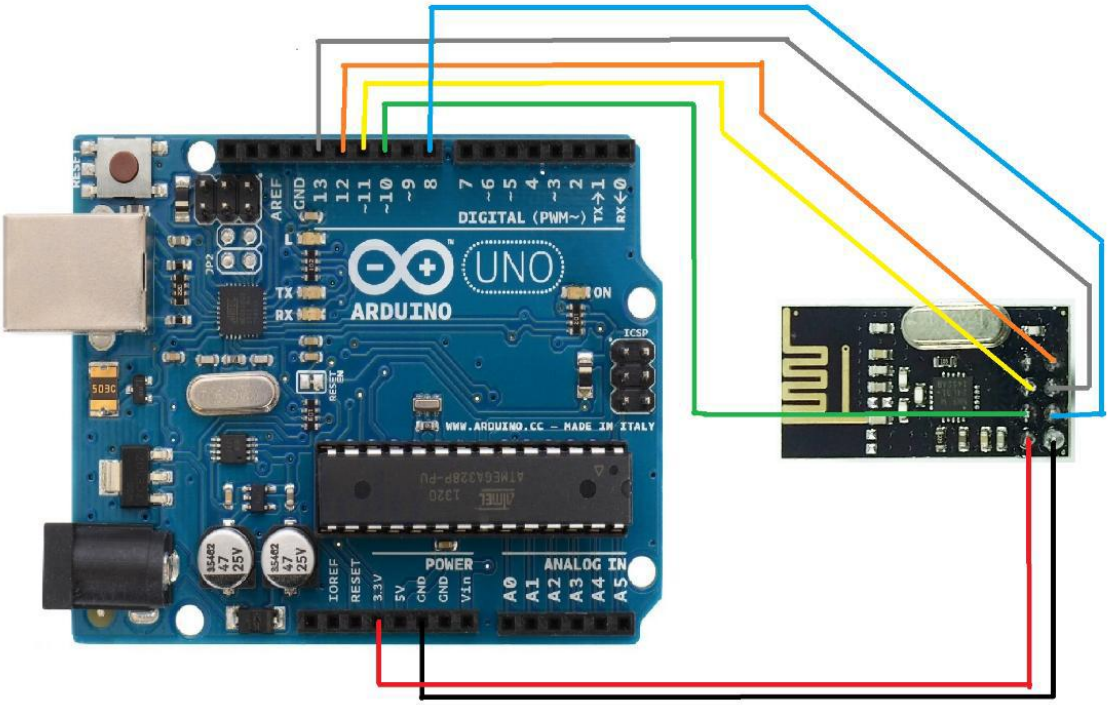
- 1 – Arduino Uno: https://www.bitstoc.com/product/1/
- Connecting Wires – https://www.bitstoc.com/product/121/
- 2 – nRF24L01 Wireless Transceiver Module

HARDWARE OVERVIEW

Below is pin-out of the nRF24 Transceiver. We will use this as reference to connect the pins to the Arduino.



