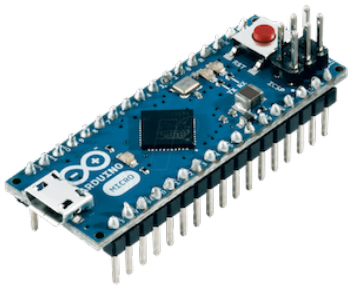
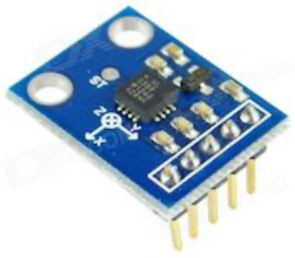
Earthquake Monitor

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:

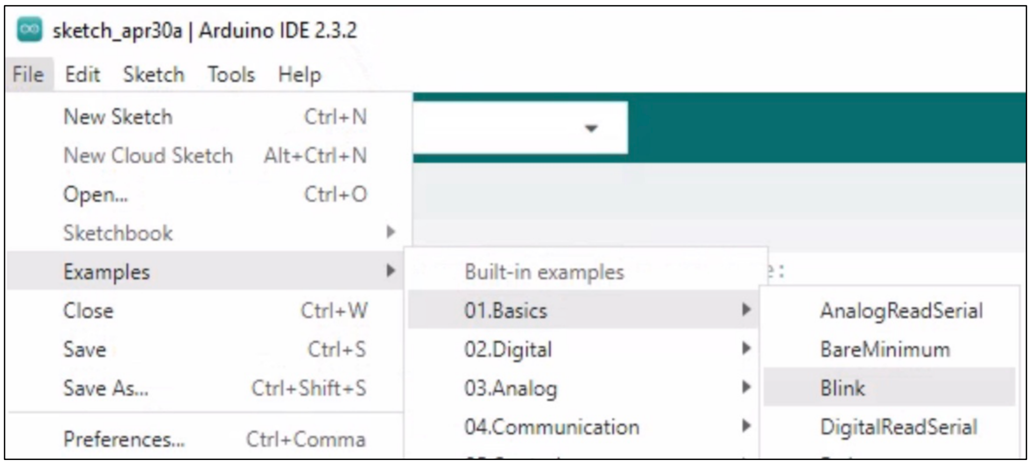

Arduino


GY-61 Accelerometer

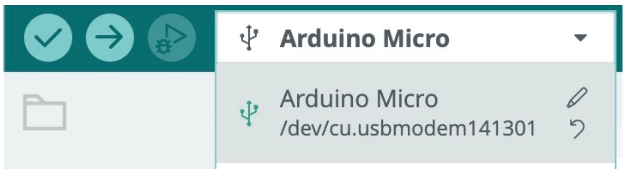
Nordic Radio

Task 1: Blink !

The first program that anyone writes on an Arduino is usually "Blink" – it is the "Hello World" of Arduino development ! First run the Arduino application and then open the "Blink" example:



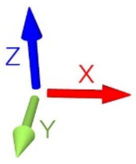
You should now see the Blink code in your editor window.
Plug your Arduino into your computer using the USB cable, then select "Arduino Micro" from the list of available devices:



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: Accelerometer 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 an accelerometer board that can be used to sense movement and vibration in three different dimensions: X Y Z



The first thing to do is to UNPLUG THE ARDUINO FROM THE USB. This is to disconnect it from the power supply, just in case you slip with a wire and accidentally short the circuit board (unlikely, but possible). Once the USB cable is unplugged, connect the accelerometer board with the ribbon cable provided with the following connections:

Accelerometer	VCC	X_OUT	Y_OUT	Z_OUT	GND
Arduino Micro	5v	A3	A4	A5	GND

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

Task 3: Accelerometer Code

We now need to read in values from the Accelerometer sensor. Update your Blink program with the following features (it might be wise to save it to your "Documents" folder first, so that you don't lose any code). In the "setup" function, set the data pins (A3, A4 and A5) to "input" mode, This is done using the "pinMode" function, for example: `pinMode(A3,INPUT);` Do this for all 3 pins !

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 each of the 3 accelerometer dimensions. This is done using the "analogRead" function, for example: `int x=analogRead(A3);` Make sure you do it for all 3 input pins (using different variable names to store the readings !)

