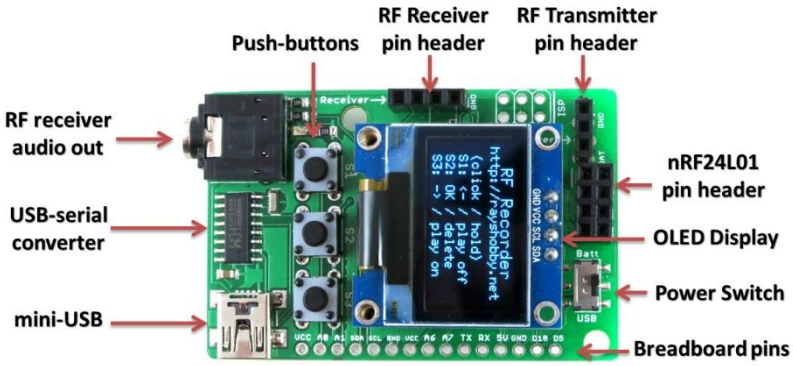
RFToy User Manual

(*Last update: Jan 27, 2014*)

Specifications

- **ATmega328p @ 3.3V, 8MHz**, with CH340G USB-serial converter and Arduino bootloader.
- **Programming using the on-board mini-USB port** (compatible with LilyPad Arduino or Arduino Pro 3.3V 8MHz)
- One **128×64 OLED** display, three tactile buttons.
- 20mm coin battery holder, and slide switch to select between USB or battery power.
- Pin headers for plugging in 433/315 MHz RF transmitter and receiver modules, and MOSFET power switches for them.
- 3.5mm audio jack to output receiver signals to a computer's line-in port, to monitor RF waves.
- Pin headers for plugging in nRF24L01 transceiver.
- Pin headers for connecting external components and/or breadboard experiments.
- Size: 1.5" x 2.3"

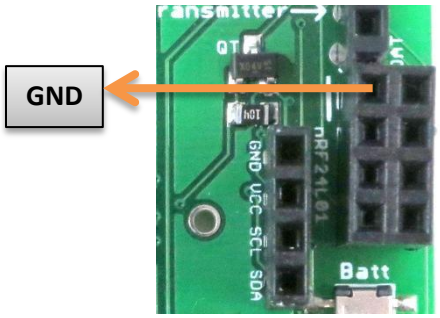
Hardware Interface

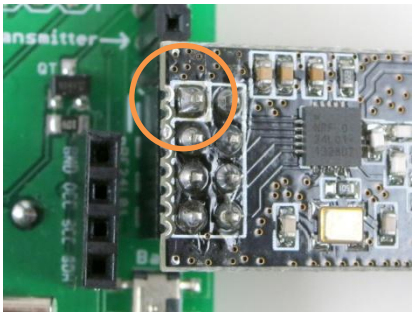


Plugging in RF Modules

nRF24L01 transceiver

To plug in nRF24L01 transceiver, note that the first pin (GND) on the 2x4 pin header is marked by a white line. This should match the same marked pin on your transceiver. See pictures below:



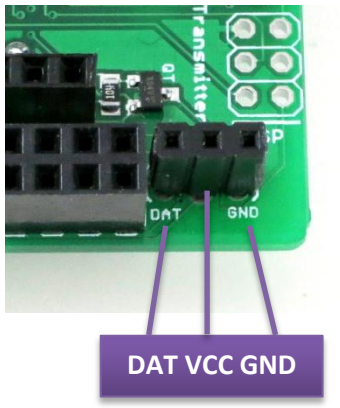


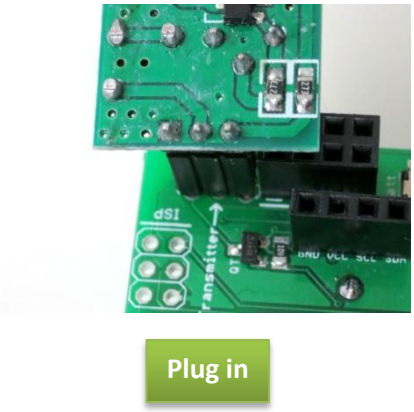
433/315MHz Receiver

To plug in a 433 or 315MHz receiver, note that the 4 pins on the receiver header are in the following order: **+5V, DAT, DAT, GND**. This order matches most common receivers. To double check, ensure the pins on your receiver match these 4 pins.

