

Starter Kit

Learning Arduino

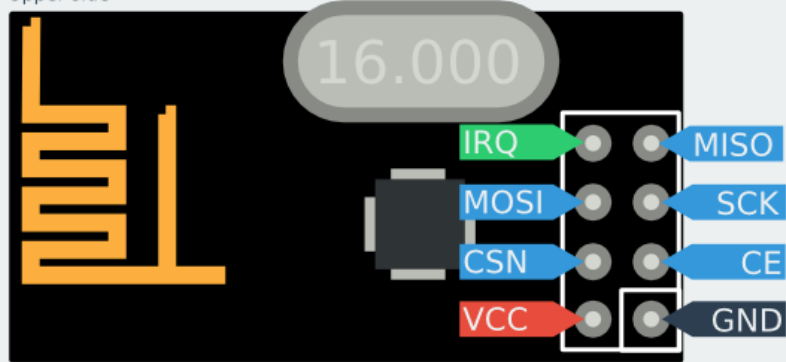
Connecting and programming nRF24L01 with Arduino and other boards

Connecting nRF24L01 and Arduino

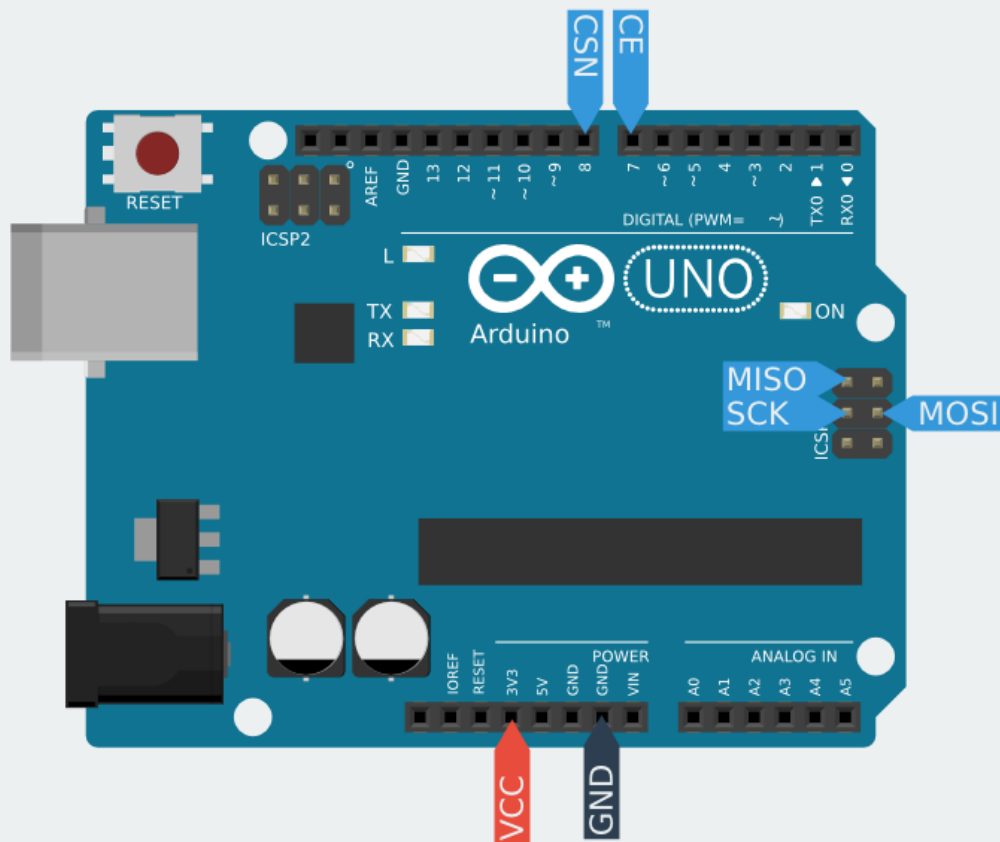
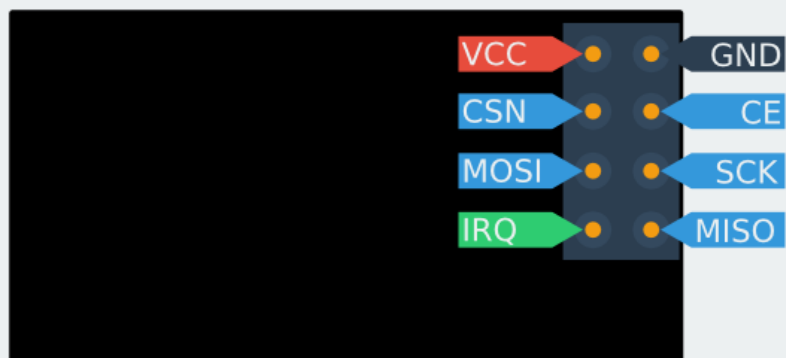
Now, when we know nRF24L01 module pinout we can now connect it to Arduino or some other board. Just connect pins on the same name on Arduino board and nRF24L01 wireless module:

