**Air Quality Sensors & Nordic Radio**

The aim of this exercise is to become familiar with the Arduino development platform, accelerometer sensors and low-power radio communications. This will be achieved through the development of a basic, networked earthquake sensor device. The components you will be using will be as follows:

| Macintosh HD:Users:slock:Desktop:nano.jpg | Image result for mq 7 s | Macintosh HD:Users:slock:Desktop:nordic.jpg |
| --- | --- | --- |
| Arduino Nano | Typical Air Quality Sensor | Nordic Radio |

**Task 1: Blink !**

The first program that anyone writes on an Arduino is usually “Blink” – it is the “Hello World” of Arduino development ! First open up the Arduino application on your laptop and then from the top-bar menu select: **File> Examples> 01.Basics> Blink**

You should now see the Blink code in your editor window.

Plug your Arduino into your laptop with a USB cable, then make sure that “**Tools> Board> Arduino Nano**” is selected, as well as “**Tools> Processor> ATMega328P(Old Bootloader)**” and that “**Tools> Port**” is the serial port that your Arduino is connected to.

If you are unsure what this means, just ask !

Next we need to upload the code onto the Arduino. This is done by clicking the “upload” button Near the bottom of the Arduino window you will see a variety of progress messages including: “Compiling sketch”, “Uploading”, “Uploading Done” or if you are really unlucky, a red error message!If everything worked OK, you should see one of the lights on the Arduino flashing on and off every second. Congratulations, you are running your first Arduino program !!

**Task 2: Sensor Hookup**

Now that you are familiar with some of the features of the Arduino development environment, we’ll do something a little bit more interesting. We are going to hook up the Arduino to a sensor and see what it’s reading.

The first thing to do is to UNPLUG YOUR ARDUINO FROM THE USB. This is just in case you slip with a wire and accidentally short the chip (unlikely, but possible).

Using jumper cables connect your sensor up to the arduino. You may have to look up the pin out for your sensor but typically its connecting 3 pins:

**Sensor to Arduino:**

**VCC -> 5V**

**AOUT -> A0**

**GND -> GND**

Once the Accelerometer board is connected, you can plug the USB cable back in.

**Task 3: Sensor Code**

We now need to read in values from the sensor. Update your Blink program with the following features (might be wise to save it to your desktop first, so that you don’t lose any code):

In the “setup” function, switch the data pin (A0) to “input” mode, This is done using the “pinMode” function: *pinMode(A0, INPUT);*

Also in the setup function, add the following line of code: *Serial.begin(9600);* This initialises serial communication so that we can send data down the USB cable, back to the computer.

Next, in the “loop” function read in the current value of the sensor. This is done using the “analogRead” function, for example: *int x = analogRead(A0);*

After you read in the value, use the “Serial.println” function to send the sensor data down the USB cable to your computer: *Serial.println(x);*

Upload your updated program onto your Arduino in the same way as before.

Once uploaded, you should be able to see the data streaming in by open up the “Serial Monitor”:

 You should get a number within the range of 0 and 1023.

**Task 6: Nordic radio Wiring**

Instead of sending your magnitude value down the serial cable to the Serial Monitor, we are now going to send it via radio to the grapher application (plugged into the projection computer). A template Arduino program has been provided to help you get started on this activity, use it as the basis for this stage of your work.

You will also need to install a radio comms library on your computer (if it is not already installed !), this can be found at:

