

11/29/2014



OÉ Gaillimh
NUI Galway

Microprocessor Systems Engineering

Digital Musical Instrument

Eoin Clancy
13388656

Edward Hanlon
12440482

Cathal Dooley
13533663

NiallRutherford
12409098

Contents

Introduction	2
User Guide	3
Design Overview	4
Detailed Design	5
Testing.....	9
Discussion and Recommendations	14
Conclusions	17
Appendices	18
Appendix 1 Individual Contributions	18
A1.1 Eoin Clancy 13388656.....	18
A1.2 Cathal Dooley 135336633.....	18
A1.3 Edward Hanlon 12440482	18
A1.4 Niall Rutherford 12409098	18
Appendix 2 Parts, materials & Costs.....	19
Appendix 3 Code Used	20
Appendix 4 Risk Assessment.....	34

Introduction

The objective of this Microprocessor Systems project was to create a digitally controlled musical instrument using an Intel Galileo microcontroller. This had to be completed on the Arduino programming platform and then demonstrated to the lecturer in front of the class. The project that our group decided to work on was a type of electronic laser "guitar" that can teach the user how to play a specific song. We settled on this idea after each coming up with a different concept for the project. The laser guitar was chosen because we felt that it created an interesting learning environment for musical beginners as well as an environment where competent musicians could create new tunes. We felt that we were proposing a new take on the traditional instrument, in that we would be able to teach the user how to play a tune on the device by using an LCD prompt.

The idea of teaching someone how to play a song had never been done before to the best of our knowledge so we decided that if we could have 2 modes on the instrument, 1 to play normally and 1 to teach, we would have an innovative new idea. It would involve having 6 lasers acting as strings would on a guitar making beams with 6 different light dependent resistors. Each individual LDR would be associated with a different note which the Arduino would play out to the attached speaker. The LCD screen would either shown which note is currently being played or which one should be played depending on the mode it was set to.

From various tests that we undertook with the Galileo microcontroller we found that the upload speed was very slow and struggled to run the code successfully. It was able to run programs that only involved one function such as writing to the LCD or taking in values from an LDR. However after two weeks of testing when we began putting together multiple functions in the code we became aware that the Galileo was very slow at uploading and returned errors to the Arduino software. Following many attempted bug fixes we decided to test the code using an Arduino Uno which successfully ran the code. We then chose to use the Arduino completely for the project as it ran all code we had previously written with no errors.

User Guide

Once the code is uploaded onto the Arduino Uno the user will be given a selection between two modes. The switch to the right of the LCD screen can be used to select between Guitar mode and Teaching mode.

In the Guitar mode the user can play whatever they want. Notes are displayed on the base of the 'guitar strings' to let the user know which note is which. The user then just breaks the desired laser with their finger to play the note. The LCD screen will also display the note they are playing.

If teaching mode is selected then the system will teach them a preprogramed song which in this case is Seven Nations Army. The LCD screen will display the note that needs to be played. Once the user has broken the correct laser the LCD will display the next note. Once the song has been played the song loops around to the start again so the user can go again.

Ideally they will improve through multiple attempts until they can play the song well and then move onto Guitar mode to play different songs by themselves. The user can switch between the two modes freely, there is no need to reset the system if they wish to change modes.

Design Overview

In order to successfully complete this project we have to set out exactly what our goals will be and how they can be achieved. We met at the end of week one it was decided that the laser guitar would be made. There were several different tasks that had to be undertaken in completing the project. They were set out as following:

- a) Connecting the lasers & LDRs
- b) Powering the speaker with an amplifier
- c) Playing notes with by breaking a laser
- d) Display different modes and notes on the LCD to the user.
- e) Finding out how to switch between different modes

We went about these tasks systematically so that we knew once each part was tested it would only take small alterations to the code to integrate them altogether.

a) Firstly we connected up the LDR's to the microcontroller using a breadboard to test the levels of light. We tested these to see the figures being read into the serial monitor with the laser fully on the LDR and just ambient light hitting it. When these values were found we set a tolerance and map each of the lasers so we knew which one of them was being triggered.

b) When we had the tolerances set for the LDR's we moved on to the second part of the design, connecting the Arduino to a speaker through an op-amp. We sourced a 64Ω impedance speaker and an LM386n-1 op-amp which were stocked in the college already and began work on setting them up. Using sample code for a speaker and amplifier connected to Arduino we found on the arduino.cc website we set up a basic circuit and tested to see if we could get any sound from the speaker. Once it was configured correctly and making a noise we experimented with the "tone()" function to play different sounds from the speaker. We then found the notes for 7 Nations Army and played it through the speaker using simple time delays to define the length of the notes.

c) The next stage of the project that we moved on to involved integrating the first two sections so by breaking a laser beam a sound was played out through the speaker. This initially started with just one laser connected up the Arduino to make it as simple as possible. Once we had successfully gotten a sound to play with one laser we added one at a time until we had a different sound for all 6 lasers.

d) Writing out messages to the LCD was the next part of the project that we tackled. We first wrote a simple "Hello World" string to the screen using an example from the Arduino website. As outlined in the testing we worked out how to get a scrolling message across the screen. Once we had this working we added to this the note that had to be played in the second line on screen.

e) We knew that we wanted to have two different modes of playing on the laser guitar, one for playing normally and one for teaching. We decided that if we could implement a switch with the Arduino we would be able to swap between each mode easily. We had a two pin flip switch and decided we would use it to choose which mode to enter upon starting the program. We connected the switch up through a

resistor to the Arduino and wrote a simple code to write one message to the LCD if the switch was in the high position or nothing if it was in low. Following this we merged the guitar function and the teaching function by using the switch.

Detailed Design

The code for the Laser Guitar was built in a modular fashion in parallel with the testing of each individual component of the system. This allowed us to create small functions of code which fulfilled the tasks they were intended for. By building our code this way, we were able to debug each small function which avoided wasting time in the future searching through numerous lines of code for errors. As we built each function, it was implemented into a larger body of code which was easily readable and reliable as each component of the program had been tested thoroughly.

