

Robot Orchestra Interim Report Group 11

Group Members:	Buruiana, Andrei	9478411
	Chanda, Joyanto	9015629
	Dimou, Theodoros	9126435
	Fumagalli, Francesco	9017237
	Petrovs, Antons	9474345
	Simpson, Joshua	8929879

Tutors: Prof. Danielle George and Dr. William McGenn

Date: 26 November 2017

Table of Contents

1. Executive Summary	3
2. Project vision	4
2.1. Introduction	4
2.1.1. Required Skills	4
2.1.2. Project Roles	5
2.2. Aims and Objectives	5
2.2.1. Aims	5
2.2.2. Objectives	6
2.3. Motivation	6
3. Literature Review	7
3.1 Keyboard	7
3.2 Xylophone	7
3.3 String instruments	8
3.4 Wind instruments	8
3.5 Stepper motors	10
3.6 Tesla Coils	11
3.7 Embedded Systems	11
3.7.1 Conductor	12
3.7.2 Instrument Microcontrollers	12
3.7.3 Wireless Link	13
3.8 MIDI	13
3.9 Music Background	15
3.10 Anvil Studio Overview	16
4. Technical Progress	16
4.1 Conductor Design	16
4.1.1 High Level Overview	16
4.1.2 MIDI2Text	17
4.2 Solenoids	18
4.3 Keyboard	19
4.4 Xylophone	21
4.5 Panpipes	23
4.6 Tesla Coils	25
4.7 Stepper motors	25
4.7.1 Hardware	25
4.7.2 Software	26
4.8 Guitar	27
5. Progress and Planning	28
5.1 Approach towards 2 nd Semester	30
5.2 Risk Management	31
5.3 Minutes and agendas	32
5.4 Logistics	32
5.5 Procurement	33
5.6 Conclusion	33
6. References	34

1. Executive Summary

This report presents the progress made by MEng Team 11 on the “New core infrastructure for the Robot Orchestra” project. The aim of the project is to expand The University of Manchester’s existing Robot Orchestra by adding at least four new robot instruments as well as an electronic conductor to control the instruments. This would increase the orchestra’s capabilities and allow for a wider selection of musical styles to be played while also being able to be used as a standalone band. Additionally, the new instruments need to be able to play at least two different recognisable songs and allow for future additions of instruments as well as for easy transportation and set up.

The individual objectives of the project are: choosing the songs to be played, selecting four instruments, designing and manufacturing them and finally, assembling them into a core orchestra that can play at least two songs together. The project is motivated by the rising interest in the existing Robot Orchestra which, if expanded, will be able to showcase the operation of multiple embedded systems to perform a shared task that combines engineering and art. Additionally, this expansion can also lead to the Robot Orchestra appealing to a wider audience and advertising engineering to people who would not normally consider it.

To date, a decision on the songs to be played has been made, *Eye of Tiger* by Survivor and *Californication* by Red Hot Chili Peppers having been chosen. Additionally, the instruments selected for the new core infrastructure are: keyboard, xylophone, panpipes, stepper motors and Tesla coils. Mechanical models of the robots for the keyboard, xylophone and panpipes have been designed and for playing the keyboard and xylophone, solenoids that press the keys have been chosen as the preferred method. After having tested various models, a decision was made on which solenoid model to use and for the panpipes, the final design has been narrowed down to two final versions and a part used by both designs has been manufactured. For the stepper motors, decisions on the driver board, microcontroller and stepper motor model have been made and currently, the motors are able to play simple songs individually. The Tesla coil instrument has been investigated and an off-the-shelf kit has been ordered and constructed, which was able to play a song when a source was plugged into the audio jack. In terms of the conductor, Raspberry Pi has been chosen as the embedded platform and software code that is able to extract the key information from the industry-standard file format used for music encoding (MIDI) and convert it to a format that can be used by the instruments designed in this project has been produced.

The report concludes that according to the progress made to date, the keyboard and the stepper motors instruments are expected to be completed by the end of the third project block with the xylophone being expected to be completed shortly after this. The conductor is expected to be completed in a similar timeframe, depending on the selected type of communications between it and the instruments. Lastly, the Tesla coil and robot panpipes are both considered as viable options for the fourth instrument, with a final decision due to be made after the testing stage.


2. Project vision

2.1. Introduction

The University of Manchester Robot Orchestra was originally created to celebrate Manchester becoming the European City of Science for 2016. Since its inception, the project has gained backing from Siemens, The Granada Foundation, EPSRC and National Instruments [1]. The Orchestra debuted to the public at the Museum of Science and Industry in Manchester and has since gone on to feature in a BBC iPlayer documentary [2]. It has proven to be a popular STEM activity, promoting engineering to younger students by showing that an array of technical engineering skills can be used to produce something that is accessible to the masses. Due to its high praise, this project has been dedicated towards developing a new stand-alone core by constructing four new instruments and a robot conductor. It will also improve the reliability, flexibility and usability of the existing orchestra.

2.1.1. Required Skills

Looking at previous instruments that have been used in the Orchestra it can be seen that an array of skills are needed in order to make each of the instruments. For example, the Glock-O-Bot [3] required technical skills in analogue circuit design, coding in C and a deep understanding of MIDI files and text conversion. Creating the expansion for the robot orchestra will require these same skills in addition to several more, since the design is more complex and four robots are being built instead of just the one. Table 2.1 below shows how the different disciplines covered by the team help in meeting the skills requirement for this project.

 Indicates where skills need to be developed.

Degree	Electronic Engineering	Mechatronic Engineering		Electrical and Electronic Engineering		
Skills	Andrei Buruiana	Antons Petrovs	Francesco Fumagalli	Joshua Simpson	Joyanto Chanda	Theodoros Dimou
Analogue Circuit Design	X	X	X	X	X	X
Electronic Circuit Design	X	X	X	X	X	X
Coding in C	X	X	X	X	X	X
Experience with Arduino			X		X	
Experience with MyRIO						
Experience with Raspberry Pi			X			
3D Modelling		X			X	X
PCB Design	X	X	X			
Basic Music Knowledge				X		
Experience in LabVIEW						

Table 2.1. Different skills met by individual team members.

The approach for this project will potentially make use of three different embedded systems: the Raspberry Pi, Arduino and National Instruments' MyRIO so previous experience working with embedded systems is crucial. As can be seen, both the Arduino and Raspberry Pi come within the scope of past experience the team has, however, no one has worked with the National Instruments (NI) MyRIO. In order to tackle this issue, an application was submitted for the National Instruments Student Project Sponsorship [4]. The main benefit of the scholarship is that NI will provide a technical engineer to assist in the use of any NI software or hardware. In addition to the NI MyRIO the project is likely to utilise LabVIEW, the team does not have much experience in utilising either of these so having an engineer from National Instruments will help in meeting this skills gap.

The bursary also has the added benefit of allowing the team to compete in the National Instruments Student Design Competition. This is a competition designed for students from different universities to submit project designs and have the chance to be showcased at the Engineering Impact Awards in London. This is also a highly technical competition which will further emphasise the technical skills that are needed in this project as well as showing how technical engineering skills can be used to create something that is enjoyed by a non-technical audience.

2.1.2. Project Roles

In addition to the technical skills that will be required to complete the project, several soft skills will be required according to the role that each member has taken on. There are six main roles that need to be fulfilled by team members:

Project Manager (Joyanto Chanda)	Main purpose of the project manager is to ensure that the project is running on schedule and can be completed on time. This is done by defining the milestones that need to be completed in order to achieve each of the objectives and overall aim of the project.
Secretary (Joshua Simpson)	Important to make sure that minutes and agendas are prepared for each team meeting. These are crucial documents since they track important decisions made by the team.
Auditor (Andrei Buruiana)	Ensure that the team does not exceed the allocated budget. They are also responsible for placing the orders on behalf of the team, whether that is through iProc or other means.
Hardware Lead (Theodoros Dimou)	Defining the team's approach to the upcoming hardware deadlines. Important to have previous experience in hardware design (e.g. hardware lead on Buggy Project).
Software Lead (Francesco Fumagalli)	Defining the team's approach to upcoming software deadlines. Important to have previous experience in software development (e.g. third year individual project had a significant software component).
Document Controller (Antons Petrovs)	Responsible for ensuring that documents across the project are the same format and follow a similar theme. Will also be responsible for assembling and formatting the reports.

Table 2.2. Team roles to be fulfilled during the project.

2.2. Aims and Objectives

2.2.1. Aims

The aim of this fourth-year project is to create four or more new robot music instruments. These robots should follow a consistent aesthetic and theme that should represent electronic and mechanical engineering so it does not fall out of line with the existing robot orchestra. They must also be controlled by an electronic conductor which will control the robots playing the instruments.

These four robots need to be able to reproduce at least two different songs or themes, but should also be designed to be able to be programmed to play a variety of other musical pieces. A high level of autonomy is expected from the robots; ideally the only interaction should be the communication between the conductor and the robots.

Apart from the flexibility in the ability to play different styles of music, this new core of instruments as well as the conductor should be mindful of future additions and should be expected to accommodate for new instruments as well as the old ones. The robot instruments must also allow being disassembled, transported and reassembled quickly and easily so that time is saved when preparing for concerts and demonstrations.

2.2.2. Objectives

After the roles were assigned to individual members planning was done for the upcoming blocks, setting the following objectives:

- Propose designs for the four new instruments:
 - Select multiple songs and assign instruments that could perform its parts.
 - Select two suitable songs that use four of the same instruments.
 - Divide the team into four groups to work on designing each instrument.
- Construct the four instruments:
 - Design and manufacture prototypes for each instrument taking into account how they will interface with the conductor.
 - Test the prototype instruments to play tunes or songs individually.
 - Redesign / remanufacture as required based upon testing.
- Design a conductor:
 - Design the conductor which will communicate with the instruments.
 - Assemble the conductor.
- Assembling the new core orchestra:
 - Test individual instruments with the conductor
 - Interface the conductor with all the instruments.
 - Synchronise the four instruments to produce coherent music.

2.3. Motivation

The work undertaken in this project is motivated by the rising interest in the university's existing Robot Orchestra with the orchestra having most recently been part of an engineering tour sponsored by the Royal Academy of Engineering as well as having been part of a BBC programme [2].

As the existing orchestra has a limited range of songs it can play, adding a conductor and new instruments that can enhance its capabilities would allow for a wider selection of musical styles to be played and appeal to a broader audience. In addition to its outreach aspect, the project also stands to illustrate the operation of multiple embedded systems in performing a common task, making use of platforms such as Arduino, Raspberry Pi and MyRIO to control the performance of the orchestra.

Additionally, due to the combination of different development platforms that the orchestra uses, its commercial exploitation by leasing the orchestra to various businesses and events will also be investigated. As companies in the technology industry can be expected to be present at several trade fairs throughout the year, the Robot Orchestra offers a good opportunity for technology companies to showcase the creative aspects of engineering as well as the applications of their products in an atypical project that has a high chance of drawing attention and attracting new customers. Moreover, the project will also represent a good reference point for future projects focused on developing robot instruments.

Lastly, since it can be difficult to explain the benefits of pursuing one's interest in science and engineering to an audience of different age categories and backgrounds, the combination of engineering and arts in the robot orchestra developed during this project can also serve as a good aid for the university to promote its courses as well as advertise engineering to people who normally would not consider it.

3. Literature Review

To gather some initial ideas for the instruments to be constructed during the project the team was split in to sub-teams and each of these sub-teams was given a specific category of robot instrument to investigate. The following section describes some examples of robots that were found, categorised by the type of instrument.

3.1 Keyboard

Early on in the project it was decided that using a keyboard would be a good choice for one of the instruments. There were several designs that were found online, but the two that stood out the most are shown below.



Figure 3.1a. J. Shauw robot keyboard [5].



Figure 3.1b. Teotronica robot keyboard [6].

These videos showed the problem of a robot piano being approached in two different ways. Figure 3.1a utilised ten robotic fingers split in to two 'hands'. One of the hands remained stationary (presumably to play chords) and the other would slide up and down a rail, able to play an array of different notes. This was quite a visually appealing design because of how closely the robot managed to replicate human movement. The design shown on the right is a much more practical and easy to implement design. It utilises solenoids to push down the keys. From Figure 3.1b it can be seen that the solenoids are staggered so that the black and the white keys can both be hit.

Comparing the two designs, the one on the right is going to be easier to make both in terms of the hardware and the software. Using a moving hand poses significant challenges, specifically in terms of the software, that limit the number of songs that can be played. If the hand had to move, then the software would have to predict not only where the next note is based but where the next four notes are also based. If it did not do this, then there is a chance that the hand would have to move for each individual note. This form of software is within the skillset of the team but due to the time constraints it is not feasible to build and still construct three other instruments. The design on the right is significantly simpler, using only a series of solenoids in a fixed position to push down on the keys. The software for this style of design is a lot simpler to create as well, this is an important factor to consider since four instruments are being created.

3.2. Xylophone

Another instrument that was considered was the xylophone, which belongs in the percussion family and more specifically in the idiophone group [7, 8]. In general, the xylophone is tuned in different musical scales according to the country of origin. For example, in Africa and Asia it belongs in the pentatonic (i.e. five keys) and heptatonic (seven keys) scale respectively. If the xylophone is part of an orchestra, it is tuned in the chromatic scale and therefore has 12 notes [9].

After researching on how to integrate the xylophone with the robotic orchestra, the team decided to design their own xylophone which would be able to play 12 notes. The idea is based on a video found on YouTube [10], where a person created a customised xylophone with eight keys. Each key had a piezoelectric sensor mounted at its center and was programmed (through an Arduino) to produce a specific sound when it was hit (Figures 3.2a and 3.2b).

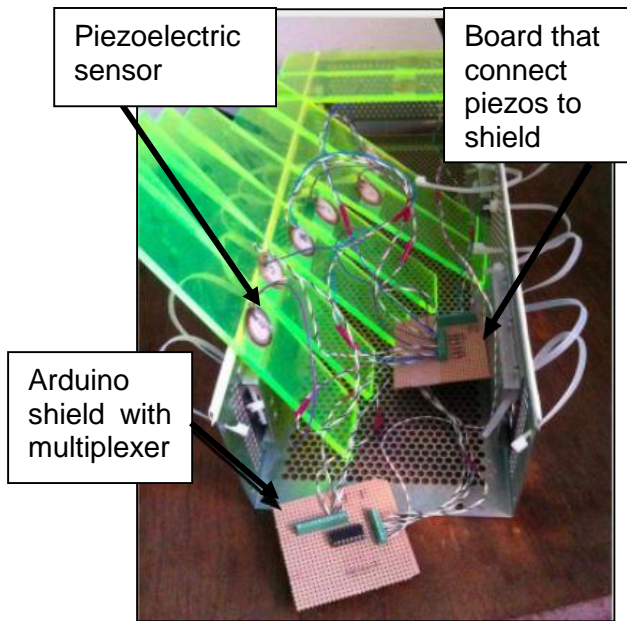


Figure 3.2a. Xylophone parts [11].



Figure 3.2b. Customised xylophone [11].

The shield contains a multiplexer and is connected to the 5V, ground, analog input and three digital outputs of an Arduino, whereas the other strip board illustrated in Figure 3.2a, is connected to the piezoelectric sensors. The Arduino is programmed to read signals from the piezoelectric sensors and it sends them back to the computer which in turn plays the corresponding note. The multiplexer is used in order to minimise the number of pins the Arduino needs to read the piezoelectric sensors.

3.3. String instruments

String instruments are musical instruments with a set of strings which vibrate and produce sound when they are pulled, hit or rubbed with a bow [12, 13].

