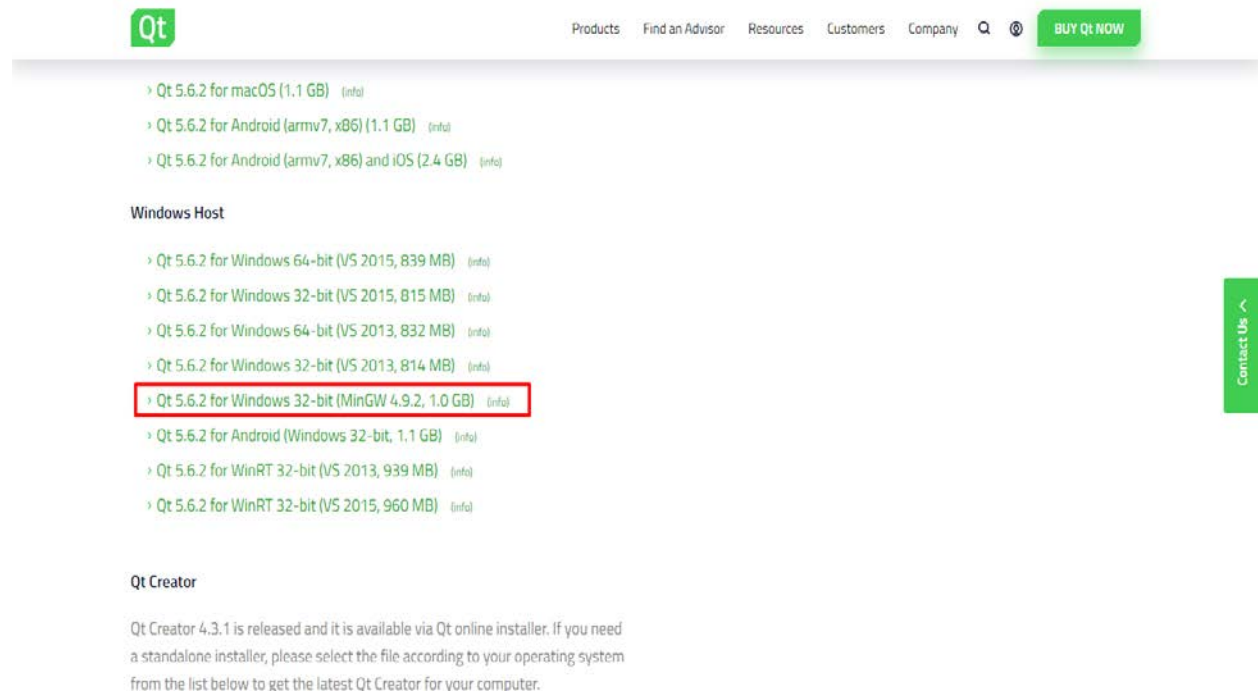


Important info on the project

INSTALLING QT WINDOWS

the gui made with qt creator.

To download link for windows with its compiler: [here](#)



link: [here](#)

INSTALLING QT LINUX

Open terminal and type the followings:

```
sudo apt-get install qt5-default  
sudo apt-get install qtcreator
```

The qt creator should now be installed, however, there is no compiler known to qt creator yet.

We should give qt creator the path of our g++.

To get path of g++ type the following in terminal

```
Whereis g++
```

The return will be the path, just copy it and past it in qt compiler path(window>>option>>build and run>>compiler then add MinGW and past the path in compiler path and hit apply. Then go to Kits tab and in compiler choose MinGW that we just created then hit enter. You're good to go.

HOW TO CONNECT PL2303TA (WINDOWS)

to able to read the serial data from Arduino for debugging purposos we need some sort of USB to TTL device to be able to read the data probarly from Arduino.

so we used pl2303ta device, other alternatives will do the same function of couse with small changes which will probably be the module's driver.

As for PL2303TA:



This is its picture with the pinout.

First of all, we need to download its driver otherwise the windows will not recognize it.

To download the driver you can directly download it from [here](#) or you can go to the module's [main page](#) and search for a newer version or older if you like.

After downloading the driver you should install it, the installation process is very straint forward so we'll leave it to you (you might need to plug the module in for installation to complete, we're not sure).

After finishing installation the driver will request to restart the computer so go ahead and restart it.

The devise should now appear in you devise manager under Ports(COM & LPT) when you plug it in.

Next we need a consul program to be able to read the serial data from Arduino so we choose [RealTerm](#) software.

You can download it directly from [here](#) or from [their site](#).
You might choose a different software if you like.

After downloading and installing your preferred Serial Terminal you should connect the device to Arduino.