To have a clear layout of what we wanted to achieve with the laser guitar, we drew up a number of flow charts, described throughout this section. The first task we faced was designing the start-up phase of the instrument. We eventually decided on the user being able to choose between two distinct modes: 'Guitar mode' and 'Song mode'. The former would allow the user to play any guitar note they wished for as long as they wanted, like a normal guitar, except there would be no fret board. The song mode allowed a person, learning how to play the instrument, to upload a pre-written song to the system and to play along with prompts which appeared on the LCD. The selection would be based on a user's decision to flick a switch upwards for the guitar mode or downwards for the song mode. See flowchart, figure 1.

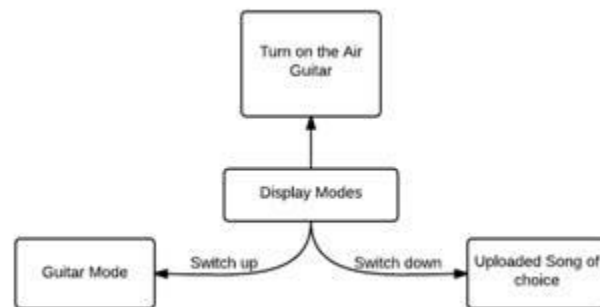


Figure 1: Outline of operation of the laser guitar

The user would also be able to determine the direction they wished to select by following the ordering of the modes on the LCD display. A sufficient amount of time was provided to the user to select their desired function, but if this time expired the laser guitar would automatically enter the mode which was corresponding to the switch direction at that time, see figure 2.

```

// include the library code:
#include <LiquidCrystal.h>
#include "pitches.h";

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int buttonState = 0;

void setup() {
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
  lcd.begin(16, 2);
  lcd.setCursor(0, 1);
  lcd.print("Guitar Mode");
  // Print a message to the LCD.
  lcd.setCursor(0, 0);
  lcd.print("Teaching Mode");
  delay(7000);
  lcd.clear();
}

```

Figure 2: Code for the setup of the instrument

The first reading of the switch occurs at the very beginning of the infinite loop which allows for the switch to have finished debouncing. This was a small issue we ran into during testing and implementing the reading of the switch state there solved our issue. As the above figure also shows, we utilised certain Arduino libraries too. The "LiquidCrystal.h" library provided us with the functionality for our LCD screen and 'pitches.h' provided us with necessary tones.

Guitar Mode

The first mode that we developed through our testing was the guitar mode. We had a basic layout of what we expected from our code and our flow chart was key in designing the overall model. We wanted each LDR to constantly be reading in values, and when the light level dropped below a certain level, which we would hard code into the program, the corresponding note of that LDR would be played out through the speaker. As figure 3 shows, we also had to implement the case where all LDRs had light shining on them, meaning no sound should be played.

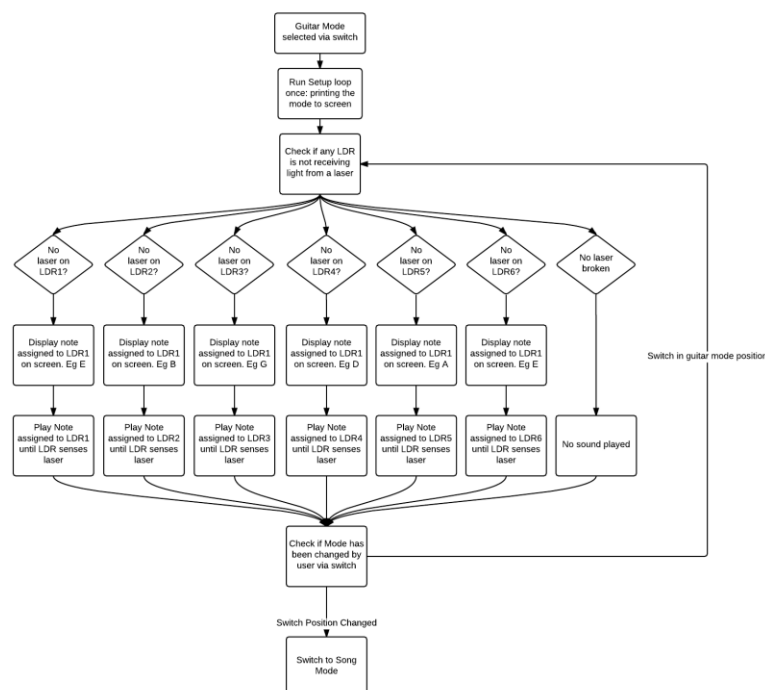


Figure 3: Outline of the operation of Guitar Mode.

Once we had built the individual functions for reading in values via the LDRs and had attached our LCD and speaker to the circuit, we created a single program which ran the guitar mode. We created actual sound through the use of the 'pitch.h' library provided by Arduino. This gave us a nice range of notes to test with and allowed us to choose notes which suited our instrument. Each LDR had a block of code similar to figure 4.

```
//laser shining = >120
//laser off = 95

if (LDR1Value<110) // light not shining on ldr, play note
{
  LDR1Value= analogRead(ldr1); //read ldr value
  LDR1Value = map(LDR1Value,0,1023,0,170); //map ldr value to a certain range
  tone(13, NOTE_E3); //The note assigned to the LDR
  lcd.setCursor(0, 0); //Setting up so note can be printed to screen
  lcd.print("E3");
  delay(1); // play for 1ms to allow for quick change between notes
}
```

Figure 4: Code for how a single note was played

The above code indicates how we checked if the laser was or was not shining on the LDR. If the laser was not shining, the corresponding note for that LDR was printed to the LCD and the note was play for 1ms. This allowed for rapid transitions between notes while playing the sound continuously if the laser remained blocked from the LDR.

We were also conscious that the user could change mode at any time, so we allowed for this to happen by simply reading the switch state before and also after the notes were played. See appendix 3 for entire code.

Song Mode

We then began developing the 'Song Mode'. We designed a flow chart to outline how our code would run once the program had been completed, see figure 5.