HARDWARE CONFIGURATION

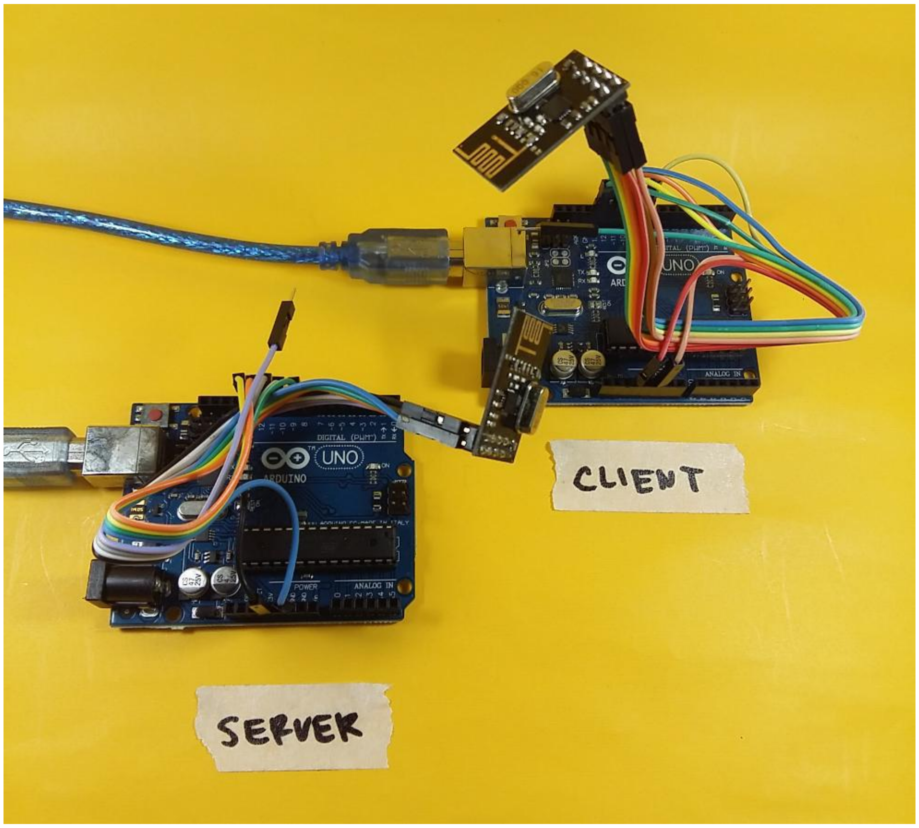


Connect the Arduino and nRF24 Module by following the diagram above. The nRF24 module and the Arduino talk to each other using the SPI communication protocol. *Note that the nRF24 input voltage is 1.9V to 3.6V, it connects to the* ***3.3V supply pin*** *of the Arduino, do not go beyond this voltage or the module will be damaged*. Review the connections from below.

The wiring of the server and client transceivers are the same. Check the table below for verify if your connection is correct.

CLIENT/SERVER circuit connections

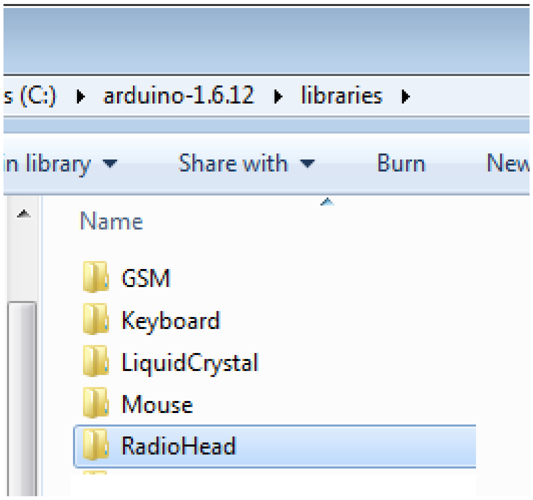
Arduino PIN VCC (3.3V)	to nRF24 Module PIN VCC
Arduino PIN GND (3.3V)	to nRF24 Module PIN GND
Arduino PIN 8 (SS)	to nRF24 Module PIN CE
Arduino PIN 10 (SS)	to nRF24 Module PIN CSN
Arduino PIN 11 (MOSI)	to nRF24 Module PIN MOSI
Arduino PIN 12 (MISO)	to nRF24 Module PIN MISO
Arduino PIN 13 (SCK)	to nRF24 Module PIN SCK
(Leave open, no connection)	nRF24 Module PIN IRQ



Actual Server and Client Setup

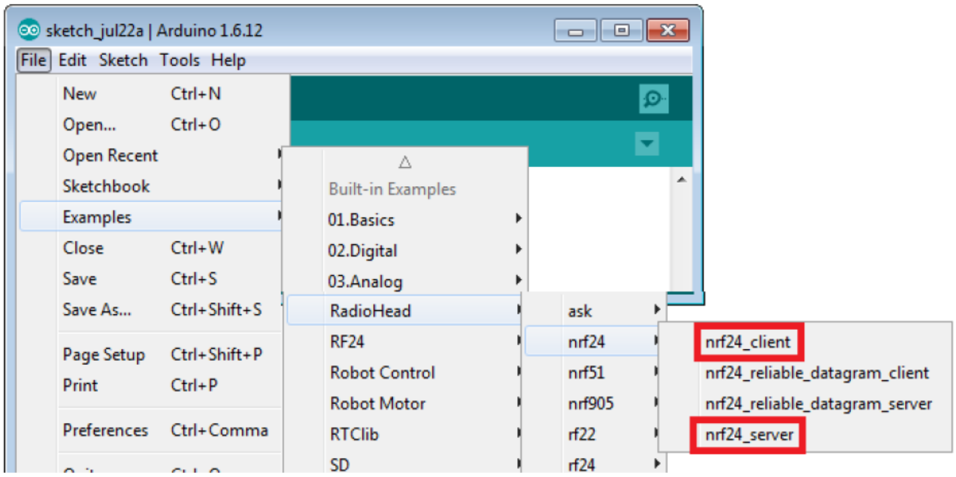
ARDUINO CODE

The great thing about the nRF24 module is it has a very good supporting Arduino library made by Radiohead. From the included **Files** folder you viewed from our tutorial link, locate and extract the folder **Radiohead** and paste it in your Arduino *librarie*s folder. Here our Arduino software files and libraries are in *Disk C:/arduino-1.6.12/libraries.* You may need to close all opened Arduino IDE and open again before you can use the Radiohead library.