Among the various string instruments, a decision was made to use the classical acoustic guitar, which is an instrument used in various different songs. Therefore, research was conducted on how to integrate the guitar with the robotic orchestra. The outcomes of the research showed that it is necessary to build two robots; the first one would control the notes that will be played on the upper part of the guitar (neck), i.e. the strings that need to be pressed for each song, whereas the second robot would control the strings that need to be played at the bottom part of the guitar (body).

3.4. Wind instruments

It was decided it would be interesting to demonstrate a wind instrument so current robot wind instruments were researched using YouTube. One issue with creating a robot wind instrument is that they require the air flow to be controlled in a specific way. For example, when playing the trumpet, the lips are vibrated in order to create a specific note or sound [14, 15]. There is a video of a robot playing the trumpet to achieve this using artificial lungs and lips [16]. Another example of a robot with artificial lungs and lips was used to play the flute [17] as seen in Figure 3.3, it was realised that designing and manufacturing a robot like this would be unrealistic due to time limit and budget.

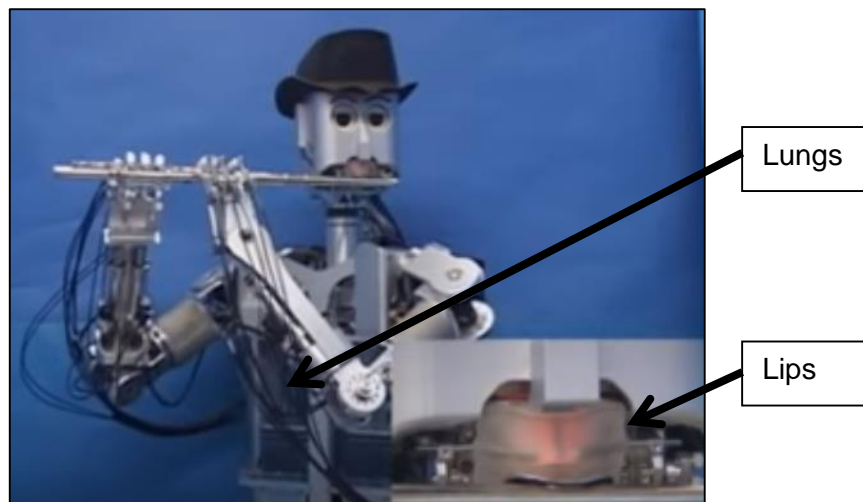


Figure 3.3. Flute playing robot with artificial lungs and lips [17].

Due to this it was decided that a wind instrument would have to be chosen in which the lip position required does not change and the airflow could be constant removing the need for artificial lungs. The instruments found to meet these requirements were the recorder and panpipes.

The panpipes are played by forming a small gap between the lips and blowing across the panpipe opening [18] and for the recorder there is just one tongue position which is maintained while blowing to produce the note [19]. Examples of robots playing the recorder [20] and the panpipes [21] were found on YouTube.

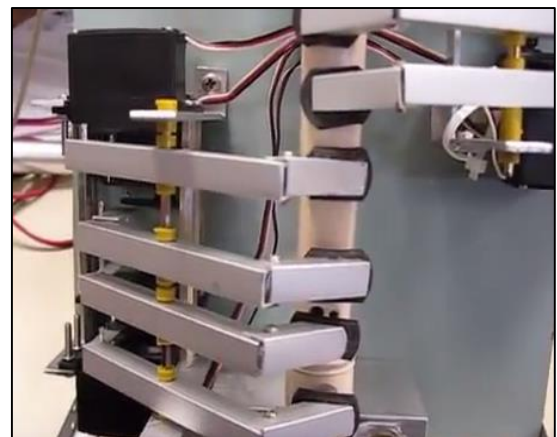
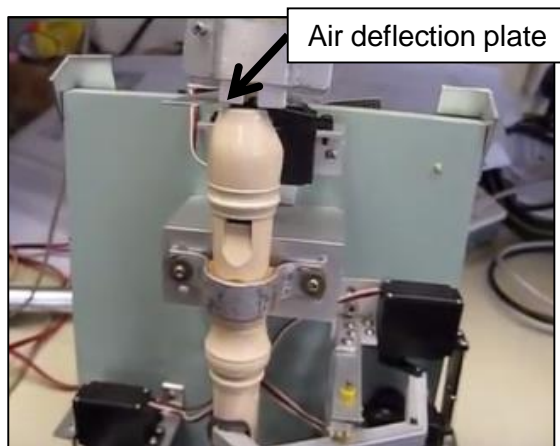


Figure 3.4a. Recorder mouthpiece section [20].

Figure 3.4b. Recorder fingers section [20].

The robot playing the recorder, as seen in Figures 3.4, uses a pump to generate air flow and does not have lips that go around the mouthpiece. One interesting thing about this instrument is that the air is produced continuously and it has a plate that moves in front of the air flow when no note is being played to deflect it away from the recorder. This removes the need for valves and the possibility of pressure building up behind the closed valve that is released when the next note is being played, which could lead to the note being produced incorrectly.

The panpipe playing robot, as seen in Figure 3.5, has one pipe that produces the air and that can be moved to change the angle at which the air is directed into the pipe, this is to create half notes which correspond to sharp and flat notes. The panpipes itself is rotated under the nozzle to line up different tubes with the nozzle to produce different notes. It also has a metal plate between the nozzle and the panpipes which is used to direct the air flow and simulate the lip position required. In this example the panpipes are used to play the vocals which could be an effective way of incorporating them into the orchestra.

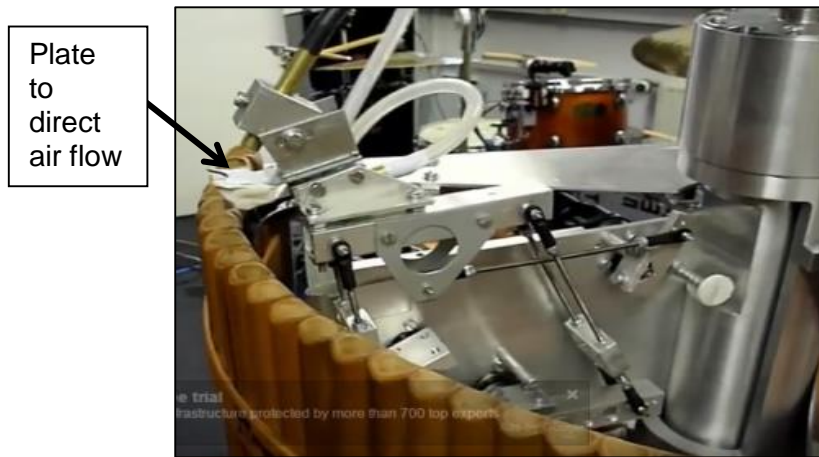


Figure 3.5. Robot panpipe example [21].

3.5. Stepper motors

A stepper motor is a DC motor that is able to convert digital pulses into equal steps of a rotation [22]. Generally, stepper motors are used in high-precision applications that use digital pulses as control signals rather than analogue voltage levels and by varying the frequency of the digital pulses, the mechanical shaft rotation can be modified. A high frequency of the pulses results in a continuous movement of the motor shaft. The advantage of stepper motors when compared to other type of high-precision motors such as servo motors is that it does not require a feedback mechanism as each individual pulse translates to the shaft rotating a precise angle [23].

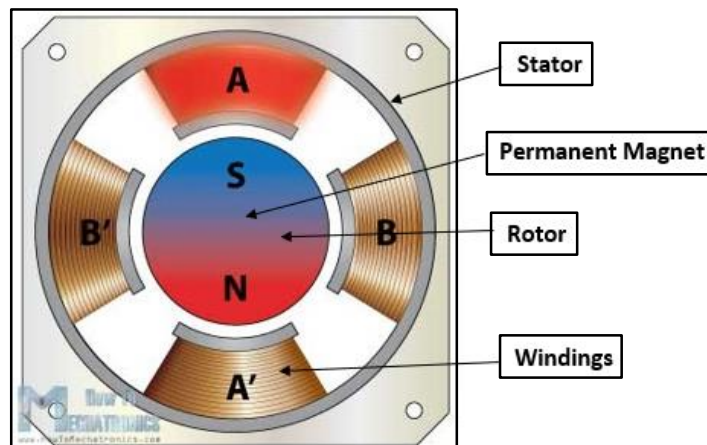


Figure 3.6. Cross-section of stepper motor [24].

Figure 3.6 presents the cross-section of a permanent magnet stepper motor. As it can be observed, the permanent magnet acts as the rotor of the motor, being surrounded by the windings of the stator.

A good way of understanding how the rotational movement is achieved is to analyse the rotor and the stator of the motor individually. The rotor comprises of two discs placed back to back, one acting as a magnetic north pole and the other as a magnetic south pole. As illustrated in Figure 3.6, the stator is placed around the rotor and is fixed. Each winding acts as an electromagnet that can be controlled independently. When activated, the stator will attract the disc of opposite polarity in the rotor, leading to the rotor's rotational movement [25].

Depending on the level of precision required, a stepper motor can be driven in 3 different modes: full step, half step and microstep. The difference between these modes is represented by the order in which the windings in the stator are energised and the effects it has on the rotational movement. For example, in half step mode, a digital pulse would produce an angular movement that is half of the one produced by the same digital pulse when the motor is operated in full step mode. The rotation of the motor can be controlled with a PWM signal (pulse-width modulated signal) of different frequencies and if this frequency is in the audible range (20 Hz - 20 KHz [26]), the motor is able to produce different musical notes.

3.6. Tesla Coils

A Tesla coil is a transformer where the secondary coil is left with an open circuit. Figure 3.7a shows a Tesla coil circuit with a spark gap. In this circuit the capacitor is charged and when the voltage over the capacitor is large enough a spark forms over the spark gap and the capacitor is connected in parallel with the primary coil and it discharges [27]. This causes a voltage to be induced over the secondary coil with the voltage level increased by the turns ratio N_2/N_1 . The turns ratio needs to be large such that the voltage over the coil is large enough to overcome the dielectric strength of the air to create streamers. The dielectric breakdown of air is 3 kV/mm [28] so to get a streamer of 10 cm a voltage of 300 kV is needed. The image in Figure 3.8 shows streamers being created as "In the Hall of the Mountain King" is being played with the Tesla coil.

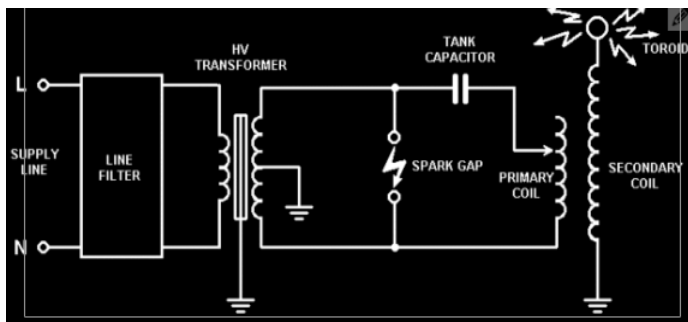


Figure 3.7a. Spark gap Tesla coil circuit [27].

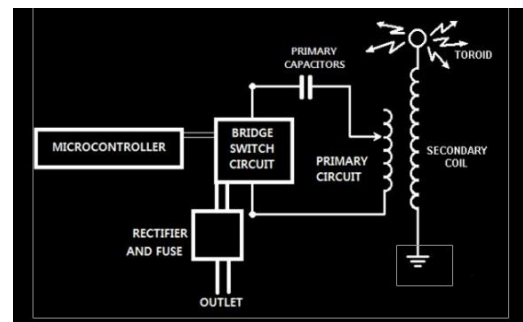


Figure 3.7b. Switching Tesla coil circuit [27].



Figure 3.8. Streamers being created by Tesla coil playing "In the Hall of the Mountain King" [29].

To make it play music the spark gap needs to be replaced by a switching circuit (Figure 3.7b) which connects and disconnects the capacitor from the primary coil (switching circuit diagram in Figure 3.7b). To get it to play music the switching frequency is modulated by the music signal which induces a voltage in the secondary at a given frequency, producing streamers at a certain frequency which will cause the air to vibrate at that frequency and produce the correct sound [30].

3.7. Embedded Systems

Embedded systems are one of the key aspects of the Robot Orchestra project. Every instrument requires a controller, and they all need to interface with the conductor. At the heart of all embedded systems lies a microcontroller so additional care needs to be taken when selecting the appropriate one. Various parameters affect the choice of microcontroller, most notably, the clock speed, number of IO pins as well as the different interfaces used to connect peripherals. Each part of the orchestra also requires certain things, for example the conductor should be able to have a graphical user interface to allow user friendly interaction. However, there is a lot of flexibility when choosing microcontrollers for the instruments and therefore choice of microcontroller for the instruments will be based on other factors discussed next.

As the project is also used to showcase specific skills as well as how to apply them whilst learning new things, the majority of the microcontrollers and peripherals will be chosen with that in mind. Moreover, using diverse programming languages, such as C, Python and LabVIEW will also allow for continual development of team members' programming skills. To conclude, apart from the conductor which has specific needs to be met, the rest of the embedded systems will be selected with the team members' personal development of skills in mind.

3.7.1. Conductor

The conductor has three tasks to perform. The first is to convert MIDI files into a format which can be interpreted by the various microcontrollers used (more in Section 3.7.2 below). The second is to transmit the converted MIDI files to the corresponding instruments, and lastly to synchronise the instruments. Furthermore, to make the conductor more elegant, a decision was made to use a GUI to allow the user to interface with the conductor. Due to these requirements, a Raspberry Pi 3 (RPI) model B single board computer was chosen. Due to the familiarity with the RPi and the complexity of the conductor the RPi was considered the best choice. The RPi is a System-On-Chip which runs various Linux distributions, including the official OS known as Raspbian [31]. It is compatible with WiFi and Bluetooth and has 40 GPIO pins [32]. It also has various other interfaces such as SPI and I2C which are commonly used to connect to peripheral devices. Python will be exclusively used to run scripts on the RPi and PyQt, a python based GUI builder to design the user interface.

3.7.2. Instrument Microcontrollers

Since four different instruments will be built, in order to use a variety of microcontroller at least two different ones will be used. As mentioned previously, since the load on the microcontrollers will be relatively low the choice will only be based on educational factors. The following four microcontrollers have been considered for the instruments.

ATmega328P (Arduino Uno)

The ATmega328P is the microcontroller used in the Arduino Uno platform. Arduino is an open-source platform consisting of both hardware and software. The hardware is based around the ATmega 8-bit microcontroller which is flashed with the Arduino bootloader, allowing Arduino programs, known as sketches to run on the microcontroller [33]. Arduinos can be used to program simple tasks as well as more complex ones with relative ease, thanks to the Arduino language which is a simplified version of C++. Arduino also has vast online support and many libraries for various peripheral devices have been written for it, allowing faster development. The ATmega328P consists of 14 digital IO pins, 6 analogue pins, SPI, I2C, USART as well as other standard peripherals such as timers and interrupts [34]. It also interfaces with USB, WiFi, Bluetooth and XBee to name a few. This gives flexibility when developing the instruments. In order to customise the Arduino board to include various other integrated circuits needed for the instruments as well as to make it as professional as possible, a breakout board for the ATmega328P was made. The Arduino bootloader was then flashed onto it to allow the user to upload Arduino sketches.