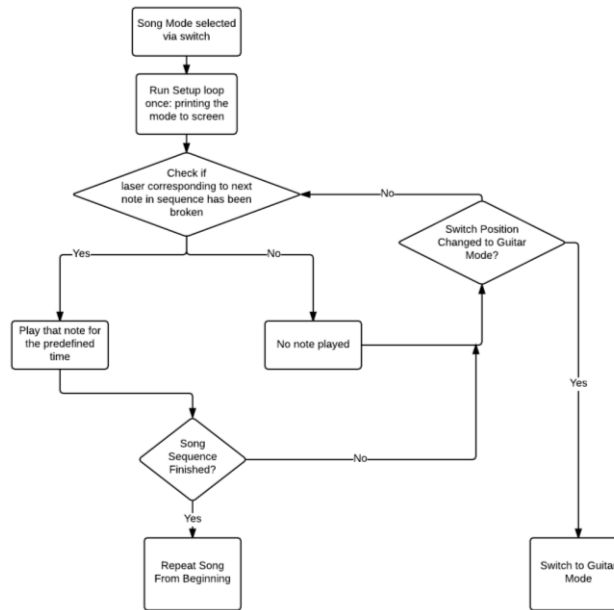


Figure 5: Outline of the operation of Song Mode.

Again, as this figure shows, we allowed the user to change the instruments mode at any point in the program by simply just reading in the state of the switch to see if it had changed to that of guitar mode.

The operation of this mode was very similar to that of the guitar mode except that the program would only play a note if it was the next note in the uploaded song, see code below in figure 6. For this to work, we used a single variable which would be incremented if the correct note was played. From there, the instrument would only play sound if the next correct note in the sequence was played. The user would be told the next note via the LCD. We also decided that it would be better for learners to have the note length coded into the program. This would allow them to initially become familiar with the song and the location of each of the notes. If they were feeling confident, they could try playing the song on the guitar mode where they would have to rely on their own timing to recreate the song.

```

if(x==0) //loop for first note
    //keeps looping here until the innerloop is executed
{
    lcd.setCursor(0, 0); //printing to screen
    lcd.print("E2");

    if (LDR1Value<110) // light not shining on LDR, play note
    {
        LDR1Value= analogRead(LDR1); //read LDR value
        LDR1Value = map(LDR1Value,0,1023,0,170);
        tone(13, NOTE_E2); //play note assigned to the LDR
        delay(600); //plays for this duration
        noTone(13); //ensures the sound is silenced once note is played

        x++; //increments the variable which only allows the
            //next note in the song sequence to be played
    }
}

```

Figure 6: Layout of how a single note is triggered in Song Mode

Testing

1. Testing the LCD Screen

1.1 Objective

The objective of this test was to connect up the LCD screen to the Galileo and make sure data was written to the screen properly.

1.2 Methods

We first connected the LCD by following a diagram provided on the Arduino website, www.arduino.cc/en/Tutorial/LiquidCrystal.

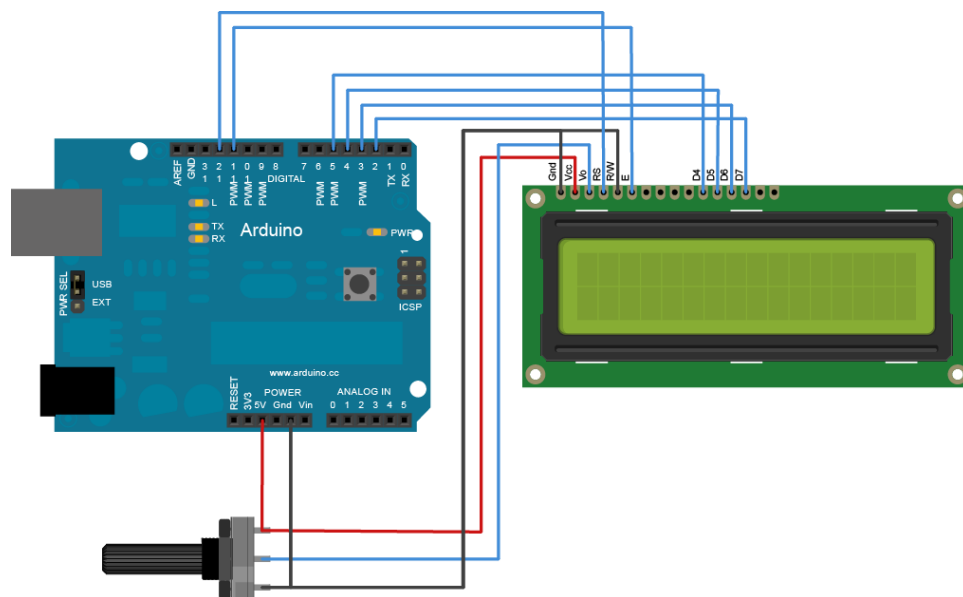


Figure 7: Circuit diagram indicating the connections between the LCD and Arduino.

Once we had connected the LCD up to our Galileo, we used one of the Arduino sketch examples to print “Hello World” out to the screen. From there we edited the code to make the text on the screen user-friendly. To achieve this we utilised scrolling messages going across the screen to display the future modes that the laser guitar would be able to enter into. We also tried printing a changing input onto the bottom line of the screen because we would need to be able to do this to display the current note that was being played on the laser guitar.

1.3 Analysis

From testing the LCD screen we were able to analyze visually how well certain effects worked on the screen and determine suitable delays how for a user to be able to read text on screen. The visual representation of code also led to some optimizations in our code so that we could save memory space on the microcontroller.

1.4 Results

Following this test we were able to say conclusively, that we knew how the screen worked and what way it would need to be connected up to the Arduino when we were putting all of the project together. As well as this we only had to edit the code that we wrote for the scrolling text and changing digits for adding it into the final code which we used.

2. Testing the LDR's

2.1 Objective

The objective of this test was to analyse the LDR values being read in by the microcontroller and map them to specific ranges of values to suit the flowcharts that we had designed.

2.2 Methods

1. We first connected the LDR's to a breadboard.
2. Then a power supply of 5v was provided to one end of each LDR.
3. Each LDR was connected to a different analog pin on the microcontroller, so that the values read in by each LDR could be distinguished.
4. LDR values were initially printed to the serial monitor on the Arduino interface.
5. Each LDR read a value between 0-1023. This was expected.
6. Each LDR was assigned a range of size 170, so that they could be distinguished from one another. E.g. (171-341 for LDR2), see code in figure 4 for an example.
7. LDR values were tested and recorded with and without the laser shining on them.
8. These values were key to the implementation of the code as the microcontroller had to know when a light was broken. This was achieved by testing for the light value to be less than that of when the laser should be shining.
9. A laser was then shone on each of the LDRs. When one laser was broken, we used the Arduino Serial monitor as a method of visually representing this by printing out, for example "LDR1 Light Broken", when the LDR connected to pin A0 recognised that no laser was shining on it. This represented the basic function of the laser guitar.