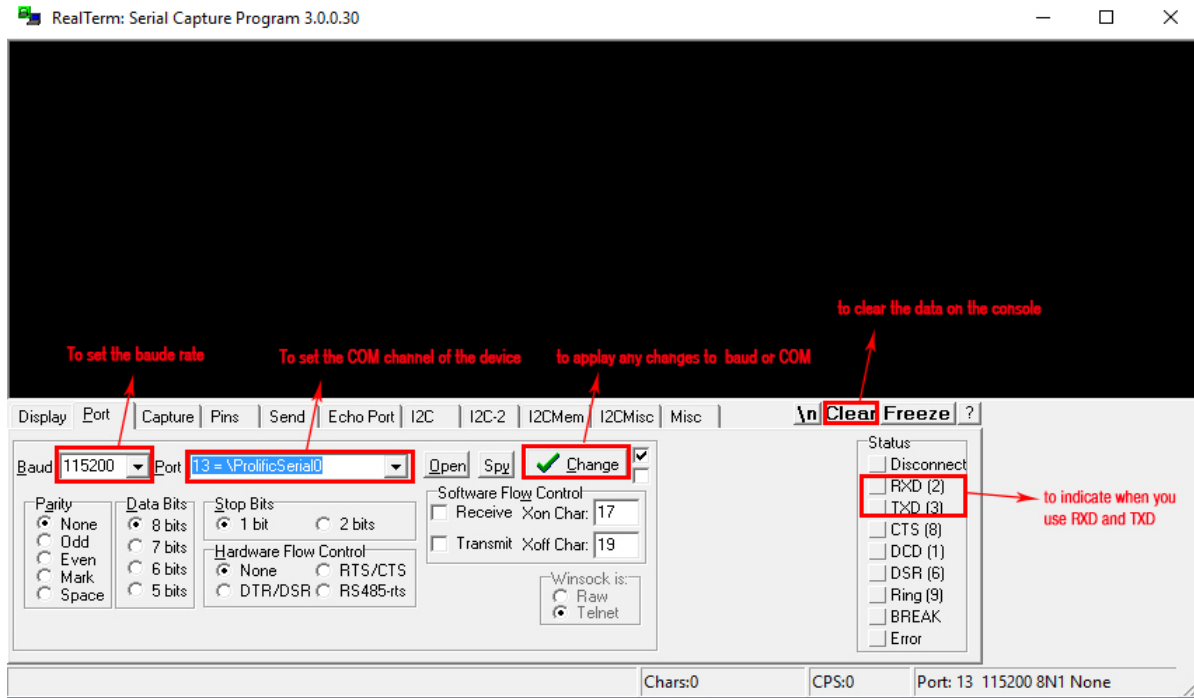
CASE 1: If you use the module to power Arduino (no external power source to Arduino board)

- Connect **+5V** of the module to **5V** or **VIN** on Arduino.
- Connect **GND** of the module to **GND** on Arduino.
- Connect **TXD** of the module to RX on Arduino.
- Connect **RXD** of the module to TX on Arduino.

CASE 2: if you use an external power source for Arduino EG. Battery)

- DO NOT** connect +5v to 5v or you might **damage** your computer or Arduino.
- Connect **GND** of the module to **GND** on Arduino.
- Connect **TXD** of the module to RX on Arduino.
- Connect **RXD** of the module to TX on Arduino.

Now open RealTerm and set your COM port and Baud rate:



This how the interface looks like.

Set the baud rate and the channel according to Arduino and you windows device manager.

HOW TO INSTALL NRF24 LIBRARY (LINUX)

Open a terminal and type the followings:

```
wget http://tmrh20.github.io/RF24Installer/RPi/install.sh
chmod +x install.sh
./install.sh
```

This will download and install everything - answer "y" to the questions about installing RF24 and RF24Network. You don't need RF24Mesh or RF24Ethernet for now, but it doesn't hurt to install them too. You might see some warnings while it's compiling the libraries but you can ignore those.

After the library is downloaded and installed you need to make some small changes to it.

In directory /home/pi/rf24libs/RF24 (assuming you didn't choose a different directory) there will be a file named "Rf24.cpp". You need to open it and insert the following include:

```
wget http://tmrh20.github.io/RF24Installer/RPi/install.sh
chmod +x install.sh
./install.sh
```

Then you need to insert the following code to make sure that the library will not send you into an infinite loop after a Hardware Fail.

You need to search for “[RF24 HARDWARE FAIL:](#)” in the .cpp file, after you find this line then you can insert the following line right after it and before [#endif](#)

```
system("cd /pi/code/rpi/code && sudo ./remover");
exit(1);
```

It’s important to note that this addition is made specifically to satisfy our needs in this project.

What this addition does is that when a hardware failure happens the library will show a single message to tell you that an error in hardware happened and then exit the entire program.

We added this because there is a problem with the SPI driver in the Rpi kernel and this is a work around it to recover from it after the SPI driver causes an error. The error is a software error that happens out of the blue with no known reason why so it will happen one time only and will reset after the program ends, this is why we forced the program to exit (to reset).

So it’s important to make sure you connected the module correctly before making the changes.

As you can see that the program calls another program (to do a specific function) before ending the program. And the “cd /pi/code/rpi/code” is to change the directory to the folder where you keep all the [code](#).