A prototype board was first developed which included USB to UART FTDI chip which allows the user to upload sketches directly from a PC through USB without any additional external components as well as allowing the Arduino to transmit back to the serial monitor on the PC. Moreover, SPI, I2C as well as all the digital and analogue pins have been broken out. A schematic can be found in Figure 1.2 of Appendix A; photographs of the final design can be found below in Figure 3.9. The prototype was designed by looking at existing ATmega328 circuits as well as reading the datasheets and application notes of the various components used [35, 36, 37, 38]. The prototype was tested and found to be working as expected. The next step will be to expand the board by adding other integrated circuits and components needed for each instrument. Another advantage of building the modules needed for the project is the cost. the overall cost of making the ATmega328P breakout was less expensive than buying an official one from the Arduino store. The final cost of making the breakout with all the components was under £5 whereas the Arduino Nano can be bought for £20.

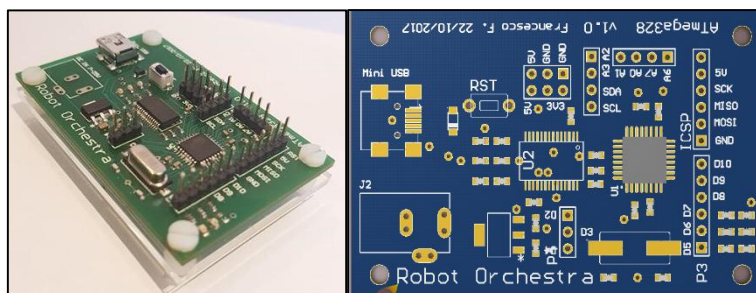


Figure 3.9. ATmega328P breakout.

NI MyRIO

The MyRIO is an 'Embedded Device' [39] developed by National Instruments. It consists of digital IO pins as well as analogue pins. It includes a Xilinx FPGA as well as a dual core ARM A9 processor. It can be programmed using both LabVIEW as well as C [39]. The MyRIO is also compatible with WiFi and Bluetooth standards and has all the peripherals which can be found on the ATmega328. The team has decided to use the MyRIO as the main driver for the xylophone. The reason for using MyRIO in the project is purely educational as the team does not have any previous experience with MyRIO and only limited experience using LabVIEW. It was seen as an opportunity become familiar with an industry standard software suite used by many engineering firms.

3.7.3. Wireless Link

The link between the conductor and the individual instruments will be wireless. The team has considered a few different options. It should be pointed out that the team is currently working on developing the instruments and therefore is not actively developing the communication between the Pi and the instruments. Preliminary research of the available options was conducted. However, an in depth look at the various options as well as testing each one will be done in semester two.

Bluetooth

The Raspberry Pi has an inbuilt Bluetooth module capable of transmitting to other Bluetooth enabled devices [32]. The Arduino would require an external Bluetooth module as it does not have an in-built one. To have the Raspberry Pi transmit to multiple devices at once a piconet is set up. Up to seven devices can be connected to the piconet as slaves, with the RPi acting as the master. The group does not have any experience setting up piconets, making Bluetooth rather challenging. However, prior to conducting further research into the feasibility of Bluetooth, it will not be ruled out.

WiFi

After researching WiFi as a transmission method the team found that it is much easier to implement than Bluetooth. The Arduino platform is capable of transmitting and receiving WiFi packets using an inexpensive additional module. Since the data transmitted will be relatively simple, a User Datagram Protocol (UDP) can be used with the RPi acting as a server and the instruments as clients. The Arduino platform provides a library which converts the Arduino into a client capable of receiving UDP packets over WiFi. The MyRIO has an inbuilt WiFi card and the MyRIO toolkit makes setting up the connection straightforward. Although this has not been tested yet.

Radio

Radio is the only method which the group has experience connecting multiple devices to. The NRF24L01+ [40] is a 2.4 GHz radio module which interfaces with both the Arduino and the Raspberry Pi. This specific module allows for up to 127 different modules to be connected simultaneously over different channels. Arduino and Python libraries allow easy development on either platform. The NRF24L01+ uses SPI to communicate with the microcontroller making it compatible with the MyRIO, however due to the high complexity of the device a library is needed for fast development and stable operation. Another benefit of using the NRF24L01+ module is that it is much cheaper than the WiFi/Bluetooth modules.

3.8. MIDI

MIDI stands for Musical Instrument Digital Interface and is used to compile all the information needed to reproduce a song under a common format. MIDI files allow specification of which note to play, for how long and with what velocity the instrument key is to be pressed [41, 42]. MIDI has become a standard interface used in the music industry and, for the sake of simplicity, will be the only format used in the project. MIDI files for a vast number of songs can be easily found online, allowing the team to easily choose from many songs that are available.

Each MIDI file starts with a *Header Chunk*. The header chunk specifies various parameters needed to decode the rest of the file. It is constructed as follow [43, 44, 45]:

Length	4 bytes	4 bytes	6 bytes		
Value	'MThd'	Length (6 bytes)	FF FF	NN NN	TT TT

Table 3.1. MIDI Header Chunk.

The top row in Table 3.1 indicates the size allocated to each section of the header. The bottom row is the actual MIDI header. The following is a breakdown of a MIDI header chunk:

- The first 4 bytes of the header chunk correspond to **'MThd'** in ASCII (4D 54 68 64 in HEX), this specifies that the incoming chunk is a header chunk.
- The next 4 bytes specify the length of the data which is to come. For header chunks this is always 6.
- The next 6 bytes of data specify three things [43, 44, 45]:
 - **FF FF** - The file format of the MIDI file. Three formats are accepted:
 - FF FF = 1: One track in the MIDI file.
 - FF FF = 2: Multiple synchronous tracks in the MIDI file. Which means that the file contains multiple tracks which all start at the same time.
 - FF FF = 3: Multiple asynchronous track in the MIDI file. Which means that the file contains multiple tracks all with different starting points.
 - **NN NN** - The number of tracks in the MIDI file.
 - **TT TT** - The timing parameter.
 - The MIDI standard defines timing in its own way, which is *delta ticks per quarter note* and essentially defines the timing for the rest of the MIDI file.

The remaining MIDI file consists of the *Track Chunk*; Table 3.2 shows its layout.

Length	4 bytes	4 bytes	Variable depending on specified length	
Value	'MTrk'	Length	Delta time	Event

Table 3.2. MIDI Track Chunk.

- The first 4 bytes of the track chunk are **'MTrk'** denoted in ASCII (4D 54 72 6B in Hex) this specifies that the incoming chunk is a track chunk.
- The next 4 bytes specify the length of the data which is to follow, which varies depending on the data.
- The following 4 bytes specify the *Delta Time*, this is the amount of time that needs to pass prior to executing the MIDI event.

The last section of the file specifies the event which is to be executed.

Length	1 byte	1 byte	1 byte
Function	Status Byte	Note Number	Note Velocity

Table 3.3. MIDI event message.

The track event can be one of three things. MIDI, meta and sysex events, all of which are always preceded by specific timing information. For this project, only MIDI events need to be considered. MIDI events contain messages sent to each individual channel (instrument), an example MIDI event message is shown in Table 3.3. MIDI events consist of three bytes, the first byte is known as the status byte and its function is to specify the type of command. A table with examples status bytes and their function can be found in Table 3.4. The second byte specifies the note to which the status byte is applied to and the third byte specifies the note velocity (loudness or softness).

A table showing the musical note to which each binary value corresponds to can be found online [43, 44, 45, 46].

Status Byte	Function
0x80	Channel 1 note off
0x81	Channel 2 note off
0x90	Channel 1 note on
0x91	Channel 2 note on
0xB0	Channel 1 Control Mode Change

Table 3.4. MIDI Status Byte example functions.

A complete MIDI track chunk containing only MIDI events would look like the following:

Length	4 bytes	4 bytes	Variable depending on specified length			
Value	'MTrk'	Length	Delta time	Status Byte	Note Number	Note Velocity

Table 3.5. MIDI Track Chunk example.

3.9. Music Background

In music a specific set of frequencies called notes are used which follow the sequence CDEFGABC. For example, if this started at C1 it would end at C2 which would have double the frequency of C1 [47]. This is called an octave.

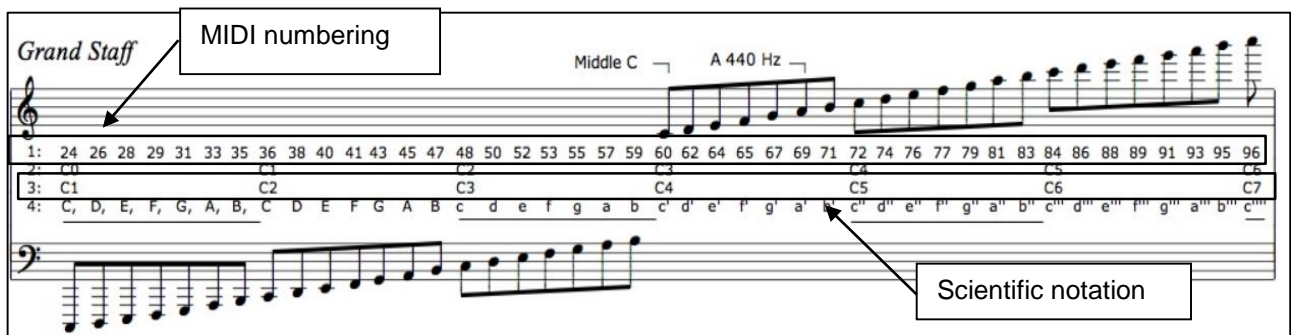


Figure 3.10. Diagram showing different musical note notations [48].

The important parts of Figure 3.10 are point 1 which is the MIDI numbering and point 3 is the scientific musical notation. These are the notations that will be referred to in this section.

An octave contains 12 notes, four of which are sharps or flats, e.g. the black keys on a piano. The frequency of each note is $2^{1/12}$ larger than the previous note's frequency. This is the reason why in the previous example C1 has half of the frequency of C2 [47]. These two notes are both classed as C's as they sound similar due to the relationship between the frequencies being a multiple of two.

A limitation of some of the instruments like the panpipes and xylophone is that they have a limited range of notes they can play. Therefore, the range of notes should be chosen to match the song choices. If the range of notes one of these instruments is required to play exceeds the range available to the instrument, it will sound out of key. For example, if the instrument has a range of C1 to E2 and the range of the music is C1 to C3 the instrument would have to revert back to C2 or C1 when trying to play C3 which would sound out of place in relation to the other notes being played. However, if an instrument had a range of C1 to E2 and was needed to play notes from the range C2 to E3 it would still work, as it is the same range of notes only an octave higher so the notes would sound similar.

3.10. Anvil Studio Overview

Anvil Studio is a MIDI and audio editing package developed by Willow Software [49]. Anvil Studio offers various tools which allow to edit MIDI files. It was also used in the development of the MIDI2Text script due to its ability to convert MIDI files into text files. The output of the MIDI2Text script were compared to those of Anvil Studio to verify proper operation of the script.

More importantly, Anvil Studio allowed the team to edit the different tracks so that unwanted tracks were removed to make the song compatible with the instruments in the orchestra. It also allowed the team to change the specific instrument on each channel, simulating the actual instruments used. Moreover, the channel can be altered in order to match pre-defined channels for each instrument rather than the default channels of the MIDI file.

The software package was critical in the decision-making process for the songs. Various instruments were altered in each track depending on requirements for each instrument, previewed each instrument individually and then played each song. Once the team was happy with the outcome, the MIDI file was saved under a new name allowing it to be imported into the Python script. The package contains 128 different instruments and sounds so the team managed to find sounds similar to the non-conventional instruments used, such as the stepper motors and the Tesla coils as well as common instruments such as the xylophone and the panpipes. Although a decision for the songs which the instruments will play has been made, Anvil Studio is currently used to edit songs used to test individual instruments and could be used in the future for adding additional songs for the orchestra to play.

4. Technical Progress

The following section describes the technical progress that has been made on the instruments mentioned in literature review during the two project blocks.

4.1. Conductor Design

The structure of the embedded systems has been mentioned briefly in Section 3.7, however this serves as a more comprehensive description. Figure 4.1 depicts a diagram showing the hierarchy of the system. First and foremost, the Raspberry Pi conductor takes input from the user through its GUI. The GUI takes in a few parameters, the file location of the MIDI file that the orchestra will play and the file location which the text output will be saved. The GUI is kept relatively simple, however if more features need to be implemented they can be added in the future. Once the MIDI file is selected the song is then converted after the user clicks on the 'Convert' button. More on the conversion can be found in Section 4.1.2. The converted file can be found in the specified location. This is useful for the user to check that the output matches with what is expected. Finally, the user clicks the 'Play' button causing the conductor to send the corresponding text file to each instrument. For simplicity, the instrument channels are static and have been predefined. The user is expected to edit the MIDI file prior to uploading it to the Pi to make sure that each channel corresponds to the correct instrument.

4.1.1. High Level Overview

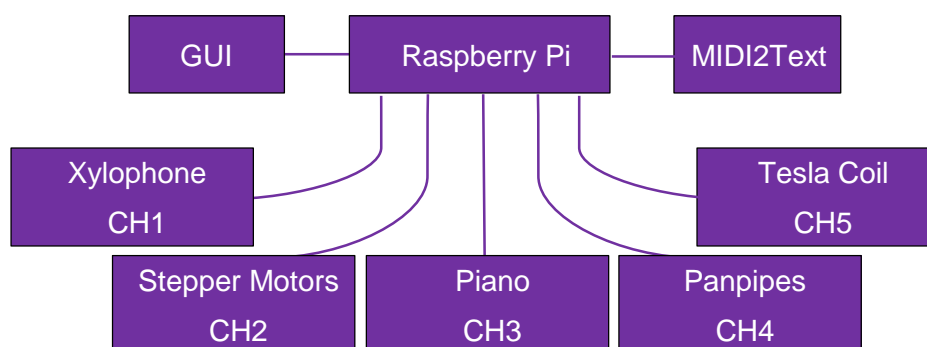


Figure 4.1. High Level System Overview.

4.1.2. MIDI2Text

The process of converting MIDI files into a format which can be interpreted by the various microcontrollers is a hard one, especially when done manually. For that reason, after having obtained a good understanding of the MIDI format, a Python script was written to automate this task. The Python script will run on the Raspberry Pi and will be controlled by the user through a GUI. Below is a flowchart with the steps the program follows to convert from a MIDI format into a text format.