2.3 Analysis

From testing the LDRs we found that they were extremely accurate to very small variations in light levels even when the values were mapped to smaller ranges.

2.4 Results

Following this test, we had developed the basic function of the musical instrument which was recognising when lasers were and were not shining. With this information we could then begin writing the code to play a musical note when a specific laser was triggered.

3. Testing the speaker

3.1 Objective

The objective of this test was to connect up the speaker to the microcontroller and make sure it was making the correct sounds for the correct readings from the LDRs (i.e. light on an LDR or no light on an LDR).

3.2 Methods

Firstly we had the speaker connected directly into the 5v and pin 13. However, we soon switched to running the speaker through an amplifier as this would produce a better sound and would also be better for the board. The amplifier we used was an LM386n-1. We found an example of how to set up the amplifier on (<http://www.instructables.com/id/LM386-Audio-Amplifier/>). We had the speaker set up with the amplifier we started mapping the LDRs to the speaker.

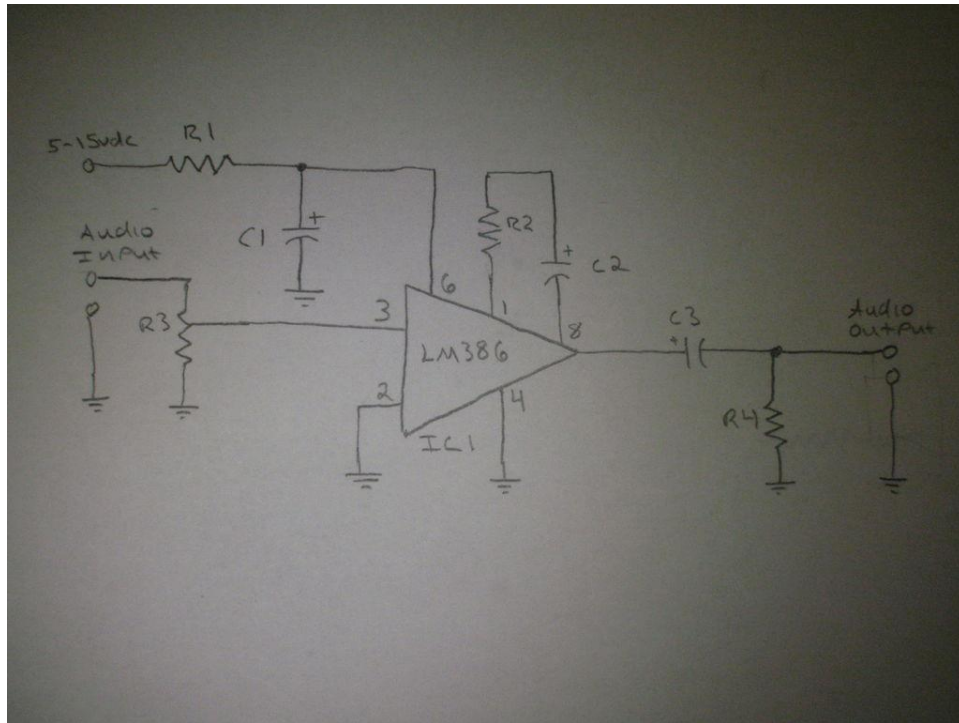


Figure 8: Layout of circuitry for the amplifier.

We had the LDRs mapped at this point so we just put a sound outputs running to pin 13 when the light was broken. Each LDR had its own note registered to it so when each is laser broken it would put out its own distinct sound.

3.3 Analysis

Once we had each break in the laser producing its own sound it was just a case of tweaking it to the sound we wanted. Much like tuning a real guitar it consisted of changing the output just a small bit and then checking the sound to really refine the sound we wanted. Once we had each sound sounding right we played a short song on it to ensure the notes were right.

3.4 Results

Once we had completed the analysis of the sound we were happy with the way it was set up and the notes it was producing. We had a good knowledge of the circuit so we knew we would have no problem moving it around if needed when we were placing into the laser guitar casing we had built.

4. Testing the switch

4.1 Objective

The objective of this was to install a switch which would allow the user to switch between two modes, Teaching mode and Guitar mode.

4.2 Methods

Firstly we installed the switch into the circuit. We connected one end of the switch into a 5v power source. The other end was set in line with a resistor and a connecting wire on the breadboard. The resistor ran to ground and the connecting wire ran to pin 7.

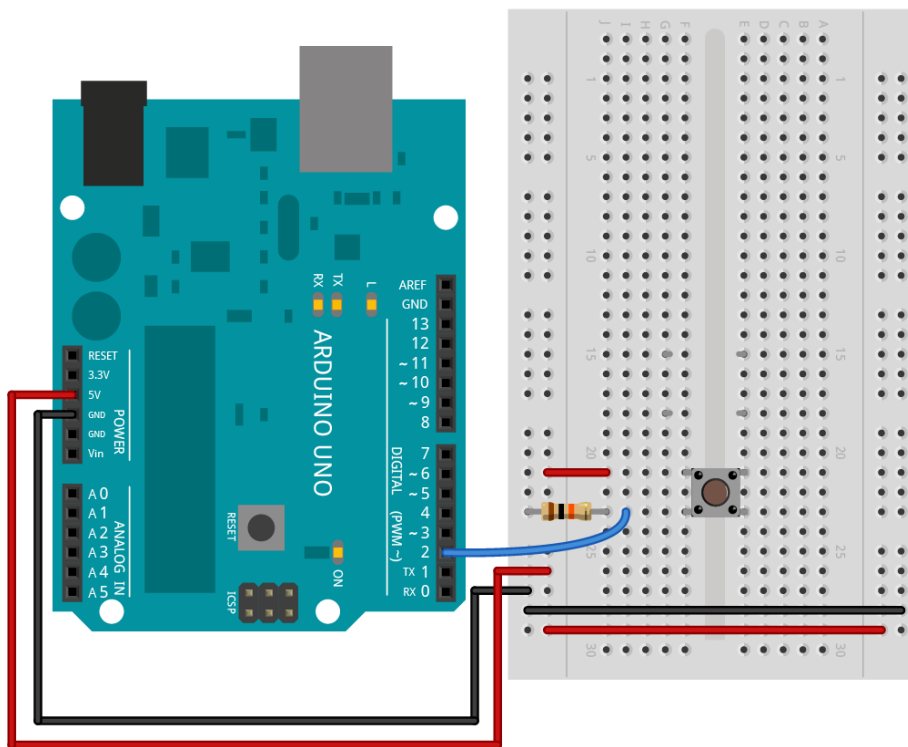


Figure 9: Connection of a switch to the Arduino board.

