VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

MechBass

Design and Implementation

James McVay

Chapter 1

Design

The design approach taken to develop each of the subsystems needed to play a single string is outlined in this section. As the only difference between each string played is their gauge, four of these single-string mechanisms are later constructed and integrated into a system to achieve the goal of being able to play four strings simultaneously. This implies the requirement of repeated construction, hence ease and repeatability of manufacture, as well as reliability and consistency of the constructed system are kept in mind during each of the design iterations. Where possible, improvements relevant to any of these considerations are made during each of the revisions. Finally, each of these units is assigned a MIDI channel and connected on a MIDI bus so that each can be individually controlled by sending MIDI messages with the appropriate channel for the desired unit.

1.1 Single-String Unit

Each string requires a way to be mounted, its pitch shifted, picked with variable velocity, damped, and the signal picked up for amplification. As the components of each string are viewed as modular, they are designed independently in an iterative manner, later integrated together into a single-string unit. A base on which all the components required for a single-string unit is needed, hence the 80/20 aluminium extrusion discussed in the pitch shifter section below is utilised as a base rail for each string-unit due to its strength and mounting flexibility.

1.1.1 Pitch Shifter

A guitar string's pitch is changed by altering the length of it that is plucked, hence a pitch shifter to implement this is required. There are two stages in realising the pitch shifter: a way of moving the mechanism that will fret the string, and the fretting mechanism itself. Existing implementations demonstrate that the pitch shifter is typically the bottleneck limiting the speed and accuracy of pitch selection, and therefore emphasis is placed on its development in this project.

In order to move along the string, a linear motion system is selected due to its simplicity and speed. Commercial solutions for a linear motion system were researched and dismissed due to their high cost, starting at $1000 per unit. Instead, a custom solution utilising 80/20, a modular aluminium extrusion solution illustrated in Figure 1.1, was pursued due to its relatively low cost, customisation options, and leniency in any imprecisions resulting from manufacturing and assembly of the system. 80/20 offers an aluminium carriage that slides along 80/20 aluminium extrusion using low friction bearing pads, and this can be used to

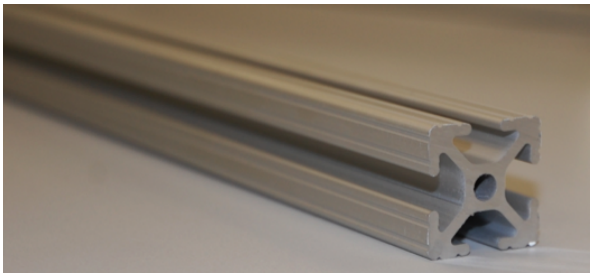

Figure 1.1: 80/20 aluminium extrusion

create a linear motion system by attaching custom brackets at either end of the extrusion. As shown in the initial design concept in Figure 1.2, these brackets house toothed pulleys, one of which is attached to a motor. These pulleys attach to the carriage through a belt, and actuating the motor moves the carriage. The carriage then provides a base on which a fretting mechanism can be constructed, as discussed later in this section.

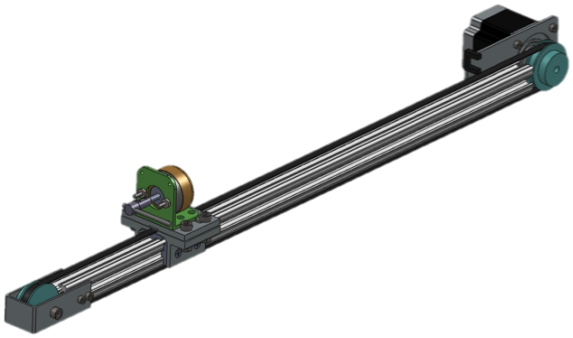

Figure 1.2: Conceptual custom linear motion system

Upon assembly using the aluminum carriage offered by 80/20, it was found that there was an undesireable amount of static friction between the pads of the carriage and the extrusion acting as a rail. Inspecting 80/20's CAD model of the carriage, it was discovered that this was not a fault, but part of the intended design of the 80/20 carriage. The amount of static friction that existed using this carriage could not be easily reduced, and for the purposes of this project, a linear motion system with minimal static friction is necessary as part of maximising the speed of the system. This resulted in a custom carriage being designed and constructed from 6 mm Perspex using a laser cutter, the final revision of which is seen in Figure 1.3. This carriage makes use of the existing bearing pads, but is not so tight on the sliding rail as to create a large amount of static friction. Thin plastic shims are then inserted behind the pads to reduce the amount of clearance between the bearing pads and the pitch shifter rail as necessary. Successive revisions of the carriage address issues discovered in previous revisions, from the way the belt is attached and tensioned, to the overall size of the carriage being revised to make it as compact and lightweight as possible.

As mentioned, a major contributing factor to the speed limit in existing solutions was that the pitch shifter would be permanently fretting the string, increasing the amount of force required to change pitch. As part of this project, the pitch shifter must include an on-demand fretting as part of the pitch shifting mechanism. Linear solenoids, rotary solenoids and hobby servos were all considered as the actuation device for this purpose, as all are devices that will reliably respond to control signals sent to them, requiring no feedback to the system driving it. Ultimately, linear solenoids were selected due to their greater simplicity,

speed, and clamping force over the other mechanisms considered. As also seen in Figure 1.3, the mechanism works by clamping a bridge attached to the two solenoids down onto the string, pulling the string onto the static fret of the carriage hence mimicking a guitarist's finger pressing the string onto a fret on the body of a bass guitar. Springs on the solenoids then release the fretting mechanism when power is no longer applied to the solenoids, allowing the carriage to move to a new position without the extra friction of continuously fretting the string. As both solenoids on either side of the carriage need to attach to the power supply from the actuator management board discussed in subsection 1.1.6, a block through which the wires for the second solenoid could pass through at the bottom of the carriage was manufactured using 3D printing. This block also provides extra structural support for the carriage, making it more rigid.