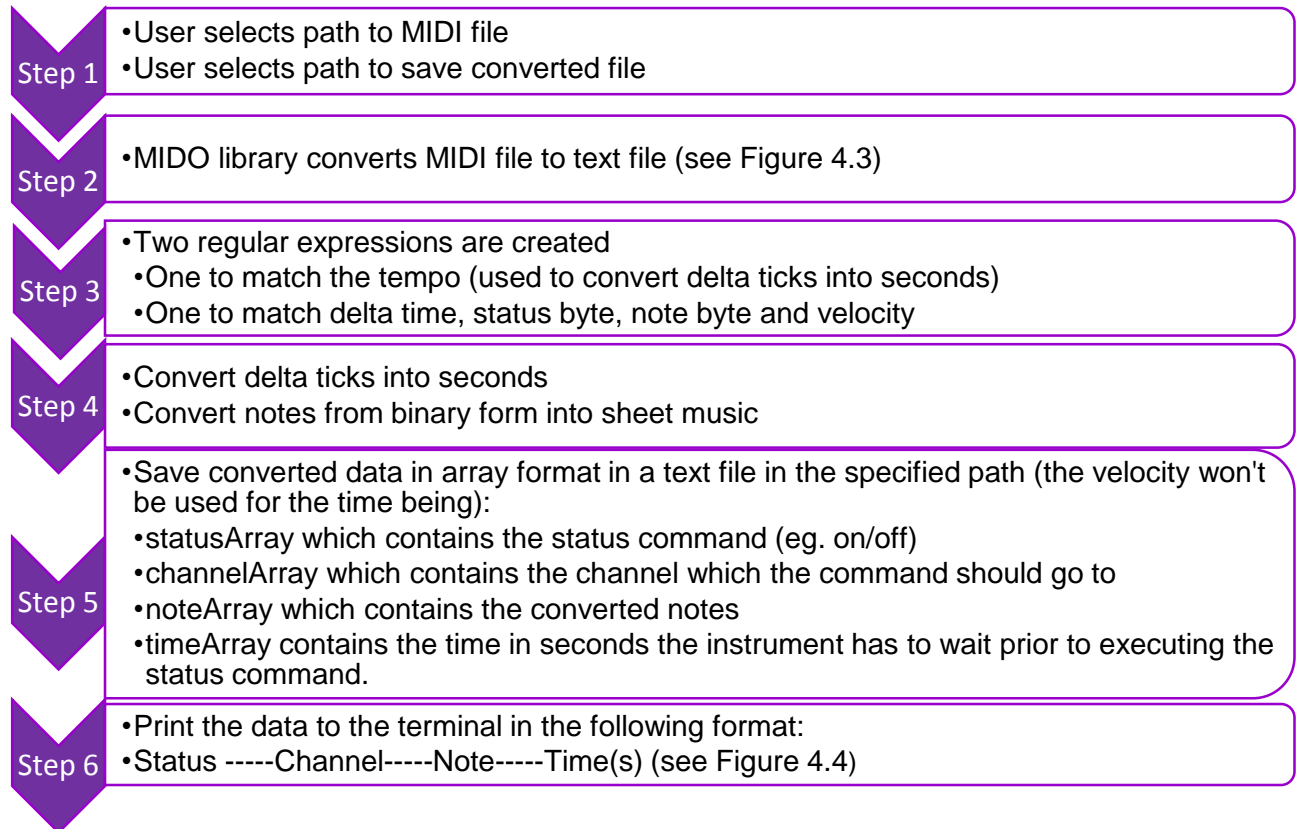


Figure 4.2. MIDI2Text flowchart.

The use of regular expressions [50] to extract each individual element gives the programmer great flexibility, since the data can be reused in various ways. The MIDI2Text script has been tested extensively and after a few revisions is currently working properly. Figure 4.5 shows the GUI. The full code can be found in Appendix B [50, 51].

```
Python 3.6.2 (v3.6.2:1d0333b5, Jul 8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\franc\Documents\MEng Project\Software\midi2text.py ====
Track 0:
(meta message time_signature numerator=4 denominator=4 clocks_per_click=480 notated_32nd_notes_per_beat=4 time=0)
(meta message sequencer_specific_data(0, 0, 63) time=0)
(meta message set tempo tempo=500000 time=0)
(meta message end_of_track time=0)
Track 1:
control_change channel=0 control=0 value=0 time=0
program_change channel=0 program=14 time=0
note_on channel=0 note=40 velocity=99 time=355
note_off channel=0 note=40 velocity=0 time=256
note_on channel=0 note=40 velocity=99 time=45
note_on channel=0 note=42 velocity=99 time=256
note_on channel=0 note=43 velocity=99 time=4
note_on channel=0 note=43 velocity=99 time=0
note_off channel=0 note=43 velocity=99 time=0
note_off channel=0 note=42 velocity=99 time=24
note_off channel=0 note=42 velocity=99 time=96
note_on channel=0 note=40 velocity=99 time=376
note_off channel=0 note=42 velocity=99 time=24
note_on channel=0 note=43 velocity=99 time=440
note_off channel=0 note=40 velocity=0 time=4
note_off channel=0 note=43 velocity=99 time=440
note_off channel=0 note=40 velocity=99 time=24
note_off channel=0 note=43 velocity=99 time=40
note_on channel=0 note=44 velocity=99 time=24
note_on channel=0 note=45 velocity=99 time=0
note_on channel=0 note=45 velocity=99 time=0
note_off channel=0 note=45 velocity=99 time=132
note_on channel=0 note=40 velocity=99 time=24
note_off channel=0 note=44 velocity=99 time=4
note_off channel=0 note=45 velocity=99 time=0
note_off channel=0 note=40 velocity=99 time=204
note_on channel=0 note=40 velocity=99 time=44
note_on channel=0 note=40 velocity=99 time=44
```

Figure 4.3. Raw Output.

```
Python 3.6.2 (v3.6.2:1d0333b5, Jul 8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\franc\Documents\MEng Project\Software\midi2textV2.5.2.py ====
Status(on/off) Channel Note Time(s)
on 0 C 1.15625
off 0 C 0.3333333333333333
on 0 C 0.0625
on 0 D 0.36979166666666663
on 0 E 0.005208333333333333
on 0 F 0.0
on 0 D 0.0
off 0 C 0.03125
off 0 E 0.125
on 0 C 0.4895833333333333
on 0 F 0.859375
off 0 C 0.005208333333333333
off 0 F 0.5833333333333333
off 0 D 0.03125
off 0 F 0.0625
on 0 E 0.03125
on 0 E 0.0
on 0 C 0.0
off 0 E 1.4739583333333333
on 0 C 0.03125
off 0 E 0.005208333333333333
off 0 C 0.0
on 0 C 0.36979166666666663
on 0 D 0.057291666666666664
on 0 D 0.1875
on 0 F 0.0
off 0 C 0.03125
on 0 D 0.0
off 0 D 0.7708333333333333
on 0 C 0.0
off 0 C 0.6145833333333333
```

Figure 4.4. Processed data.

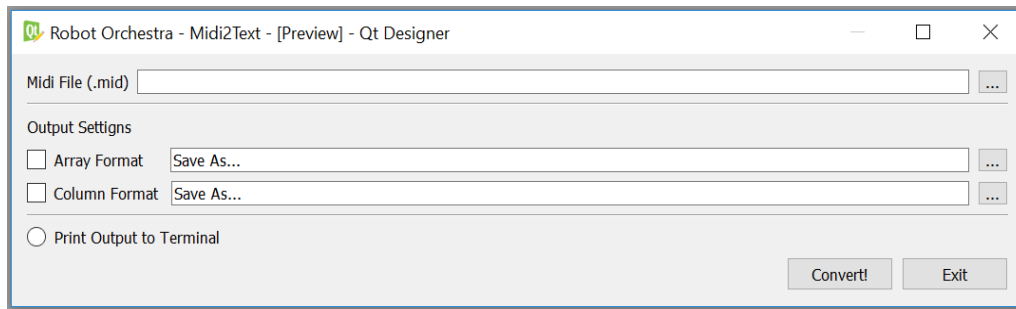


Figure 4.5. MIDI2Text GUI.

4.2. Solenoids

Solenoids will be utilised in both the robot keyboard and robot xylophone as shown in Section 4.3 and Section 4.4 respectively. They are common electronic components that come in various forms, electromechanical, pneumatic and hydraulic [52]. The solenoids used in this project are electromechanical, they were selected as they were cheap to purchase and were simple to implement in the robot keyboard and xylophone. A significant amount of force would not be needed so pneumatic and hydraulic (which are more expensive as well) were not considered.

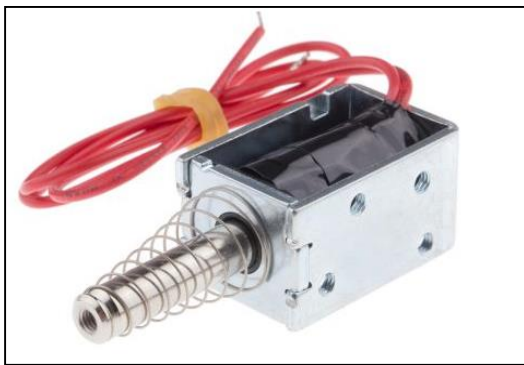


Figure 4.6a. Pull action solenoid from Solentec Ltd [53].

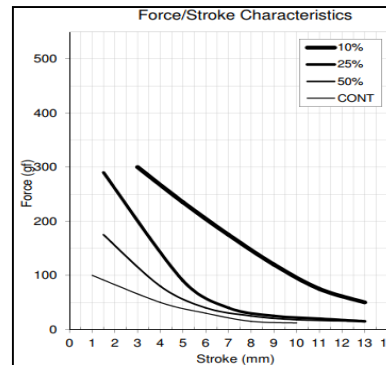


Figure 4.6b. Solentec solenoid's force/stroke characteristic [53].

Electromechanical solenoids consist of two main components: a coil and plunger or slug that is placed inside the coil (this is usually made out of steel) [52]. According to Ampere's Law [54] when a current is passed through the solenoid a magnetic field is created in and around the coil which forces the plunger to be drawn in. As can be seen in Figure 4.6a a spring is used to hold the plunger in place, this is so that when a current is not being passed through the solenoid and no force is acting on the plunger it returns to its normal position outside of the coil, this is typical of a 'Pull' type solenoid. 'Push' type solenoids are also available where the spring holds the plunger inside the coil when the coil is not energised, when a current is passed through the plunger is forced out of the coil. A solenoid is dictated by two parameters, the duty cycle of the solenoid and the force produced according to the duty cycle. The duty cycle is calculated by the following formula:

$$\text{Duty Cycle (\%)} = \frac{\text{On Time}}{\text{On Time} + \text{Off Time}} * 100 \quad (1) [53]$$

In this case the "On Time" refers to the amount of time that the coil is energised and the "Off Time" refers to the time that it is not. Figure 4.6b shows how the force exerted by the solenoid reduces as the duty cycle increases i.e. as the rate at which the solenoid is energised and de-energised increases the force exerted by the plunger decreases. This is because as the solenoid is energised and de-energised it begins to gain heat which increases the losses in the solenoid, so less force is delivered. The stroke is the amount of the plunger that is outside of the coil when it is energised. It can be seen that as the duty cycle decreases the force that can be exerted by the solenoid increases, this means that if the solenoid is not switched on and off very quickly then more force can be expected to be produced.

4.3. Keyboard

The main aim of the keyboard design was to be as simple as possible, since the design for both the xylophone and panpipes was quite complicated. With this in mind, the first design was created.

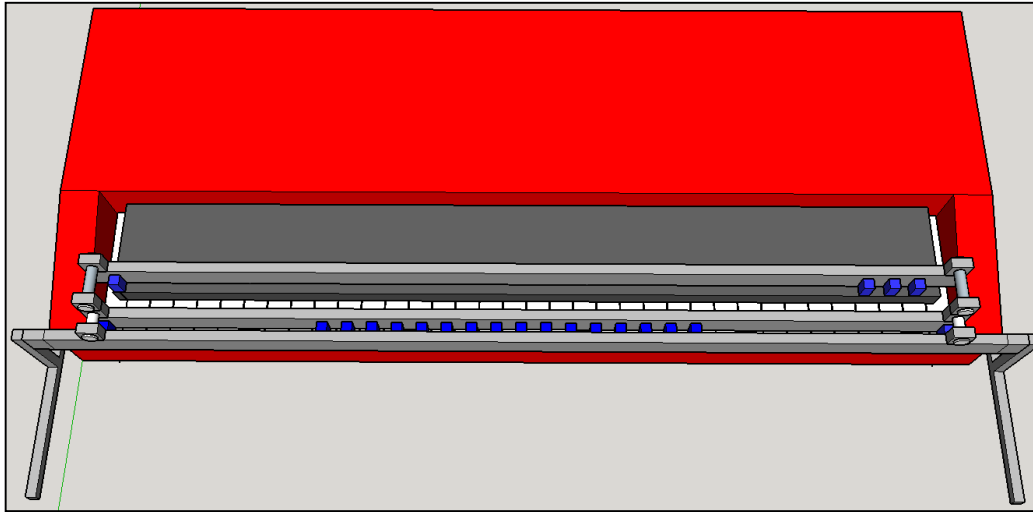


Figure 4.7. Initial Keyboard Design shown using Google SketchUp.

The first design utilised three metal panels and can be seen in Figure 4.7. Panels 2 and 3 were used to screw the solenoids into. The solenoids are represented by the blue boxes in Figure 4.7, they would be screwed into the panel and spaced out so the plunger of each solenoid fell in line with a key. It was important to have two separate rails of solenoids since the robot needed to be able to play both the black keys and white keys. Panel 1 acted as a support for the other two panels and was attached to the legs of the frame, various other views of the design can be seen in Figures 3.2 and 3.3 of Appendix C. This design had the advantage of being simple in construction and not needing many steps in order to be complete.

This model was completed at the end of the first project block (13/10/17) but at this stage the keyboard that was going to be used in the demonstration had not been ordered, this meant that the design could not be properly dimensioned. It also meant that the exact solenoid needed for the design could not be chosen since the force needed to push down the key was unknown. Since it was impossible to determine the exact solenoid that was needed some common 5V solenoids were ordered from SparkFun Electronics [60]. These were ordered in low numbers since it was expected they would have to be changed in the second project block.

When the testing phase of the second project block began a circuit to connect the solenoids to a digital I/O pin on the Arduino was created which energise the coil and pulls the plunger. This circuit was used to test whether the preliminary solenoids ordered in the first project block were strong enough to push the key down on the keyboard. It was quickly found that a 5 V solenoid was simply not strong enough to push the key down, this was to be expected since they had been ordered without any measurement of the forces needed.

To measure this force some M10 nuts were weighed and then placed, one by one, on a single key until a sound was triggered. It was found that when the weight of the nuts equated to ≈ 70 gf this was strong enough to trigger a sound from the keyboard. A solenoid that is able to produce ≈ 105 gf (1.5×70 gf [55]) should then be used for the keyboard design. In addition to this, the maximum duty cycle had to be determined for the solenoids, this would be dependent on the specific song being played since each of the songs had the keyboard on for different amount of time. Using Anvil Studio, the piano track for each of the songs was looked at, from this it could be determined that the song that would produce the greatest duty cycle would be *Eye of the Tiger* by Survivor. Since one solenoid controls one note with this design it was important to identify the note that would be need to be 'on' the most during the song. Through using Anvil Studio again to study the *Eye of the Tiger* MIDI file the note C# was determined to be 'on' the most during the song. By using Equation 1 and measuring the seconds that the C# was being played throughout the song the maximum duty cycle was determined (Equation 2) to be 30%.

$$\text{Duty Cycle (\%)} = \frac{162 \text{ seconds}}{162 \text{ seconds} + 385 \text{ seconds}} * 100 \quad (2)$$

Using this and the required force, a new more appropriate solenoid was selected. Initially the MCSMO-0630S12STD from *Multicomp* was selected since the Stroke Vs Force characteristic met the requirements needed for this project (Figure 3.1 from Appendix C [56]). In order to test this solenoid a new circuit had to be created to interface the solenoid with the Arduino board since it cannot deliver enough current to meet the needs of the solenoid. At approximately 30% duty cycle the solenoid consumes 9.6 W of power at 12 V DC. This means that it needs 800 mA to operate at this power, the Arduino is only capable of supplying 40mA from one of its digital IO pins. In order to power the solenoid a circuit was created using a transistor that allows the solenoid to be powered from an external power supply while being connected to the Arduino (see Figure 4.12 in Section 4.4).

Using this circuit, it was determined that this solenoid was strong enough to push one of the keys down on the keyboard. However, this solenoid has since been replaced with the 905-9931 from *Solantec Limited*. This is because the solenoid from *Multicomp* did not come with a spring attached to hold the plunger in place when it was not energised. The solenoid from *Solantec* comes fixed with a spring and has very similar stroke vs force characteristics, using this solenoid will save significant time as individual springs will not have to be fixed on to the solenoids manually.

It was also decided that the initial model designed (Figure 4.7) would have to be modified in order to make it more adjustable to the height of the keyboard. The previous design had a fixed height for the panels, so, in order to overcome this threaded rod was used in replacement of the metal legs as seen in Figure 4.8. These rods would be fed through a Bosch Bar (Figure 4.9a).