Each sets of code had already been wrote so it was just a case of setting the program to run into the correct one depending on the way the switch was set. Each set of code would be called in whether pin 7 was high or low. A menu was also printed onto the LCD screen to let the user know which way the switch should be for each mode.

4.3 Analysis

Once we had the switch set up we turned on the Arduino and tested switching between the two modes. We decided to put a longer delay on the menu at the start to make it more user friendly.

4.4 Results

We found this was a very good addition to our project. The extra mode on the guitar gave the user more freedom to play what they wanted rather than just playing one tune. Therefore this will mean the user will get much more from this device.

Discussion and Recommendations

Discussion

This project proved very beneficial in terms of working as a group to produce a desired end product. The task to build a digitally based musical instrument seemed quite challenging at first but as we worked forward, it became very interesting and was a great learning experience. It gave all involved a greater knowledge of circuits, wiring, coding, and indeed, music. Creating a new and unique musical instrument required innovation and thought and once the ideas began to flow we quickly decided on the 'laser guitar'. Brainstorming, sharing and commenting on different ideas with the group is a valuable technique which we will certainly use in similar situations later on in life. From an early stage, we began meeting frequently to discuss plans and to work on the instrument. This certainly worked in our favour in terms of staying on top of the work and ensured we usually met most of our deadlines. We were rarely under time pressure or in any state of panic. This project highlighted the importance of good time-keeping.

The best way to learn about something is through testing which was imperative in this project. This task allowed us to attain an in-depth knowledge of circuits and how they relate and integrate with the microcontroller and a computer program such as Arduino. We achieved this knowledge through trial and error, testing and observations. We learned about the relationships between the circuit elements such as the resistor, capacitor and power and also how to wire the system to work correctly. At first, problems arose with the circuits which were proving to be tricky but with persistence we worked it out and benefited from continuous testing and modification.

Having had some previous experience with the Arduino computer program meant we had familiarity with the language and how to implement it into our project. After searching the internet we managed to get a good basis of code which was a roundabout version of what we were trying to achieve. We implemented this code into our own project; however, we were constantly coming up with new ideas for the project throughout its course so the code had to be regularly updated. Figuring out how to take these new ideas and convert it into language that your microcontroller will understand and in turn carry out is great practice in process of writing code. Once again, this part of the project consisted of trials and testing and became a great learning experience.

Working in a group of people proves very constructive. It teaches you to work in conjunction with your colleagues and how to gel as a unit to accomplish the overall goal. It emphasized the importance of taking responsibility of your own individual role and contribution to the overall task. It thought us how to adapt to the timetable of the rest of the group to create a work timetable that suits everyone. Group

work is also essential in developing good communication skills in a working environment, for example, voicing your opinion and finding common ground when it comes to making new decisions. Working together in these sort of circumstances certainly provided us with valuable techniques which we will surely adapt into future work.

Another positive aspect that we derived from undertaking this project was an increased awareness of music. To get our instrument to play a song we had to research different songs and notes and how the notation is accessed by the Arduino via a musical library. We learned a lot about scales, octaves and the notes on each guitar string which was very interesting.

Clearly, this project enhanced our knowledge of electronics and microprocessor systems. It was a very useful learning curve and we benefited highly throughout the course of the task.

Recommendations

We would rate this project as an overall success and upon completion we found we worked efficiently during the task. In terms of recommendations for future work, there are a few steps we would take and a few things we would change to achieve optimum standards of work and results.

Time-keeping is essential in maintaining steady progress and to guarantee all deadlines are met. Therefore creating a timetable to suit the group and fit the needs of the task is pivotal in producing the best work, and is something we would recommend for future work.

It is always a good decision to plot a list of equipment needed to complete the task. This way you can have everything you need from the beginning and won't run into any stoppages during the course of the project due to lack of equipment, which may slow down the whole momentum of the group. Fortunately, we gathered ourselves very early on and made sure we had access to all the essentials we needed. This meant we could begin work straight away without any lost time.

One regret we have, was that we decided very late to switch from the Intel Galileo microcontroller to the Arduino Uno. We ran into some execution problems with the Galileo which prevented us from uploading our code for quite a while. Due to this problem we missed out on many days of work which was very frustrating. Fortunately, we switched to the Uno and everything was back in order, the code uploaded and executed perfectly. Also the Uno was smaller and more portable than the Galileo and didn't require an extra cable for the power source. This problem caused us to change our timetable and

could have been resolved a lot quicker had we used the Uno following our initial problems with the Intel Galileo. From this, we learned that for future work, if a piece of equipment is not functioning correctly and causing a loss of valuable working time in the lead up to a deadline, it is wise to find alternative equipment rather than wasting time on the dysfunctional piece.

Another important recommendation is to take health and safety precautions when carrying these sort of projects. This is very important because anything can happen if you are not careful. We had to be cautious in the making of our project as we were using lasers which have to be used very carefully since they were quite powerful. Therefore, health and safety must always be considered when undertaking this kind of work.

Conclusions

After a lot of trial and error we finally made an electronic instrument that we were all happy with. When it was finished we played it, all the lasers were registering properly with the correct notes playing and the LCD screen was printing the right messages for each mode.

We feel we succeed with our aim of making the 'guitar' as user friendly as possible. The project covered both the teaching aspect while also giving freedom to more advanced musicians. From testing the device ourselves we find it is ideal for beginners as it can walk the user through a song slowly and can be quicker as the user improves. The guitar mode is also ideal for advance musicians as the user can play any set of notes themselves. This means the device can play a lot of songs if the user knows them and can even be used to create new music

The choice to switch to the Arduino Uno from the Intel Galileo worked out well as we found the project began to move along quicker as we had fewer problems with errors. Also the more compact size made it easier to fit it into the casing we had made for it.