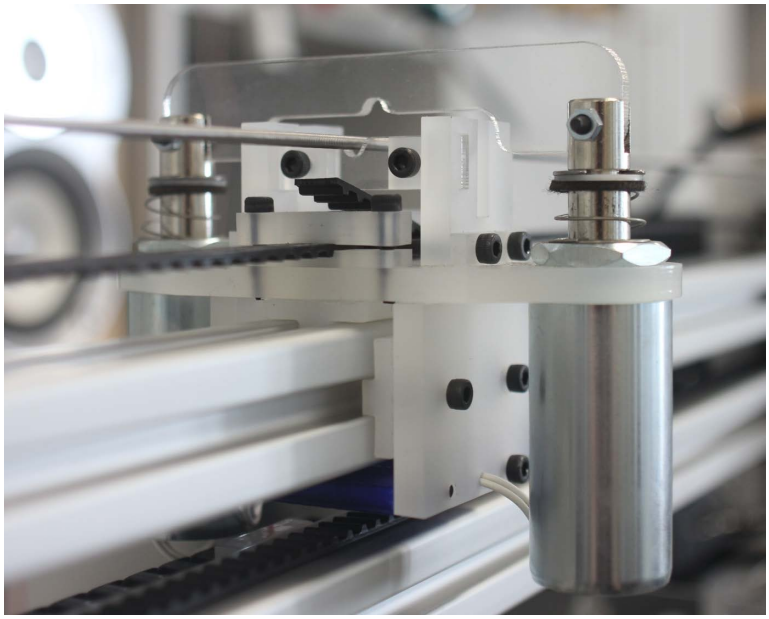

Figure 1.3: Pitch shifter carriage

Brackets to attach to the linear motion rail were designed to house the pulleys and motors that drive the system, as well as limit switches. An open loop system was decided upon due to its simplicity in terms of control, as it requires no explicit feedback from the linear motion system in order to know the position of the carriage. This is possible as the system has been designed with low friction in mind, hence by using a stepper motor and assuming this low friction ensures there is no slippage in the system, the position of the carriage will always be known in terms of stepper motor steps from what is deemed the home position. Limit switches are used for alignment of the system upon startup, as well as to prevent the carriage being over driven and potentially cause damage in the event the system unexpectedly loses track of the carriage position. The switch closest to the motor is used for alignment of the system, positioning the carriage to the home position upon start-up. The second limit switch contained within the brackets at the other end of the linear motion rail is used as a safety so that should the system become misaligned and the carriage hit the second switch unexpectedly, the system will recalibrate itself.

The bracket to which the motor is attached is designed to accommodate NEMA 23 form factor stepper motors, one of the most common sized motors used for driving linear motion systems that are available with a variety of torque and maximum speed ratings. Toothed pulleys are selected that have a diameter to ensure that the drive belt will clear the profile of the 80/20 rail and attach to the carriage. The first pulley driving the system is mounted to

the shaft of the motor and positioned using a grub screw, while the idler pulley is mounted on a stainless steel rod suspended by flanged bearings in the mounting brackets.

Seen in Figure 1.4, E-Clips attach to grooves in the rod position it between two bearings, with the idler pulley fixed on the rod using a grub screw. After functional testing of the pitch shifter mechanism using a low cost NEMA 23 stepper motor with a top speed of 200 RPM, a more powerful, NEMA 23 stepper motor with a top speed of 600 RPM was selected to drive the linear motion system component in order to significantly increase the speed of the pitch shifter three fold.

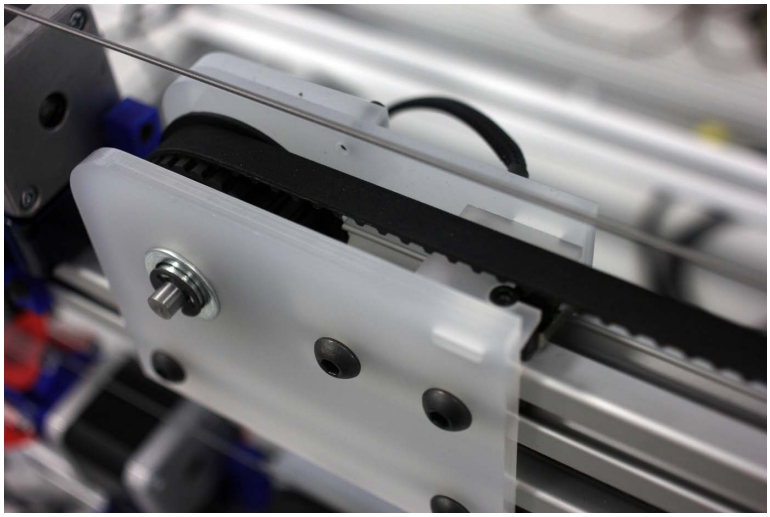

Figure 1.4: Idler pulley configuration

As the fretting system is powered on a carriage that moves, a way of supplying this power to the carriage that would not become tangled or caught is neccessary. Typical linear motion systems utilise what is known as a cable carrier. Comprising of a chain-like structure that is hollow in the middle, the carrier runs alongside the carriage of the system, with the excess resting on an adjacent platform. The wires used to power the solenoids are fed through the cable carrier, guided and enclosed during movement to prevent them becoming tangled.

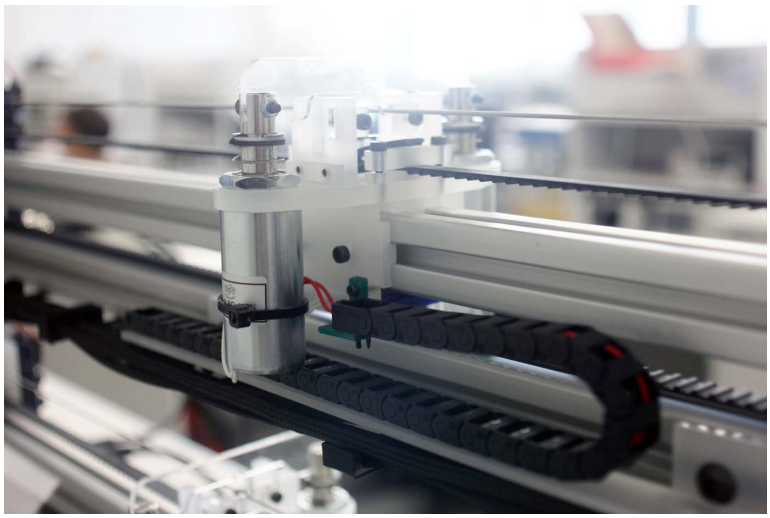

Figure 1.5: Cable carrier for the fretting mechanism

As shown in Figure 1.5, suitable cable carriers and attachment components are utilised in the pitch shifter, with a bracket to attach one end of the carrier to the carriage manufactured by 3D printing. A platform attached to the base rail of the single-string unit provides a mounting point for the other end of the carrier, as well as a place for the excess carrier to rest while in use.

1.1.2 String Mounting

Each single-string unit needs to support and tension a bass string using the traditional machine head used on a guitar. Like all components in this project, the string mount and tensioner went through numerous design iterations. As a result, the machine head used to tune the string is integrated into the top of the driving end of the pitch shifter's brackets.