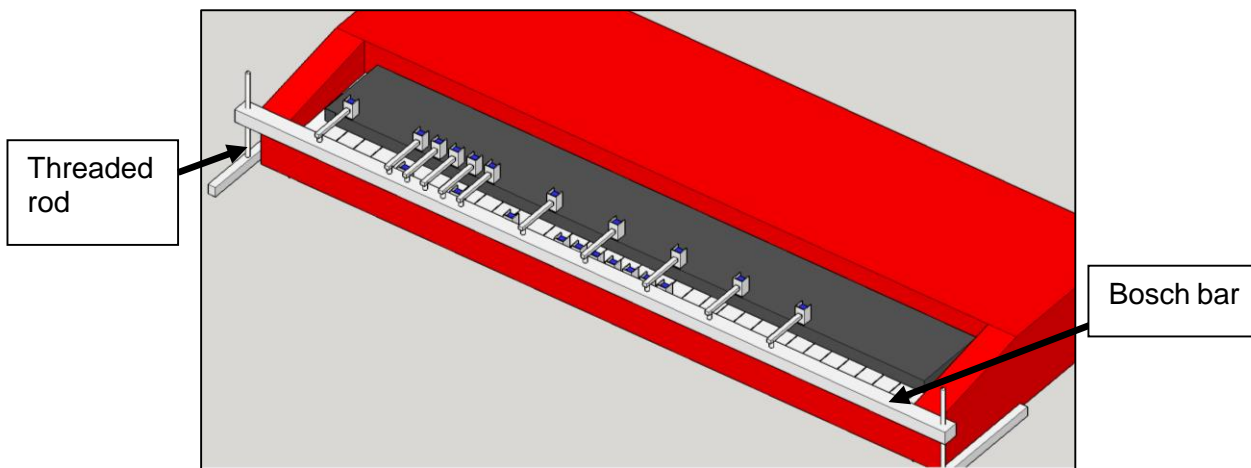


Figure 4.8. New robot keyboard design.

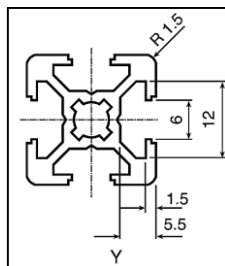


Figure 4.9a. Cross-section of Bosch bar [57].



Figure 4.9b. T-Slot nut [58].

The reason for choosing a Bosch bar is largely to do with simplicity of design. From looking at Figure 4.9a it can be seen that the Bosch bar being used has grooves of a specific dimension. T-Slot Nuts can be purchased which fit in to Bosch bar and can slide along the bar. Clamps will be designed for the solenoids which can be screwed in to individual T-Slot nuts which, depending on whether that particular solenoid will be playing a black or white key, will be a specific distance away from the Bosch bar. From Figure 4.8, the solenoids intended to play the black keys are mounted on the top surface of the bar and supported by a bracket. The solenoids playing the white keys are mounted to the front surface.

Also, to simplify the design further, it was found that only 19 notes were needed in order to play both *Eye of the Tiger* and *Californication* so 19 clamps will be designed and fixed to the Bosch bar. The design can be expanded to play all 61 notes by making more clamps for the solenoids and purchasing more T-slot nuts to fix them to the bar.

4.4. Xylophone

The model of the xylophone has 12 keys, where the notes will be programmed so that they can vary between the octaves according to the song that will be played. In addition to this, keys can be reprogrammed to play other sounds.

Model 1

In the initial design, the 12 bars were held by two vertical plates, joined together using the friction-tight method, with two square-form bars, each one located at the sides of the xylophone (Figure 4.10). Moreover, piezoelectric sensors were attached at the top middle of the keys which would produce a signal triggering the corresponding sound to be played through the speaker. Finally, a square arch was modeled, which held one solenoid at its top center, and with the aid of wheels would be programmed to travel across the xylophone (Figure 4.10). Transparent Perspex would be used to manufacture the xylophone. This would allow the audience to inspect the interior of the xylophone, which would be fitted with LEDs along with the circuitry needed for the device to function, providing an overall better visual appeal.

The arch was designed in three different sections which would be joined using L-shaped mounting brackets. Additionally, a rod was placed between the parallel plates of the arch as a support, preventing it from toppling as well as guiding it to travel across the xylophone. As for the arch's locomotion, stepper motors were the preferred method to be used since steppers have more torque than servos [59]. A stepper motor would have been mounted on each side of the arch along with the wheel and controlled using an Arduino in order to move the arch across the xylophone. Overall, two Arduinos would have been used, one to control the two wheels, and therefore the arch; and the other to receive the signals from the piezoelectric sensors and translate them into sounds. The pieces for the xylophone were laser cut and SolidWorks 2016 was used to model them.

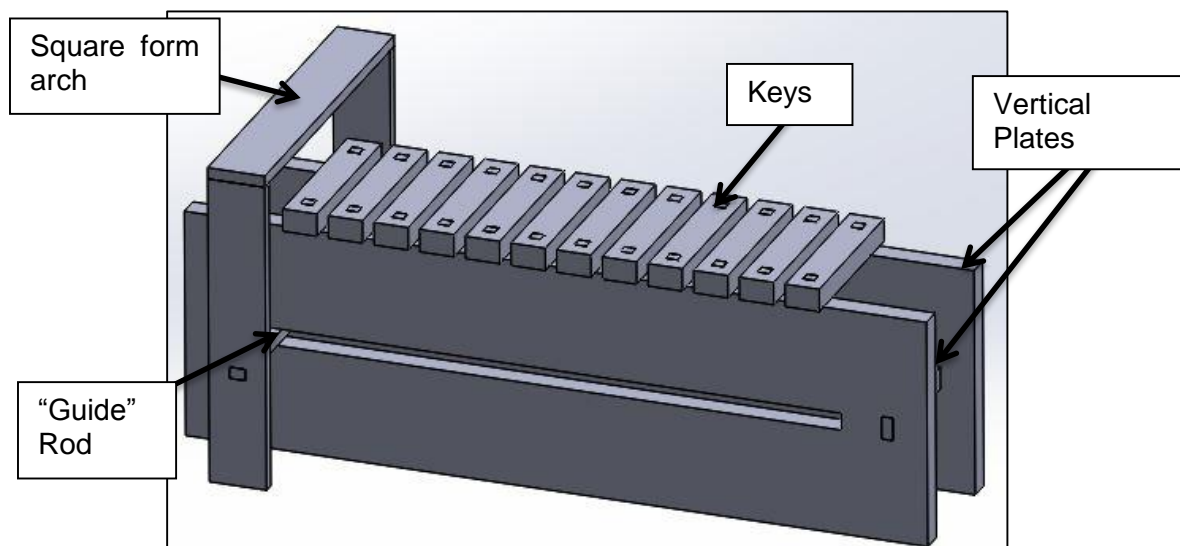


Figure 4.10. Model of Xylophone.

Model 2

After completing the first design of the xylophone, it was decided that the first model is not feasible within the given time frame as the team wanted to produce a prototype before the end of semester one.

For that reason, the design had to be simplified. The most significant change was the removal of the arch which was replaced by a Bosch bar that was placed across the instrument, and was supported by two threaded rods (at each end of the xylophone). Additionally, with the aid of threaded struts, 12 solenoids were mounted on the bottom side of the Bosch bar, each one assigned to a key (Figure 4.11), which will be programmed to hit the piezoelectric sensors according to the song being played. The last change was to modify the design of the key notes so that they could be similar to the ones found on an actual xylophone. In total, one Arduino will be used, which will control the number of times each solenoid will hit the piezoelectric sensor, according to the song being played.

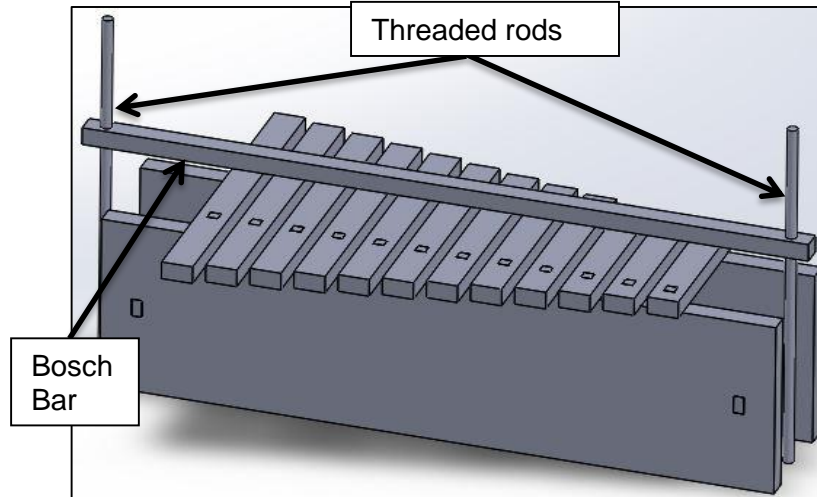


Figure 4.11. Model 2 of the xylophone.

Each solenoid which will be used, needs a 5 V DC power supply and requires a current of 1.1 A [60]; the aforementioned was calculated using the following equation:

$$P = V * I \quad (3)$$

so $5.5 \text{ W}[60] = 5 * I \rightarrow I = 1.1 \text{ A}$, where the power was taken from the datasheet.

However, the Arduino can only supply a maximum of 40 mA [35] and therefore the use of a Darlington transistor is necessary in order to supply the current through the solenoid. For this reason, a TIP120 NPN transistor was chosen, with a maximum base current of 120 mA. In order to calculate the required base current, the following formula was used found on the TIP120 datasheet [61]:

$$I_C = 250 * I_B \rightarrow I_B = \frac{I_C}{250} \quad (4)$$

where $I_C = 1.1 \text{ A}$, which is the current required by the solenoid; therefore, $I_B = 4.4 \text{ mA}$. In order to provide a base current of 4.4 mA, a resistor was placed between the Arduino and the transistor, with a value of 0.94 k Ω . The value of the resistor was calculated by using the formula:

$$R = \frac{V_{\text{arduino}} - V_{BE}}{I_B} = \frac{5 - 0.85}{4.4 * 10^{-3}} = 943 \Omega \quad (5)$$

A schematic diagram of a solenoid has been created and simulated using the NI Multisim software, which can be seen below in Figure 4.12. On the diagram, the voltage of the Arduino is labeled as "V1", the 0.94 k Ω resistor as "R1", the external DC voltage source as "V2" and the transistor as "Q1". Since the Multisim software does not have a representation of a solenoid, a resistor was used labeled as "R2" with a value of 4.55 Ω since it was assumed that the impedance of the solenoid is $\frac{5 \text{ V}}{1.1 \text{ A}} = 4.55 \Omega$. For simplicity, a free-wheeling diode was omitted from the simulation but will be included in the final design.

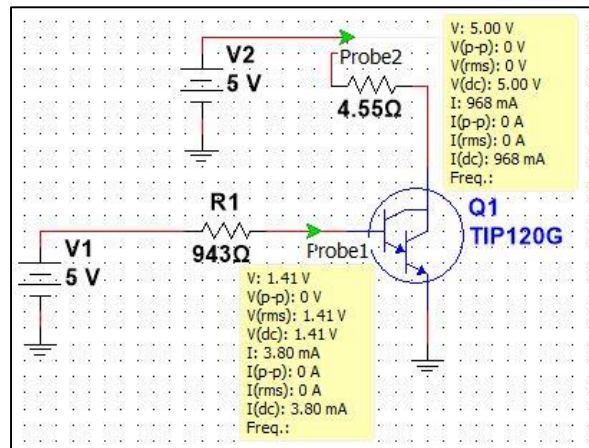


Figure 4.12. Solenoid Schematic.

At this point, all the components of the xylophone have been received from the mechanical workshop and have been assembled together. The remaining actions that need to be tackled are: to mount the solenoids to the Bosch bar and test them, to attach the piezoelectric sensors on the center of the keys and finally to align the solenoids and the piezoelectric sensors.

4.5. Panpipes

Between the recorder and the panpipes, the panpipes were the preferred choice, since they were more suitable to play the vocals. The video by TeamDare [21] shows one way of making a robot playing panpipes where the mechanism in the video shows the panpipes moving sideways and the nozzle only moving up and down. The decision for the panpipe design was to reverse the moving components and instead make the nozzle move sideways and the panpipes remain stationary. This simplified the design because the panpipes would require a larger and more sophisticated design to be moved whereas the nozzle is much smaller and lighter.

Below, Figure 4.13 shows the initial design for the panpipe setup. The panpipes are held up and fixed and the yellow nozzle constantly blows the air at the pipes with a small servo motor at the end which blocks the airflow to the pipes when needed. The nozzle is on a platform that moves sideways using a stepper motor operated track which will be controlled by a microcontroller such as an Arduino.

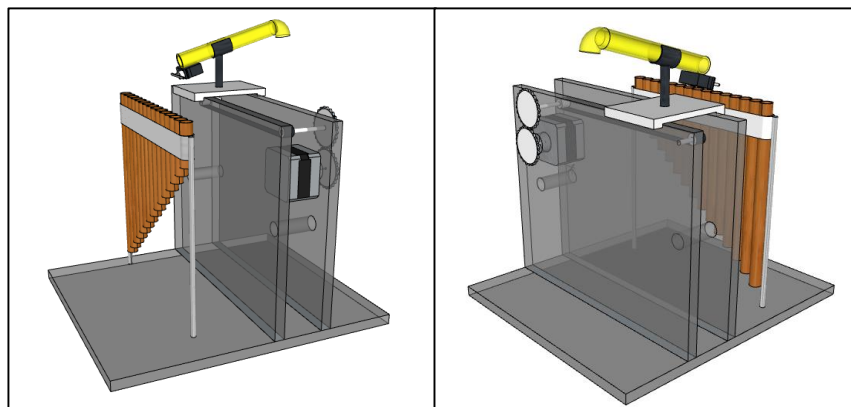


Figure 4.13. Initial 3D design for the panpipes front (left) and back (right) view.

This setup can be broken down into a list of the main components:

- **The panpipes** are 18 cm across with 15 pipes. They came tied together with string forming a slight curve. It is best if the curve is eliminated so that there is no variation in distance between the nozzle and avoiding a curved track. The plan to eliminate the curve is by untying the pipes and making a custom holder for the pipes.
- **The track** will resemble a treadmill and will have the platform with the nozzle attached on top of it. It must be long enough to allow the nozzle platform to go from the first to the last pipe. It was proposed that the track would be driven by a stepper motor which is controlled by an Arduino. This was based off of a previously designed guitar track shown in Appendix D.

- **The nozzle** is going to be mounted on top of the platform which is attached to the track. The air is provided by a mattress air pump which will be constantly on. To block the air in order to hit timed notes, a servo will be attached to the end of the nozzle with a small piece of plastic on the tip, rotating 90 degrees to stop the air and -90 degrees to allow the air to pass again.
- **The frame** will be made out of 5mm thick Perspex. It will house the track and the lift up the platform with the nozzle to the required height. The stepper motor will be attached from the side.

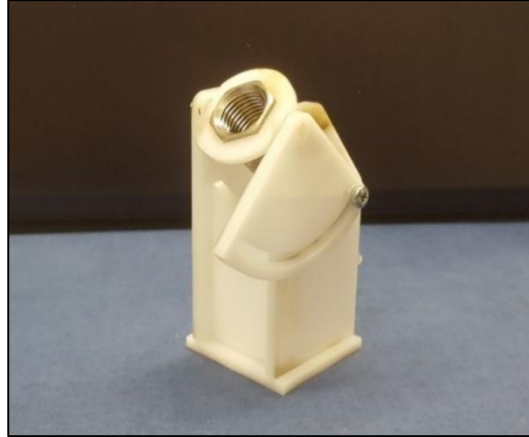


Figure 4.14. Nozzle which directs the airflow to the panpipes.