nRF24L01+ module - connecting to Arduino

Upper side



Down side



Connecting nRF24L01 and Arduino4

Schematic is very universal and fits for all the Arduino's: UNO, DUE, MEGA, Leonardo, Yun etc. (Arduino 1.0 (R3) standard, but also with older boards)

SPI signals are in the ICSP connector. For connecting we suggest using [female/female jumper wires \(type FF\)](#). The rest of the signals can be connected using a [female/male jumper wires \(type FM\)](#).

Connect power pins from nRF to Arduino as shown below:

nRF24L01	ARDUINO
VCC	3.3V
GND	GND

CE and CSN pins can be connected to any digital pins. Then in RF24 library, you can specify which pins you used. I chose pins 7 and 8 because I will use them in the examples.

On Arduino UNO boards SPI pins are connected with some digital pins. While using modem you must remember that these digital pins won't be available.

- MOSI is connected to the digital pin 11
- MISO is connected to the digital pin 12
- SCK is connected to the digital pin 13
- SS (not used, but also blocks) is connected to the digital pin 10

The Arduino MEGA 1280 and 2560 have a similar situation.

- MOSI is connected to the digital pin 51
- MISO is connected to the digital pin 50
- SCK is connected to the digital pin 52
- SS is connected to the digital pin 53

On the Arduino DUE, Yun and Leonardo SPI pins are on ICSP connector, and are independent of the digital pins.

Programming nRF24L01

Having module connected, we need to program it. First program you probably know, we'll make traditional "Hello World".

We will make one device (with the modem), will send the string to the other device. The second device will send the received string to a stationary computer and then will display it in the Arduino Serial Port Monitor.

In this project we used RF24 library, which can be found on Github: RF24 library on Github. You only need to click on "Download ZIP" button and it'll start downloading all necessary things. You can install the library in Arduino IDE using Sketch-> Import library-> Add library. Another way is to extract the zip file to your Arduino home directory: Arduino/libraries on Linux or Documents/ Arduino/libraries in Windows.

Transmitter program will look like:

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8);

const byte rxAddr[6] = "00001";

void setup()
{
  radio.begin();
  radio.setRetries(15, 15);
  radio.openWritingPipe(rxAddr);

  radio.stopListening();
}

void loop()
{
  const char text[] = "Hello World";
  radio.write(&text, sizeof(text));

  delay(1000);
}
```

At the beginning of the sketch we inform the program that we'll use libraries.

- SPI.h – to handle the communication interface with the modem
- nRF24L01.h – to handle this particular modem driver
- RF24.h – the library which helps us to control the radio modem

Next, we need to create an object called “**radio**”

```
RF24 radio(7, 8);
```

This object represents a modem connected to the Arduino. Arguments 7 and 8 are a digital pin numbers to which signals CE and CSN are connected. If you have connected them to other pins can change this arguments. Then I create a global array called “**rxAddr**”.