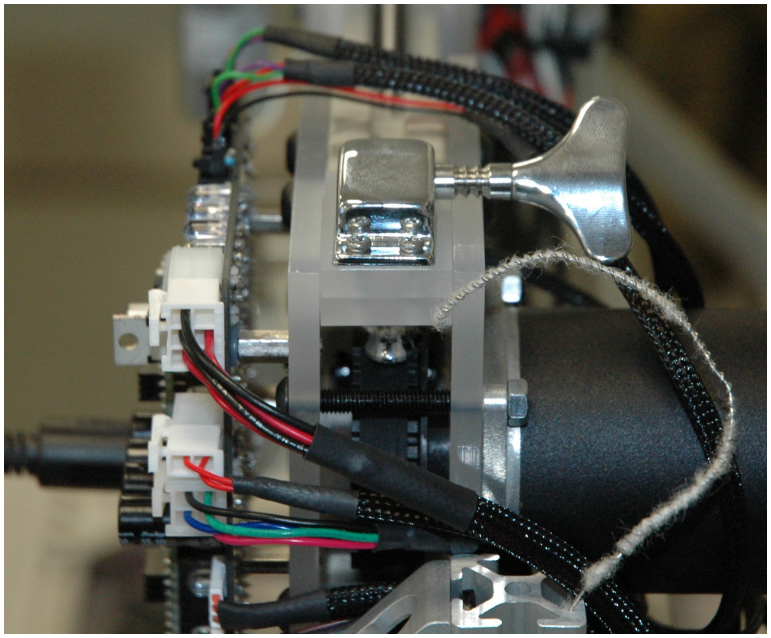

Figure 1.6: Mounting of the machine head

The string runs over a perspex bridge and through a short length of 80/20 into a standard bass machine head, all integrated into the top of one end of the pitch shifter's bracket as shown in Figure 1.6, . The block of 80/20 between the bridge and machine head is added to increase rigidity of the assembly. Similarly on the retaining end of the string mount, the string passes through a plate behind a length of 80/20 extrusion and through the extrusion, then over a perspex bridge. The retaining ball of the string holds the string in place behind the plate, enabling the tensioning and tuning of the string through the machine head at the other end of the string. The mount is designed for flexibility in the gauge of string that can be mounted, with 0.035 and 0.110 gauge strings being tested and tuned to their standard tuning with this system. A typical bass guitar utilises four strings with gauges between 0.035 and 0.110, hence the string mounting system is capable of supporting all of the string gauges found on a typical bass guitar.

1.1.3 Picking

A string is sounded by picking it, and the height at which the pick crosses the string has a direct effect on the dynamics of the sound made. Being able to control this parameter,

known as picking velocity, is one of the explicit goals of this project. Previous implementations utilised solenoids or some form of motor to actuate the pick against the string. In Vindriis' BassBot, an analysis of different picking mechanisms was conducted, and a pickwheel housing 8 picks attached to a NEMA 23 form factor motor was deemed superior to the smaller NEMA 17 based pickwheel, and solenoid mechanisms. While it was clear that wheel-based picking mechanisms were more reliable, only two types of motor were compared in Vindriis' analysis. The NEMA 17 motor included in this comparison was of low torque and limited speed, severely reducing the maximum speed at which it could reliably pick the string.

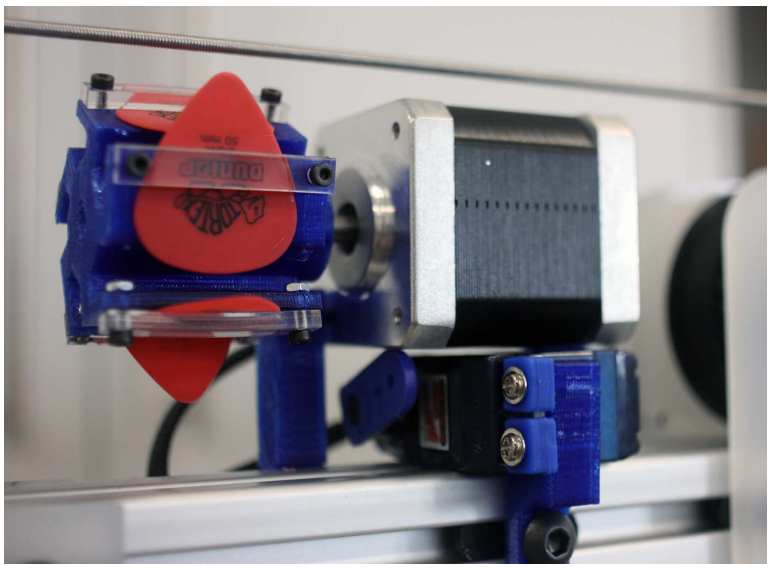

Figure 1.7: A NEMA 17 pickwheel with servo velocity control

An effective method of adjusting the height at which the string is picked is to mount the motor driving the pickwheel on a pivot, and place a servo motor under the motor. The arm of the servo then moves the motor around the pivot point, adjusting the relative height at which each pick can strike the string. Due to the way the string is mounted in a single-string unit, it is physically not possible to place a servo and the NEMA 23 factor stepper motor used in Vindriis' project directly below the string. To overcome this issue without reducing the maximum speed at which the string is picked, a NEMA 17 motor with similar specifications as the NEMA 23 deemed best in Vindriis' project is used to drive the pickwheel. This allows enough room for mounting of both the motor and the velocity adjustment servo beneath the string as shown in Figure 1.7, without sacrificing picking speed.

The brackets used to attach the stepper motor and servo are designed with the appropriate dimensions, and manufactured using the same 3D printed employed in creating components for the pitch shifter. Also shown attached to the stepper motor is a pickwheel produced in the same manner, with five fins to which guitar picks can be mounted, and a grub screw to secure the wheel on to the shaft of the stepper motor. This is achieved using laser cut pieces of perspex bolted onto each fin, clamping the pick between the perspex and the fin. This enables population of the pick wheel with any standard guitar pick, which is important as there is a large variety of guitar picks which result in different characteristics of the string's tone when picked.

To pick the string, the stepper motor must simply be rotated the correct number of steps. As the motor rotates 1.8° per step, then it takes $\frac{360}{1.8} = 200$ steps per entire revolution. With five picks on the pick wheel, the motor must turn $\frac{200}{5} = 40$ steps to pick the string once.

Torque is of higher priority than speed in selecting the servo for velocity adjustment, as it is expected that the servo would be given the velocity at the same time as the pitch shifter commanded to move to a new position, inherently taking longer than the servo to complete the action.

1.1.4 Damper

In order to mute the string on demand, some form of damper is required. Simply put, the damper needs to touch the string to reduce its vibration. Many of the existing implementations either had no damper, or a rudimentary solenoid based damper that is either on or off. These implementations offer very little control over the amount of damping, with no analysis of the resulting effectiveness. A finer control over damping is desired for the system, resulting in a servo based approach.

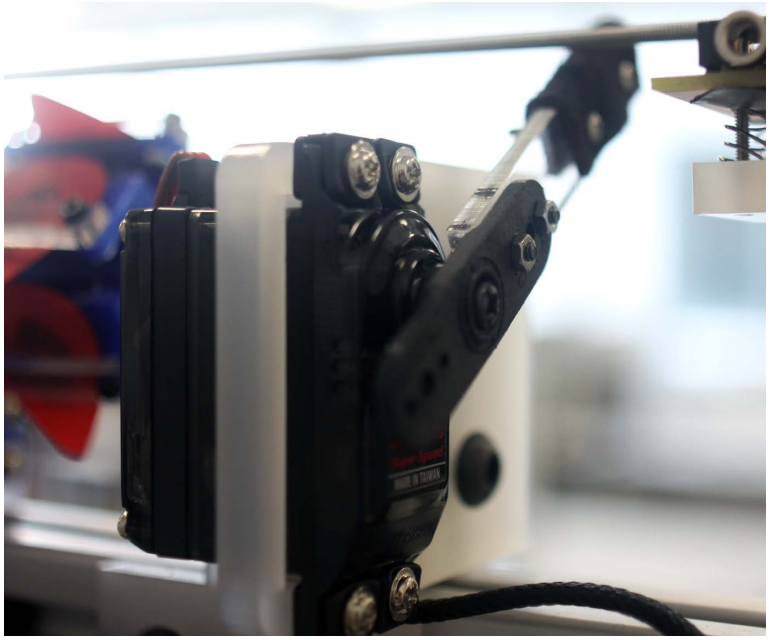

Figure 1.8: Damper mechanism

As shown in Figure 1.8, a servo with an extended arm covered in wool felt is employed. The position of this arm is controlled by changing the angle of the servo, in turn getting closer to the string and damping its vibration by touching, or moving further away such that it is completely undamped. Speed in damping is regarded as the most important factor in servo selection, as the system should respond to a MIDI *noteOff* command as quickly as possible. Hence a servo with the highest speed for less than $20 is utilised for the damper component.