Appendices

Appendix 1 Individual Contributions

A1.1 Eoin Clancy 13388656

I was primarily involved in the testing of the components and the development of the code for our laser guitar. At the beginning of the project I did a lot of work on trying to get the instrument to play real guitar sounds, which would have provided better quality sound than the digital notes. However, the processing of the serial code to a MIDI input into a music program proved to be too difficult. I took the project home at weekends to get the speaker and amplifier set up and also helped with mapping the LDR's for the code. I wrote the majority of the code used which proved to be a success for the testing of the laser guitar and I was satisfied that I managed to get the two modes incorporated into one seamless program which also provided a nice interface to the user.

A1.2 Cathal Dooley 135336633

I helped out with collecting the necessary pieces for our instrument and setting up the breadboard circuits. I helped in the testing of the LDR's to map them and also to align them in our finished design. I have a background in music, so I was able to advise the other members on the best notes to use in the two modes of the laser guitar.

A1.3 Edward Hanlon 12440482

Once we were given the project I did research of different types of ideas for the project and presented my ideas when we met. Once we had the idea set and started I help with setting up the circuits we were using and with the testing. I sourced a suitable LCD for the project. I also did the soldering for the project. I soldered wires onto a speaker we had been using and I also soldered the pins we needed for the LCD. I then set up the circuit for the LCD. I wrote code for the LCD and so did one of the other members. However once we decided to include a switch the other member's code was more suited. While the code was being tidied up I set up the circuit of the switch for testing. I also built the casing for the 'guitar' and fitted the device into it.

A1.4 Niall Rutherford 12409098

At the start of the project I came up with an idea of playing music by moving your hand over piano like keys but not actually touching them. When the laser guitar was eventually chosen I wrote code to take readings from the LDR's and play a note using a "for" loop. I also wrote the code when we were testing the LCD screen to get a scrolling message and changing numbers on it. However Eoin decided to use his own code for both of these sections. In the report I contributed to the introduction, design overview, testing, commenting on the code and writing the risk assessment in the appendices.

Appendix 2

Parts, materials & Costs

No.	Part	Source	Cost
1	Intel Galileo	Supplied by college	N/A
1	Speaker	Supplied by college	N/A
1	Connecting wires	Supplied by college	N/A
1	LM368n-1 Amplifier	Supplied by college	N/A
1	Switch	Supplied by college	N/A
	Resistors	Supplied by college	N/A
	Capacitors	Supplied by college	N/A
1	Arduino Uno		
1	LCD screen	Maplin	€29.99
6	Lasers	Maplin	€6.50 each
6	LDRs	Maplin	€2.79 each

Appendix 3

Code Used

```

int ldr1=0;
int ldr2=1;
int ldr3=2;           //initialising all the pins that the LDR's are connected to
int ldr4=3;
int ldr5=4;
int ldr6=5;

int LDR1Value=0;
int LDR2Value=0;
int LDR3Value=0;
int LDR4Value=0;      //setting each of the readings for the LDR's initially to 0
int LDR5Value=0;
int LDR6Value=0;
int loop1counter =0;
int loop2counter = 0;
const int buttonPin = 7; //initialising the pin that the switch is connected to
int x = 0;             //initialising an integer to count through the loops in 7nation army

#include "pitches.h";
#include <LiquidCrystal.h> // include the library code:
#include "pitches.h";

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // initialize the library with the numbers of the interface pins
int buttonState = 0;

void setup()

```

```

{
    pinMode(buttonPin, INPUT);  //initially write to the screen the user's options
    lcd.begin(16, 2);           //of choosing to learn or just free play
    lcd.setCursor(0, 1);
    lcd.print("Guitar Mode");
    // Print a message to the LCD.
    lcd.setCursor(0, 0);
    lcd.print("Teaching Mode");
    delay(7000);
    lcd.clear();
}

void loop()
{
    buttonState = digitalRead(buttonPin);    //check the state of the switch
    if(buttonState == HIGH)                  //if it is in the high state write to screen that it is
    {                                         //in guitar mode
        buttonState = digitalRead(buttonPin);
        if(loop1counter == 0){
            // set up the LCD's number of columns and rows:
            lcd.clear();
            delay(500);
            noTone(13);
            delay(1);
            lcd.begin(16, 2);
            lcd.setCursor(0, 1);
            lcd.print("Now Playing:");
        }
    }
}

```

```

    // Print a message to the LCD.
    lcd.setCursor(0, 0);
    lcd.print("GUITAR MODE");
    delay(2000);           //delay the message on screen for 2 seconds
    lcd.clear();           //clear the message on screen before the first note comes
    loop1counter++;        //exit this loop so it stops showing guitar mode
}
loop2counter = 0;

LDR1Value= analogRead(ldr1);           //setting each pin to respectively
LDR1Value = map(LDR1Value,0,1023,0,170); //named LDR1Value
                                         //each value is then mapped so

LDR2Value= analogRead(ldr2);           //when reading on the serial
LDR2Value = map(LDR2Value,0,1023,171,341); //monitor bug finding.
                                         //also mapped so light change

LDR3Value= analogRead(ldr3);           //can trigger a noise
LDR3Value = map(LDR3Value,0,1023,342,512);

LDR4Value= analogRead(ldr4);
LDR4Value = map(LDR4Value,0,1023,513,683);

LDR5Value= analogRead(ldr5);
LDR5Value = map(LDR5Value,0,1023,684,854);

LDR6Value= analogRead(ldr6);
LDR6Value = map(LDR6Value,0,1023,855,1025);

lcd.setCursor(0, 1);           //print out to the screen
lcd.print("Now Playing:");

```

```

if (LDR1Value<110) // light not shining on ldr, play note
{
    LDR1Value= analogRead(ldr1);           //play a note related to ldr1 if the
    LDR1Value = map(LDR1Value,0,1023,0,170); //first laser beam is broken
    tone(13, NOTE_E3);
    lcd.setCursor(0, 0);
    lcd.print("E3");                       //print out to the screen the note
    delay(1);                             //being played
}