From testing and the video [21], it was found that the air needs to hit the tops of the panpipes at a certain angle to provide the best sound, so the mechanical workshop made a support for the nozzle that allows the angle to be adjusted as shown in Figure 4.14.

As this panpipe design has not been completed or tested yet, there will be variations in some components such as the spacing between each pipe of the panpipes, the height of the frame, the distance between the panpipes and the nozzle, and the type of stepper motor that controls the track.

A change to the design was proposed where the 15 pipes would be mounted in a circle around the nozzle and the nozzle would rotate in the middle using a servo motor. This would simplify the design by removing the moving platform setup which has a complicated design as well as the stepper motor, making the new setup smaller and requiring fewer components. The new design is showing in Figure 4.15 below.

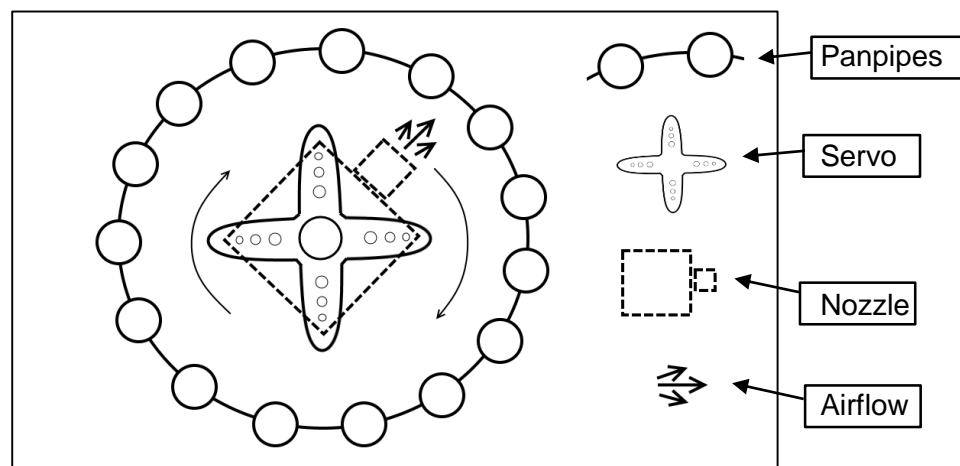


Figure 4.15. New 360-degree panpipe design.

The next step will be the testing of the panpipes sound with the air pump blowing the air. This will determine the feasibility of the panpipes as an instrument and will help determine which design will be easier to implement.

4.6. Tesla Coils

To explore the feasibility of using a Tesla coil as an instrument three Tesla coil kits were bought to test (shown in Figure 4.16). The turns ratio between the primary and secondary coil is 350, as the power supply is 18 V, the maximum voltage over the secondary coil is 6300 V. As the electrical breakdown of air is 3 kV/mm [28], streamers with lengths of $\frac{6300 \text{ V}}{3000 \text{ V/mm}} = 2.1 \text{ mm}$ can be expected from the unconnected end of the secondary coil.

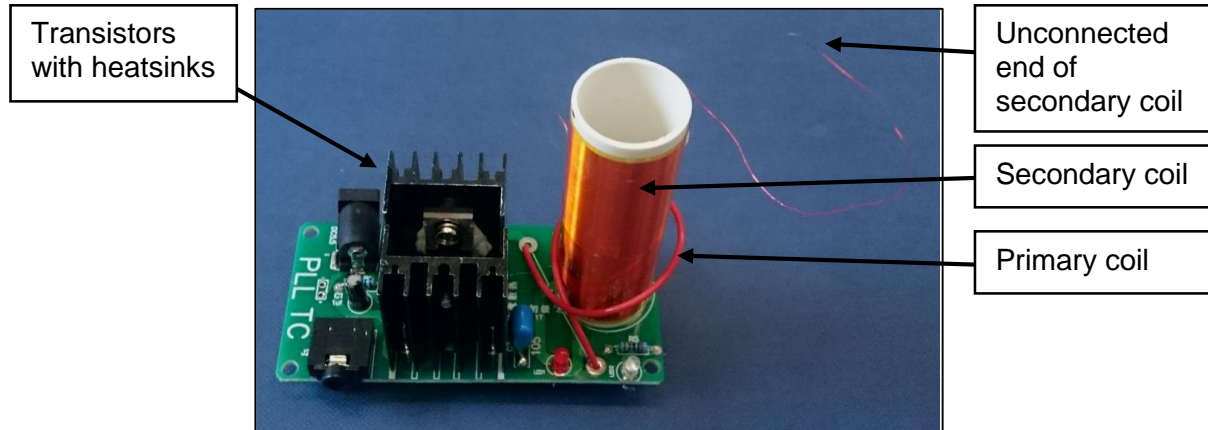


Figure 4.16. Built Tesla coil.

To play music through the Tesla coil a smartphone is connected using a 3.5 mm auxiliary input. The Tesla coil was able to produce reasonably high sound quality however, it was quiet so a microphone and speaker would have to be setup if a Tesla coil this size was being used. The volume could also be increased by making the Tesla coil larger (as shown in Figure 3.8 of Section 3.6 [29]) but the sound quality is lower as the interference from the streamers is higher.

To find out more about safety regarding Tesla coils a meeting was arranged with Dr. Vidyadhar Peesapati who is a researcher at the high voltage laboratory at the University of Manchester. The risk surrounding Tesla coils is low as although the voltage is high the current in the coil is very low and a cage could be placed around the coil to mitigate any risk. It could also be arranged to use the high voltage laboratory to test the Tesla coil during development and there would be an opportunity to use the large Tesla coil in the high voltage laboratory for the demonstration day. Currently, a decision is being made on whether to go forward with the panpipes or a Tesla coil. The deadline for this decision is 02/02/2018.

4.7. Stepper motors

4.7.1. Hardware

For producing the required musical notes, a decision has been made to use the NEMA 17 stepper motor. This is because in the early stages of the project, this type of motor has been bought for testing due to its low price and its specifications which allowed for easy interfacing with inexpensive driver boards (12 V rated voltage and 1.5 A rated current [62]). Further to testing, the motors were considered well suited for the required tasks and decided to keep them for the next prototyping stages.

In order to control the motor, the Allegro A4988 driver board was chosen as it was recommended by multiple sources and according to [63], it could operate with voltages between 8 V and 35 V while supplying a current of up to 2 A. The control algorithm was implemented on an Arduino Uno microcontroller board. Figure 4.17 presents the wiring diagram for the stepper motor control circuit.

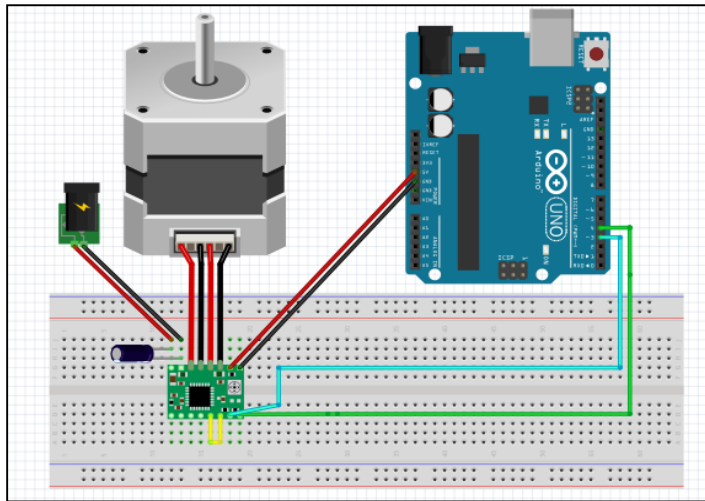


Figure 4.17. Stepper motor control circuit.

As the aim of controlling the motor was to produce various frequencies in the audible range, precise control of the rotation was not required.

As a result, the pins of the driver board that control the step resolution of the motor (full step, half step, etc.) have been left disconnected, leaving the motor to operate in the full step mode. Additionally, the STEP and DIR pins of the driver board have been connected to ports 3 and 4 of the Arduino.

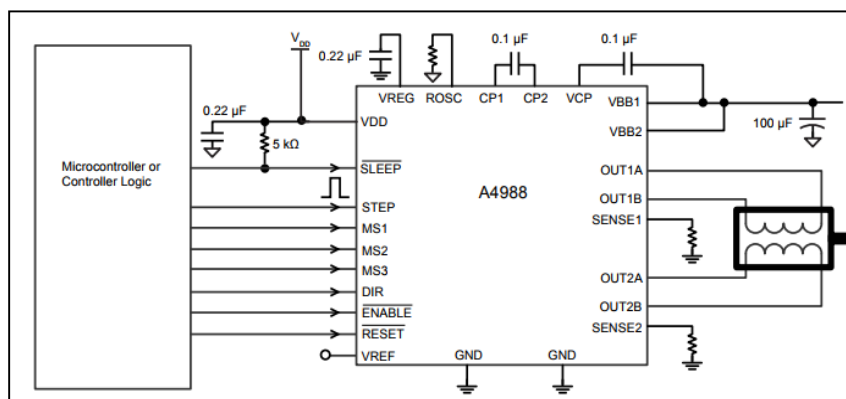


Figure 4.18. A4988 Pinout [63].

Figure 4.18 presents the A4988 pinout and its required connections. The STEP pin is used for controlling the movement of the motor with each pulse sent to this pin being converted to a step rotation and the DIR pin is used for controlling the direction in which the motor spins. Additionally, a decoupling 100 µF capacitor has been used as indicated in Figure 4.18 in order to protect the board against voltage irregularities. In setting up the control circuit, the guide presented in [24] has been used for ensuring the wiring was done properly so as not to damage the components.

4.7.2. Software

The stepper motors are driven by an ATmega328P Arduino microcontroller and an A4988 stepper driver. Outputting a square wave from the Arduino to the stepper driver causes the stepper to rotate. If the frequency of the square wave is matched to that of individual notes, then the sound produced by the stepper matches that of the note. By combining various notes together at different time intervals musical pieces can be played.

Timers

For example, the note A4 has a frequency of 440 Hz [64]. The period of a musical note can be calculated by using the following equation:

$$T = \frac{1}{f} \quad (6)$$

where T = period (in seconds) and f = frequency (in Hz).

In the case of A4, the period is approximately 2272 μs . In order to generate a square wave of this frequency, the signal can be driven to a logical high for half the period, followed by a logical low for the other half of the period, thus obtaining a PWM signal with a 50% duty cycle. The pseudocode for producing this square wave is presented in Appendix E.

The current version of the stepper motor circuit works by making use of the `millis()` and `delayMicroseconds()` functions as presented below. The prototype works well when controlling one or two motors, being able to play simple songs such as “Happy Birthday” or “Frere Jacques”. However, timing errors could become noticeable when increasing the number of motors as the code executes sequentially. A maximum deviation of 10 Hz has been detected which may not be obvious in higher octaves (higher frequencies) but can pose a risk for notes in lower octaves (lower frequencies). Depending on the feasibility of successfully using more motors in the same way, the future circuit could make use of dedicated ICs to generate the required frequencies.

The Arduino Uno microcontroller has a series of pre-defined libraries that are useful for creating accurate delays and implementing timed actions on its GPIO (general purpose input/output) pins. If no additional IC is to be used, the functions `delayMicroseconds()` and `millis()` are particularly useful in implementing these timed actions. In the case of the stepper motors, the function `delayMicroseconds()` is used twice, once after driving the output pin high and once after driving it low, generating thus a square wave with the period determined by the function’s parameter. Additionally, to play each note for a specific amount of time, the function `millis()` is used to calculate the elapsed time at one instance in the code execution and based on its returning value, decide whether to keep playing the same note or switch to the next note to be played. However, this method has some limitations. If multiple stepper motors are to play a song simultaneously, making use of the `delayMicroseconds()` function might result in a noticeable delay in the song as the execution of the entire code is paused when this function runs. Additionally, using the `millis()` function eliminates the option of using the PWM to drive the stepper motors as the function required for using the PWM pins, `analogWrite()`, can make the `millis()` function imprecise and thus, lead to a timing error [65].

555 Timer

The LM555 timer is an IC capable of generating specific frequencies at various duty ratios by connecting two resistors and two capacitors [66]. Multiple ICs would be used, one for each note and since the IC has an enable pin, the Arduino would be able to turn it on/off with ease. This is a good solution as a lot of the software is now simplified due to the hardware used. However, more hardware is needed making it more expensive to produce. A closer investigation on the components needed to produce some of the frequencies resulted in resistor values in the microohm magnitude. This poses two issues, first, it is hard to procure micro-ohm resistors and most importantly the datasheet recommends resistor values of 1 K Ω to 1 M Ω [66]. This method has therefore been ruled out.

Digital Signal Synthesizer

A Digital Signal Synthesizer (DSS) is an IC capable of generating a vast range of frequencies. The DSS considered for this application is the AD9837. The AD9837 is capable of producing frequencies between 0-5 MHz, with a 0.02 Hz resolution making them incredibly accurate [67]. The IC requires no external components (apart from a decoupling capacitor) and the frequency is set by three internal registers. The IC communicates with the microcontroller via SPI [67]. This is advantageous as it can be easily reprogrammed to another frequency, allowing one IC to play multiple notes. As expected the IC is more expensive than the 555 timer. The AD9837 offers a promising solution, and will most probably be used. The breakout board for the AD9837 has been designed and further testing is needed, see Figure 1.1 in Appendix A.

4.8. Guitar

This section was included in the report to show work on the guitar which had been carried out during Semester 1. However, after consideration it was decided not to include the guitar in the robotic orchestra due to the complexity of the design and the limited time available.

Six servos would be placed above the strings along the bottom end of the guitar, where the player’s fingers would normally strike the strings, this would allow the servos to mimic the motion of the hand playing the guitar.

Furthermore, six rods would be placed along the neck of the guitar capable of moving along it. Each rod would be controlled by a stepper motor allowing for accurate control over the rod's position. The rods would permanently be in contact with the strings, thereby removing the need of an additional system required to lower the rods to press the strings. The aforementioned system would be controlled by an Arduino microcontroller. Figure 10.1 in Appendix J shows the design of the guitar.

5. Progress and Planning

Since this project has a lot of different components an effective and detailed plan was crucial to be successful. The primary method for organising this project has been using a Gantt chart and conducting team progress reviews every 3 weeks.

One of the things that had to be avoided when organising the schedule for this project is creating all of the instruments and the conductors in a serial manner, i.e. finish one and then create the next. The total allocated time to the project is 17 weeks (85 days, not including weekends) so it crucial to have the instruments being built in parallel to one another. This ensures that if there is a delay in one of the instruments (i.e. some components are taking longer than expected to be delivered) then progress can be made on a different instrument, ensuring that team members are not being underutilised.