1.1.5 Optical Pickup

A typical bass guitar uses a magnetic pickup to capture the vibration of the string. However due to the stepper motors and other electronics that are involved in this project, a traditional magnetic pickup cannot be used due to the amount of electromagnetic noise these devices generate. Instead an optical pickup is employed to capture the vibration of the string for amplification.

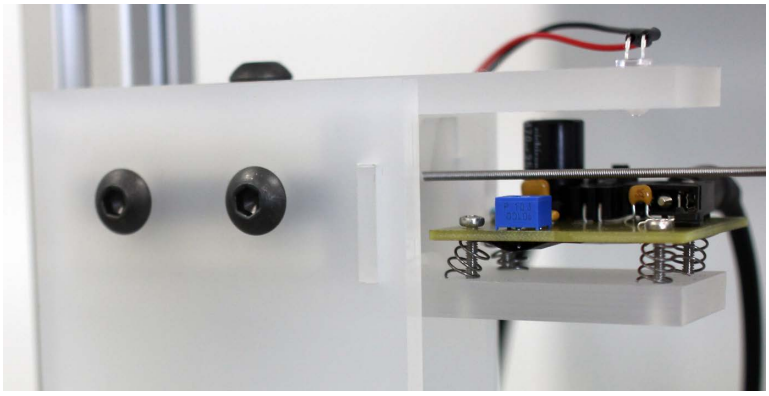

Figure 1.9: Optical pickup

The optical pickup is integrated into the retaining end of the string support mechanism, synonymous to where the pickup would be located on a typical bass guitar. As shown in Figure 1.9, the top plate houses an IR emitter, while a PCB containing an IR sensitive phototransistor is affixed to the bottom plate using springs and screws. The springs and screws allow adjustment of the distance of the phototransistor relative to the string, as different string gauges have an effect on the sensitivity and response of the pickup, an aspect that should be adjustable for the target environments of the system.

In order to supply the optical pickups with power for operation, a simple 3.3V linear dropout regulator is employed with decoupling capacitors. Additional decoupling capacitors are present on each of the optical pickup boards to ensure that noise on the power supply to each is kept to a minimum. This is important, as any noise present in the power supply is likely to have an impact on the amount of noise present in the output from the pickups.

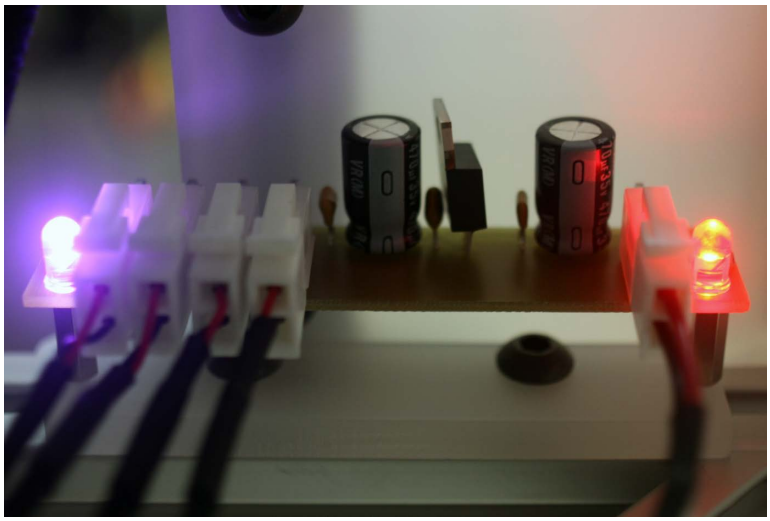

Figure 1.10: Power supply for the optical pickups

Shown in Figure 1.10, the power supply takes a 5 V input on the right, and supplies 3.3 V through 4 connectors on the left, one for each pickup. Similarly to the actuator management board in the next subsection, LEDs give a visual indication as to whether the supply is operating correctly for ease of debugging during operation.

1.1.6 Actuator Management

The ability to control all of the components that comprise a single-string unit necessitated the design, manufacture, and population of the acutation management board PCB shown in Figure 1.11 to provide the control to achieve the desired functionality. The design uses an Atmel ATMega328 microcontroller as the core in order to allow firmware programming for each board through ISP (In System Programming) using an AVR ISP MK II and the Arduino IDE. Arduino compatibility is desired as there are a large number of libraries available for this particular IDE, such as the AccelStepper library for acceleration and deceleration of stepper motors, the Servo library for control of servos, and MIDI libraries for implementation of the MIDI standard. Various microcontrollers were considered, however the ATMega328 was selected due to its simplicity to implement, requiring as little as a crystal and power source for operation, while offering all of the functionality that is required for control of each single-string unit.

In the first revision of the board, flexibility as to what could be connected was one of the main considerations. While it had been decided that the pickwheel and pitch shifter would be driven using stepper motors, the method for fretting the pitch shifter and actuating the string damper had not been decided. This led to a board design that contained the flexibility that allowed the attachment of either solenoids or servos. By the final design, revision 3, the form of actuators for these components had been decided upon and the PCB tailored to suit.

Instead of designing a stepper driver for the pickwheel and pitch shifter modules, commercial drivers were selected and used as daughter boards on the management PCB. This is done as reliable and accurate open loop control of the stepper motors is required, and using existing driver boards abstracted away the complexity of controlling the stepper motors in order to achieve this at minimal cost. The motors employed for both the pitch shifter and picking mechanisms can operate on up to 30 V, and the drivers are able to handle an absolute maximum of 35 V. 24 V was therefore selected as the supply voltage for the stepper motors. Operating close to the maximum of the drivers was undesirable as transients that could easily be generated by the stepper motors when being braked could be greater than 35 V, damaging the drivers. A combination of lower voltage, a bank of decoupling capacitors on this stepper motor supply rail, and transient voltage suppressors in close proximity to the motor supply connection to the motor driver daughter boards are used to protect the drivers from damage. To enable debugging, LEDs are attached to the stepper motor driver control lines to enable the visual indication of their status. As board space is limited, a single RGB LED for each driver board indicates the status of each of its three control lines; step, enable, and direction.