s (C:) ▸ arduino-1.6.12 ▸ libraries ▸
n library ▾ Share with ▾ Burn New
Name

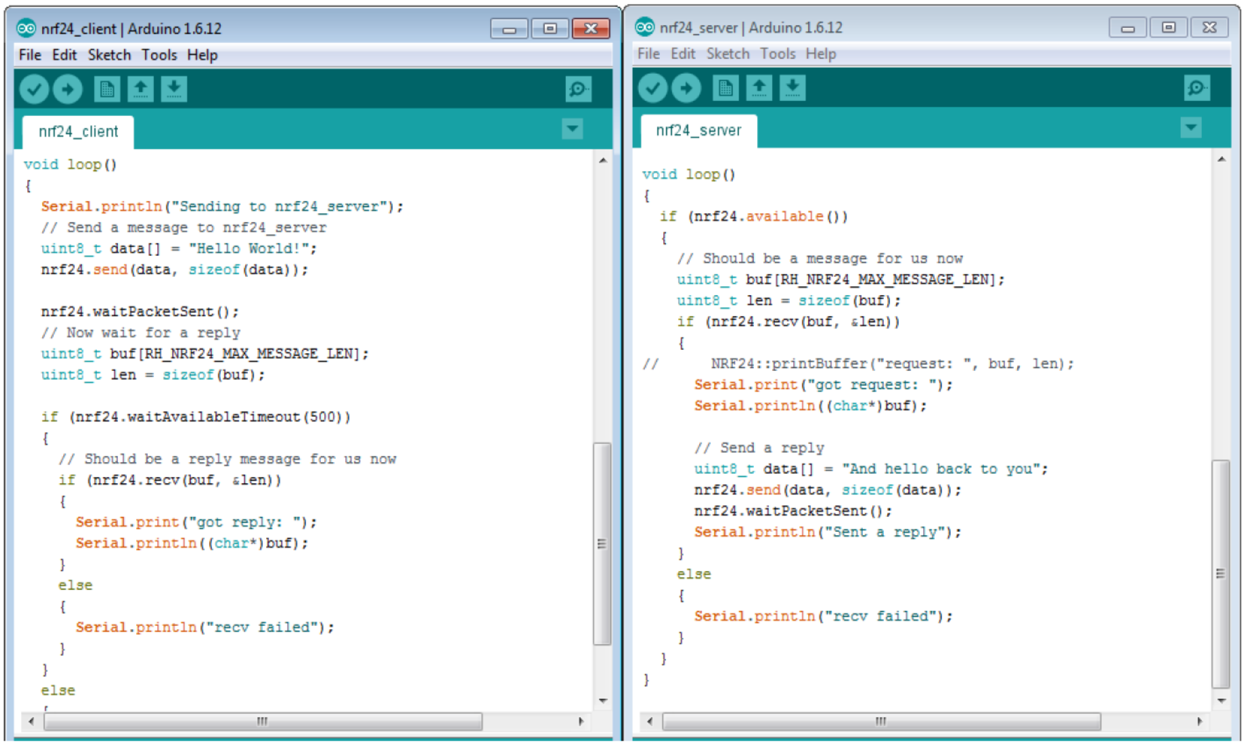
GSM
Keyboard
LiquidCrystal
Mouse
RadioHead

Open your Arduino IDE. Go to **File>Examples>RadioHead>nrf24**



From the nrf24 selection choose the **nrf24_client** Arduino code to Compile and Upload to your Client Circuit and the **nrf24_server** Arduino code to the Server Circuit.

The other two code selection with **reliable_datagram** can be used if you want to create multiple number of nodes like for example 6 clients to connect in 1 server. You may try these codes in your future projects.



```
void loop()
{
  Serial.println("Sending to nrf24_server");
  // Send a message to nrf24_server
  uint8_t data[] = "Hello World!";
  nrf24.send(data, sizeof(data));

  nrf24.waitPacketSent();
  // Now wait for a reply
  uint8_t buf[RH_NRF24_MAX_MESSAGE_LEN];
  uint8_t len = sizeof(buf);

  if (nrf24.waitAvailableTimeout(500))
  {
    // Should be a reply message for us now
    if (nrf24.recv(buf, &len))
    {
      Serial.print("got reply: ");
      Serial.println((char*)buf);
    }
    else
    {
      Serial.println("recv failed");
    }
  }
  else
```

```
void loop()
{
  if (nrf24.available())
  {
    // Should be a message for us now
    uint8_t buf[RH_NRF24_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);
    if (nrf24.recv(buf, &len))
    {
//      NRF24::printBuffer("request: ", buf, len);
      Serial.print("got request: ");
      Serial.println((char*)buf);

      // Send a reply
      uint8_t data[] = "And hello back to you";
      nrf24.send(data, sizeof(data));
      nrf24.waitPacketSent();
      Serial.println("Sent a reply");
    }
    else
    {
      Serial.println("recv failed");
    }
  }
}
```

A preview of the code above for the Client and Server.