```
const byte rxAddr[6] = "00001";
```

In this array we wrote the address of the modem, that will receive data from Arduino. Address has value “00001”, but if you want you can change it to any other 5-letter string. The address is necessary if you have a few modems in the network, thanks to the address, you can choose a particular modem to which you are sending the data. In the “setup” function we call the method “**radio.begin ();**”. It activates the modem.

Next we call “**radio.setRetries(15, 15);**” function. It shows how many times the modem will retry to the send data in case of not receiving by another modem. The first argument sets how often modem will retry. It is a multiple of 250 microseconds. $15 * 250 = 3750$. So, if the recipient does not receive data, modem will try to send them every 3.75 milliseconds. Second argument is the number of attempts. So in our example, modem will try to send 15 times before it will stop and finds that the receiver is out of range, or is turned off.

The method of “**radio.openWritingPipe (rxAddr);**” sets the address of the receiver to which the program will send data. Its argument is an array previously made with the receiver address.

The last method in the “setup” function is “**radio.stopListening ();**”. It switch the modem to data transmission mode.

In the “loop” function, we start with creating a string that we want to send using modem.

```
const char text[] = "Hello World";
```

It's an array of characters/letters to which we assigned a "Hello World" text. Then, using the method of **"radio.write (& text, sizeof (text));"** we send text through the radio to the modem (the address of the modem was set up earlier using "openWritingPipe"). First argument is an indication of the variable that stores the data to send. That's why we used "&" before the variable name, so we can make an indicator from this variable. The second argument is the number of bytes that the radio will take from a variable to be sent. Here we used the function **"sizeof ()"**, which automatically calculates the number of bytes in a "text" string.

Through this method, you can send up to 32 bytes at one time. Because that is the maximum size of a single packet data modem. If you need confirmation that the receiver received data, the method "radio.write" returns a "bool" value. If it returns "true", the data reached the receiver. If it returns "false" this data has not been received.

The **"radio.write"** method blocks the program until it receives the acknowledgment or until you run out of all attempts to transmit established methods set in **"radio.setRetries"**.

The last part of the "loop" function is **"delay (1000);"**. It blocks the program for 1000 milliseconds, or one second. It makes the program will send "Hello World" every second to the receiver.

The receiver

The program of the receiver in the second modem device will look like this:

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8);
```

```
const byte rxAddr[6] = "00001";

void setup()
{
  while (!Serial);
  Serial.begin(9600);

  radio.begin();
  radio.openReadingPipe(0, rxAddr);

  radio.startListening();
}

void loop()
{
  if (radio.available())
  {
    char text[32] = {0};
    radio.read(&text, sizeof(text));

    Serial.println(text);
  }
}
```

The program looks quite similar to the program of the transmitter. First we selected libraries which will be used and then we create a “radio” object with selected control pins. In the next line you can see a table with the address of the receiver – the same as in the transmitter. At the beginning of the “setup” function we set the object “Serial” for communication Arduino with the computer.

```
while (! Serial);
```

This part is waiting for the Arduino USB port switches to serial COM port when you connect USB cable. That is true for Arduino with ATmega32u4 – like Leonardo, for ATmega328 based boards, which have separate chip for USB/Serial communication Serial is available always. The method of “**Serial.begin (9600);**” sets the baud rate with the computer via USB / COM.

The next part of the function is to set the nRF24L01 modem. Like before we used “**radio.begin ();**” method. The next line of the program is “**radio.openReadingPipe (0, rxAddr);**”, which determines the address of our modem which receives data. The first argument is the number of the stream. You can create up to 6 streams that respond to different addresses. We created only address for the stream