As the solenoids are high current devices requiring a 24 V supply, the 5 V 40 mA output of the microcontroller clearly cannot power such a device. An N-Channel MOSFET is used to switch the solenoids as desired, with the MOSFET's controlled via the output of the microcontroller and power supplied from the power rail for the solenoids. Flyback diodes are employed to minimise flyback present in inductive loads when its supply voltage is suddenly removed. Without such protection, transient voltages generated by the flyback would damage the MOSFET, and in-turn the microcontroller itself.

As discussed, each single-string unit is controlled through MIDI messages, with each unit being assigned a channel. As such, the standard method for interfacing with MIDI devices as defined by the MIDI Manufacturers Association is implemented on the actuator management PCB with an optoisolator used as specified to isolate the MIDI *In* from the UART pins of the microcontroller. However it is also desired to retain serial functionality to allow control of each unit through serial commands during testing. This resulted in a header that could either be attached to an external UART to USB converter to enable serial commu-

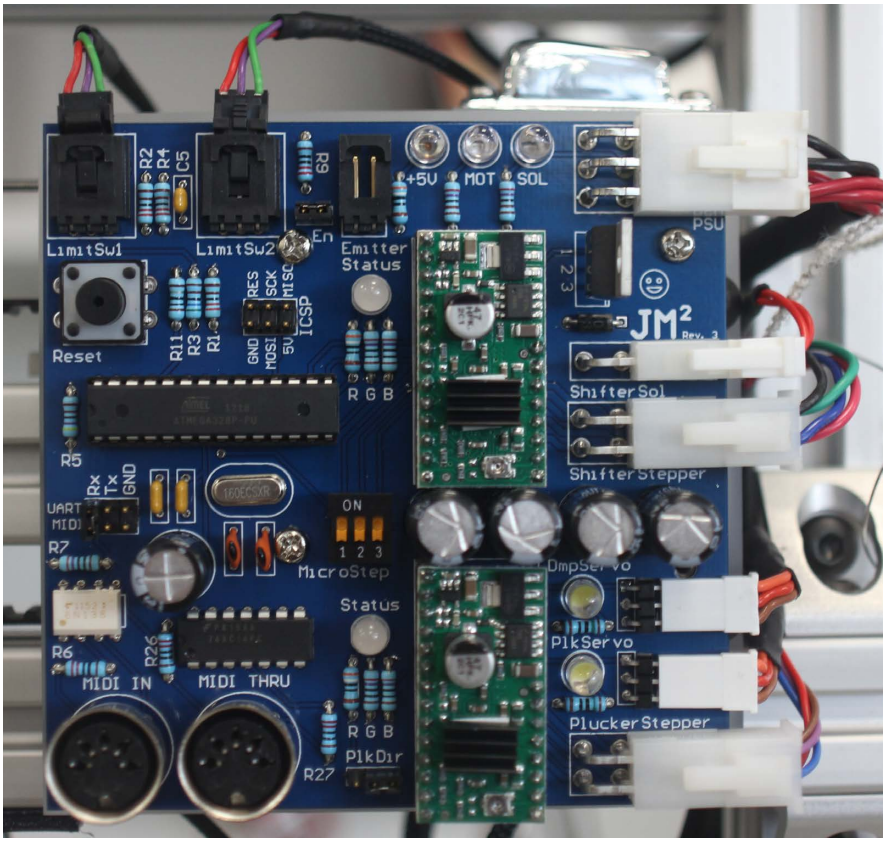

Figure 1.11: Actuator management PCB

nication, or jumpered on the board to enable MIDI functionality. To allow the connection of single-string units in a bus arrangement, a MIDI *Thru* port is implemented to pass the MIDI signals onto the next PCB in the chain.

Finally, three voltage rails exist on the board, one for logic and servo power, another for the solenoids, and the last for the stepper motors. These are supplied through a single connector from the power supply discussed in the next section of this Chapter, with indicator LEDs for each rail to show that the board is receiving all the power necessary for correct operation. For modularity and to ease the connection of different components to the management PCB, suitable connectors capable of handling the relevant currents for each actuator are employed on the board, with the mating connectors used on each actuator.

1.2 Four String System

The complete MechBass system comprising four single-string units requires a power supply capable of supplying all four units, a frame on which each of the units can mounted together in some logical form, and a simple way of interfacing the system with standard MIDI software and hardware.

1.2.1 Power Supply

As mentioned previously, three power rails are required for each unit. When designing mechatronics systems, it is good practice to over specify power related components. Firstly, 5 V is needed for powering the logic of each actuator management PCB and servos, with

peak currents of 1 A per servo and 500 mA for logic. Overall, $(1 + 1 + 0.5) \times 5 \times 4 = 50$ W of 5 V power is required for the system, leading to the selection of a 100 W power supply unit. 24 V is required for both the solenoids and stepper motors, however the supplies for each are isolated to minimise transients on the supply line. Each solenoid has a rated current of 0.5 A, resulting in a total requirement of $0.5 \times 2 \times 24 \times 4 = 96$ W of power. Taking into account peak current requirements, a 24 V 150 W power supply is employed to supply all of the solenoids. Finally, 24 V had previously been selected for both of the stepper motors employed in each single-string unit. As the pitch shifter and picking mechanisms are never actuated at the same time, the largest current requirement of the two motors is considered, being that of the pitch shifter. As this motor has a peak current requirement of 2.8 A, a power supply capable of at least $2.8 \times 24 \times 4 = 268.8$ W is needed. Again this power supply is over specified, with a 450 W 24 V unit being employed.

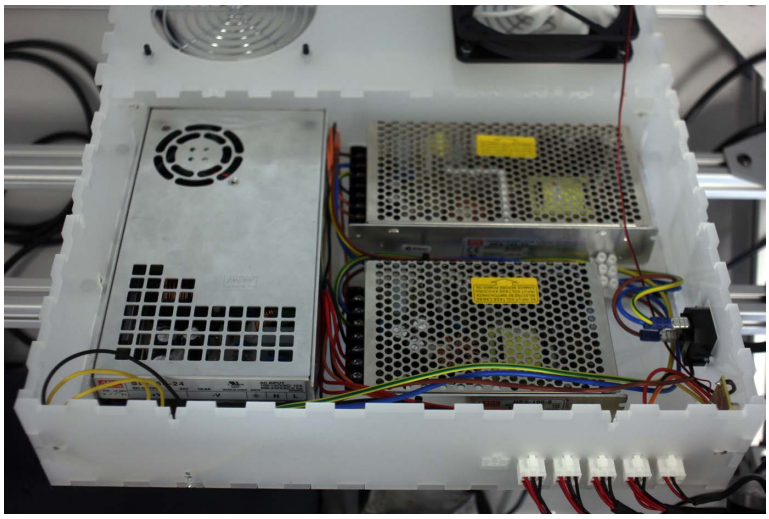

Figure 1.12: Power supply module

Each actuator management PCB possesses a six pin Molex connector through which all three power rails are supplied and to which each of the three power supplies selected must be connected. As the power supplies must be secured for safety (so as to not have mains power exposed), a power supply module to house all three units and supply their power through a matching six pin Molex connector is designed. As seen in Figure 1.12 with the top lid removed, the three power supplies are contained within a perspex enclosure, with the output of each connected to a PCB with six Molex connectors that are externally accessible. This allows cables that connect between each of these Molex connectors and those on the actuator management PCBs to supply all of the necessary power for correct operation of a single-string unit. An IEC socket at the back of the power supply supplies mains power to each of the power supply units inside, fused for protection against short circuits. Thermal management is implemented via a low noise fan and exhaust grills on the top of power supply that prevent the individual units from over heating during operation. Finally, a small PCB with LEDs representing the status of each power supply unit is mounted next to the IEC socket at the back of the power supply case to give a visual indication that all units are working correctly.