```

```
//laser off 262
```

```
//laser on 300
```

```

else if (LDR2Value<290) // light not shining on ldr, note played
{
    LDR2Value= analogRead(ldr2);           //same as above but for each note
    LDR2Value = map(LDR2Value,0,1023,171,341);
    tone(13, NOTE_B2);
    lcd.setCursor(0, 0);
    lcd.print("B2");
    delay(1);
}

```

```
//laser off 380
```

```
//laser on 420
```

```

else if (LDR3Value<410) // light not shining on ldr, note played
{
    LDR3Value= analogRead(ldr3);

```



```

        LDR3Value = map(LDR3Value,0,1023,342,512);
        tone(13, NOTE_G2);
        lcd.setCursor(0, 0);
        lcd.print("G2");
        delay(1);
    }

//laser off 550
//laser on 595
    else if (LDR4Value<585) // light not shining on ldr, note played
    {
        LDR4Value= analogRead(ldr4);
        LDR4Value = map(LDR4Value,0,1023,513,683);
        tone(13, NOTE_D2);
        lcd.setCursor(0, 0);
        lcd.print("D2");
        delay(1);
    }

//laser off 755
//laser on 790
    else if (LDR5Value<790) // light not shining on ldr, note played
    {
        LDR5Value= analogRead(ldr5);
        LDR5Value = map(LDR5Value,0,1023,684,854);
        tone(13, NOTE_A1);
        lcd.setCursor(0, 0);
        lcd.print("A1");
        delay(1);
    }

```

```

    }

//laser off 892
//laser on 925
    else if (LDR6Value<920) // light not shining on ldr, note played
    {
        LDR6Value= analogRead(ldr6);
        LDR6Value = map(LDR6Value,0,1023,855,1024);
        tone(13, NOTE_E1);
        delay(1);
    }

    else if (LDR1Value > 110 && LDR2Value >290 && LDR3Value > 410 && LDR4Value > 585 &&
LDR5Value > 790 ) && LDR6Value > 920)
    {
        LDR2Value= analogRead(ldr2);           //make sure no notes
        LDR2Value = map(LDR2Value,0,1023,171,341); //are played when all the
        LDR1Value= analogRead(ldr1);           //laser beams are solid
        LDR1Value = map(LDR1Value,0,1023,0,170);
        Serial.println("in else st");
        noTone(13);
        lcd.setCursor(0, 0);
        lcd.print("00");
        delay(1);
    }

    buttonState = digitalRead(buttonPin);
}

//*****second part of the program, to play 7 nations army***** //

```

```

else{
buttonState = digitalRead(buttonPin);
    if(loop2counter == 0)
    {
        // set up the LCD's number of columns and rows:    //set up as before show on the screen that
        Serial.begin(9600);                                //the mode is now teaching mode and song
        lcd.clear();                                       //playing is 7 nations army
        delay(500);
        noTone(13);
        delay(1);
        lcd.begin(16, 2);
        lcd.setCursor(0, 1);
        lcd.print(" Now Playing:");
        // Print a message to the LCD.                    //scroll the message across the screen so it
        lcd.setCursor(0, 0);                             //can be all seen
        lcd.print("Seven Nations Army");
        delay(1000);
        for (int positionCounter = 0; positionCounter < 3; positionCounter++)
        {
            // scroll one position left:
            lcd.scrollDisplayLeft();
            // wait a bit:
            delay(400);
        }
        delay(700);
        for (int positionCounter = 0; positionCounter < 3; positionCounter++)
        {
            // scroll one position right:
            lcd.scrollDisplayRight();
            // wait a bit:

```

```

        delay(400);

    }

    delay(1000);

    lcd.clear();

    loop2counter++;           //exit the loop and don't return unless the
    }                         //loop is broken again

    loop1counter = 0;

// set the cursor to column 0, line 1
// (note: line 1 is the second row, since counting begins with 0):

    lcd.setCursor(0, 1);
    lcd.print("Play note: ");

LDR1Value= analogRead(ldr1);           //set all the ldr values again
LDR1Value = map(LDR1Value,0,1023,0,170); //remap each LDRvalue to useable
                                         //values

LDR2Value= analogRead(ldr2);
LDR2Value = map(LDR2Value,0,1023,171,341);

LDR3Value= analogRead(ldr3);
LDR3Value = map(LDR3Value,0,1023,342,512);

LDR4Value= analogRead(ldr4);
LDR4Value = map(LDR4Value,0,1023,513,683);

LDR5Value= analogRead(ldr5);

```

```
LDR5Value = map(LDR5Value,0,1023,684,854);
```

```
LDR6Value= analogRead(ldr6);
```

```
LDR6Value = map(LDR6Value,0,1023,855,1025);
```

```
if(x==0)                                //set a new loop to go through each note in the song
{
    //x increments after each not is played

    lcd.setCursor(0, 0);
    lcd.print("E2");                    //the note that should be played is printed to the screen
    if (LDR1Value<110)                  // light not shining on ldr, play note
    {
        //the program only advances if specific laser is broken

        LDR1Value= analogRead(ldr1);
        LDR1Value = map(LDR1Value,0,1023,0,170);
        tone(13, NOTE_E2);              //if the user goes for the wrong string
        delay(600);                     //nothing is played
        noTone(13);
        x++;
    }
}

}
```

```
else if(x==1)                          //the loop is incremented onto the next note
```

```
{
    lcd.setCursor(0, 0);
    lcd.print("E2 again");
    if (LDR1Value<110) // light not shining on ldr, play note
    {
        LDR1Value= analogRead(ldr1);
```

```

        LDR1Value = map(LDR1Value,0,1023,0,170);
        tone(13, NOTE_E2);
        delay(400);
        noTone(13);
        x++;
    }

}

else if(x==2)
{
    lcd.clear();
    lcd.setCursor(0, 1);
    lcd.print("Play note: ");
    lcd.setCursor(0, 0);
    lcd.print("G2");
    if (LDR2Value<290) // light not shining on ldr, note played
    {
        LDR2Value= analogRead(ldr2);
        LDR2Value = map(LDR2Value,0,1023,171,341);
        tone(13, NOTE_G2);
        delay(300);
        noTone(13);
        x++;
    }
}

else if(x==3)
{