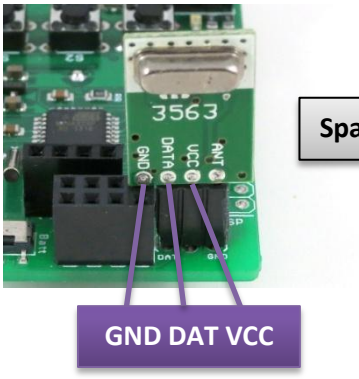
433/315MHz Transmitter

To plug in a 433 or 315MHz transmitter, note that the 3 pins on the transmitter header are in the following order: **DAT, VCC, GND**. This order matches most common transmitters. To double check, ensure the pins on your receiver match these 3 pins. You will need to bend the pins on the transmitter straight, and plug it in facing down. See pictures below:





NOTE: Some transmitters may have different pin orders. For example, one of SparkFun's transmitters uses this pin order: **GND, DAT, VCC**. In this case, you can make use of the **GND** pin on the nRF24L01 header to match the pin order:

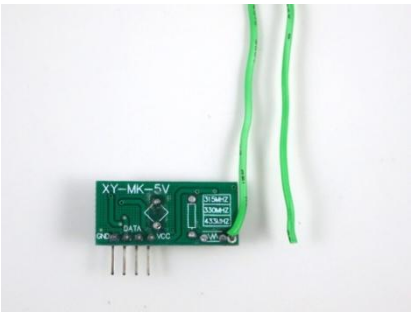


SparkFun's Transmitter

Operating Voltage, Antenna, and Transmission Range

- The **nRF24L01** transceiver's typical operating voltage is **1.9V to 3.6V** (all pins are 5V tolerant). It typically has built-in PCB antenna. The transmission range is 10 to 30 meters. If longer transmission range is needed, you can get nRF24L01 modules with built-in amplifier, which can extend the transmission range to hundreds of meters.
- The **433/315MHz transmitter's** operating voltage is **2.7V to 5V**. The higher the voltage, the longer the transmission range. The transmitter does NOT have built-in antenna, so you will need to solder a wire to the antenna pin. The recommended wire length is 17cm (6.7 inch) for 433MHz transmitter, and 24cm (9.4inch) for 315MHz transmitter. The wire length is calculated based on 1/4 of the wave length. Because the antenna is quite long, you can fold it down in half or curl it. See pictures below for example.
- The **433/315MHz receiver's** operating voltage is **5V**. It generally does not work if the voltage falls below 4.5V. Therefore when using the receiver you either need to power the circuit with USB, or use a battery pack that can provide 4.5V to 5V voltage. Similar to the transmitter, the receiver does NOT have built-in antenna. So you need to solder a wire as antenna, using the same recommended length as above.

nRF24 with amplifier



RF receiver with solder-on wire antenna

Transmitter with solder-on wire antenna

Programming RFToy

RFToy has a built-in ATmega328p microcontroller and USB-serial converter. The microcontroller is uploaded with the Arduino bootloader. It runs at 3.3V, 8MHz, and is **compatible with Lilypad Arduino or Arduino Pro 3.3V 8MHz w/ 328**. When programming the RFToy, plug in a mini-USB cable to RFToy's mini-USB port, then run Arduino, select **Tools -> Board -> Lilypad Arduino w/ ATmega328**; next, select the **Serial Port** (see below), and finally click on the **Upload** button to flash a program to RFToy.