1.2.2 Assembly Frame

As each single-string unit is built upon a length of 80/20 extrusion acting as base rail, a frame capable of attaching all four of these rails and supporting their weight is required. A

vertical stack was selected to enable ease of access to all four units when mounted, as well as providing an appealing visual appearance. A frame based upon 80/20 components was designed, due to ease of assembly and mounting compatibility with the existing base rails of each single-string unit.

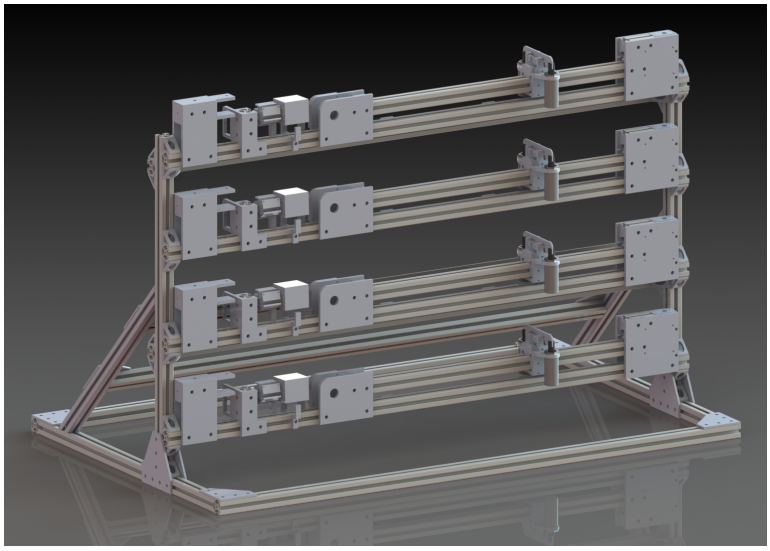

Figure 1.13: Render of the assembly frame

As seen in the render in Figure 1.13, a base comprising from 80/20 extrusion and suitable brackets is designed on to which two lengths of extrusion are vertically mounted. Each of the single-string units are attached across these vertical lengths, with cross members at 45 degrees between the vertical extrusions and the base for extra support. Lengths of extrusion between the two cross members also offer a suitable mounting place for the power, the connectors of which are in the position closest to the actuator management PCBs for supplying the power to each of the single-string units once mounted.

1.2.3 Complete Control

Finally, complete control of the system using standard MIDI is needed. As discussed, each single-string unit is assigned a MIDI channel, with MIDI *In* and MIDI *Thru* connectors present on each actuator management board. Hence, a MIDI bus can be created by connecting the output from a MIDI controller to any of the actuator management boards, and daisy chaining the remaining boards by connecting the MIDI *Thru* of the previous board in the bus to the MIDI *In* of the next.

The design of the system offers two mechanisms for control of the MechBass system. First, the control notation can be targeted, with the commands for each single-string unit being manually written and sent to the channel corresponding to the targeted unit. While this is suitable for testing purposes, it may not always be sufficient for musicians. It is common for MIDI notations to comprise a lone complete control sequence that is targeted for a generic device, resulting in the entire notation being transmitted on a single MIDI channel. In order to make these type of notations compatible with the MechBass system, the individual MIDI messages comprising the sequence must be assigned on the fly to a MIDI channel corresponding to a single-string unit capable of carrying out the command. This can be achieved using the software ChucK, with a program to act as an interface between the MIDI controller and MechBass to translate the *noteOn* commands based on which single-string unit is capable of playing each desired note.

Chapter 2

Implementation

Following their design, the subsystems are then manufactured and implemented into a complete four string system. The important aspects of construction at each stage are outlined, followed by the steps undertaken in setting up the hardware and actuator management firmware.

2.1 Construction

Construction of the system begins with the assembly frame outlined and designed previously, to which the base rails of each single-string unit are attached.

2.1.1 Assembly Frame

As designed in subsection 1.2.2, lengths of 80/20 extrusion are cut to length and assembled using the specified brackets. The rectangular base of the frame is constructed first, squared using a T-square and measuring tape before tightening of the bolts of each bracket that keeps the base together. Next, the vertical extrusions to which each single-string unit is mounted are attached using their brackets, horizontally aligned with each other to within a 1 mm tolerance using a caliper, and the 45 degree extrusions that support these vertical mounting extrusions are attached and tightened. Once the assembly frame is constructed, the base rails for each single-string unit are mounted onto the vertical extrusions, spaced evenly as shown in Figure 2.1.

Figure 2.1: Constructed assembly frame

Horizontal extrusions are then attached between the 45 degree angled supports. As seen in Figure 2.2, the system's power supply module mounts to these two extrusions between the angled supports. Given that the brackets for the assembly frame are not removed and stay aligned, the assembly frame can be disassembled to a state where it can be packed flat in 15 minutes, and reassembled in a similar time.

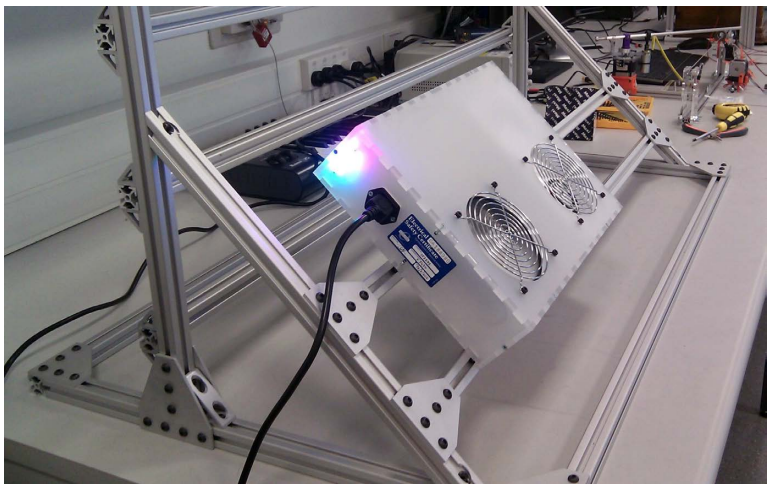

Figure 2.2: Power supply mounted on horizontal extrusions

Finally, all the bolts of the frame are re-tightened, and the level of each of the single-string units base extrusions checked before moving onto their construction.