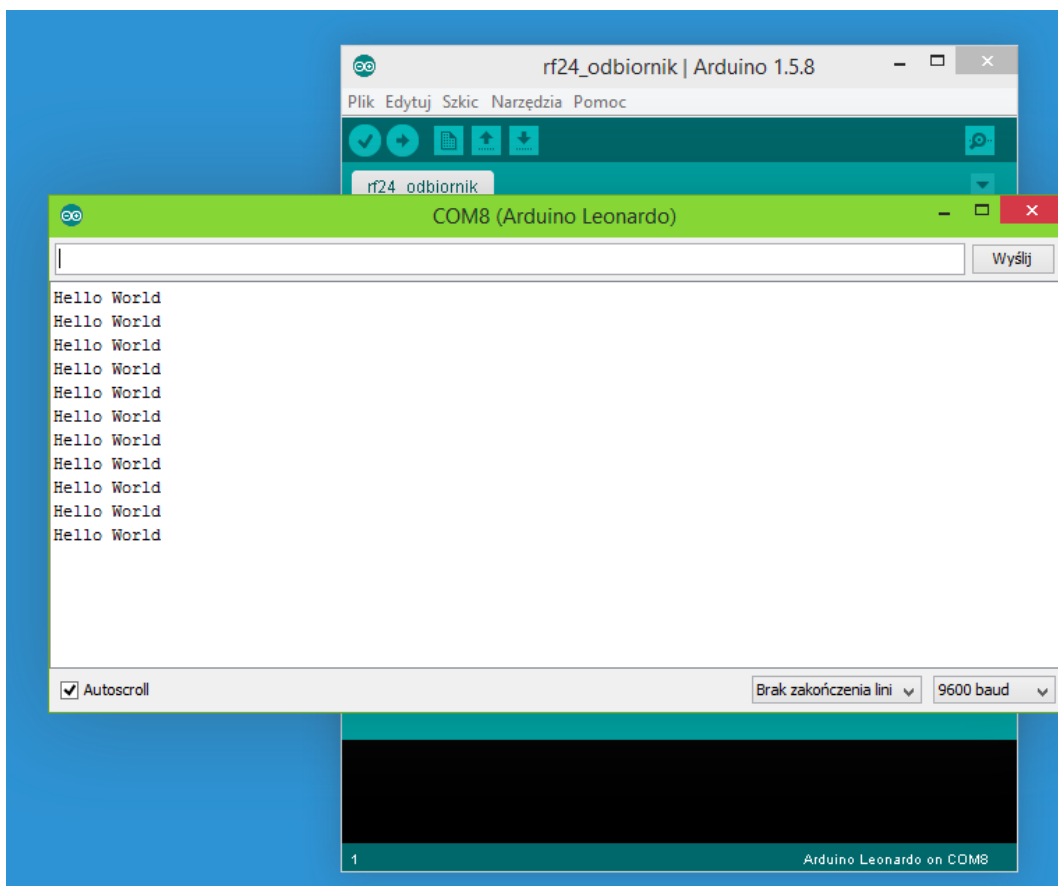
number 0. The second argument is the address to which the stream will react to collect the data. In this example we set the address assigned to a "rxAddr" array.

The next step is to enable receiving data via modem using the method "**radio.startListening ();**". From that moment the modem waits for data sent to the specified address. In the "loop" function, program performs the following operations. First checks whether any data have arrived at the address of the modem using the method "**radio.available ();**". This method returns a "true" value if we received some data, or "false" if no data.

```
char text[32] = {0};
```

If the data was received, then it creates a 32-element "char" type array called "text" and filled with zeros (later the program will fill it with the received data). To read the data we use the method "**radio.read (&text, sizeof (text));**". The first argument is an indicator of the variable to which you want to save the data received by the modem. To present a variable as an indicator we applied the "&" character in front of its name. The second argument is the amount of data to be stored in a variable. Here, we again have used the "**sizeof ()**" function, which automatically calculates the size of the "text" array.

When data is received, it send's it to the "**Serial.println (text);**" method. Then the received text is being sent to a computer, where you can see it in the "Serial Port Monitor" using the Arduino IDE. If you did everything ok and there are no mistakes in connections, you should see the same values in your Serial Port Monitor:



Serial Port Monitor showing communication

- RF24 library on Github
- nRF24L01+ module on Nettigo

Stay tuned for next articles. We have connected nRF24L01 with Arduino, we will cover [Teensy 3.1](#) and smaller μ like [ATtiny](#)

This entry was posted in [howto](#) and tagged [arduino](#), [howto](#), [nrf24l01+](#) on 2014/12/04 [<http://starter-kit.nettigo.eu/2014/connecting-and-programming-nrf24l01-with-arduino-and-other-boards/>] by [nettigo](#).