After each value you read in, use the "Serial.println" function to send X, Y and Z acceleration data down the USB cable to your computer. For example: `Serial.println(x);` If you want to put a blank line between any values, just use `Serial.println(“ “);`

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 see a stream of data scrolling past in the serial monitor window – if you tap the accelerometer board with your finger, you should see the numbers change !

Task 4: Gravity Baseline

In addition to sensing vibration, the accelerometer will also detect the pull of gravity. Depending on the orientation of your sensor board, this might be in X, Y, Z, or a combination of different directions! In your setup function, read the initial values from X, Y and Z and remember them as a "baseline" (you can create three integer variables at the top of your program to store this information in). You will need to subtract these initial values from your later readings (made in the loop function) in order to factor out the effects of gravity. Your aim is to adjust the raw input readings so that when the accelerometer is at rest (i.e. not being shaken) your adjusted reading are close to zero.

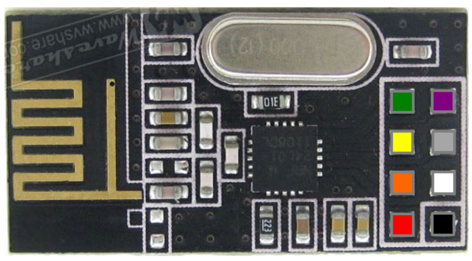
Task 5: Magnitude calculation
Sending 3 separate values down the serial cable is a bit unnecessary – we only really need to send a single value (the total size of the vibration). Perform a calculation on the Arduino to resolve the three separate readings into a single absolute "magnitude" value (hint: you might need to use Pythagoras to do this). Finally, change your code so that you only send the single value down the serial line. Check in serial monitor to see if everything is working as you might expect.

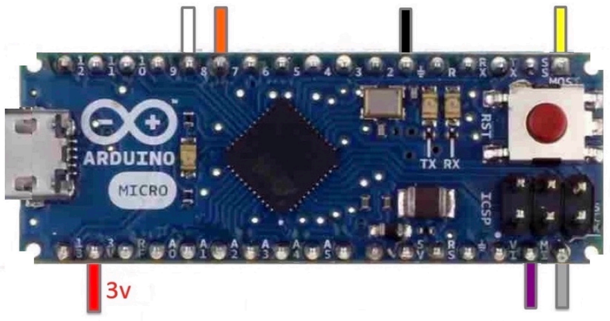
Task 6: Nordic radio
Instead of sending your magnitude value down the serial cable to the Serial Monitor, we are now going to send it via radio to the grapher application (that is plugged into the projection computer). A template Arduino program has been provided to help you get started on this activity, use it as the basis for your work. To get the template to work, you will need to install the MIRF radio communication library on your computer. Both the template and MIRF can be found here:

http://people.cs.bris.ac.uk/slock/quake

Download and extract the content of this zip file, then move the `EarthquakeSensorTemplate` folder into your "Documents" folder and the `Mirf` folder into "Documents/Arduino/Libraries". Note that you might have to restart the Arduino application in order for it to find the Mirf library.

To connect a Nordic radio to your Arduino, you will need a specially constructed 8-wire connection cable. Wire up the Nordic radio to your Arduino following the below wiring colour scheme:





Connect green wire to any *empty* row on the breadboard

Task 7: Sending sensor data via radio
Merge the code from your original "Blink" file into the `EarthquakeSensorTemplate` file. Add your previously written code to the "loop" function, so that it reads data from your accelerometer, factors out the effects of gravity, calculates the magnitude and finally inserts the calculated value into the message packet, ready for sending via the Nordic radio to the "GRAPH" device. In order for the grapher to successfully receive your message, you need to obey the following rules:

- Your message must be an array of 9 bytes
- The starting byte must be a special "message begins" byte 0x00
- The next 5 bytes (bytes 1-5) must be the characters of the unique name for your device
- 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

In order to split the magnitude value into two separate bytes, you will need to use the highByte() and lowByte() functions. The grapher expects to see the high byte first, followed by the low byte in. Make sure that you change the name of the device to a unique 5 character string - this is so we know where the reading has come from when it is displayed on the projector screen.