2.1.2 Multiple Single-String Units

The parts necessary for the construction of each of the four single-string units are manufactured using 3D printed PLA (polyactic acid) and laser cut acrylic, with holes drilled and tapped to size in the relevant parts as designed. Actuator cables are extended and sleeved, with the appropriate mating connections added for attachment to the management PCB. Next, each of the brackets are bolted on to each of the base rails. Starting with the pitch shifter brackets and working down to the retaining end of the string support, actuators are attached to the brackets as required. With the pitch shifter mounted, the pulleys are attached to their respective stepper motors and idler rods, and the belt for each carriage installed and tensioned.

With the physical components attached to all four single-string units, the actuator management PCBs are populated and tested sans the motor drivers so as to not risk damaging them as they the most expensive components on the board. Once populated, the power lines are tested for shorts to ground, as well as between one another through resistance checking with a multimeter. Next, the PCB's are connected to the power supply and programmed with test code that visually shows each of the output lines working through the debugging LEDs. Following this, the motor drivers are installed into their sockets on the PCBs, and each management PCB is then attached to the front acrylic plate used for pitch shifter and string support components. With the PCBs mounted, all of the actuators and limit switches for each single-string unit are connected respectively, with each PCB being connected to the system power supply.

Finally, the management PCBs are interconnected using MIDI cables, taking the MIDI *Thru* from one into the MIDI *In* of the next, and the MIDI *In* of the PCB at the bottom start of this MIDI bus. This is the input to the system, and will be attached to the appropriate

control software and hardware once the system is configured for complete control of the MechBass system.

Once the construction of the system is complete with all units configured identically, the cables for each single-string unit create a complication. As each of the single-string units are vertically stacked, the cables from the units above droop down onto those below.

With large number of moving components and hence points for these cables to get tangled, a way of managing all of the cables in the system is necessary. Small cable clips are therefore designed that are made to snap in to the channels of the 80/20 extrusion, and attach to the bottom of each single-string units base extrusion to guide the cables as required.

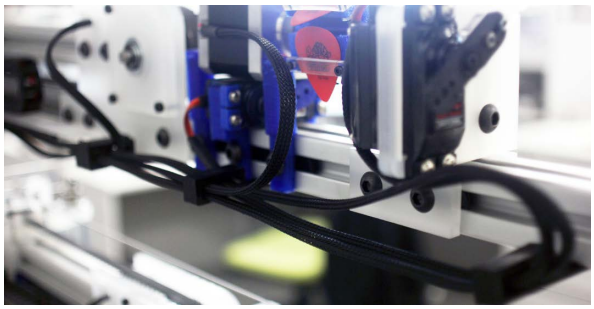

Figure 2.3: Clips for cable management

Shown in Figure 2.3, the cable clips effectively contain and guide all of the cabling from the actuators down to the management PCB for connection. This prevents the issues identified from occurring, while enabling easy removal of a single-string unit from the system.

2.2 Configuration

With the physical construction of the system complete as shown in Figure 2.4, it must be configured for use by implementing the firmware to control each single-string unit, mounting and tuning of the strings correctly, and writing interface software to enable the control of the entire system with standard MIDI software.

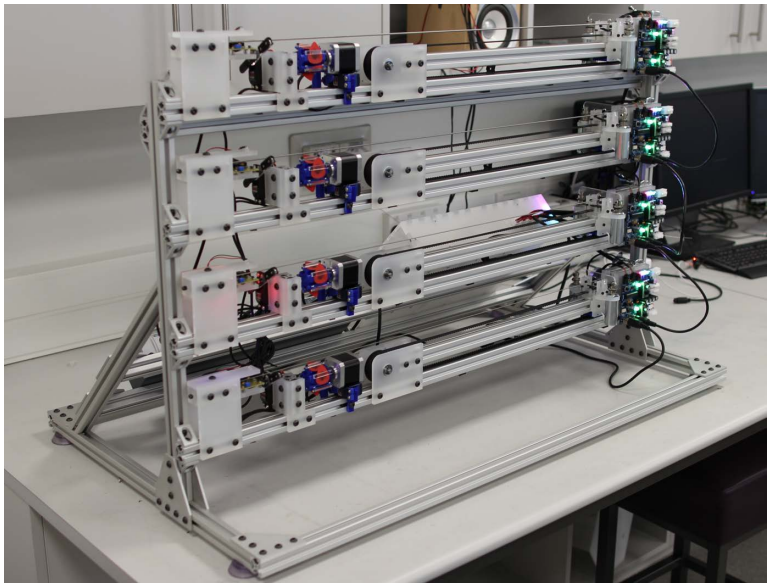

Figure 2.4: The constructed system

2.2.1 Actuator Management Firmware

To control each of the actuators that comprises a single-string unit, the firmware for the actuator management board must implement the desired functionality as discussed below. Upon power up, the pitch shifter must be moved to a specified home position before waiting for MIDI messages (handled by the MIDI library for Arduino), to execute a *noteOn* or *noteOff*. To find the home position, the shifter is stepped towards the goal until an interrupt from the homing switch is received. At this point, the shifter moves 15 steps away from the the home switch to what is designated as the home position.

In order to move the pitch shifter to the correct position for a given note, a lookup table is required containing the stepper position for a given fret. While this can be done by measuring the different frequencies sounded by a pick of the string with the pitch shifter at each of the possible steps from home, all of the parameters of the mechanical system are known, and the positions of the frets can instead be derived from these. The equation that determines the location of each fret from the nut on a normal bass guitar is given as frets are spaced along the neck at a distance from the bridge of $x = \frac{L}{k^n}$, where L is the scale length, x is the fret number, and $k = \sqrt[12]{2}$, the ratio of the spacing of two consecutive frets. In the case of a single-string unit in MechBass, the home position of the pitch shifter is the nut, as it is the point at which a string is fretted for tuning to its open string frequency. The distance between the bridge on the pitch shifter carriage at this position and the bridge on the retaining end of the string support (the open string length) is 822 mm. From this, the distance in millimetres for each fret n from the home position is given by $x = \frac{822}{1.05946^n}$, and in turn, the distance from the home position to move to given a desired fret is found as $822 - x$. The last step in creating the lookup table for the fret positions is to convert these distances from the home position to be in steps instead of millimetres. Given that the stepper motor driving a pitch shifter takes 200 steps to rotate 1 revolution, and the attached pulleys driving the belt of the pitch shifter have a diameter of 50 mm, the linear displacement of the pitch shifter for each step of the stepper motor is found to be $\frac{50 \times \pi}{200} = 0.78$ mm.