```

```

    lcd.setCursor(0, 0);
    lcd.print("E2");
    if (LDR1Value<110) // light not shining on ldr, play note
    {
        LDR1Value= analogRead(ldr1);
        LDR1Value = map(LDR1Value,0,1023,0,170);
        tone(13, NOTE_E2);
        delay(300);
        noTone(13);
        x++;
    }
}

else if(x==4)
{
    lcd.setCursor(0, 0);
    lcd.print("D2");
    if (LDR3Value<410) // light not shining on ldr, note played
    {
        LDR3Value= analogRead(ldr3);
        LDR3Value = map(LDR3Value,0,1023,342,512);
        tone(13, NOTE_D2);
        delay(300);
        noTone(13);
        x++;
    }
}

```

```
else if(x==5)
{
    lcd.setCursor(0, 0);
    lcd.print("C2");
    if (LDR4Value<585) // light not shining on ldr, note played
    {
        LDR4Value= analogRead(ldr4);
        LDR4Value = map(LDR4Value,0,1023,513,683);
        tone(13, NOTE_C2);
        delay(800);
        noTone(13);
        x++;
    }
}

else if(x==6)
{
    lcd.setCursor(0, 0);
    lcd.print("B1");
    if (LDR5Value<790) // light not shining on ldr, note played
    {
        LDR5Value= analogRead(ldr5);
        LDR5Value = map(LDR5Value,0,1023,684,854);
        tone(13, NOTE_B1);
        delay(400);
        noTone(13);
        x++;
    }
}
```



```
}
```

```
//allows this note to be played even though it shouldn't be played
```

```
else if(x<1000)
```

```
{
```

```
    if (LDR6Value<920) // light not shining on ldr, note played
```

```
    {
```

```
        LDR6Value= analogRead(ldr6);
```

```
        LDR6Value = map(LDR6Value,0,1023,855,1024);
```

```
        tone(13, NOTE_E1);
```

```
        delay(1);
```

```
    }
```

```
}
```

```
    else if (LDR1Value > 110 && LDR2Value >290 && LDR3Value > 410 && LDR4Value > 585 &&
LDR5Value > 790) && LDR6Value > 920)
```

```
{
```

```
    LDR1Value= analogRead(ldr1);
```

```
    LDR1Value = map(LDR1Value,0,1023,0,170);
```

```
    LDR2Value= analogRead(ldr2);
```

```
    LDR2Value = map(LDR2Value,0,1023,171,341);
```

```
    LDR3Value= analogRead(ldr3);
```

```
    LDR3Value = map(LDR3Value,0,1023,342,512);
```

```

    LDR4Value= analogRead(ldr4);
    LDR4Value = map(LDR4Value,0,1023,513,683);

    LDR5Value= analogRead(ldr5);
    LDR5Value = map(LDR5Value,0,1023,684,854);

    LDR6Value= analogRead(ldr6);
    LDR6Value = map(LDR6Value,0,1023,855,1025);
    Serial.println("in else st");
    noTone(13);           //no tone is played if all of the lasers are hitting their targets
    delay(1);
}

    Serial.println(x);           //once x is incremented as far as 7 the loop is reset to
//the start
    if (x==7)                   //so you can play the song again
    {
        x=0; //resets the riff
    }
    buttonState = digitalRead(buttonPin); //checking the state of the switch in case it
                                           //has changed

}
}

```



College of Engineering and Informatics, Risk Assessment.

To Be Completed for Individual Projects e.g. Research, FYP's etc.

List <u>ALL</u> Lab Location(s): EN 3001,3002,3003	Project Title: Microprocessor Systems Engineering PBL trigger music project.	Duration (Dates) :- 17 th October 2014 - 28 th November 2014
Person(s) at risk: Student/Researchers Names:- Eoin Clancy, Cathal Dooley, Edward Hanlon, Niall Rutherford, Martin Glavin, Barbara Oliveira		

Hazard and hazard outcome. (state the specific harm that can occur)	Likelihood (1 to 4) (With Controls in place)	Severity (1 to 3) (With Controls in place)	Controls required (State Specific control, type of PPE, etc.) See over for additional controls.
Lasers used to play music. If lasers move they could go in someone's eyes.	1	2	Additional equipment:The lasers will be secured in a box pointing at light dependent resistors. They will be unable to move from the position they are fixed in.
PCB handling can cause scratches to hands from PTH component legs	4	1	Use Gloves and carefully lift the Galileo board.

Likelihood:

1= Very Unlikely/Yearly

2= Unlikely/During a Semester

Severity 1 2 3

{ 34 }

Comments:

Place an "x" in the matrix for each hazard

Overall Risk assessment is the highest

Likelihood			
		X	
	X		

Severity:

1= Slight Harm

2= Moderate Harm

Hierarchy of Controls: (Refer to Lab Safety Statement for basic controls)

Details of Engineering Controls Required: (e.g.)

- Additional equipment or apparatus available and approved.
- Additional Safety Signs in the work area.
- Raw Materials defined and approved.
- Any additional Noise, Vibration hazards.
- Any additional Occupational Health hazards defined and controls in place.

Details of Administrative/Procedural Controls required: (Add/Delete as required)

- Standard Operating Procedures in place for lab experiments.
- Provide induction training for all labs.
- Biological Hazard Protocols approved.
- Chemical Risk Assessments done.
- Relevant MSDS available.
- Are "Out of Hours" activities required and approved.
- Any specific Emergency Procedures are in place and approved.
- "Fieldwork" approved – Is specific Risk assessment approved.

Any Specific additional PPE required (other than that listed below):

(DELETE/ADD as appropriate)



Hearing Protection, Eye Protection, Dust Mask, Protective Footwear, Face Protection, Lab Coat, Gloves

Personal Protective Equipment (*PPE*) required; (General Requirements)



General Rules:

- No Loose Hair, No Loose Clothes, No Dangling Jewellery
- Adequate Footwear required (No open toes)
- Wash your hands after the laboratory session.
- No Eating or Drinking.

DO NOT ENTER THE LAB OR OPERATE MACHINERY

Prepared By: Niall Rutherford

Approved by (Research/Project Supervisor(s): _____

Signed by Researcher/Student(s) - _____

(To confirm that they have read and understand the contents of this document)