Serial Port and Driver:

RFToy uses CH340G USB-to-Serial converter, which is a low-cost alternative to the popular FTDI chip. For:

- **Windows 7, 8, 8.1 and Linux:** no driver installation is needed.
- **Windows XP:** download and install driver http://raysfiles.com/drivers/ch341ser.exe
- **Mac OS:** download and install driver http://raysfiles.com/drivers/ch341ser_mac.zip

On Windows, the Serial Port name is **COM?** where ? is a number assigned to the USB-serial chip. On Linux, the Serial Port name is **/dev/ttyUSB?** where ? is a number. On Mac, the Serial Port name is **tty.wch ch341g xxx.**

RFToy Arduino library:

The **RFToy Arduino library** can be downloaded from: http://github.com/rayshobby/rftoy. You can either clone this Github repository, or download it as a zip: https://github.com/rayshobby/RFToy/archive/master.zip

The library has embedded the following open-source library code:

- Mirf library: for interfacing with nRF24L01 transceiver

- **VirtualWire** library: for interfacing with 433/315MHz RF transmitter and receiver
- **RCSwitch** library: for interfacing with RF remote power sockets
- **U8g** library: for interfacing with 128x64 OLED display

The library provides several starter demos. For details, please check the description in each demo therein.

Pin Assignment

Internally assigned pins:

- **D2**: RF receiver DATA pin
- **D7**: RF transmitter DATA pin
- **D4/5/6**: buttons S1/S2/S3
- **D8**: RF transmitter power (active high, pulled high by default)
- **D9**: RF receiver power (active high, pulled high by default)
- **D13**: indicator LED
- **D16/17**: nRF24L01 CSN/CE pins
- **D11/12/13**: nRF24L01 SPI pins (can be shared with other SPI devices)
- **SDA/SCL**: OLED I2C pins (can be shared with other I2C devices)
- **D0/1**: Serial RX/TX pins

Spare Pins: **D3, D10, A0, A1, A6, A7**

Breadboard Use:

RFToy has a row of 0.1”-pitch pin headers, which can be used to connect external components or for breadboard experiments.

Power Options

RFToy can be powered either by USB or a battery. The **Power Switch** is used to select the power source, between **USB** or **Batt**.

- **When using USB**, the +5V from USB will power the RF receiver and transmitter, and is stepped down to 3.3V to power the microcontroller and nRF24L01.
- **When using battery**, the battery voltage directly powers all components. RFToy also has a coin battery holder at the back of the PCB. It fits a standard 20mm coin cell, such as the CR2032 (3V), or LIR2032 (3.7V rechargeable). **Keep in mind that the RF receiver required at least 4.5V, so it won’t work with a 3V to 3.7V battery**.
- **If using an external battery**, you can solder the battery’s positive and negative wire to VCC and GND respectively. **Keep in mind that the battery voltage should not exceed 5V**. A standard 3.7V Lithium battery or 3V to 4.5V battery pack should work fine.

Reduce Power Consumption

For long-term sensing and environment monitoring projects, you should keep the microcontroller in power-down sleep most of the time to reduce power consumption. The nRF transceiver supports sleep mode too. The RF transmitter and receiver do not have a sleep mode, but RFToy uses separate MOSFETs to provide power to these two modules, so they can be programmably powered down to minimize power draw. Specifically, pins D9 and D8 are used to power down these two modules.

The RFToy library has a demo (**vwSender**) which shows how to combine power-down sleep, watchdog timer, and the RF receiver’s power pin to increase the battery life for long-term use. While sleeping, this demo draws less than 100uA current. It can be further reduced by removing the OLED display.

Using Audio-Out

The on-board 3.5mm audio jack can be used to outputs the RF receiver signal to a computer's line-in port. You can then use audio recording software, such as the open-source **Audacity** software, to record and analyze the RF signals. This is useful to reverse engineer ad-hoc RF protocols, the encoding pattern of which is not known ahead of time. This approach has been used to reverse engineer the RF signals from several off-the-shelf wireless sensors, such as temperature, humidity, rain, and soil moisture sensors. For details, please refer to this **blog post**.

Links and Resources

- **RFToy Homepage:** http://rftoy.rayshobby.net
- **RFToy Arduino library:** http://github.com/rayshobby/rftoy
- **RFToy Hardware Files:** http://github.com/rayshobby/rftoy-hw