38 thoughts on “Connecting and programming nRF24L01 with Arduino and other boards”



Kumaran

2015/02/03 at 16:27

I was having problem with NRF24 for almost 2 weeks now (Arduino and Raspberry Pi). This is due to many misleading information out there that doesn't work or confusing or too much information that not required for basic "Hello World" program.

After spending lots of time trying sample scripts that doesn't work, I was almost gave up. Finally, found your website with super simple script and worked like charm on first attempt.

Your graphical guide (clean and clear) helps a lot in getting the pins connected without mistakes.

Step by step explanation on script lines helps to understand it's function more clearer. In fact it's a great "Hello World" tutorial that everyone can get started without issue.

Now only I know that only few lines of codes needed and it's enough to test NRF24 "Hello World" program.

Keep up your good work. Thank you.

**nettigo**

Post author

2015/02/03 at 18:19

Hi Kumaran,

We are glad You found these posts useful :)

**Joseph Chrzempiec**

2015/02/18 at 00:09

Hello i tried your sketch and it works thank you. But the problem I'm having is that the delay receiving it on the serial monitor is very slow it's like instead of 1 second i get like a 5 second delay. i try to take out the delay that seems to go faster but then i get instead of a hello world once i get it like 3 times at once. can someone please help me out? thank you.

**nettigo**

Post author

2015/02/19 at 12:49

@Jospheh

Hard to say what is wrong.

If there are radio interferences nrf makes few tries to send – maybe this is the case? Something is interfering – can You make test in another environment?

Second guess is that after some work we find that Mirf library (<https://github.com/stanleyseow/arduino-nrf24l01>) is much better in terms of stability. We will write some examples in future, but right now You can take a look at code <https://github.com/nettigo/TinnySensorModule> This sketch is for ATtiny84, but inside is folder with code for UNO to receive data.



Luke

2015/03/12 at 03:42

Hi, I tried your code and it works for one or two lines but then it starts outputting gobbeltgook forever.

I'm not sure what's going on here. At first I had the pins you point out as 7 and 8 as 9 and 10 so I changed them to 7 and 8 but it does the same thing.

**nettigo**

Post author

2015/04/02 at 15:22

@Luke

So it works, first two times and later goes bananas? Frankly I don't have idea... You use exactly code as in examples on our blog?



Topcorner

2015/04/14 at 18:39

new to all this – getting ARduino IDE 1.6.3 compile error shown below. I have a feeling something is pointing to the wrong area so it is not picking up all the libraries but cannot figure it out. I've tried changing the in the include statements to “xxx” but no change. Would greatly appreciate any help you can offer.

Arduino: 1.6.3 (Windows 7), Board: “Pro Trinket 3V/12MHz (FTDI)”

Using library SPI in folder:

C:\Users\Top\AppData\Roaming\Arduino15\packages\arduino\hardware\avr\1.6.2\libraries\SPI

Using library RF24-master in folder: C:\Users\Top\Documents\Arduino\libraries\RF24-master (legacy)

C:\Users\Top\AppData\Roaming\Arduino15\packages\arduino\tools\avr-gcc\4.8.1-arduino2\bin/avr-g++

-c -g -Os -w -fno-exceptions -ffunction-sections -fdata-sections -fno-threadsafe-statics -MMD -

mmcu=atmega328p -DF_CPU=12000000L -DARDUINO=10603 -DARDUINO_AVR_PROTRINKET5FTDI -

DARDUINO_ARCH_AVR -

IC:\Users\Top\AppData\Roaming\Arduino15\packages\arduino\hardware\avr\1.6.2\cores\arduino -

IC:\Users\Top\AppData\Roaming\Arduino15\packages\arduino\hardware\avr\1.6.2\variants\eightanalog

inputs -IC:\Users\Top\AppData\Roaming\Arduino15\packages\arduino\hardware\avr\1.6.2\libraries\SPI -