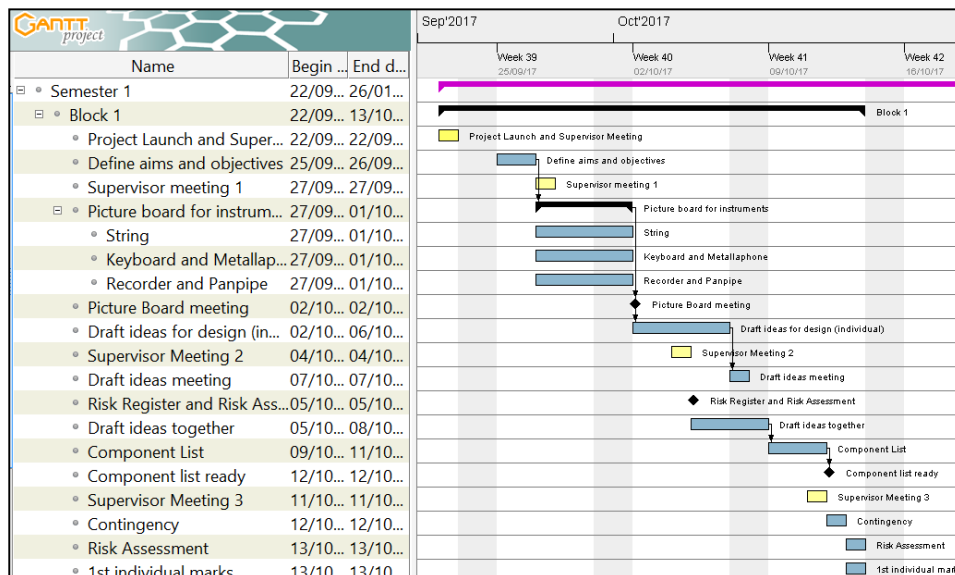


Figure 5.1. Initial version of the Gantt chart for Project Block 1.

Figure 5.1 shows the initial plan for Project Block 1. The initial stages of the project were open ended so one of the first tasks was to define the aims and objectives of the project, this was followed by defining the different categories of instrument to be designed during this project which helped in narrowing down the criteria for the research period. The research period started on the 27/09/17 through to 01/10/17 (5 days) and culminated in a picture board meeting, which helped in defining the initial four robot instruments that were going to be built: guitar, panpipes, keyboard and xylophone with two backup instruments in the Tesla coil and stepper motors. Deciding on the instruments to be made was a significant step in the project because it now meant more definitive milestones could be set and a more detailed plan could be created to ensure progress.

A milestone was created for the end of the first project block which was to have a *Component list ready* (Figure 5.1). This meant that all necessary electronic components and hardware for the first prototypes of the instruments could be ordered to be worked on in the second project block. Initially, this was going to be the only goal for the end of the first project block. However, specifically after *Supervisor Meeting 2*, it was realised that the instrument designs that were a product of the picture board meeting required many individual steps to be achieved and that starting with simpler designs would be beneficial to the project, these designs could then be adapted to include the additional features from the initial designs. In addition to this, having simpler designs made the *Construct the 4 instruments* (Section 2.2.2) objective more achievable.

The robot guitar was also dropped in favour of a significantly simpler stepper motor design, primarily because the initial designs produced were too complex to be completed within the given timeframe. The panpipe design is currently under debate as to whether it will be progressed, it may be replaced with the Tesla coil. This will be determined after the testing period with the panpipes is over and the feasibility of the robot within the given time frame can be better understood.

In addition to this, one of the objectives was to *Select 2 suitable songs [...]* (Section 2.2.2). It was realised that designing the hardware without thought towards the songs that they are being designed to play will make the designs more complicated. Deciding on the songs in the early stages of the project ensured that the designs can be made as simple as possible.

Several new tasks and milestones were added to tackle the issues that were highlighted during the second supervisor meeting, these can be seen in Appendix F. It was decided that the second week should end with both the songs and hardware choices being made meaning that the hardware could be designed with specific song choices (and therefore notes) in mind. This made the design for some of the robots simpler, for example the keyboard was designed to only play 19 different notes instead of the 61 it was capable of. To decide on the song choices a Facebook poll was created where individual team members could vote on different choices, since there are only six members it was easy to get the votes from each member and the poll only had to be live for a day. The advantage of using a Facebook poll for this is that it allows individual members to add their own suggestions as well as easily tracking the number of votes each song achieves. The next day a meeting was held to discuss the most voted song choices and integrate them with the MIDI files of the songs using Anvil Studio. In order to make sure that the *Component list ready* milestone could still be completed on time the songs and instruments needed to be selected by the end of the second week. This is because, in order to create a component list, 3D models were needed which could take a significant amount of time to develop. The milestone for the first project block was achieved and a component list was constructed and ordered which gave 3 weeks for the components to arrive, ready to be used at the start of the next project block.

During the first module block the priorities for the group switched from making significant progress on the project to focusing on individual modules. The work done on the project during this time focused mostly on preparing for the upcoming tasks of the second project block and preparing for the first significant project deadline – *Interim Report*. It was clear that in order to achieve this milestone a significant amount of work would need to be executed. So, in order to pre-empt the workload, and ensure that no time was wasted planning during the second project block, several meetings were held where the approach to achieving the deadline for the interim report was discussed. The first meeting (*Interim Report Meeting – Contents Page*) was used to create a contents page for the report which was used in the next meeting (*Designate Interim Report Roles*) to designate individual sections to individual team members. There was another meeting (*Interim Meeting 3*) which took place two days before the start of the second block. This was deliberately close to the start as it was used to discuss the plan and milestones for the second project block.

The second project block was more complex to schedule because there were more milestones and they often required the same people working on them. The milestones for the end of the second project block were:

- Software for stepper motors created
- Keyboard software simulation created
- Construct the hardware for the xylophone
- Keyboard hardware submitted for manufacture
- Submit the First Interim Report

To make sure that these milestones were achieved several of the tasks were run in parallel. Also, in project block two the team was split in to a hardware team (four members) and software team (2 members). The slight bias towards the hardware team is due to the higher amount of work to be completed on the design and construction of the robot instruments. The reason for choosing the stepper motor software to be created first is because of the low hardware requirements that the stepper motors had, the circuits were quite simple and one could be connected to the Arduino with relative ease. The software development could start from early on in the project block.

Fortunately, all of the components that were ordered in the first project block arrived on time so the drive boards and power supply necessary for the steppers to operate were ready to be used by the software team. Several of the components had been delivered for the keyboard and xylophone so a testing phase began. Each of the testing phases was started with a goal in mind. For example, the *Solenoid Testing for Keyboard* task (Figure 5.2) was created to check whether the solenoids that had been ordered in the first project block were strong enough to push down the keys, if they were not then completing this phase would ensure there was enough time to order more in and still make progress on the keyboard hardware.

Highlighted in blue in Figure 5.2 is the approach towards the interim report. A draft of the report was to be made ready by the start of the last week of the block, 20/11/17 (a week before the deadline). This ensured that there was enough time to conduct a table read and perform any edits. It was debated whether a week may be too long and that the time may be better spent working on the technical progress of the project and having more time to write the report, pushing the deadline for the draft to Wednesday (22/11/17). Due to the size of the report and the amount of time it would take to conduct a table read the initial date was settled on and remained as the deadline for the first draft. This approach worked well, the interim report was submitted on time and there was ample time to edit the document and have it ready. Spacing the editing out over the course of a week resulted in the team still being able to meet the deadlines for modules outside of the project.

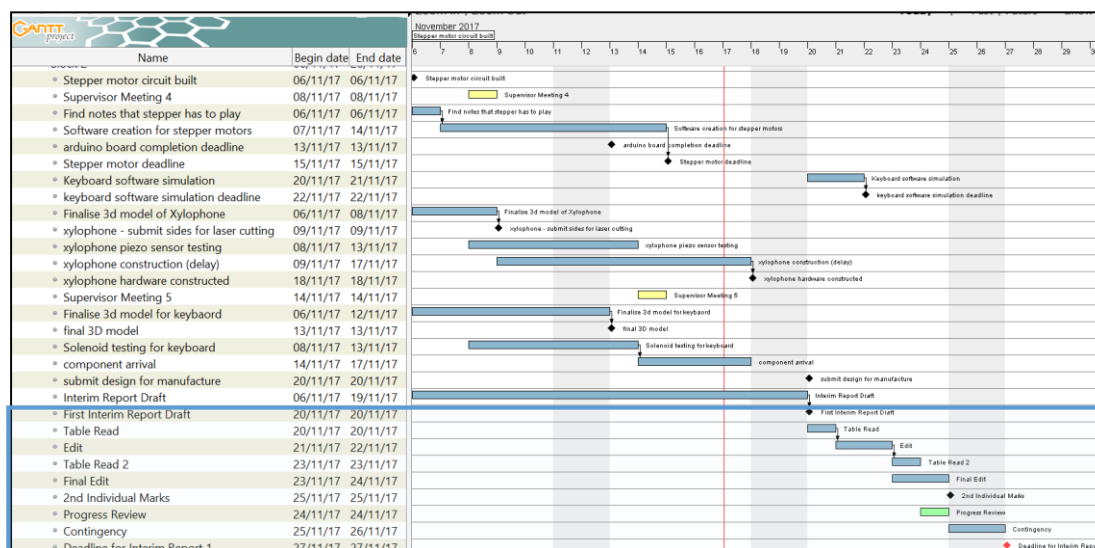


Figure 5.2. Gantt chart for Second Project Block.

However, the *Keyboard software simulation* milestone (due for 20/11/17) was not completed on time. This was partially due to the interim report taking priority over other workload during the last week and due to the size of the sub teams. In order to address the latter, in the second semester the software and hardware team will contain three members each. This will provide more resources to the software team so that two instruments can be worked on at the same time, instead of one.

5.1. Approach towards 2nd Semester

As discussed in Section 5, both the hardware and software teams will be evened out in the second semester which should help to achieve both the hardware and software goals in the upcoming project blocks. Below is a summary of the remaining tasks for each of the instruments. The Gantt charts for the blocks in semester 2 can be seen in Appendix G.

Keyboard: The software simulation for the keyboard will take significant amount of time to complete so will form the main part of the third project block. It is worth spending a significant amount of time on this section because both the Xylophone and Keyboard will use very similar software – so building the keyboard software with this in mind will reduce the amount of time spent on the Xylophone. The Keyboard simulation software will start development at the beginning of the second week of project block 3. It is currently scheduled to be completed by 12/02/18 (week three of the third project block). By this time the hardware for the keyboard will also be constructed so the remainder of the block will be spent on integrating the software simulation with the hardware.

Xylophone: The hardware for the Xylophone has now been delivered and constructed. The next stage would be to write the software and integrate it with the hardware. As mentioned in Section 4.4 the hardware was given a new design which was significantly simpler than the previous design. This also has the added benefit of the software between the Xylophone and Keyboard now being very similar in structure. Because of this, finalising the Xylophone should not be a very long task once the Keyboard instrument is completed. Adapting the software of the Keyboard to suit the needs of the Xylophone hardware will take place at the start of the fourth project block.

Panpipes and Tesla Coil: The plug for the air pump used on the panpipe was delayed by a week which has delayed the testing to the start of the next project block. The Tesla Coil currently looks like it would make a good addition to the orchestra, however more research needs to be conducted in order to determine how the Tesla Coil will be scaled up, since the volume of the off-the-shelf kit purchased was not high enough. In Figure 7.1 of Appendix G, there are tasks to perform some preliminary tests on the panpipes with the air pump and do some research on scaling the Tesla Coil up to operate at a higher voltage. These are scheduled to be completed partway through the second week of the third project block (07/02/18), by the end of the third project a block a decision will be made whether to move forward with the Tesla coil or the robot panpipes.

Stepper Motors: The current software iteration of the software allows for one stepper motor to be controlled and play a song. However, the initial design involved having four stepper motors synchronised with each other. This is now being explored by researching various timers. Before the start of the third project block a decision will be made on what method will be best suited for achieving multiple stepper motors at once. The beginning of the third project block (30/01/18 – 06/02/18) has been dedicated towards achieving the *Multiple Stepper Motor Software Deadline* (06/02/18). It was found that the noise from the steppers was amplified when fixed on to a wood surface. To capitalise on this, a wooden speaker box will be designed in the first week of the third project block and be fixed to the stepper in the third week. The entire stepper motor instrument has a completion date of 14/02/18 (third week of project block three).

Conductor Design: The main task to be completed with the conductor is to decide the method of communication between the Raspberry Pi and the four robot instruments. A deadline has been set for the end of the third project block (15/02/18) to make this decision. Once this has been a step-by-step will be created to create the conductor by the end of the fourth project block. By the end of the first week in the fourth project block the hardware for the instruments should be complete, this means that more resources can be applied to designing and constructing the conductor.

5.2. Risk Management

In order to identify the potential risks that project faces a risk register was established, which allowed some mitigations to be put in place. The full risk register can be seen in Appendix H.

Risks 1, 5 and 9 (Appendix H) are not from direct action of the team and have to be mitigated. One of the ways in doing this is to add contingency time at the end of each block. From Figure 5.2 and Appendices F and G, it can be seen that this was done, giving two days at the end of each block for tasks to overrun. During the second project block the testing phase for the panpipes was supposed to begin at the start of the second week, so that the feasibility of the panpipes could be determined. However, since the plug that was needed to power the air pump was delayed until the end of the second week the testing phase was delayed.

The contingency time attached to the end of the second project block had to be used to compensate for the delay, giving two additional days to work on the testing for the panpipe. Risk 5 should not be much of an issue (shown by its *Probability being equated to 1*) since a locked cupboard has been provided, all built equipment shall be stored in there and the key is removed from the room.

Risk 2 (Appendix H) has the potential to cause significant setbacks in the project, especially since the team has been split into two teams which have a focus on either software or hardware. However, the risk is somewhat reduced by having several members in each team and more than one person on each task. This ensures that more than one person is familiar with each task that is currently ongoing with the project so if someone was to become unavailable then another member will be able to pick up from the last member that left it. Also, the team has weekly meetings where the whole team is present (this is in addition to the supervisor meetings).

The primary goal of these meetings is to check on progress made in the previous week but they also allow members to update each other on specific tasks. This means that if another member has to pick up the work of another there is only, at the maximum, a five-day gap in the workload.

Risk 4 is reduced by using Google Drive. Since Google Drive is a completely online file storage system every file can be accessed by each individual member. This acts as a good back up to the local files stored on individual laptops.

Risk 7 is due to none of the team members having previously used a MyRIO (Table 2.1). In order to mitigate this risk two actions were taken. Firstly, the National Instruments Student Scholarship was applied for, where one of the benefits is being provided with a National Instruments Engineer. Then a training session was organised with a National Instruments Engineer to provide an introduction to the MyRIO and to demonstrate its capability. In order to further reduce this risk all members of the team will be present at the training session so that everyone can be assumed to be at the same level for the MyRIO after. This ensures that individual team members can provide support to others should there be problems using it.

Risk 8 is mitigated by constructing components lists before ordering a significant amount of components (such as at the end of the first project block). This gives the opportunity to review the items being ordered and ensure that they are within budget, it also gives time to look for alternatives to the products ordered. Some may be cheaper if factors such as delivery time are more flexible. For example, at the end of the first project block there was a three-week period that the items could be delivered in so cheaper alternatives for the solenoids could be selected. These were exactly the same solenoids that were going to be ordered but because they had a longer delivery time the total cost per solenoid was cheaper.

Risk 10 is in regards to only one member of the team having skills in one particular element of the project. For example, only one member of the team has previous experience in using the Raspberry Pi, which means that should a person become unavailable it may be hard to keep progressing on any tasks relating to the RPi. In order to mitigate this, any task involving the RPi will involve more than one member of the team, ensuring that each member is able to use the RPi for this project.

5.3. Minutes and agendas

A project meeting is held each week with the team and project supervisors Danielle George and William McGenn. For the meeting an agenda is produced by the Secretary and is distributed a day before the meeting. Before it is sent out it is reviewed by the team to make sure everyone's work for that week has been included so all progress made can be presented effectively.

Minutes are taken during the meeting which are typed up and uploaded to the shared drive within two days so the team can look at the actions required for that week. The minutes include an ongoing actions list, a section detailing actions from the previous week (confirming if they had been completed or not) and a section containing a summary of the topics discussed detailing the actions that the meeting produced.