<http://people.cs.bris.ac.uk/slock/mirf.zip>

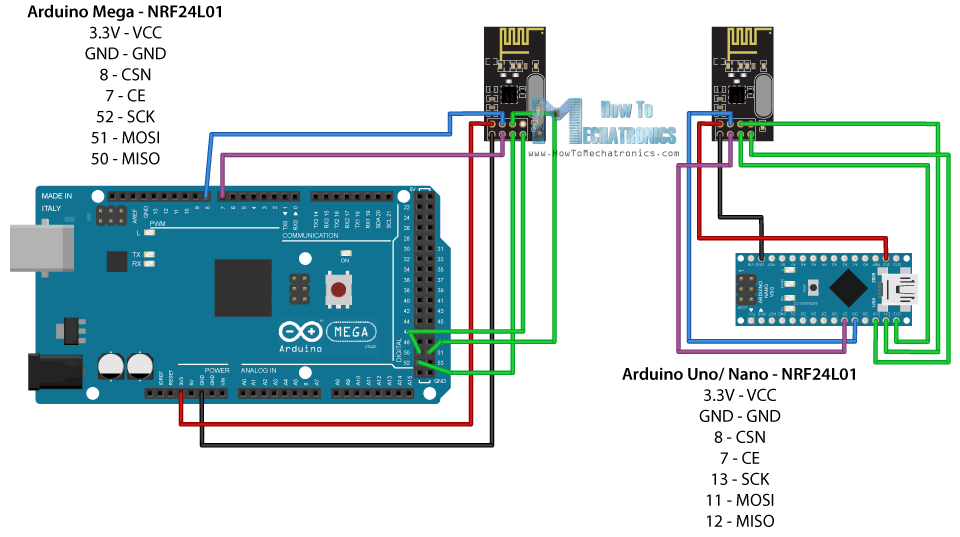
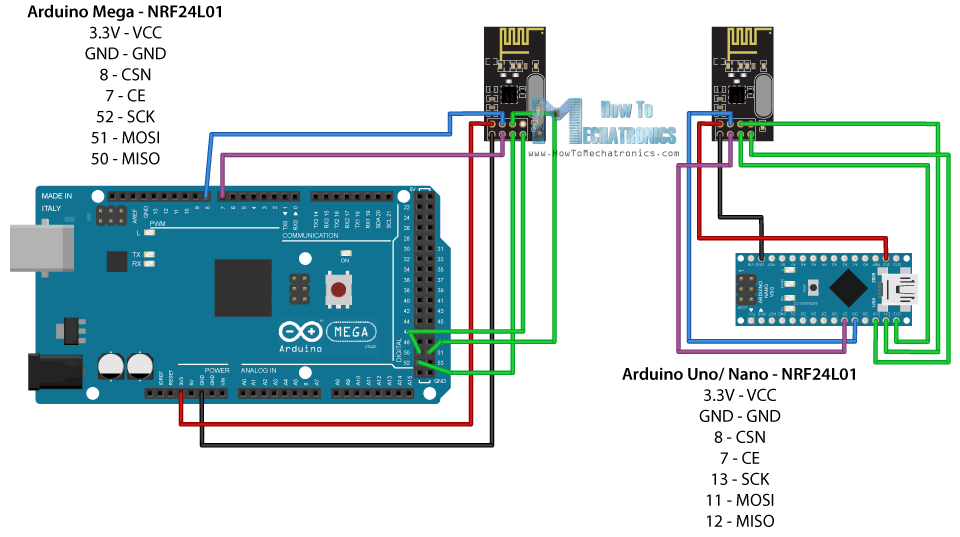
Unzip it and copy the whole folder into: My Documents/Arduino/Libraries

You will need a Nordic radio and a specially constructed 8-wire connection cable.

First, wire up the Nordic radio to the Arduino following the below wiring colour scheme:

|  |  |
| --- | --- |

MAKE SURE THAT YOU USE THE 3.3V POWER PIN – AND NOT THE 5V PIN !!!!!!!!!!!!!!!!!!!!!!!!!!!!



**Task 6: Nordic radio Code**

Start with the following code:

*#include <SPI.h>*

*#include <Mirf.h>*

*#include <nRF24L01.h>*

*#include <MirfHardwareSpiDriver.h>*

*float stationaryX, stationaryY, stationaryZ;*

*byte id[5] = "SIMON";*

*int payloadSize = 9;*

*void setup()*

*{*

*Mirf.spi = &MirfHardwareSpi;*

*Mirf.init();*

*Mirf.setRADDR(id);*

*Mirf.setTADDR("GRAPH");*

*Mirf.payload = payloadSize;*

*Mirf.channel = 90;*

*Mirf.config();*

*pinMode(A0, INPUT);*

*Serial.begin(9600);*

*}*

*void loop() {*

*byte message[payloadSize];*

*int x = analogRead(A0);*

*Serial.println(x);*

*message[0] = ;*

*message[1] = ;*

*message[2] = ;*

*message[3] = ;*

*message[4] = ;*

*message[5] = ;*

*message[6] = ;*

*message[7] = ;*

*message[8] = ;*

*Mirf.send(message);*

*while (Mirf.isSending()) delay(1);*

*}*

*}*

In order for the grapher to successfully receive your message, you need to obey the following rules:

* Your message must always contain a total of 9 bytes
* The first byte must be a special “message begins” byte 0x00
* The next 5 bytes must be the characters of a unique name for your device (e.g. “STEVE”)
* The next 2 bytes must be your magnitude (integers are stored in 2 bytes on these Arduinos)
* The final byte must be a special “message ends” byte 0xFF