IC:\Users\Top\Documents\Arduino\libraries\RF24-master

C:\Users\Top\AppData\Local\Temp\build8004962604088768678.tmp\RF24_Hello_World_tx.cpp -o

C:\Users\Top\AppData\Local\Temp\build8004962604088768678.tmp\RF24_Hello_World_tx.cpp.o

RF24_Hello_World_tx.ino: In function ‘void setup()’:

RF24_Hello_World_tx.ino:13:31: error: invalid conversion from ‘const byte* {aka const unsigned char*}’
to ‘uint64_t {aka long long unsigned int}’ [-fpermissive]

In file included from RF24_Hello_World_tx.ino:3:0:

C:\Users\Top\Documents\Arduino\libraries\RF24-master\RF24.h:324:8: error: initializing argument 1 of
‘void RF24::openWritingPipe(uint64_t)’ [-fpermissive]

void openWritingPipe(uint64_t address);



nettigo

Post author

2015/04/14 at 19:05

Hi!

Thank You for trying our examples. RF24 was our first choice when we started projects with nRF modules, however now we prefer <https://github.com/aaronds/arduino-nrf24l01> Mirf library. Less features but more stable.

These errors are related to changes in newer IDE and compiler. I think if You fallback to 1.5.x it will work.

We plan to rewrite examples to Mirf library using newer IDE, but I can not give any estimate when it can happen.



Topco

2015/04/14 at 19:37

Thank you for the info. I'll keep plugging away at it.



Ed

2015/04/15 at 00:05

Since NRF24L01 sends max of 32 bytes at a time, how do you go about sending a set of information that has 150 bytes?

What if instead of test you are sending numbers?

I am trying to send a bunch of data from some sensors.

Thanks



nettigo Post author

2015/04/23 at 22:19

@Ed

There is no support for that AFAIK, Your software has to split message into smaller chunks and put them back on the other side...



Maciek

2015/05/02 at 18:58

Nie działa. Ani na Arduino IDE 1.0.5-r2, ani na 1.7.2

```
radio.openReadingPipe(0, rxAddr);
```

sketch_may02c.ino: In function 'void setup()':

sketch_may02c:15: error: invalid conversion from 'const byte*' to 'uint64_t'

sketch_may02c:15: error: initializing argument 2 of 'void RF24::openReadingPipe(uint8_t, uint64_t)'



spikes556

2015/05/14 at 01:41

I was having trouble with the Arduino Micro and the NRF24I01. I found the solution on this web page

<http://minspan.blogspot.com/2014/06/nrf24l01-rf24-radio-on-arduino-micro.html>

Basically, the ICSP are not hooked up or hooked up wrong. You will have to use pin 11 for MISO, pin 9 for SCK, pin 10 for MOSI. The mentioned website explains it further.



Divyesh

2015/05/16 at 02:22

Hello,

Very nice and simple Hello world example. It is pretty useful as a beginner.

No I have a question for you. Have tried to set a network of more than one nrf24l01?

i am working on the project where I am making a network of nrf24l01 transceivers, such that one can broadcast the text message from one transceiver at the base station to all the other transceivers connected to it. After receiving the message, the transceiver should reply back to the base station in same manner.

I need your some guidance in the coding. What should be the code at base station and what could be the code at other node?

Please help..It's urgent!!!

Thank you



afzal rehmna

2015/05/24 at 07:56

can u post the schematics for the both the transmitter and the reciever i am confused

should i use 2 arduino and 2 nrf101 for communications ?????? plz help some



TeDe123

2015/06/11 at 09:38

Hi,

Thanks a lot for this tutorial. It's great and probably the best in whole web.

I've run this exaple on my Arduino Due boards. I've just needed to write `#define RF24_DUE` in the code to make it works. Everything is correct and i am sending and receiving messegas on my boards.

Now I have to use the nrf24l01 transmission with my usb host shield which uses SPI too. Does somebody knows how to initialize SPI in code for two devices – usb host shield and nrf24l01 in this case, (maybe some hardware changes are needed?)