5.4. Logistics

WhatsApp was chosen as the communication platform and two groups were set up with one with the team and one with the team and supervisors. Meetings were arranged either using the WhatsApp group or at the end of a previous meeting. The storage platform that is to be used for sharing and saving document was chosen to be google drive as all members of the team had access to it.

The project group was split up into two with a hardware side and a software side. In the hardware group was Joyanto, Josh, Anton and Theodore and on the software side was Andrei and Francesco. Joyanto as initially in the software group but it was realised that initially more work on hardware was needed to get the robots built ready for programming.

5.5. Procurement

The team has an allocated budget of £1500 and so far, £506 have been spent on musical instruments, electronic components and building materials. The full expenditure record is presented in Appendix I. Although this represents approximately a third of the budget, it is worth mentioning that a substantial amount was spent in the beginning on buying the musical instruments (keyboard, stepper motors, etc.) and the components required to set them up (ex: power sources). These expenses accounted for approximately £250 and a key aspect to be taken into account is that these were nonrecurring expenses, thus reducing the risk of a future budget exceeding.

The current financial plan leaves £200 to be used as a reserve for unexpected expenses that accounts for possible component failures or improper equipment being bought during the prototyping stages, leaving a current usable balance of approximately £800. In terms of future expenses, the biggest pressure on the budget is posed by the solenoids used for pushing the keys of the keyboard as these are expensive components and a large number is required for the project. A solenoid costs approximately £10 per piece and it is estimated that 25 will have to be bought in the next stages, thus running a cost of £250. To minimise this risk, single solenoids will be tested in advance to ensure that the large order will meet the project requirements. Another large expected expense is that of the building materials necessary for constructing the mechanical ensemble of the robots, the cost being estimated at £200. Considering the available balance, the cash reserve and the future large expected expenditures, approximately £350 are left to be used for electronic components. As these have a generally low price, the project is expected to be finished within the allocated budget.

5.6. Conclusion

This report outlines the progress that has been made towards constructing a new core the Manchester Robot Orchestra. Currently three instruments are confirmed to contribute to the four with the last being decided. The stepper motor design involves having four stepper motors synchronised together to play MIDI files. They spin at specific frequencies to mimic notes being played. The keyboard utilises a Bosch bar held up by threaded rods, a row of solenoids will be fitted in to the bar and will be used to press down on the keys. The xylophone utilises a similar design to the keyboard, some solenoids will be mounted on to a Bosch bar which is supported by two threaded rods. The solenoids will hit piezoelectric sensors which will trigger a sound to be played from the speaker. This is quite a unique instrument because the sound being played can be changed according to the song as well as the type of instrument being played.

The fourth instrument is being decided between a Tesla coil and robot panpipes. The panpipes are still in their preliminary stage; the testing phase has been delayed due to a plug for the air pump arriving late. This has since arrived so the testing phase will begin in the third project block. The Tesla coil is a recent addition and is currently being researched to better understand the feasibility of making it one of the four instruments. Once the testing phase for the panpipes is complete and the design for the Tesla coil is researched further, a decision will be made by the end of the second week of the third project block regarding which design will be progressed.

The third project block should see the completion of two of the instruments – the keyboard and the stepper motors. The xylophone should be completed shortly after in the fourth project block since it shares similar software traits to the keyboard. The construction for the conductor will begin in the third project block, a decision will be made regarding the type of communication that is needed between the conductor and instruments.

6. References

- [1] The University of Manchester, "European City of Science | The University of Manchester | Science and Engineering," The University of Manchester, May 2016. [Online]. Available: <http://www.se.manchester.ac.uk/about-us/ecos/> [Accessed November 2017].
- [2] BBC, "BBC Radio 1 - Can a robot replace Ed Sheeran?," BBC, 3 March 2017. [Online]. Available: <http://www.bbc.co.uk/programmes/p04sxvgw> [Accessed October 2017].
- [3] L. Mather, "Glock o Bot," YouTube, 13 June 2016. [Online]. Available: <https://www.youtube.com/watch?v=4toW8Tc9GPI> [Accessed October 2017].
- [4] National Instruments, "Student Project Sponsorship - National Instruments United Kingdom," National Instruments, 12 August 2017. [Online]. Available: <http://uk.ni.com/studentponsorship> [Accessed November 2017].
- [5] J. Shauw, "Dream wedding played by piano robot," YouTube, 7 November 2010. [Online]. Available: <https://www.youtube.com/watch?v=lwuiOwZrzCM> [Accessed October 2017].
- [6] Teotronica, "robot playing piano," YouTube, 14 September 2009. [Online]. Available: https://www.youtube.com/watch?v=HHcgTbs7_Os [Accessed October 2017].
- [7] Bowles, E.A. and Marcuse, S. (2017). *Percussion Instrument*. [online] Available at: <https://www.britannica.com/art/percussion-instrument> [Accessed November 2017].
- [8] The editors of Encyclopedia Britannica. (2012). *Idiophone*. [online] Available at: <https://www.britannica.com/art/idiophone> [Accessed November 2017].
- [9] Estrella, E. (2017). *What is a Chromatic Scale?* [online] Available at: <https://www.thoughtco.com/what-is-a-chromatic-scale-2456561> [Accessed November 2017].
- [10] Jenna deBoisblanc, "Arduino: how to build an electronic MIDI xylophone", 3 May 2011. [Online]. Available: <https://www.youtube.com/watch?v=92VIEDtQKVI> [Accessed November 2017].
- [11] Deboisblanc, J.(n.d.). *Xylophone*. [online] Available at: <http://jdeboi.com/xylophone/> [Accessed November 2017].
- [12] Cambridge Dictionary. (n.d.). *String Instrument*. [online] Available at: <https://dictionary.cambridge.org/dictionary/english/string-instrument> [Accessed November 2017].
- [13] Halfpenny, E. and Grame, T. (2017). *Stringed Instrument*. [online] Available at: <https://www.britannica.com/art/stringed-instrument> [Accessed November 2017].
- [14] Learnitonlinetoday, "How to Play Trumpet- Lesson #1 Beginner," Learnitonlinetoday, 9 January 2010. [Online]. Available: <https://www.youtube.com/watch?v=2SjH0qhQpoU> [Accessed November 2017].
- [15] WikiHow, "wikiHow to Play the Trumpet," WikiHow, [Online]. Available: <https://www.wikihow.com/Play-the-Trumpet> [Accessed November 2017].
- [16] A. Lim, "Toyota Trumpet Robot (Subtitles in English and Japanese)," 4 January 2010. [Online]. Available: <https://www.youtube.com/watch?v=6fctULDctuA&t=107s> [Accessed November 2017].
- [17] BotJunkie, "Waseda Flutist Robot," 1 November 2008. [Online]. Available: <https://www.youtube.com/watch?v=jx8U1FgILCE> [Accessed November 2017].
- [18] WikiHow, "wikiHow to Play the Panpipe or Pan Flute," [Online]. Available: <https://www.wikihow.com/Play-the-Panpipe-or-Pan-Flute> [Accessed November 2017].

- [19] "how to play the recorder," arta recorder, [Online]. Available: <http://www.arta-recorder.org/> [Accessed November 2017].
- [20] Cu cselab, " recorder12_1.wmv", Computer Engineering CUHK, 5 Mar 2012. [Online]. Available: <https://www.youtube.com/watch?v=Bi6ShZp7poM> [Accessed November 2017].
- [21] T. Dare, "TeamDARE's robot band plays 'Rolling in the deep'," TeamDare, 27 August 2011. [Online]. Available: <https://www.youtube.com/watch?v=RKpKyqGX2Nk> [Accessed November 2017].
- [22] O. Engineering, "What is a stepper motor? - Principles, types and controllers", *Omega.co.uk*, 2017. [Online]. Available: https://www.omega.co.uk/prodinfo/stepper_motors.html [Accessed November 2017].
- [23] "Stepper Motors | NEMA Stepper Motors & Controllers", *Circuitspecialists.com*, 2017. [Online]. Available: <https://www.circuitspecialists.com/stepper-motor> [Accessed November 2017].
- [24] Stepper Motor - How It Works – "HowToMechatronics", *HowToMechatronics*, 2017. [Online]. Available: <http://howtomechatronics.com/how-it-works/electrical-engineering/stepper-motor/> [Accessed November 2017].
- [25] "How do stepper motors work?", *Explain that Stuff*, 2017. [Online]. Available: <http://www.explainthatstuff.com/how-stepper-motors-work.html> [Accessed November 2017].
- [26] M. Pilhofer and H. Day, *Music theory for dummies*.
- [27] TheHomebrewGuru, "Build and Code a Musical Tesla Coil With a Microcontroller," instructables, [Online]. Available: <http://www.instructables.com/id/Build-a-Musical-Tesla-Coil-like-a-Pro/> [Accessed November 2017].
- [28] Admin, "Breakdown voltage/Dielectric strength (Breakdown of Insulator)," *The Physics Fact book*, 15 September 2014. [Online]. Available: <https://electronicspani.com/breakdown-voltagedielectric-strength/> [Accessed November 2017].
- [29] E. Goodchild, "'In the Hall of the Mountain King" - Played on Musical Tesla Coils," Youtube, [Online]. Available: https://www.youtube.com/watch?v=8LAhKkPUo_A [Accessed November 2017].
- [30] "The musical tesla coils," physics central- physics buzz blog, [Online]. Available: <http://physicsbuzz.physicscentral.com/2007/11/musical-tesla-coils.html> [Accessed November 2017].
- [31] "www.raspberrypi.org," Raspberry Pi Foundation, [Online]. Available: <https://www.raspberrypi.org/downloads/raspbian/> [Accessed November 2017].
- [32] "Raspberry Pi Foundation," [Online]. Available: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/> [Accessed November 2017].
- [33] "Sparkfun.com," [Online]. Available: <https://learn.sparkfun.com/tutorials/what-is-an-arduino> [Accessed November 2017].
- [34] "Arduino.com," [Online]. Available: <https://store.arduino.cc/usa/arduino-uno-rev3> [Accessed November 2017].
- [35] "Atmel," [Online]. Available: http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf [Accessed November 2017].
- [36] "Atmel," [Online]. Available: http://www.atmel.com/Images/Atmel-2521-AVR-Hardware-Design-Considerations_ApplicationNote_AVR042.pdf [Accessed November 2017].

- [37] "Arduino," [Online]. Available: <https://www.arduino.cc/en/uploads/Main/ArduinoNano30Schematic.pdf> [Accessed November 2017].
- [38] "ftdichip.com," 2015. [Online]. Available: http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf [Accessed November 2017].
- [39] "National Instrument," [Online]. Available: <https://www.ni.com/en-gb/shop/select/myrio-student-embedded-device> [Accessed November 2017].
- [40] "Nordic Semiconductors," [Online]. Available: <https://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P> [Accessed November 2017].
- [41] "midi.org," MIDI Organisation, [Online]. Available: <https://www.midi.org/specifications/item/the-midi-1-0-specification> [Accessed November 2017].
- [42] J. Gibson, "Indiana.edu," Indiana University, 2013. [Online]. Available: <http://www.indiana.edu/~emusic/361/midi.htm> [Accessed November 2017].
- [43] T. M. Association, "midi.org," MIDI Organisation, [Online]. Available: <https://www.midi.org/articles/an-intro-to-midi> [Accessed November 2017].
- [44] S. Richard, "Petesqbsite," [Online]. Available: <http://www.petesqbsite.com/sections/express/issue18/midifilespart1.html> [Accessed November 2017].
- [45] "CSIE.ntu.edu.tw," [Online]. Available: <https://www.csie.ntu.edu.tw/~r92092/ref/midi/> [Accessed November 2017].
- [46] "electronics.dit.ie," [Online]. Available: http://www.electronics.dit.ie/staff/tscarff/Music_technology/midi/midi_note_numbers_for_octaves.htm [Accessed November 2017].
- [47] C. O. 2. Jarrod Hart (Los Olivos, "Musical Notes Explained Simply," theprovincialscientist, 25 October 2011. [Online]. Available: <http://theprovincialscientist.com/?p=1576> [Accessed November 2017].
- [48] "MIDI numbering, the Helmholtz Pitch Notation System,," theoretically correct, [Online]. Available: <http://www.theoreticallycorrect.com/Helmholtz-Pitch-Numbering/>. [Accessed November 2017].
- [49] "Anvil Studio," Willow Software, [Online]. Available: <https://www.anvilstudio.com/> [Accessed November 2017].
- [50] A. Sweigart, in AUTOMATE THE BORING STUFF WITH PYTHON, San Francisco, No Starch Press, 2015.
- [51] O. M. Bjørndalen, "mido.readthedocs.io," [Online]. Available: <https://mido.readthedocs.io/en/latest/> [Accessed November 2017].
- [52] Society of Robots, "Actuators - Solenoids," Soceity of Robots, 9 August 2006. [Online]. Available: http://www.societyofrobots.com/actuators_solenoids.shtml [Accessed October 2017].
- [53] BICRON Electronics Company, "Standard & Customer Solenoids for OEM Application," BICRON Electronics Company, Connecticut, 2011. Available: http://www.bicronusa.com/pdfs/BICRON_DC_Solenoids.pdf [Accessed October 2017].

- [54] H. A. Radi and J. O. Rasmussen, Principles of Physics For Scientists and Engineers, Berkley, CA, USA: Springer, Berlin, Heidelberg, 2013.
- [55] BICRON Electronics Company, "Standard & Customer Solenoids for OEM Application," BICRON, 2011. [Online]. Available: http://www.bicronusa.com/pdfs/BICRON_DC_Solenoids.pdf [Accessed October 2017].
- [56] multcomp, "MCSMO-0630S12STD Technical Data Sheet," Farnell, Leeds, 2011.
- [57] R. Components, "3842992888/3000 Bosch Rexroth Aluminium Strut 20 x 20 mm, 6mm Groove , 3000mm L," Rexroth Bosch Group, Germany.
- [58] R. Components, "3842523135 Bosch Rexroth Strut Profile T-Slot Nut , M4 Thread strut profile 20 mm, Groove Size 6mm," RS Components, Lohr Am Main.
- [59] Burris, M. (2017). Stepper Motor vs. Servo Motors- Selecting a Motor. [online] Available at: <https://www.lifewire.com/stepper-motor-vs-servo-motors-selecting-a-motor-818841> [Accessed November 2017].
- [60] Shenzhen Zonhen Electric Appliances. (n.d.). Specification. [online]. Available at: http://www.robotshop.com/media/files/pdf2/zho-0420s-05a4.5_specification.pdf.
- [61] Semiconductor Components Industries. (2014). Plastic Medium- Power Complementary Silicon Transistors. [online] Available at: <https://www.onsemi.com/pub/Collateral/TIP120-D.PDF> [Accessed November 2017].
- [62] "NEMA17 Stepper Motor - 34mm", SPOOL3D, 2017. [Online]. Available: <https://spool3d.ca/nema17-stepper-motor-34mm/> [Accessed November 2017].
- [63] Allegro MicroSystems, "DMOS Microstepping Driver with Translator and Overcurrent Protection", A4988 datasheet, 2014.
- [64] Frequencies of Musical Notes, A4 = 440 Hz", Pages.mtu.edu, 2017. [Online]. Available: <https://pages.mtu.edu/~suits/notefreqs.html> [Accessed November 2017].
- [65] "analogWrite()", Arduino.cc, 2017. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/> [Accessed November 2017].
- [66] "Texas Instruments Website," 2015. [Online]. Available: <http://www.ti.com/lit/ds/symlink/lm555.pdf> [Accessed November 2017].
- [67] "Analog Devices Website," [Online]. Available: <http://www.analog.com/media/en/technical-documentation/data-sheets/AD9837.PDF> [Accessed November 2017].