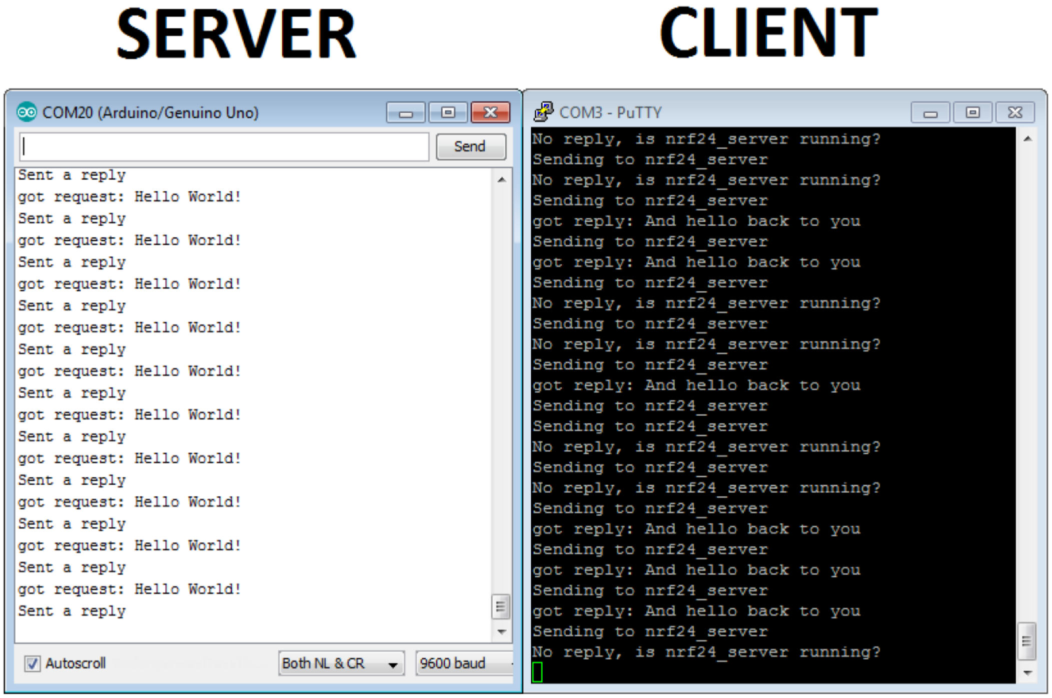
Compile and Upload each code and upload them to your assigned Client and Server Arduino set-up.

OUTPUT

Your Arduino codes are now running on both the Client and Server circuit. To view the exchanges of data open a Serial Terminal for the Server and Client circuit.

Here our Server circuit is running at COM20 port so we open our Arduino Serial Monitor at COM20 port. Using another terminal software like **Putty** (download it here http://www.putty.org/) or **Tera Term** (download it here https://ttssh2.osdn.jp/) open your Client circuit terminal. Here our Client circuit is using the COM3 Arduino port.

The Client circuit sends a streamed/packetized **"Hello World"** text to the Server circuit. Once the Server receives this text it prints out in the Arduino Serial Terminal what data it received like **got request: Hello World!** The Server will then send a reply to the Client and the Client prints out the received data in the terminal which is **got reply: And hello back to you**.



```
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
got request: Hello World!
Sent a reply
```

```
No reply, is nrf24_server running?
Sending to nrf24_server
No reply, is nrf24_server running?
Sending to nrf24_server
got reply: And hello back to you
Sending to nrf24_server
got reply: And hello back to you
Sending to nrf24_server
No reply, is nrf24_server running?
Sending to nrf24_server
No reply, is nrf24_server running?
Sending to nrf24_server
got reply: And hello back to you
Sending to nrf24_server
Sending to nrf24_server
No reply, is nrf24_server running?
Sending to nrf24_server
No reply, is nrf24_server running?
Sending to nrf24_server
got reply: And hello back to you
Sending to nrf24_server
got reply: And hello back to you
Sending to nrf24_server
got reply: And hello back to you
Sending to nrf24_server
No reply, is nrf24_server running?
```

APPLICATIONS and REFERENCES

Wireless Remote Using 2.4 GHz NRF24L01 by Akshay Jha:
https://www.instructables.com/id/Wireless-Remote-Using-24-Ghz-NRF24L01-Simple-Tutor/

Create a Private Chatroom using Arduino, nRF24L01 and Processing by B.Aswinth Raj:
https://circuitdigest.com/microcontroller-projects/chatroom-using-arduino-and-processing

https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/