I found the information in datasheet that shield uses pin 10 as a CS PIN, so i've changed it for nrf to pin 4. When i did that and stopped using port 10 for nrf24l01 my program doesn't work anymore. I cannot even write anything on terminal using `Serial.println()` function.

If i use pin 10 for nrf CS again Serial port works, but nrf transmission doesn't work.

Maybe somebody could help me.



AhmarSultan

2015/06/12 at 12:48

I have an UNO and on the other hand a MEGA. The UNO works fine as a transmitter when I connect it with an external power supply where the MEGA is connected as the receiver to the computer. But when i do the vice versa of it (MEGA as transmitter and UNO as receiver) and i connect MEGA to the external power supply, no packets are received and no change is captured. Can you suggest me a solution to my problem?

I will be grateful.

Thanks.



Brian

2015/07/12 at 23:49

These instructions worked great!

I have the ADDICORE nRF24L01+ kit and had this running within an hour. The kit comes with Male-Female connectors so it was easier to connect from the RF board to the Arduino on the digital input pins (11, 12, 13) instead of the ICSP pins.

How do I send numeric data? I need to send a byte, not text.

Thanks in advance to whoever can post some simple Arduino code to answer this question.



Brian

2015/07/13 at 13:40

I figured out one way to send a number instead of a string:

```
//define  
byte msg[1];  
  
void loop()  
msg[0] = any_byte_number;  
radio.write(msg, 1);
```



Dario

2015/07/13 at 14:53

Hi!

Your explanation is complete and straightforward, much more than any other confusing website on the web

Thanks a lot!



Hiloliddin

2015/08/10 at 19:28

Hello

I have error in compiling here:

```
radio.openWritingPipe(rxAddr);
```

error message is here:

Arduino: 1.6.6 Hourly Build 2015/08/07 05:34 (Windows 8.1), Board: "Arduino Uno"

transmitt.ino: In function 'void setup()':

```
transmitt:15: error: invalid conversion from 'const byte* {aka const unsigned char*}' to 'uint64_t {aka long long unsigned int}' [-fpermissive]
```

In file included from transmitr.ino:5:0:

```
C:\Users\A\Documents\Arduino\libraries\RF24\RF24.h:324:8: error: initializing argument 1 of 'void
RF24::openWritingPipe(uint64_t) [-fpermissive]
```

```
void openWritingPipe(uint64_t address);
```

```
^
```

```
invalid conversion from 'const byte* {aka const unsigned char*}' to 'uint64_t {aka long long unsigned
int}' [-fpermissive]
```

This report would have more information with
"Show verbose output during compilation"
enabled in File > Preferences.

can help please

thanks



Shamil Mowlana

2015/09/07 at 09:28

@Hiloliddin In both transmitter and receiver use rxAddr[6] in the reading and writing pipes. Add the square brackets [6].



Plinio Bueno

2015/09/12 at 08:38

Great tutorial. I sufered a lot with this damn radio and now you made it as easy as possible. Thank you very much!



Roberto Lo Giacco

2015/09/21 at 00:19

Hi, I'm creating a series on using the nRF24 to create a sensor network and I would appreciate your feedback.

The series, 2 out of the planned 7 posts out, starts at

<https://rlogiacco.wordpress.com/2015/09/06/nrf24-walk-through-introduction/>



Bibin

2015/09/28 at 04:54

Nice tutorial. Is it possible to interface/connect these nRF24L01 modules to an android app ?



Luka

2015/10/12 at 13:23

Hello,

how can i change a String to char. So instead of :

```
const char text[] = "Hello world";
```

i could send:

```
String hello="Hello World";
```

```
const char text[] = hello;
```

Thank you!



Emin

2015/10/14 at 20:45

hi, i just copy paste your code but i got this message "invalid conversion from 'const byte* {aka const unsigned char*}' to 'uint64_t {aka long long unsigned int}' [-fpermissive]" from arduino compiler. So, I

search for it. After that I wrote "const uint64_t rxAddr = 0xE8E8F0F0E1LL;" instead of "const byte rxAddr[6] = "00001""; . I open serial monitor and I saw repeatedly "yy". I have problem with NRF24 for almost 3 days. I am using arduino Leonardo. Please help me for solving this problem. I am going crazy.