To allow simple adjustment of the lookup table should the values for the open string length and pulley diameters vary from what is measured, the equations for populating the look up table are carried out upon initialisation with the values for each of the parameters defined in the header of the firmware. Further, a MIDI note offset needs to be calculated dependant on what the open string note of a unit is as in the system there are four strings; G, D, A, and E, and while the single-string unit for each of these strings will have frets at the same positions, each unit has a different root note. This results in the range of MIDI notes playable on each unit differing, and as the look up table begins at index 0, an offset is needed to find the correct index in the table from the MIDI note. For example, the G string's root note corresponds to MIDI note 43 and the position of this note is at index 0 in the look up table. As a result, 43 must be subtracted from any MIDI note sent to this unit to find the correct index in the look up table containing the step position the pitch shifter must move to. Further, the number of frets physically possible is limited, hence the table is built to only contain the number of frets physically possible, and the noteOn routine prevented from attempting to look up values that outside the array bounds. Finally, each single-string unit should only respond to messages on a MIDI channel assigned to it. Each of the single-string units G-A are assigned a MIDI channel between 1 to 4, with the G string being assigned channel 1, up to the A string which is assigned channel 4.

The *velocity* value of a *noteOn* message is an 8-bit field corresponding to 128 levels of pick volume. As the velocity of a pick is controlled via the servo underneath the stepper motor driving the pickwheel, these 128 levels must be mapped to the range of possible positions on this servo. The Servo library for Arduino is used to control the servo, allowing

angles of 0 through to 180 degrees to be set on a servo motor. Through experimentation, it is found that the minimum value for the servo is 140, and the maximum value is 120. From this, the 128 levels of velocity representable in MIDI is mapped to the 20 possible servo positions by $\frac{V \times 20}{128}$, where V is the value of the velocity field in the MIDI message. As division is inherently slow on microcontrollers, bit shifting is instead employed, resulting in this mapping acheived using the expression $V \times 20 >> 7$. Similarly, the angles corresponding to damper on and damper off for the servo actuating the damper mechanism are experimentally found to be 146 and 143 respectively, with these values used to control the damper through the servo library as required. These 20 velocity are deemed sufficient, as the majority of Western music notation makes use of no more than eight of the corresponding dynamic values.

The stepper motors employed for the picking mechanism and pitch shifter both have a maximum speed of 600 RPM, or 2000 steps per second. In driving motors with a load, it is undesirable to actuate them in an on-off fashion, that is, from full speed to a complete stop and vice versa, as it is likely the motors will slip due to the applied load. In turn this has the potential to generate large amounts of operating noise, contributes to wear and tear on the system, and most importantly degrades the accuracy of the open loop system employed for the pitch shifter. Instead, the stepper motors should be more gently accelerated and deccelerated, gradually increasing and decreasing their speed during a movement. The *AccelStepper* library is employed to handle such control, allowing parameters for maximum speed and maximum acceleration to be supplied. The maximum speed is that specified by the data-sheets of the stepper motors, while the maximum acceleration is found experimentally.

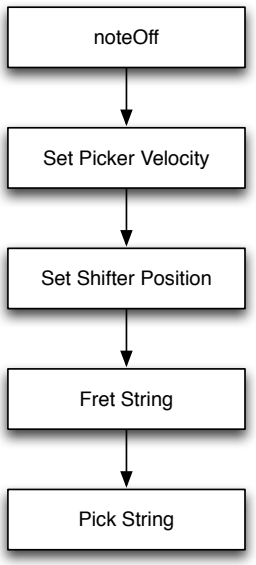

Figure 2.5: Execution of a *noteOn*

Once initialized and a *noteOn* message received by the MIDI handler, a callback function in the firmware steps through the states shown in Figure 2.5. A *noteOff* is first called which simply releases the pitch shifter fret by deactivating the solenoids, and enables the mute by applying the damper to the string. This is done explicitly at the start of a *noteOn* command to ensure that the string damper is not engaged, adding undesirable friction on the string during a pitch shift. Next, the velocity for the pickwheel is set, and the pitch shifter stepped to the position retrieved from the lookup table for the given note, before fretting the string.

Lastly, the damper is deactivated to move away from the string, and the pickwheel sent the appropriate number of steps to pick the string once. The setting of the velocity for the pickwheel is executed first, as this simply changes the duty cycle of the 50 Hz PWM signal being sent to the servo underneath the pickwheel stepper motor. This is quick to execute, and allows the servo to rotate to the appropriate position for the desired velocity before the picking of the string is executed as the last step of a *noteOn* sequence.

Finally, the capability of prepositioning the pitch shifter before a note is to be played is desirable for reducing the latency of a *noteOn*. The *noteOn* routine is modified such that when a *noteOn* with a *velocity* of 1 is received, the pitch shifter moves to the relevant fret, but does not play the note. A second *noteOn* with the same note at a later time then allows the note to be played with minimal latency, as the pitch shifter does not have to change position. With the firmware complete, the system is capable two modes of operation; on-the-fly and prepositioning. On-the-fly allows notations created without consideration for the MechBass system to be played, while prepositioning enables a lower latency performance using MIDI *noteOn* messages with a velocity of 1 to move the pitch shifter to the correct position before the actual note is played.

2.2.2 Control Software

With the firmware prepared on each of the single-string units of MechBass, one form of control of the system is already possible through MIDI. By knowing what notes each unit is capable of, compositions can be tailored to comprise of separate tracks for each single-string unit, and these tracks can then be sent to the appropriate MIDI channel for each unit. However it is also desirable for MechBass to play notations that have been been composed as a single track and hence can only be sent to a single MIDI channel. To allow this, an interface to map each of these MIDI messages to a MIDI channel corresponding to a single-string unit capable of playing the note is necessary.

Software written for ChucK is used to implement this, with ChucK acting as a MIDI device for the composition or notation software and then providing a MIDI out through any compatible hardware controller to the MechBass. Using the previous information about what MIDI note range each single-string unit is capable of, incoming MIDI messages are remapped on-the-fly by the software to the correct channel for playback on MechBass, being output on the MIDI hardware attached to MechBass.

2.2.3 Physical Tuning

The maximum speed a stepper motor can achieve is primarily determined by the current that is allowed to flow through its coils, and this current limit is controlled by a limiting trimpot on the stepper motor driver boards. From the datasheet for each motor it is found that at 24 V, both the picking mechanism and pitch shifter's motors obtain their maximum speed when a current limit between 2.2 A and 2.6 A is set. For each stepper motor, a multimeter is placed in series with one of the coils to measure the current with the coil active, and the trimpot on the respective driver adjusted to set the limit to 2.4 A.

With the system configured, tuning of the strings through the machine heads is necessary. Strings of typical gauges, 0.045, 0.065, 0.085, and 0.105, are mounted to the corresponding G, D, A, and E single-string units, being tensioned loosely with the resepctive machine head. Existing ChucK software for pitch detection is modified and run to continuously sample the audio input and calculate the frequency using a Fast Fourier Transform (FFT). The output of each pickup is attached to an audio input configured for use in ChucK, and a hardware MIDI controller attached to the MIDI input of the unit currently being tuned. With the

MIDI controller set to repeatedly send a *noteOn* command corresponding to the open string note of the currently attached unit, the current frequency of the string is output by ChucK. The string is accordingly tensioned using the machine head until the desired frequency is reached for the given string, being 98.0 Hz, 73.4 Hz, 55.0 Hz, and 41.2 Hz for the G, D, A, and E strings respectively, as these correspond to the typical tuning of a bass guitar.