Jose

2015/10/29 at 06:23

Thank you for such a clear explanation on how to use this RF module. Got it up and running within half an hour with two Arduino UNOs.

Do you know if it's possible to have one module set up as a transmitter while you have 2 or more receivers?

Again, thanks for your time in putting this guide together!



aymeric

2015/11/27 at 14:12

Hello,

Thanks for the clear script !

It works fine, but as soon as I plug-on my gps shield, without changing anything in the code.. it stops working. .I suppose this is coming from the CSN/SS pin (7) that I need to put to HIGH, but I have had no success so far.

Can someone help please ?

Thank you !



nettigo

Post author

2015/11/27 at 14:33

@aymeric

What pins are used by gps shield? Is it SPI also? Does it have fixed CSN/SS pin? Remember – on SPI SS pin is used to select which slave device has access to SPI bus. And this is reason why NRF and GPS have to use different CSN/SS pins



K Srinivas Pavan Kumar

2015/12/16 at 08:31

Hi, I have tried your code .

The transmitter side it is working fine but I don't know whether it is transmitting or not, and at receiver side it is showing me this error can anyone can solve this issue

At transmitter side I have used Arduino pro mini

At receiver side I used Arduino Uno

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 1 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 2 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 3 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 4 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 5 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 6 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 7 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 8 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 9 of 10: not in sync: resp=0x35
```

```
avrdude: stk500_rcv(): programmer is not responding
```

```
avrdude: stk500_getsync() attempt 10 of 10: not in sync: resp=0x35
```

Problem uploading to board. See <http://www.arduino.cc/en/Guide/Troubleshooting#upload> for suggestions.

This report would have more information with
"Show verbose output during compilation"
enabled in File > Preferences.



nettigo Post author

2015/12/16 at 14:41

Hello,

This error show some problem with communication between UNO and computer. Is this missing bootloader on UNO, or wrong drivers in computer I can not tell. Can You upload Blink (File/Examples/01. Basics/Blink)? If You disconnect everything from UNO can You program it? Did You connect something to pins 0/1 in UNO?

tegiz



2015/12/17 at 10:56

I have 2 Arduino mega 2560.
2 NRF24L01 + PA + LNA.
I tried almost all the way. Can I sent a text message to another Arduino.
If you have any tips, how to resolve it

Thank

**nettigo** Post author

2015/12/17 at 12:27

To be honest, I have lying few NRF24L01+PA+LNA. I connected them but for first time I was unable to get them running. Had no time to debug them, but I have them on TODO list. If there will be some outcome, I'll post them on this blog, but don't know when.

**nettigo** Post author

2016/02/25 at 11:59

We have recently add PA+LNA versions to our offer: <https://nettigo.eu/products/nrf24l01-module-wireless-communication-for-arduino-2> and from first tests – they are great. Works exactly as drop-in replacement for 'regular' NRF, have better range. How much is 'better'? Can not say exactly for now. We will do some tests in open area and will see how range have improved. But from indoor test – when tested across two concrete ceilings we get excellent results. With regular NRF modems we got 50%-70% packets lost. Replacing them with PA+LNA modems get ideal transmission w/o any loses.

**Mike**

2016/01/18 at 05:57

Thank you for the information it was great help for me. I am very new to arduino and got several nrf24l01 to play with and needed someplace to start. I used a Nano and Uno with your example and it worked good. I had to change the pin numbers some for the Nano because I used a shield that connected to different Dout numbers but it worked first try.



heng

2016/02/13 at 15:17

HI
i have tried every tutorial for nrf24l01 but none work for me so I tried this which connects differently. I cannot upload because of error on this:
`const byte rxAddr[6] = "00001";`
it is no longer accepted on 1,5.5 or 1,6.4 IDE versions
can you please update?
thanks



Summit Sayem

2016/03/07 at 16:26

Thanks a lot! Works perfect
